

A Privacy-Preserving Logistics Information System with Traceability

Quanru Chen¹, Jinguang Han^{2(✉)}, Jiguo Li^{3,4}, Liquan Chen⁵, and Song Li¹

¹ College of Information Engineering, Nanjing University of Finance and Economics, Nanjing, China

cqrqx@qq.com, lisong@nufe.edu.cn

² Jiangsu Provincial Key Laboratory of E-Business, Nanjing University of Finance and Economics, Nanjing, China

jghan22@gmail.com

³ College of Computer and Cyber Security, Fujian Normal University, Fuzhou, China

⁴ Fujian Provincial Key Laboratory of Network Security and Cryptology, Fuzhou, China

ljg1688@163.com

⁵ School of Cyber Science and Engineering, Southeast University, Nanjing, China

Lqchen@seu.edu.cn

Abstract. Logistics Information System (LIS) is an interactive system that provides information for logistics managers to monitor and track logistics business. In recent years, with the rise of online shopping, LIS is becoming increasingly important. However, since the lack of effective protection of personal information, privacy protection issue has become the most problem concerned by users. Some data breach events in LIS released users' personal information, including address, phone number, transaction details, etc. In this paper, to protect users' privacy in LIS, a privacy-preserving LIS with traceability (PPLIST) is proposed by combining multi-signature with pseudonym. In our PPLIST scheme, to protect privacy, each user can generate and use different pseudonyms in different logistics services. The processing of one logistics is recorded and unforgeable. Additionally, if the logistics information is abnormal, a trace party can de-anonymize users, and find their real identities. Therefore, our PPLIST efficiently balances the relationship between privacy and traceability.

Keywords: Privacy Protection · Multi-signature · Pseudonym · Traceability · Logistics Information System.

1 Introduction

In recent years, with the rapid development of e-commerce, online shopping has become a popular trend. Online shopping is an interactive activity between a buyer and a seller, where after completing an order by a buyer, the product is delivered via a logistics system [26]. Logistics system helps to reduce product cost and save shopping time.

Unfortunately, the current LISs [33] cannot effectively protect users' privacy information. Users' personal information is clearly visible on the express bill and the LIS database [48]. Some data breaches in LISs released users' personal information, including addresses, phone numbers, transaction details, etc. If a user's personal information is leaked and maliciously collected, she may be at high risk of identity forgery and property fraud, in addition to the risk of being harassed by spam messages. Therefore, it is interesting and important to consider the privacy issues in LISs.

Furthermore, since a product is delivered by multiple logistics stations, it is important to record the whole logistics process and make the process unforgeable. Additionally, to prevent users from conducting illegal transactions, users can be de-anonymized [35,42] and traced.

In this paper, we propose a privacy-preserving logistics information system with traceability (PPLIST). Compared with the existing LISs, our scheme has the following advantages:

- 1) Users can anonymously use the logistics services in our PPLIST scheme. Users generate and use different pseudonyms in different logistics services. Even the internal staff of a logistics company can not directly obtain the information of users' identities, our PPLIST effectively protects users' personal information.
- 2) In the case that the identity of a user needs to be released, a trace party can de-anonymize a user and find his identity. This properties prevent users from conducting illegal logistics via a logistics system.
- 3) Our PPLIST scheme is efficient. Multi-signature is applied to record the delivery process and reduces the storage space.

Contributions: Our main contributions in this paper are summarised as follows: 1) The definition and security model of our PPLIST scheme are formalised; 2) A PPLIST scheme is formally constructed; 3) The security of our PPLIST scheme is formally reduced to well-known complexity assumptions; 4) Our PPLIST scheme is implemented and evaluated.

1.1 Related Work

In this subsection, we introduce the work which is related to our PPLST scheme, including LIS, privacy protection in LIS, multi-signature and pseudonym.

Logistics Information System LIS is a subsystem and the nerve center of logistics systems. As the control center of the whole logistics activities, LIS has many functions. The main functions of LIS are as follows: collect, store, transmit, process, maintain and output logistics information; provide strategic decision support for logistics managers; improve the efficiency of logistics operations [28].

Bardi et al. [6] pointed out that the choice of LIS directly affected the logistics cost and customer-service level. Lai et al. [27] showed that LISs is very important for a company to manage product inventory and predict the trend of customers'

online shopping. In addition, Ngai et al. [37] claimed that LIS is an information system that can promote a good communication between the companies and the customers. An LIS adoption model was proposed in [37] to examine the relationship among organizational environment, perceived benefits and perceived barriers of LIS adoption. In [15], Closs and Xu argued that the important source of enterprise competitive advantages was logistics information technology. Their research showed that companies with advanced logistics information technology and LIS performed better than other companies.

LISs have been proposed and applied into various application scenarios [1,2]. Amazon [1] is one of the first companies to provide e-commerce services. Amazon has a logistics system, which realizes the organization and operation of the whole logistics activities. Amazon has also added special technology, One-Click [14], in their LIS, which can automatically store the information of customers. Therefore, customers do not input their person information in each shopping. In addition, Amazon's LIS has the following functions [16]: order confirmation in time, smooth logistics process, accurate inventory information and optional logistics methods, etc. Amazon has become a business to consumer (B2C) e-commerce [25] company.

Taobao [2] is a consumer to consumer (C2C) e-commerce [51] platform. Taobao entrusts all logistics activities to a third party logistics company, but takes a series of measures to ensure the security of logistics activities. For instance, Taobao implements the network real-name system (NRS) [43] in their LIS, and has set up a special customer-service department to solve products logistics problems. Besides, Taobao has the functions of timely confirmation of orders and delivery within the specified time.

Privacy Protection in LIS Although the LIS of e-commerce platform brings convenience to people's life, it also brings great challenges to privacy protection. LIS stores a large number of users' personal information. Once the information is leaked, it will result in serious threaten to the life and safety of users. Some privacy protection methods in LIS have been proposed, such as [17,32,41,29,45,47,49]. We compare our scheme with these systems in Table 1.

Léauté et al. [32] proposed a scheme to ensure the privacy of users while minimizing the cost of logistics operation. The scheme formalizes the problem as a Distributed Constraint Optimization Problem (DCOP) [19], and combines various techniques of cryptography. But the disadvantage of this scheme [32] is that the anonymization of users is not considered. In [49], Frank et al. proposed a set of protocols for tracking logistics information, which is a light-weight privacy protection mechanism.

To solve the problem of privacy leakage caused by stolen express order number, Wei et al. [47] proposed a k-anonymous model to protect logistics information. However, the method only protects a part of users' personal information, because the names and telephone numbers of receivers are directly printed on the express bills for delivery.

Table 1. The Comparison between Our Scheme and Related Schemes

Systems	Anonymity	Traceability	Security Proof
Gao et al.[17]	×	×	✓
Léauté et al.[32]	×	×	✓
Qi et al.[41]	×	✓	×
Liu et al.[29]	✓	×	✓
Laslo et al.[45]	×	✓	×
Wei et al.[47]	✓	×	×
Frank et al.[49]	×	✓	×
Our PPLIST	✓	✓	✓

To improve the security of [47], Qi et al. [41] proposed a new logistics management scheme based on encrypted QR code [45]. After a courier scans the encrypted QR code by using an APP, the logistics information of products in the database is automatically updated through GPRS or Wi-Fi. The APP provides an optimal delivery route for couriers. However, the problem of [41] is that users' personal information is still visible to the internal staff of express companies. In addition, Laslo et al. [45] proposed a traceable LIS based on QR code. However, this scheme does not consider privacy protection.

Furthermore, Gao et al. [17] proposed a secure LIS, named LIP-PA, which can protect the logistics process information between different logistics stations, but the protection of users' personal information is not considered well. Hence, the privacy of users in LISs [17,41,45] was not fully considered.

Liu et al. [29] designed an LIS based on the Near Field Communication (NFC) [50] technology. In [29], users' personal information was hidden in tags, and only authorized people can access information. However, because of the limitation of computation power, the scheme cannot perform complex encryption and decryption processes.

In summary, above schemes addressed the privacy issues in LIS, but these schemes did not consider the track of delivery process and the trace of illegal users. However, these are important issues in LISs. Therefore, to solve these problems, we propose a new privacy-preserving LIS called PPLIST.

