A Scalable and Auditable Secure Data Sharing Scheme with Traceability for Fog-Based Smart Logistics

Yanbo Yang, Jiawei Zhang, Ximeng Liu, *Senior Member, IEEE* and Jianfeng Ma, *Member, IEEE*

***Abstract*—Smart Logistics (s-Logistics) has become more and more popular driven by intelligent Internet of Things (IoT) which deploys pervasive smart devices in s-Logistics systems. The explosive growth of s-Logistics data collected by these resource-limited IoT devices enables Fog-based s-Logistics that provides data outsourcing and sharing services via multiple clouds within small latency. Nevertheless, it also gives rise to prominent security risks of user privacy leakage considering malicious users and data integrity violation with untrusted cloud servers, which are severe to s-Logistics systems and cannot be addressed by simple encryption. To solve these issues, in this paper, we propose an efficient Large Universe and Traceable Privacy-preserving Data Sharing (LUTPDS) for Fog-based s-Logistics. It simultaneously achieves data access control, data integrity protection, key escrow and abuse resistance, user privacy preserving and scalability. We devise a large universe and multi-authority Ciphertext-Policy Attribute-Based Encryption (CP-ABE) scheme in which access policy hiding mechanism is used for user privacy preserving, while white-box tracing and certificateless public data integrity auditing techniques are employed to resist key abuse and escrow problems. In addition, online/offline encryption and verifiable outsourced decryption are leveraged for high efficiency and cloud encryption is utilized to extend to multiple clouds. In the end, we formally prove the security of our scheme for indistinguishability of chosen plaintext attack (IND-CPA) security and traceability. Detailed performance evaluation with extensive experiments shows that our scheme is practicable for s-Logistics compared with existing schemes.**

***Index Terms*—Fog-based Smart logistics, multi-authority CP-ABE, public data integrity auditing, verifiable outsourced decryption, traceability.**

I. Introduction

NOWADAYS, Smart Logistics (s-Logistics) plays an essential role in modern e-commerce and retail industry and has become more and more popular in people's daily lives due to its intelligence and convenience [1]. More than 86% of consumers respect s-Logistics as an important part of their online shopping experience [2]. Recently, many pioneering s-Logistics platforms, e.g., Amazon, Alibaba, etc, have been advanced into a new era with the help of the Internet of Things (IoT) technique which deploys pervasive smart IoT devices in supply chains and transportation systems to gather significant data (i.e, s-Logistics data) from cargo, workflow and transportation vehicles for status monitoring, tracking, analysis, route optimization, predication, etc [3]. As a result, the s-Logistics data are growing exposively in these systems and it is no longer appropriate to store these data locally in IoT devices [4]. Thus, how to store and share the s-Logistics data of big volume and high value is a serious problem for s-Logistics besides its advantages.

To deal with the difficulties and satisfy the requirements of short time-delay in s-Logistics applications, many solutions adopt the architecture of Cloud-Fog-IoT shown in Fig 1 for data storage and sharing in s-Logistics systems (i.e, Fog-based s-Logistics) [5]. It not only enables small latency in data access with Fog Computing but also can alleviate local burden of resource-constraint IoT devices by providing unlimited resources and convenient data outsourcing and sharing services via cloud [6], [7]. However, the scalability problems and extensive security concerns have been more and more prominent. The resources of a single cloud may be insufficient for the fast growing s-Logistics data [8]. Moreover, as to the security concerns, for one thing, s-Logistics data are outsourced in cloud and shared across large amounts of users which may cause data leakage via unauthorized access [9]. For another thing, users' data are maintained by cloud which can easily reveal the private information embedded in s-Logistics data or even violate the integrity of these data by deleting those unfrequently accessed records for cost saving or hiding the fault of data loss caused by software or hardware malfunctions [10], [11]. Therefore, it is urgent to protect the confidentiality, fine-grained access control and integrity of users' data in Fog-based s-Logistics.