Multi-Signature Multi-signature, also called multi-digital signature, is an important branch of digital signature. Multi-signature is suitable to the case where multiple users sign on a message, and a verifier is convinced that each user participated in the signing [8].

Itakura [23] first proposed the concept of multi-signature, and proposed a multi-signature scheme with fixed number of signatures. Then, many multi-signature schemes were proposed [7,22,34,39,38,40]. The multi-signature generation time of schemes [23,40] is linear with the number of signers. Okamoto et al. [38] proposed a multi-signature scheme, but it, like scheme [39], only allows each signer in a group to sign the message. It's inflexible. Furthermore, Ohta and Okamoto [39] formalized the security model of multi-signature. However, this

scheme did not consider the security of the key generation process, so its security is not strong. Based on [39], Micali et al. [34] proposed a formal and strong security model for multi-signature. Bellare and Neven [7] proposed a new scheme and proved its secure in the plain public-key model. This scheme improved the efficiency of previous multi-signature schemes.

Since it enables multiple signers to collaboratively sign on a message, multi-signature has been used into various application scenarios, such as [36,46,4,9]. Shacham [30] proposed a sequential aggregate multi-signature scheme. The scheme computed the final multi-signature by sequentially aggregating the signatures from multiple signers. However, the data transmission of [30] is large. To solve this problem, Neven [36] presented a new sequential aggregate multi-signature scheme based on [30]. The scheme of [36] reduces signing and verification costs effectively.

Tiwari et al. [46] proposed a secure multi-proxy multi-signature scheme. It does not need pairing operations, and reduces the running time. The scheme is also secure against the attack of selected messages. However, Asaar et al.[4] found the scheme in [46] is insecure, and proposed an identity-based multi-proxy and multi-signature scheme without pairing. The security of this scheme was reduced to the RSA assumption in the random oracle model by using the Forking Lemma technique [5].

Recently, Dan et al.[9] proposed a new multi-signature scheme. Signature compression and public-key aggregation were used in the scheme. Therefore, when a group of signers signed a message, the verifier only needs to verify the final aggregate signature. The advantage of this scheme is that the size of final aggregate signature is constant and independent of the number of signers. Furthermore, this scheme is secure against rogue-key attacks. When constructing our PPLIST, we apply the scheme [9] to record the whole logistics process and reduces the storage cost.

Pseudonym Pseudonym is a method that allows users to interact anonymously with other organizations. Because pseudonym is unlinkable, it can effectively protect the information of a user's identity [44] among multiple authentications. The common pseudonym generation techniques are as follows [13]: 1) Encryption with public key; 2) Hash function; 3) Keyed-hash function with stored key; 4) Tokenization.

Chaum [10] found that pseudonym enables users to work anonymously with multiple organizations, and users can use different pseudonyms in different organizations. Because of the unlinkability of pseudonym, no organization can link a user's pseudonyms to her identities. Later, Chaum and Evertse [11] presented a pseudonym model scheme based on RSA. However, the scheme needs a trusted center to complete the sign and transfer of all users' credentials.

To reduce the trust on the trusted center, Chen [12] proposed a scheme based on the discrete logarithm assumption. The scheme also needs a trusted center, but the trusted center is only required for pseudonym verification. Although

Chen’s scheme is less dependent on the trusted center than the scheme [11], the trusted center was still required.

In order to enable users to have the initiative in the pseudonym system, Lysyanskaya et al. [31] proposed a new scheme. In this scheme, a user’s master secret key was introduced. If the master secret keys are different, the information of users’ identities must be different. In addition, the pseudonym certificate submitted by a user to an organization only corresponds to the user’s master public key and does not disclose the information of his master secret key.

Pseudonym has been applied in some schemes [20,24] to protect users’ privacy. To reduce the communication cost of traditional pseudonym systems in Internet of Vehicles, Kang et al. [24] proposed a privacy-preserved pseudonym scheme. In this scheme, the network edge resources were used for effective management, and the communication cost was effectively reduced.

In [20], Han et al. proposed an anonymous single sign-on (ASSO) scheme. In this scheme, pseudonym was applied to protect users’ identities. A user uses his secret key to generate different pseudonyms, and obtains a ticket from a ticket issuer anonymously without releasing anything about his real identity. Furthermore, a user can use different pseudonyms to buy different tickets and the ticket issuer cannot know whether two tickets are for the same user or two different users. In our PPLIST scheme, to protect users’ privacy, we apply the pseudonym developed in [20] to enable users to use logistics services anonymously and unlinkably.

1.2 Paper Organisation

The remainder of this paper is organised as follows. Section 2 presents the preliminaries used in our scheme, and describes the formal definition and security model of our PPLIST scheme. Section 3 provides the construction of our scheme. The security proof and implementation of our scheme are presented in Section 4 and Section 5, respectively. Finally, Section 6 concludes this paper.

2 Preliminaries

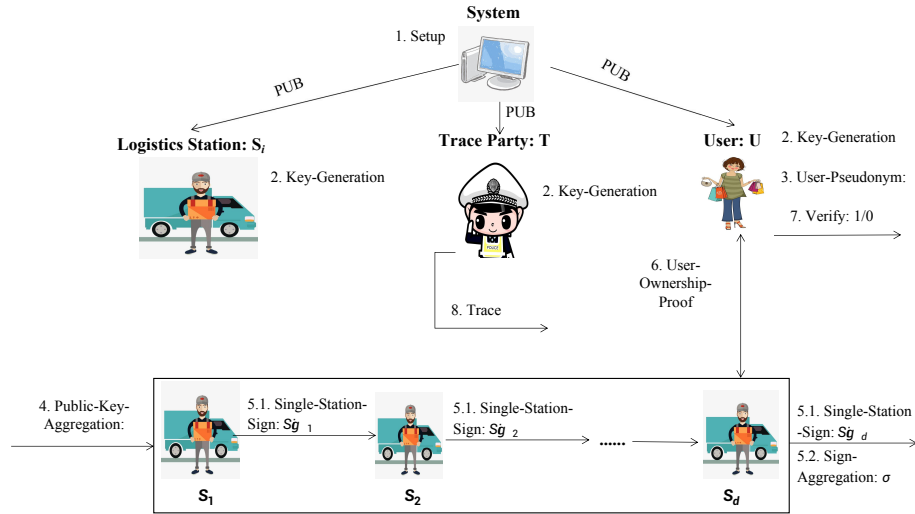
In this section, the preliminaries used throughout this paper are introduced, including bilinear group, complexity assumptions, formal definition and security model. Table 2 summaries the notations used in this paper

The framework of our PPLIST is presented in Fig. 1. The system first generates the public parameters PUB . Then, each entity (e.g. logistics station, user and the trace party) generates its secret-public key pair. Prior to ordering a service, the user generates a pseudonym by using his secret key. The system determines the delivery path, and then generates the aggregated public key of the selected logistics stations. After that, each selected logistics station S_i generates its single signature Sig_i on the product information, pseudonym and aggregated public key, and then passes it to the next selected logistics station. Finally, the last selected logistic station generates its signature and the aggregate signature

Table 2. Notation Summary

Notation	Explanation	Notation	Explanation
1^l	A security parameter	<i>Pseudonym</i>	The pseudonym of U
S_i	The i -th logistics station	PUB	Public parameters
U	User	PPT	Probable polynomial-time
T	The trace party	$\mathcal{B}(1^l)$	A bilinear group generator
YA	The aggregation of AgY	$x \xleftarrow{R} X$	x is randomly selected from X
AgY	A set of selected public keys	H_1, H_2, H_3	Cryptographic hash functions
σ	The aggregation of signatures	Sig_i	The i -th single signature
π	The proof of user's ownership	I	A set consisting of the indexes of selected logistics stations
d	The number of elements in I		
q	A prime number		

σ . To obtain a product, the user needs to prove that he is the owner by generating a proof of the knowledge included in the pseudonym. The user can verify whether the product is delivered correctly by checking the aggregate signature σ . In the case that the identity of a user needs to be traced, the trace party can use his secret key to de-anonymous the pseudonym, and find the user's identity.

**Fig. 1.** The Framework of Our PPLIST Scheme

2.1 Bilinear Group

Let G_1, G_2, G_ι be cyclic groups with prime order q . A map $e : G_1 \times G_2 \rightarrow G_\iota$ is a bilinear map if it satisfies the following properties: **(1) Bilinearity:** For all $g \in G_1, h \in G_2, a, b \in \mathbb{Z}_q$, $e(g^a, h^b) = e(g^b, h^a) = e(g, h)^{ab}$; **(2) Non-degeneration:** For all $g \in G_1, h \in G_2$, $e(g, h) \neq 1_\iota$, where 1_ι is the identity element in G_ι ; **(3) Computability:** For all $g \in G_1, h \in G_2$, there exists an efficient algorithm to compute $e(g, h)$.

Let $\mathcal{B}(1^l) \rightarrow (e, q, G_1, G_2, G_\iota)$ be a bilinear group generator which takes as input a security parameter 1^l and outputs a bilinear group $(e, q, G_1, G_2, G_\iota)$.

A function $\epsilon(x)$ is negligible if for any $k \in \mathbb{N}$, there exist a $z \in \mathbb{N}$ such that $\epsilon(x) < \frac{1}{x^z}$ when $x > z$.

2.2 Complexity Assumptions

Definition 1 (Computational Diffie-Hellman (CDH) Assumption [21]). Let $\mathcal{B}(1^l) \rightarrow (e, q, G_1, G_2, G_\iota)$, and g_1, g_2 be generator of G_1, G_2 , respectively. Suppose that $\alpha, \beta \xleftarrow{R} \mathbb{Z}_q$. Given a triple $(g_1^\alpha, g_1^\beta, g_2^\beta)$, we say that the CDH assumption holds on $(e, q, G_1, G_2, G_\iota)$ if all PPT adversaries \mathcal{A} can output $g_1^{\alpha\beta}$ with a negligible advantage, namely $\text{Adv}_{\mathcal{A}}^{\text{CDH}} = \text{PR}[\mathcal{A}(g_1^\alpha, g_1^\beta, g_2^\beta) \rightarrow g_1^{\alpha\beta}] \leq \epsilon(l)$.

Definition 2 (Discrete Logarithm (DL) Assumption [18]). Let G be a cyclic group with prime order q , and g be a generator of G . Given $Y \in G$, we say that the DL assumption holds on G if all PPT adversaries can output a number $x \in \mathbb{Z}_q$ such that $Y = g^x$ with a negligible advantage, namely $\text{Adv}_{\mathcal{A}}^{\text{DL}} = \text{PR}[Y = g^x | \mathcal{A}(q, g, G, Y) \rightarrow x] \leq \epsilon(l)$.

2.3 Formal Definition

A PPLIST scheme is formalized by the following eight algorithms:

Setup $(1^l) \rightarrow \text{PUB}$. The algorithm takes the security parameters 1^l as input and outputs the public parameters PUB .

Key – Generation. This algorithm consists of the following sub-algorithms:

- 1) *Station – Key – Generation* $(1^l) \rightarrow (SK_{S_i}, PK_{S_i})$. This algorithm is executed by each logistics station S_i . S_i takes the security parameters 1^l as input, and outputs his secret-public key pair (SK_{S_i}, PK_{S_i}) , where $i = 1, 2, 3, \dots, n$.
- 2) *User – Key – Generation* $(1^l) \rightarrow (SK_U, PK_U)$. This algorithm is executed by a user U . U takes the security parameters 1^l as input, and outputs his secret-public key pair (SK_U, PK_U) .
- 3) *Trace – Key – Generation* $(1^l) \rightarrow (SK_T, PK_T)$. This algorithm is executed by a trace party T . T takes the security parameters 1^l as input, and outputs his secret-public key pair (SK_T, PK_T) .

User – Pseudonym(PUB, SK_U, PK_T) $\rightarrow Pseudonym$. This algorithm is executed by U . U takes as input his secret key SK_U , the public key PK_T of the trace party and the public parameters PUB , and outputs a pseudonym $Pseudonym$.

Public – Key – Aggregation($PUB, PK_{S_{k_1}}, PK_{S_{k_2}} \dots, PK_{S_{k_d}}$) $\rightarrow YA$. Let I be a set which consists of the indexes of some selected logistics stations. This algorithm takes as input the public parameters PUB and the public keys $PK_{S_{k_1}}, PK_{S_{k_2}}, \dots, PK_{S_{k_d}}$ of selected logistics stations, and outputs the aggregated public key YA .

Sign. This algorithm consists of the following sub-algorithms:

- 1) *Single – Station – Sign*($PUB, SK_{S_{k_i}}, Pseudonym, YA, m$) $\rightarrow Sig_i$. This algorithm is executed by each selected logistics station S_{k_i} . S_{k_i} takes as input its secret key $SK_{S_{k_i}}$, the aggregated public key YA , product information m and the public parameters PUB , and outputs a signature Sig_i , where $i = 1, 2, 3, \dots, d$.
- 2) *Sign – Aggregation*($PUB, Sig_1, Sig_2, \dots, Sig_d$) $\rightarrow \sigma$. This algorithm takes as input the public parameters PUB and signatures Sig_i , and outputs a final signature σ .

User – Ownership – Verify($PUB, SK_U, PK_T, Pseudonym$) $\leftrightarrow S_i(PUB) \rightarrow (\pi, 1/0)$. This algorithm is executed between S_i and U .

- 1) U takes as input his secret key SK_U , the T 's public key PK_T , his pseudonym $Pseudonym$ and the public parameters PUB , and outputs a proof π .
- 2) The verifier takes as input the public parameters PUB , and outputs 1 if the proof π is valid; otherwise, it outputs 0 to show the proof is invalid.

Verify($PUB, \sigma, Pseudonym, YA, m$) $\rightarrow 1/0$. This algorithm takes as input the public parameters PUB , the final signature σ , the pseudonym $Pseudonym$, the aggregated public key YA and product information m , and outputs 1 if signature is valid; otherwise, it outputs 0 to show it is an invalid signature.

Trace($PUB, \sigma, SK_T, Pseudonym, YA, m$) $\rightarrow PK_U/\perp$. This algorithm is executed by T . T takes as input his secret key SK_T , the pseudonym $Pseudonym$, the aggregated public key YA , the final signature σ , product information m and the public parameters PUB , and outputs U 's public key PK_U if the signature σ is valid; otherwise, it outputs \perp to show failure.

2.4 Security Requirements

The security model of our scheme is defined by the following two games.

Unforgeability. This is used to define the unforgeability of signature, namely even if users, the trace party and the other stations collude, they cannot forge a valid signature on behalf of the selected logistics stations. This game is executed between a challenger \mathcal{C} and a forger \mathcal{F} .

Setup. \mathcal{C} runs $Setup(1^l) \rightarrow PUB$ and sends PUB to \mathcal{F} .

Key-Generation Query.

- 1) \mathcal{F} asks the public key of stations. \mathcal{C} runs $Station - Key - Generation(1^l) \rightarrow (SK_{S_i}, PK_{S_i})$ and sends the station's public key PK_{S_i} to \mathcal{F} .
- 2) When \mathcal{F} asks a user's secret-public key pair, \mathcal{C} runs $User - Key - Generation(1^l) \rightarrow (SK_U, PK_U)$ and sends (SK_U, PK_U) to \mathcal{F} . Let GPU be a set of users' public key.
- 3) When \mathcal{F} asks the secret-public key of the trace party, \mathcal{C} runs $Trace - Key - Generation(1^l) \rightarrow (SK_T, PK_T)$ and sends (SK_T, PK_T) to \mathcal{F} .

User-Pseudonym Query. \mathcal{F} submits a SK_U and the public key PK_T of the trace party, \mathcal{C} runs $User - Pseudonym(PUB, SK_U, PK_T) \rightarrow Pseudonym$ and sends $Pseudonym$ to \mathcal{F} . Let UPQ be a set of pseudonyms of users.

Public-Key-Aggregation Query. Let I be a set which consists of the indexes of some selected logistics stations and let d be the number of elements in the set I . \mathcal{F} submits a group of selected stations' public keys. \mathcal{C} runs $Public - Key - Aggregation(PUB, PK_{S_{k_i}}) \rightarrow YA$, where $i = 1, 2, \dots, d$. \mathcal{C} returns YA to \mathcal{F} .

Sign Query. \mathcal{F} adaptively submits selected station's secret key $SK_{S_{k_i}}$, the aggregation of public key YA , and U 's pseudonym $Pseudonym$ and the product information m to ask for a single signature Sig_i up to ϱ times.