As a countermeasure to data confidentiality and fine-grained access control, many cloud data sharing schemes [12]–[15] have been proposed using Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [16]. In general CP-ABE schemes, data is encrypted for confidentiality and stored in cloud together with specific access policies for fine-grained access control [17]. Nevertheless, existing CP-ABE schemes still confront many limitations when directly leveraged in s-Logistics system. First of all, the s-Logistics systems are usually in large scale which requires large attribute universe. Gao et al. [18] presented a CP-ABE scheme for data sharing supporting large attribute universe but suffered from severe low efficiency and

J. Zhang and J. Ma are with the School of Cyber Engineering and the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, China (e-mail: zjw8512@126.com;jiangqixdu@xidian.edu.cn;jfma@mail.xidian.edu.cn).

Y. Yang is with the School of Information Engineering, Inner Mongolia University of Science & Technology and the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, China (email: yangyanbo@imust.edu.cn)

X. Liu is with the College of Computer and Big Data, Fuzhou University, Fuzhou 350116, China, and the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, China (e-mail: snbnix@gmail.com)

J. Zhang is the co-first author and corresponding author.

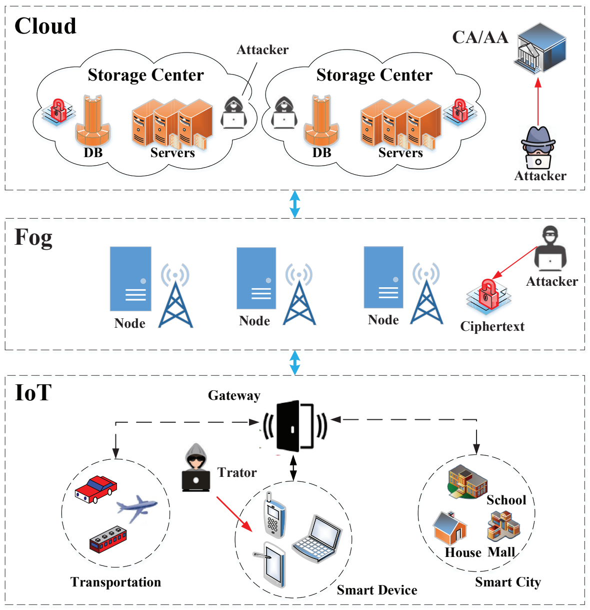


Fig. 1: An example of Cloud-Fog-IoT architecture

super complexity with predict encryption mechanism. Meantime, the studies in [19], [20] proposed two large universe CP-ABE schemes for data sharing but failed to resist the key escrow and abuse problems caused by centralized attribute authority. Zhang et al. [21] solved the problems in large universe CP-ABE schemes by integrating multi-authority and white-box tracing mechanism but it faces the bottleneck of low efficiency in computation. The work [22] proposed another multi-authority CP-ABE with large attribute universe, but it cannot support flexible and expressive access policy. Thus, practical data sharing scheme in s-Logistics should provide efficient, key escrow and key abuse free mechanism with large attribute universe.

Furthermore, the cleartext access policies stored in cloud in conventional CP-ABE schemes may reveal user privacy [19]. For example, in s-Logistics systems, the access policy "{Name = Lee, Position = Doctor, Ientity = 1234-56789, Address = 77 Massachusetts Avenue}" declares the authorization for access but reveals the user's privacy including the identity, address information and cloud servers or attackers can infer the privacy of authorized users, which is unacceptable for users. Zhang et al. [23] proposed a partially policy-hidden CP-ABE scheme for user privacy preserving but incured low efficiency with composite ordered bilinear groups. To improve efficiency, [24], [25] proposed two privacy-preserving CP-ABE schemes based on prime ordered bilinear groups and Wang et al. [26] further improved their efficiency with outsourced computing technique. However, all of these schemes still suffered from key escrow problem. Fan et al. [27] solves the problem with multi-authority mechanism, but it lacks the support for large universe and integrity protection for shared data in untrusted cloud which may bring about security risks to s-Logistics systems. Although the schemes in [11], [28]–[31] proposed several excellent data shariing schemes with public data integrity protection mechanism in centralized or decentralized way, they cannot guarantee the fine-grained access control and high efficiency.