Output. \mathcal{F} outputs a forged signature Sig'_i , a final signature σ' , U 's pseudonym $Pseudonym$ and the product information m' , the public keys of selected logistics stations AgY and the aggregated public keys YA' . \mathcal{F} wins the game if $PK_{S_i} \in AgY$, \mathcal{F} has not conducted signature query on the message m' , and $Verify(PUB, \sigma', Pseudonym, YA', m') = 1$.

Definition 3. A privacy-preserving logistics information system with traceability is $(\varrho, \epsilon(l))$ unforgeable if all probabilistic polynomial-time (PPT) forger \mathcal{F} who makes ϱ signature queries can only win the above game with a negligible advantage, namely

$$Adv = Pr \left[Verify(PUB, \sigma', Pseudonym, YA', m') = 1 \right] \leq \epsilon(l) \quad (1)$$

Traceability. This is used to formalise the traceability of our scheme, namely an attacker \mathcal{A} cannot frame a user who did not use the logistics services. We suppose that at least one station is honest. This game is executed between a challenger \mathcal{C} and an attacker \mathcal{A} .

Setup. \mathcal{C} runs $Setup(1^l) \rightarrow PUB$ and sends PUB to \mathcal{A} .

Key-Generation Query.

- 1) \mathcal{A} can ask for the public key of each station. \mathcal{C} runs *Station – Key – Generation*(1^l) $\rightarrow (SK_{S_i}, PK_{S_i})$ and sends the station's public key PK_{S_i} to \mathcal{F} .
- 2) When \mathcal{A} asks a user's secret-public key pair, \mathcal{C} runs *User – Key – Generation*(1^l) $\rightarrow (SK_U, PK_U)$. Let the secret-public key pair of U^* be (SK_{U^*}, PK_{U^*}) . \mathcal{C} sends other users' secret-public key pair (SK_U, PK_U) and PK_{U^*} to \mathcal{A} . Let GPU be a set consisting of users's public keys.
- 3) When \mathcal{A} asks the secret-public key pair of the trace party, \mathcal{C} runs *Trace – Key – Generation*(1^l) $\rightarrow (SK_T, PK_T)$ and sends (SK_T, PK_T) to \mathcal{A} .

User-Pseudonym Query. \mathcal{A} submits a user's SK_U and the public key PK_T of the trace party, \mathcal{C} runs *User – Pseudonym*(PUB, SK_U, PK_T) $\rightarrow Pseudonym$ and sends $Pseudonym$ to \mathcal{A} . Let UPQ be a set of pseudonyms of users.

Public-Key-Aggregation Query. Let I be a set which consists of the indexes of some selected logistics stations and let d be the number of elements in the set I . \mathcal{A} submits a group of selected stations' public keys. \mathcal{C} runs *Public – Key – Aggregation*($PUB, PK_{S_{k_i}}$) $\rightarrow YA$, where $i = 1, 2, \dots, d$. \mathcal{C} returns YA to \mathcal{A} .

Sign Query. \mathcal{A} adaptively submits a selected station's secret key $SK_{S_{k_i}}$, the aggregation of public key YA , and U 's pseudonym $Pseudonym$ and the product information m to ask for a single signature Sig_i up to ϱ times.

Output. \mathcal{A} outputs a tuple $(\sigma', Pseudonym', YA', m')$. \mathcal{A} wins the game if $Trace(PUB, \sigma', SK_T, Pseudonym', YA', m') \rightarrow PK'_{U^*}$ with $PK'_{U^*} \notin GPU$ or $PK'_{U^*} \neq PK_{U^*} \in GPU$.

Definition 4. A privacy-preserving logistics information system with traceability is $(\varrho, \epsilon(l))$ traceable if all probabilistic polynomial-time (PPT) adversary \mathcal{A} who makes ϱ signature queries can only win the above game with a negligible advantage, namely

$$Adv = Pr \left[\begin{array}{l} PK'_{U^*} \notin GPU \text{ or} \\ PK'_{U^*} \neq PK_{U^*} \in GPU \end{array} \middle| \begin{array}{l} \text{Trusted – Party – Trace} \\ (PUB, \sigma', SK_T, Pseudonym', \\ YA', m') \rightarrow PK'_{U^*} \end{array} \right] \leq \epsilon(l) \quad (2)$$

3 Construction of Our Scheme

In this section, we introduce the construction of our scheme. We firstly present a high-level overview, and then describe the formal construction of our scheme.

3.1 High-Level Overview

The high-level overview of our scheme is as follows.

Setup. The system generates the corresponding public parameters PUB .

Key-Generation. Suppose that there are n logistics stations. Each S_i , U and T generate their secret-public key pairs (x_{s_i}, Y_{s_i}) , (x_u, Y_u) and (x_t, Y_t) , where $i = 1, 2, 3, \dots, n$.

User-Pseudonym. In order to protect privacy in a delivery process, U generates a pseudonym $Pseudonym$ by using his secret key SK_U and T 's public key PK_T .

Public-Key-Aggregation. According to product information, the system determines the logistics process by selecting a set of logistics stations $S_{k_1}, S_{k_2}, \dots, S_{k_d}$. Let $AgY = \{Y_{S_{k_1}}, Y_{S_{k_2}}, \dots, Y_{S_{k_d}}\}$ be a set consisting of the public keys of the selected logistics stations. For each service, a table $Table$ is built to record its delivery information. The system uses the public key of each S_{k_i} and the set AgY to generate h_i to resist the rogue key attacks, where $i = 1, 2, 3, \dots, d$. Then, the system generates the aggregated public key YA .

Sign. Each selected logistics station S_{k_i} uses his secret key $x_{s_{k_i}}$ to generate a signature Sig_i on U 's pseudonym $Pseudonym$ and the product information m , and sends Sig_i to the next logistics stations. Finally, the last logistic station S_{k_d} use his secret key x_{k_d} to generate a single signature Sig_d on U 's pseudonym $Pseudonym$ and the product information m , and compute the aggregated signature $\sigma = \prod_{i=1}^d Sig_i$. S_{k_d} also adds σ to the table $Table$.

User-Ownership-Verify. When U proves to the last logistic station S_{k_d} that he is the owner of the product, he proves that his secret key x_u is included in the pseudonym $Pseudonym$ by executing a zero-knowledge proof with S_{k_d} . If the proof is correct, U is the owner of the product; otherwise, he is not the owner of the product.

Verify. When U receives a product, he checks whether the product was delivered correctly by checking the validity of the aggregate signature σ . If it is, the product is delivered correctly; otherwise, there are some problems in the delivery process.

Trace. Given $(\sigma, Pseudonym, AgY, m)$, in the case that a user needs to be de-anonymized, the trace party T first checks whether the signature is correct or not. If it is incorrect, T aborts; otherwise, T uses his secret key x_t to de-anonymize the Pseudonym and get U 's public key Y_u .

3.2 Formal Construction

The formal construction of our PPLIST scheme is formalised by the following eight algorithms:

Setup. The system runs $\mathcal{B}(1^l) \rightarrow (e, q, G_1, G_2, G_\iota, g_1, g_2)$ with $e : G_1 \times G_2 \rightarrow G_\iota$. Let g_1 be a generator of G_1 and g_2 be a generator of G_2 . Suppose that $H_1 : \{0, 1\}^* \rightarrow G_1$, $H_2 : \{0, 1\}^* \rightarrow Z_q$ and $H_3 : \{0, 1\}^* \rightarrow Z_q$ are cryptographic hash functions. The public parameters are $PUB = (e, q, G_1, G_2, G_\iota, g_1, g_2, H_1, H_2, H_3)$.

Key-Generation.

- 1) *Station – Key – Generation.* Each logistics station S_i selects $x_{s_i} \xleftarrow{R} Z_q$ and computes $Y_{s_i} = g_2^{x_{s_i}}$. The secret-public key pair of S_i is (x_{s_i}, Y_{s_i}) , where $i = 1, 2, 3, \dots, n$.
- 2) *User – Key – Generation.* Each U selects $x_u \xleftarrow{R} Z_q$ and computes $Y_u = g_2^{x_u}$. The secret-public key pair of U is (x_u, Y_u) .
- 3) *Trace – Key – Generation.* T selects $x_t \xleftarrow{R} Z_q$ and computes $Y_t = g_2^{x_t}$. The secret-public key pair of T is (x_t, Y_t) .

User-Pseudonym. To generate a pseudonym for a product information m , U firstly computes $k = H_3(x_u \parallel m)$ and then computes $C_1 = g_2^k, C_2 = Y_t^k \cdot g_2^{x_u}$. The pseudonym is $Pseudonym = (C_1, C_2)$.

Public-Key-Aggregation. Let $AgY = \{Y_{S_{k_1}}, Y_{S_{k_2}}, \dots, Y_{S_{k_d}}\}$ be a set consisting of the public keys of the logistics stations which will deliver the product to the user. The system firstly computes $h_i = H_2(Y_{S_{k_i}} \parallel AgY)$, and then computes $YA = \prod_{i=1}^d Y_{S_{k_i}}^{h_i}$. Let $(Pseudonym, m, AgY, YA)$ be a record of the product information m . The system adds it into the table *Table*.

Sign. When receiving a product, each S_{k_i} computes $Sig_i = H_1(C_1 \parallel C_2 \parallel m)^{h_i \cdot x_{S_{k_i}}}$. S_{k_i} sends Sig_i to $S_{k_{i+1}}$ for $i = 1, 2, 3, \dots, d-2$. Finally, S_{k_d} computes Sig_d and $\sigma = \prod_{i=1}^d Sig_i$. Subsequently, S_{k_d} adds it into the record of m in the table *Table*.

User-Ownership-Verify. To prove the ownership of the product to the last logistics station S_{k_d} , U and S_{k_d} work as follows.

- 1) U selects $v_1 \xleftarrow{R} Z_q, v_2 \xleftarrow{R} Z_q$ and computes $V_1 = g_2^{v_1}, V_2 = Y_t^{v_1} \cdot g_2^{v_2}$.
- 2) U sends C_1, C_2, V_1, V_2 to S_{k_d} . S_{k_d} selects $c \xleftarrow{R} Z_q$, and returns it to U .
- 3) U computes $r_1 = v_1 - c \cdot k$, and $r_2 = v_2 - c \cdot x_u$, and returns (r_1, r_2) to S_{k_d} .
- 4) S_{k_d} verifies $V_1 \stackrel{?}{=} g_2^{r_1} \cdot C_1^c$, and $V_2 \stackrel{?}{=} Y_t^{r_1} \cdot g_2^{r_2} \cdot C_2^c$. If these equations hold, it outputs 1 to show that U is the owner of the product; otherwise, it outputs 0 to show that U is not the owner of the product.

Verify. U verifies $e(\sigma, g_2^{-1}) \cdot e(H_1(C_1 \parallel C_2 \parallel m), YA) \stackrel{?}{=} 1_{G_t}$. If the equation holds, it outputs 1 to show that the delivery process is correct; otherwise, it outputs 0 to show that there are some errors in the delivery.

Trace. In the case that the identity of U who selected the product m needs to be revealed, T searches in the table *Table*, and finds the record $(\sigma, Pseudonym, YA, m)$ firstly. Then, T verifies $e(\sigma, g_2^{-1}) \cdot e(H_1(C_1 \parallel C_2 \parallel m), YA) \stackrel{?}{=} 1_{G_t}$. If it is not, T quits the system immediately; otherwise, T computes $Y_u = C_2 / C_1^{x_t}$, and confirms the identity of user.

4 Security Analysis

In this section, the security of our scheme is formally proven.

Theorem 1. *Our privacy-preserving logistics information system with traceability (PPLIST) is $(\varrho, \epsilon(l))$ -unforgeable if and only if the $(\epsilon(l)', T)$ -computational Diffie-Hellman (CDH) assumption holds on the bilinear group $(e, q, G_1, G_2, G_\iota)$ and H_1, H_2 are two random oracles and H_3 is a cryptographic hash function, where ϱ is the number of signature queries made by the forger \mathcal{F} , and $\epsilon(l)' \geq \frac{1}{q} \cdot \frac{1}{q-1} \cdot \frac{1}{\varrho} \cdot \epsilon(l)$.*

Proof. Suppose that there exists a forger \mathcal{F} that can break the unforgeability of our scheme, we can construct an algorithm \mathcal{B} which can use \mathcal{F} to break the CDH assumption. Given $(A, B_1, B_2) = (g_1^\alpha, g_1^\beta, g_2^\beta)$, the aim of \mathcal{B} is to output $g_1^{\alpha\beta}$.

Setup. \mathcal{B} selects $H_3 : \{0, 1\}^* \rightarrow \mathbb{Z}_p$. The public parameters are $PUB = (e, q, G_1, G_2, G_\iota, g_1, g_2, H_3)$.

- \mathcal{B} responds to the queries of \mathcal{F} about the random oracle H_1 .
 - 1) \mathcal{F} queries the hash function H_1 of a pseudonym (C_1^k, C_2^k) and a message m_k . \mathcal{B} selects $t_k \xleftarrow{R} \mathbb{Z}_q$, and sets $H_1(C_1^k \| C_2^k \| m^K) = g_1^{t_k}$, where $k = 1, 2, \dots, n$. \mathcal{B} sends $g_1^{t_k}$ to \mathcal{F} and adds $(C_1^k, C_2^k, m_k, g_1^{t_k})$ into the table $Table_1$.
 - 2) \mathcal{F} queries the hash function H_1 of a pseudonym (C_1^*, C_2^*) and a message m^* . \mathcal{B} sends g_1^α to \mathcal{F} , and adds $(C_1^*, C_2^*, m^*, g_1^\alpha)$ into the table $Table_1$.
- \mathcal{B} responds to the queries of \mathcal{F} about the random oracle H_2 . Let j is the index of $Y_{s_j} \in AgY$.
 - 1) when $i \neq j$, \mathcal{B} selects $h_i \xleftarrow{R} \mathbb{Z}_q$ and sets $h_i = H_2(Y_{s_i} \| AgY)$. \mathcal{B} returns h_i to \mathcal{F} and adds (Y_{s_i}, AgY, h_i) into the table $Table_2$.
 - 2) when $i = j$, \mathcal{B} selects $h_j \xleftarrow{R} \mathbb{Z}_q$ and sets $h_j = H_2(Y_{s_j} \| AgY)$. \mathcal{B} returns h_j to \mathcal{F} , and adds (Y_{s_j}, AgY, h_j) into the table $Table_2$.

Key-Generation Query.

- 1) *Station-Key-Generation Query.* \mathcal{B} picks a station S_j from S_1, S_2, \dots, S_n . For the i -th logistics station key generation query, \mathcal{B} selects x_{s_i} , and computes $Y_{s_i} = g_2^{x_{s_i}}$ where $i \neq j$. \mathcal{B} returns Y_{s_i} to \mathcal{F} . For the j -th logistics station key generation query, \mathcal{B} returns B_2 to \mathcal{F} .
- 2) *User-Key-Generation Query.* \mathcal{B} selects $x_u \xleftarrow{R} \mathbb{Z}_q$, and compute $Y_u = g_2^{x_u}$. \mathcal{B} sends the secret-public key pair (x_u, Y_u) to \mathcal{F} .
- 3) *Trace-Key-Generation Query.* \mathcal{B} selects $x_t \xleftarrow{R} \mathbb{Z}_q$, and compute $Y_t = g_2^{x_t}$. \mathcal{B} sends the secret-public key pair (x_t, Y_t) to \mathcal{F} .

User-Pseudonym Query. \mathcal{F} submits a product information m . \mathcal{B} computes $k = H_3(x_u \parallel m)$ firstly, then computes $C_1 = g_2^k$, $C_2 = Y_t^k \cdot g_2^{x_u}$. \mathcal{C} sends (C_1, C_2) to \mathcal{F} .

Public-Key-Aggregation Query. \mathcal{F} submits a group of logistics stations' public-key and sets $AgY = \{Y_{s_1}, Y_{s_2}, \dots, Y_{s_d}\}$. \mathcal{B} searches in $Table_2$, and gets $h_i = H_2(Y_{s_i} \parallel AgY)$, where $i = 1, 2, 3 \dots d$. \mathcal{B} computes $YA = \prod_{i=1}^d Y_{s_i}^{h_i}$, and sends YA to \mathcal{F} .