As a summary, when adopting Cloud-Fog-IoT architecture for s-Logistics data sharing, data security including access control and integrity protection is essential and incurs several problems. To break through the aforementioned limitations, in this paper, we propose a Large Universe and Traceable Privacy-preserving Data Sharing (LUTPDS) scheme to address the problems of data sharing in s-Logistics system including scalability, user privacy leakage in access policy, key-escrow and key abuse problems, bounded attribute universe, data integrity violation and low efficiency. In particular, our main contributions are listed as follow:

- LUTPDS achieves scalability, key escrow and key abuse resistance, unbounded attribute universe and user privacy preserving in access policies at the same time for s-Logistics data sharing in multiple clouds by devising a multi-authority and fully hidden access policy CP-ABE scheme with white-box user tracing mechanism.
- To guarantee the data integrity and eliminate the high computation and storage burden in resource-limited IoT devices of s-Logistics systems, LUTPDS integrates the certificateless public data integrity auditing mechanism and online/offline technique together with verifiable outsourced decryption into the designed CP-ABE scheme in a non-trivial way.
- We present formal security analysis for LUTPDS to show that it is static secure in random oracle model and full traceable. Moreover, the detailed performance evaluation with extensive simulation experiments by implementations demonstrates that LUTPDS is practicable and suitable for Fog-Based s-Logistics.

Therefore, with the proposed LUTPDS for Fog-based s-Logistics data sharing, the data access control can be guaranteed in the first place with features of fine granularity, distributed environment, large attribute universe and attribute privacy preserving, together with data integrity protection and higher efficiency for practical applications.

The rest of this paper is organized as follows. We review some related work in Section II. In Section III, we present the summary of notations and definitions used in our proposal. Section IV gives the system and threat model of our scheme together with formal system definition and security model. Based on this, we describe in detail the constructions of our proposed scheme in Section V. In Section VI, we analyze the security of our scheme and present its performance evaluation. Finally, the conclusion of our work is summarized in Section VII.

II. Related Work

In this section, we will introduce some related work in the field of smart logistics and ABE-based data access control. Then, we give a comparison between some state-of-the-art data access control schemes with our proposal.

These days, Smart Logistics has become prevalent all over the world and the necessity in people's daily lives with its high quality and convenience in shopping, corresponding and cargo transporting, etc. Benefited from IoT, mobile communication, cloud computing, s-Logistics is developing into a new era. Lee et al. [32] proposed a smart logistics system based on IoT for warehouse management. Then, Wen et al. [33] studied the robotics control and communication in smart logistics and Su et al. [34] researched the vehicle routing for smart logistics systems. To deal with the problem of the sharing and outsourcing of s-Logistics data of huge volume to reduce the heavy burden in IoT devices and improve the quality of service, the works in [1] and [4] present the idea of Cloud-based s-Logistics systems. However, most of the existing s-Logistics technologies do not consider the severe security and privacy concerns on s-Logistics data sharing which need to be urgently addressed for further application.

The promising CP-ABE enables confidentiality and flexible fine-grained access control in data sharing [16]. Nevertheless, its centralized authority incurs single point failure and key escrow problem, which motivates multi-authority CP-ABE or decentralized CP-ABE [35]–[37], but these schemes also face the bottleneck of bounded attributes. Thus, Rouselakis et al. [38] proposed large universe CP-ABE with the feature of unbounded attributes and flexibility, which motivates the following works, e.g. [19], [20]. To achieve both key escrow and abuse resistance and large universe, recently, Zhang et al. [21] proposed a CP-ABE scheme by combining multi-authority CP-ABE [39] and large universe together with white-box traceability, and Banerjee et al. [22] presented another decentralized CP-ABE supporting large universe and with constant-size key and ciphertext. However, they still suffer from the issue of low efficiency.

To reduce the computation cost in encryption, the first online/offline ABE scheme was proposed by [40], in which the encrytion process is divided into two phrase: online and offline phrase that takes on all major computation costs of online while the resource-constrained devices are being charged. Based on this, the schemes in [41], [42] designed the fully secure and verifiable mechanism for online/offline encryption, respectively and Li et al. [43] developed it into intermediate pool technique. With respect to the cost for decryption, Green et al. [44] introduced the idea of outsourced decryption into ABE. Later, Lai et al. [45] developed a verifiable outsourced decryption ABE scheme to guarantee the correctness of outsourced decryption in untrusted cloud and motivated the following works in [37], [46] to alleviate the burden in end devices for decryption.