Sign Query. \mathcal{B} responds to the queries of \mathcal{F} about the single signature Sig_i :

- 1) Since $i \neq j$, \mathcal{F} asks about the single signature of the logistics station S_i on a pseudonym $(C_1, C_2) \neq (C_1^*, C_2^*)$ and the message m^* . \mathcal{B} computes $Sig_i = g_1^{\alpha \cdot h_i \cdot x_{s_i}}$. \mathcal{B} sends Sig_i to \mathcal{F} .
- 2) Since $i \neq j$, \mathcal{F} asks about the single signature of station S_j on a pseudonym (C_1, C_2) and a message $m \neq m^*$. \mathcal{B} computes $Sig_i = g_1^{t_i \cdot h_i \cdot \beta}$. \mathcal{B} sends Sig_i to \mathcal{F} . \mathcal{F} can ask for many times.
- 3) \mathcal{F} asks about the single signature of station S_j on a pseudonym (C_1^*, C_2^*) and the message m^* . \mathcal{B} aborts.

Output. \mathcal{F} outputs a forged final signature σ' . According to the above situations, \mathcal{F} can make q queries of random oracles and ϱ signature generations, respectively. By using Forking lemma technique, for two queries of the random oracle H_2 on the j -th station, \mathcal{B} selects h_j and h'_j with $h_j \neq h'_j$. For other selected stations, \mathcal{B} sets h_i and h'_i with $h_i = h'_i$. Hence, $YA/YA' = \prod_{i=1}^d Y_{S_{k_j}}^{h_j - h'_j}$. If \mathcal{F} can forge a valid signature, \mathcal{B} have $Sig_j = g_1^{\alpha \cdot \beta \cdot h_i}$ and $Sig'_j = g_1^{\alpha \cdot \beta \cdot h'_i}$, respectively. Then, \mathcal{B} computes $\sigma/\sigma' = g_1^{\alpha \cdot \beta \cdot (h_i - h'_i)}$. Therefore, \mathcal{B} can compute $g_1^{\alpha \cdot \beta} = (\sigma/\sigma')^{1/(h_i - h'_i)}$ and break the CDH assumption.

Since \mathcal{F} needs to make two hash queries to get different values of h_i and h'_i , the advantage is $(\frac{1}{q} \cdot \frac{1}{q-1})$. Furthermore, \mathcal{F} queries the single signature of station S_j on the pseudonym (C_1^*, C_2^*) and the message m^* with the advantage $\frac{1}{\varrho}$. Therefore, the advantage with which \mathcal{B} can break the CDH assumption is

$$Adv_{\mathcal{B}}^{CDH} \geq \frac{1}{q} \cdot \frac{1}{q-1} \cdot \frac{1}{\varrho} \cdot \epsilon(l) \quad (3)$$

Theorem 2. *Our privacy-preserving logistics information systems with traceability (PPLIST) is $(\varrho, \epsilon(l))$ -traceable if the computational Diffie-Hellman (CDH) assumption holds on the bilinear group (e, q, G_1, G_2, G_i) with the advantage at most $\epsilon_1(l)$, the discrete logarithm (DL) assumption holds on the group G_2 with the advantage at most $\epsilon_2(l)$, and H_1, H_2, H_3 are random oracles, where ϱ is the number of signature queries made by the forger \mathcal{F} , and $\epsilon(l) = \max\{(\frac{1}{2} \cdot \frac{1}{q} \cdot \frac{1}{\varrho} \cdot \epsilon_1(l)), (\frac{1}{2} \cdot \epsilon_2(l))\}$.*

Proof. Suppose that there exists an adversary \mathcal{A} that can break the traceability of our scheme, we can construct an algorithm \mathcal{B} which can use \mathcal{A} to break the CDH assumption or DL assumption.

Setup. The public parameters are $PUB = (e, q, G_1, G_2, G_\iota, g_1, g_2)$.

- \mathcal{B} responds to the queries of \mathcal{A} about the random oracle H_1 .

- 1) \mathcal{A} queries H_1 on a pseudonym (C_1^k, C_2^k) and a message m_k . \mathcal{B} selects $t_k \xleftarrow{R} \mathbb{Z}_q$, and sets $g_1^{t_k} = H_1(C_1^k \parallel C_2^k \parallel m_k)$, where $i = 1, 2, \dots, n$. \mathcal{B} sends $g_1^{t_k}$ to \mathcal{A} , and adds $(C_1^k, C_2^k, m_k, g_1^{t_k})$ to the table $Table_1$.
- 2) \mathcal{A} queries H_1 of a pseudonym (C_1^*, C_2^*) and a message m^* . \mathcal{B} sets $g_1^\alpha = H_1(C_1^* \parallel C_2^* \parallel m^*)$. \mathcal{B} sends g_1^α to \mathcal{A} , and adds $(C_1^*, C_2^*, m^*, g_1^\alpha)$ to the table $Table_1$.

- \mathcal{B} responds to the queries of \mathcal{A} about the random oracle H_2 . Let j be the index of $Y_{s_j} \in AgY$.

- 1) when $i \neq j$, \mathcal{B} selects $h_i \xleftarrow{R} \mathbb{Z}_q$ and sets $h_i = H_2(Y_{s_i} \parallel AgY)$. \mathcal{B} returns h_i to \mathcal{A} , and records it into the table $Table_2$.
- 2) when $i = j$, \mathcal{B} selects $h_j \xleftarrow{R} \mathbb{Z}_q$ and sets $h_j = H_2(Y_{s_j} \parallel AgY)$. \mathcal{B} sends h_j to \mathcal{A} and adds (Y_{s_j}, AgY, h_j) into the table $Table_2$.

- \mathcal{B} responds to the queries of \mathcal{A} about the random oracle H_3 . When receiving a query (x_u, m) on H_3 , \mathcal{A} selects $k \xleftarrow{R} \mathbb{Z}_p$ and returns it to \mathcal{A} . \mathcal{B} adds (x_u, m, k) into the table $Table_3$.

Key-Generation Query.

- 1) *Station – Key – Generation Query.* \mathcal{B} picks a logistics station S_j from S_1, S_2, \dots, S_n . For the i -th logistics station key generation query, \mathcal{B} selects x_{s_i} , and computes $Y_{s_i} = g_2^{x_{s_i}}$ where $i \neq j$. \mathcal{B} returns Y_{s_i} to \mathcal{A} . For the j -th logistics station key generation query, \mathcal{B} returns B_2 to \mathcal{A} .
- 2) *User – Key – Generation Query.* \mathcal{B} selects $x_u \xleftarrow{R} \mathbb{Z}_q$, and compute $Y_u = g_2^{x_u}$. \mathcal{B} retains the private key x_u , sends the public key Y_u to \mathcal{A} , and sets GPU to store the public key Y_u of all users. Y_{u^*} is the public key of user U^* .
- 3) *Trace – Key – Generation Query.* \mathcal{B} selects $x_t \xleftarrow{R} \mathbb{Z}_q$, and compute $Y_t = g_2^{x_t}$. \mathcal{B} sends the secret-public key pair (x_t, Y_t) to \mathcal{A} .

User-Pseudonym Query. \mathcal{A} submits a user's secret key x_u and a product information m . \mathcal{B} first searches in the table $Table_3$ and gets $k = H_3(x_u \parallel m)$. Then \mathcal{B} computes $C_1 = g_2^k$, $C_2 = Y_t^k \cdot g_2^{x_u}$. \mathcal{C} sends (C_1, C_2) to \mathcal{A} .

Public-Key-Aggregation Query. \mathcal{A} submits a group of logistics stations' public-keys and sets $AgY = \{Y_{s_1}, Y_{s_2}, \dots, Y_{s_d}\}$. \mathcal{B} searches on the table $Table_2$, and gets $h_i = H_2(Y_{s_i} \parallel AgY)$ where $i = 1, 2, 3, \dots, d$. \mathcal{B} computes $YA = \prod_{i=1}^d Y_{s_i}^{h_i}$. \mathcal{B} sends YA to \mathcal{A} .

Sign Query. \mathcal{B} responds to the queries of \mathcal{A} about the single signature Sig_i :

- 1) when $i \neq j$, \mathcal{A} asks about the single signature of station S_i on a pseudonym $(C_1, C_2) \neq (C_1^*, C_2^*)$ and the message m^* . \mathcal{B} computes $Sig_i = g_1^{\alpha \cdot h_i \cdot x_{s_i}}$. \mathcal{B} sends Sig_i to \mathcal{A} .
- 2) when $i \neq j$, \mathcal{A} asks about the single signature of station S_j on a pseudonym (C_1, C_2) and a message $m \neq m^*$. \mathcal{B} computes $Sig_i = g_1^{t_i \cdot h_i \cdot \beta}$. \mathcal{B} sends Sig_i to \mathcal{A} .
- 3) \mathcal{A} asks about the single signature of station S_j on a pseudonym (C_1^*, C_2^*) and the message m^* . \mathcal{B} aborts.

Output. \mathcal{A} outputs a final signature σ' . We consider the following two types of attackers. Suppose that the secret-public key pair of U' is (x'_u, Y'_u) , and \mathcal{A} only knows Y'_u . In Type-1 case, \mathcal{A} outputs a final signature σ' containing a new pseudonym (C'_1, C'_2) . In Type-2 case, \mathcal{A} also outputs a final signature σ which is a signature on a used pseudonym.

Type-1: If there is a new pseudonym $(C'_1, C'_2) \notin UPQ$. \mathcal{C} uses (C'_1, C'_2) to compute $Y'_u = C'_2 / (C'_1)^{x_t}$, and gets Y'_u , $Y'_u \notin GPU$. \mathcal{A} forged a single signature Sig_i and a final signature σ' , where $Sig_i = H_1(C'_1 \parallel C'_2 \parallel m)^{h_i \cdot x_{s_i}}$, and $\sigma' = \prod_{i=1}^d Sig_i$. Hence, \mathcal{B} can use \mathcal{A} to break CDH assumption, The detailed proof is shown in *Theorem 1*.

Type-2: If there is a pseudonym (C'_1, C'_2) , namely $(C'_1, C'_2) \in UPQ$. \mathcal{B} uses (C'_1, C'_2) to compute $Y'_u = C'_2 / (C'_1)^{x_t}$, and gets Y'_u with $Y'_u \neq Y_{u^*} \in GPU$. If \mathcal{A} can prove $C'_1 = g_2^{k'}$, $C_2 = Y_t^{k'} \cdot g_2^{x'_u}$. By using the rewinding technique, \mathcal{B} can extract the knowledge of (x'_u, k') from \mathcal{A} . Therefore, given (Y'_u, g_2) , \mathcal{B} can output a x'_u such that $Y'_u = g_2^{x'_u}$. Hence, \mathcal{B} can use \mathcal{A} to break the DL assumption.

By the proof of unforgeability, the advantage with which \mathcal{B} can break the CDH assumption is $(\frac{1}{q} \cdot \frac{1}{q-1} \cdot \frac{1}{\ell} \cdot \epsilon_1(l))$. In the situation of *Type-1*, \mathcal{B} can break the CDH assumption with the advantage $(\frac{1}{2} \cdot \frac{1}{q} \cdot \frac{1}{q-1} \cdot \frac{1}{\ell} \cdot \epsilon_1(l))$. In the situation of *Type-2*, \mathcal{B} can break the DL assumption with the advantage $\frac{1}{2} \cdot \epsilon_2(l)$. Hence, $\epsilon(l) = \max\{(\frac{1}{2} \cdot \frac{1}{q} \cdot \frac{1}{q-1} \cdot \frac{1}{\ell} \cdot \epsilon_1(l)), (\frac{1}{2} \cdot \epsilon_2(l))\}$.

5 Experiment and Evaluation

In this section, we introduce the implementation and evaluation of our PPLIST scheme.

5.1 Runtime Environment

The performance of our PPLIST scheme is measured on a Lenovo Legion Y7000P 2018 laptop with an Intel Core i7-8750H CPU, 500GB SSD and 8GB RAM. The scheme is implemented in Microsoft Windows 10 System using E-clipse Integrated environment, Java language and JPBC library [3].

In our implementation, we apply the Type F curve. For the hash functions $H_1 : \{0, 1\}^* \rightarrow G_1$, $H_2 : \{0, 1\}^* \rightarrow Z_q$ and $H_3 : \{0, 1\}^* \rightarrow Z_q$ required by our scheme, we used *SHA-256* and the “newElementfromHash()” method in the JPBC library.

Our scheme is implemented in the following three cases: 1) $n = 20, d = 10$; 2) $n = 100, d = 50$; 3) $n = 200, d = 100$. The experimental results are shown in Table 3.

Table 3. Times(ms)

Phase	n=20,d=10	n=100,d=50	n=200,d=100
Setup	522	519	506
Station-Key-Generation	137	578	1031
User-Key-Generation	7	5	4
Trace-Key-Generation	8	5	3
User-Pseudonym	27	12	12
Public-Key-Aggregation	217	745	1400
Sign	122	486	953
User-Ownership-Verify	112	106	97
Verify	176	144	141
Trace	186	144	141

5.2 Timing

The setup phase is a process run by the system. It takes 522ms, 519ms and 506ms to setup the system in case 1, case 2 and case 3, respectively. According to the data, it can be observed that the running time of the three cases is roughly the same in the setup phase.

The key pair generation phase is run by logistics stations, user and trace party. It takes 137ms, 578ms and 1031ms to generate the key pair of logistics stations in case 1, case 2 and case 3, respectively. In the user key pair generation phase, it takes 7ms, 5ms and 4ms in case 1, case 2 and case 3, respectively. For trace party to generate key pair, it takes 8ms, 5ms and 3ms in case 1, case 2 and case 3, respectively.

The pseudonym generation phase is run by the user. It takes 27 ms, 12ms and 12 ms in case 1, case 2 and case 3, respectively. Observing the experimental data, it is not difficult to find that the running time of the public key aggregation phase is proportional in the number of logistics stations. It takes 259ms, 745ms and 1400ms to aggregate public keys in the three cases, respectively. The signature phase is run by logistics stations. The times to generate a multi-signature in case 1, case 2 and case 3 are 122ms, 486ms and 953ms, respectively.

In the user ownership verification phase, a user proves the ownership by interacting with the last logistics station. It takes 112ms, 106ms and 97ms in case 1, case 2 and case 3, respectively. In signature validation phase, it takes

186ms, 144ms and 141ms to verify a multi-signature in case 1, case 2 and case 3, respectively. To trace a user, it takes 176ms, 144ms and 141ms in case 1, case 2 and case 3, respectively. We implement our scheme in three different cases. The experiment results show the efficiency of our scheme.

6 Conclusions

In this paper, to protect users' privacy in LIS, a PPLIST was proposed. In our scheme, users anonymously use logistics services. Furthermore, a trace party can de-anonymized users to prevent illegal logistics. Additionally, the whole logistics process can be recorded and is unforgeable. We formalize the definition and security model of our scheme, and present a formal construction. We formally proved the security of our scheme and implemented it.

In our scheme, a buyer can prove the ownership of a product by proving the knowledge included in the pseudonyms. Our future work is to improve the flexibility of this work to enable an owner to designate a proxy to prove the ownership of products on behalf of him.

Acknowledgment

This work was partially supported by the National Natural Science Foundation of China (Grant No. 61972190, 62072104, 61972095) and the National key research and development program of China (Grant No. 2020YFE0200600). This work was also partially supported by the Postgraduate Research & Practice Innovation Program of Jiangsu Province (Grand No. KYCX20_1322) and the Natural Science Foundation of the Fujian Province, China (Grant No. 2020J01159).

References

1. Amazon. <https://www.amazon.cn>
2. Taobao. <https://www.taobao.com>
3. Angelo, D.C., Vincenzo, I.: JPBC:Java pairing based cryptography. In: ISCC 2011. pp. 850–855. IEEE (2011)
4. Asaar, M.R., Salmasizadeh, M., Susil, W.: An identity-based multi-proxy multi-signature scheme without bilinear pairings and its variants. *The Computer Journal* **58**(4), 1021–1039 (2015)
5. Bagherzandi, A., e, J.H.C., Jareck, S.: Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In: CCS 2008. pp. 449–458. ACM (2008)
6. Bardi, E.J., Raghunathan, T.S., Bagchi, P.K.: Logistics information systems: The strategic role of top management. *Journal of Business Logistics* **15**(1), 71–85 (1994)
7. Bellare, M., Neven, G.: Multi-signatures in the plain public-key model and a general forking lemma. In: CCS 2006. pp. 390–399. ACM (2006)
8. Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In: PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer (2003)

9. Boneh, D., Drijvers, M., Neven, G.: Compact multi-signatures for smaller blockchains. In: ASIACRYPT 2018. LNCS, vol. 11273, pp. 435–464. Springer (2018)
10. Chaum, D.: Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM* **28**(10), 1030–1044 (1985)
11. Chaum, D., Hendrik Evertse, J.: A secure and privacy-protecting protocol for transmitting personal information between organizations. In: CRYPTO 1986. LNCS, vol. 263, pp. 118–167. Springer (1986)
12. Chen, L.: Access with pseudonyms. In: CPA 1995. LNCS, vol. 1029, pp. 232–243. Springer (1995)
13. Chrisos, M.: Pseudonymization Techniques: How to Protect Your Data. <https://www.techfunnel.com/information-technology/pseudonymization-techniques-how-to-protect-your-data/> (2019)
14. Christin, N., Yanagihara, S.S., Kamataki, K.: Dissecting one click frauds. In: CCS 2010. pp. 15–26. ACM (2010)
15. Closs, D.J., Xu, K.: Logistics information technology practice in manufacturing and merchandising firms – An international benchmarking study versus world class logistics firms. *International Journal of Physical Distribution & Logistics Management* **30**(10), 869–886 (2000)
16. Correll, N., Bekris, K.E., Berenson, D., Brock, O., Causo, A., Hauser, K., Okada, K., Rodriguez, A., Romano, J.M., Wurman, P.R.: Analysis and observations from the first Amazon picking challenge. *IEEE Transactions on Automation Science and Engineering* **15**(1), 172–188 (2018)
17. Gao, Q., Zhang, J., Ma, J., Yang, C., Guo, J., Miao, Y.: LIP-PA: A logistics information privacy protection scheme with position and attribute-based access control on mobile devices. *Wireless Communications and Mobile Computing* **2018**(1), 1–14 (2018)
18. Gordon, D.M.: Discrete logarithms in $GF(P)$ using the number field sieve. *SIAM Journal on Discrete Mathematics* **6**(1), 124–138 (1993)
19. Grunshpoun, T., Grubshtein, A., Zivan, R., Netzer, A., Meisels, A.: Asymmetric distributed constraint optimization problems. *Journal of Artificial Intelligence Research* **47**(1), 613–647 (2013)
20. Han, J., Chen, L., Schneider, S., Treharne, H., Wesemeyer, S., Wilson, N.: Anonymous single sign-on with proxy re-verification. *IEEE Transactions on Information Forensics and Security* **15**(1), 223–236 (2020)
21. Hanaoka, G., Kurosawa, K.: Efficient chosen ciphertext secure public key encryption under the computational Diffie-Hellman assumption. In: ASIACRYPT 2008. LNCS, vol. 5350, pp. 308–325. Springer (2008)
22. Horster, P., Michels, M., Petersen, H.: Meta-Multisignature schemes based on the discrete logarithm problem, pp. 128–142. IFIP, Springer (1995)
23. Itakura, K., Nakamura, K.: A public-key cryptosystem suitable for digital multisignatures. *NEC Research & Development* **71**(1), 1–8 (1983)
24. Kang, J., Yu, R., Huang, X., Zhang, Y.: Privacy-preserved pseudonym scheme for fog computing supported internet of vehicles. *IEEE Transactions on Intelligent Transportation Systems* **19**(8), 2627–2637 (2018)
25. Kim, D.J., Song, Y.I., Braynov, S.B., Rao, H.R.: A multidimensional trust formation model in B-to-C e-commerce: A conceptual framework and content analyses of academia/practitioner perspectives. *Decision Support Systems* **40**(2), 143–165 (2005)
26. Korth, B., Schwede, C., Zajac, M.: Simulation-ready digital twin for realtime management of logistics systems. In: IEEE Big Data 2018. pp. 4194–4201. IEEE (2018)

27. Lai, K., Ngai, E., Cheng, T.: Information technology adoption in Hong Kong's logistics industry. *Transportation Journal* **44**(4), 1–9 (2005)
28. Liu, F., Shu, P., Lui, J.C.S.: AppATP: An energy conserving adaptive mobile-cloud transmission protocol. *IEEE Transactions on Computers* **64**(11), 3051–3063 (2015)
29. Liu, S., Wang, J.: A security-enhanced express delivery system based on NFC. In: *ICSICT 2016*. pp. 1534–1536. IEEE (2016)
30. Lysyanskaya, A., Micali, S., Reyzin, L., Shacham, H.: Sequential aggregate signatures from trapdoor permutations. In: *EUROCRYPT 2004*. LNCS, vol. 3027, pp. 74–90. Springer (2004)
31. Lysyanskaya, A., Rivest, R.L., Sahai, A., Wolf, S.: Pseudonym systems. In: *SAC 1999*. LNCS, vol. 1758, pp. 184–199. Springer (1999)
32. Léauté, T., Faltings, B.: Coordinating logistics operations with privacy guarantees. In: *IJCAI 2011*. pp. 2482–2487. Morgan Kaufmann (2011)
33. Marko, A., Marjan, M., Vlada, S.: Logistics information system. *Vojnotehnicki glasnik* **58**(1), 33–61 (2010)
34. Micali, S., Ohta, K., Reyzin, L.: Accountable-subgroup multisignatures: Extended abstract. In: *CCS 2001*. pp. 245–254. ACM (2001)
35. Narayanan, A., Shmatikov, V.: De-anonymizing social networks. In: *S & P 2009*. pp. 173–187. IEEE (2009)
36. Neven, G.: Efficient sequential aggregate signed data. *IEEE Transactions on Information Theory* **57**(3), 1803–1815 (2011)
37. Ngai, E., Lai, K.H., Cheng, T.: Logistics information systems: The Hong Kong experience. *International Journal of Production Economics* **113**(1), 223–234 (2008)
38. Ohta, K., Okamoto, T.: A digital multisignature scheme based on the fiat-shamir scheme. In: *ASIACRYPT 1991*. LNCS, vol. 739, pp. 139–148. Springer (1991)
39. Ohta, K., Okamoto, T.: Multisignature schemes secure against active insider attacks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* **E82-A**(1), 21–31 (1999)
40. Okamoto, T.: A digital multisignature scheme using bijective public-key cryptosystems. *ACM Transactions on Computer Systems* **6**(4), 432–441 (1988)
41. Qi, H., Chenjie, D., Yingbiao, Y., Lei, L.: A new express management system based on encrypted QR code. In: *ICICTA 2015*. pp. 53–56. IEEE (2015)
42. Qian, J., Li, X., Zhang, C., Chen, L.: De-anonymizing social networks and inferring private attributes using knowledge graphs. In: *INFOCOM 2016*. pp. 1–9. IEEE (2016)
43. Qu, Z., He, P., Hou, L.: Studies on internet real-name system and network action surveillance system. In: *EDT 2010*. pp. 469–472. IEEE (2010)
44. Stallings, W.: Handling of personal information and deidentified, aggregated, and pseudonymized information under the California consumer privacy act. *IEEE Security & Privacy* **18**(1), 61–64 (2020)
45. Tarjan, L., Senk, I., Tegeltija, S., Stankovski, S., Ostojic, G.: A readability analysis for QR code application in a traceability system. *Computers & Electronics in Agriculture* **109**(4), 1–11 (2014)
46. Tiwari, N., Padhye, S., He, D.: Efficient ID-based multiproxy multisignature without bilinear maps in ROM. *Annals of Telecommunications* **68**(3–4), 231–237 (2013)
47. Wei, Q., Li, X.: Express information protection application based on k-anonymity. *Application Research of Computers* **31**(2), 555–567 (2014)
48. Wei, Q., Wang, C., Li, X.: Express information privacy protection application based on RSA. *Application of Electronic Technique* **40**(7), 58–60 (2014)

49. Xu, F.J., Tan, C.J., Tong, F.C.: Auto-ID enabled tracking and tracing data sharing over dynamic B2B and B2G relationships. In: RFID-TA 2011. pp. 394–401. IEEE (2011)
50. Ye, L., Wang, Y., Chen, J.: Research on the intelligent warehouse management system based on near field communication (NFC) technology. *International Journal of Advanced Pervasive and Ubiquitous Computing* **8**(2), 38–55 (2016)
51. Yong, L., Jing, P., Ma, B., Bo, Y.: The analysis of logistic bottleneck of taobao.com in C2C model and its countermeasures. In: ICEE 2010. pp. 5537–5539. IEEE (2010)