Since user privacy risk is a long-time concern in CP-ABE schemes considering the cleartext access policy in ciphertexts [47], the hidden policy CP-ABE schemes come as a solution. To hide the sensitive information in access policy, Zhang et al. [23] proposed a CP-ABE scheme with partially hidden access policy supporting large universe and expressive policy. To reduce its high computation with composite bilinear group, Cui et al. [48] proposed an identical scheme based on prime order groups with verifiable outsourced decryption. For higher security, Zhou et al. [49] proposed a fully hidden policy CP-ABE but introduces high computation cost. Then, the works in [27], [50] proposed two efficent and key escrow free CP-ABE schemes that supports fully hidden policy. However, these schemes cannot guarantee the data integrity in regard to untrusted cloud. Recently, Cui et al. [28] raised a key escrow free public data auditing scheme in data sharing. Then, Garg et al. [31] improved the efficiency in public data auditing and the works in [11], [29], [30] make some progress in solving untrusted auditor. However, they cannot address the fine-grained access in data sharing and have no ability of scalability for multiple clouds to hold explosive growing s-Logistics data.

TABLE I: Function Comparison

Scheme	$\mathbf{F}_1$	$\mathbf{F}_2$	$\mathbf{F}_3$	$\mathbf{F}_4$	$\mathbf{F}_5$	$\mathbf{F}_6$	$\mathbf{F}_7$	$\mathbf{F}_8$	$\mathbf{F}_9$
Scheme [21]	✓	×	×	×	×	✓	✓	✓	×
Scheme [50]	✓	✓	×	×	×	×	✓	×	×
Scheme [43]	✓	×	×	✓	✓	×	✓	×	×
Scheme [28]	×	×	✓	×	×	×	×	×	×
Scheme [31]	×	×	✓	×	×	×	×	×	×
Scheme [29]	×	×	✓	×	×	×	×	×	✓
Scheme [37]	✓	×	×	✓	✓	×	✓	×	×
Scheme [19]	✓	✓	×	×	✓	✓	×	✓	×
Scheme [27]	✓	✓	×	✓	✓	×	✓	×	×
LUTPDS	✓	✓	✓	✓	✓	✓	✓	✓	✓

Note. $\mathbf{F}_1$: Fine-grained access control; $\mathbf{F}_2$: User privacy preserving; $\mathbf{F}_3$: Data integrity auditing; $\mathbf{F}_4$: Online/offline encryption; $\mathbf{F}_5$: Outsourced decryption; $\mathbf{F}_6$: User tracing; $\mathbf{F}_7$: Multiple authorities; $\mathbf{F}_8$: Large attribute universe; $\mathbf{F}_9$: Scalability to multiple clouds.

To seek a solution for data sharing in Fog-based s-Logistics that can support data confidentiality, fine-grained access control, data integrity, efficient encryption and decryption, key escrow and key abuse free, large attribute universe and scalability at the same time, we devise our scheme LUTPDS, while most of existing schemes cannot satisfy these requirement of s-Logistics – see Table I for the advantages of LUTPDS.

III. Preliminaries

In this section, we provide the notations and definitions used in our work including access structure and complexity assumptions.

A. Notations

Throughout this paper, we use the following notations described in Table II.

TABLE II: Notation Descriptions

Notations	Descriptions
Z_p	an integer set modulo a prime number p
$[l1, l2]$	set $\{l1, l1+1, \cdots, l2\}$ containing consecutive integers
$[n]$	the set that contains $\{1, \cdots, n\}$
$\|S\|$	the size of the set S
U_a, U_A	attribute universe and authority universe
GPR	the global public key
PK_{GID}	public key of user GID
SK_{GID}	secret key of user GID
$DK_{GID,S}$	user decryption key
$TK_{GID,S}$	user transformation key
IT	intermediate ciphertext pool for encryption
CT, CT_t	final ciphertext and transformed ciphertext
$CT_t^{'}$	part of transformed ciphertext for data integrity auditing
$CT_t^{''}$	transformed ciphertext components for user decryption

B. Access Structure

Definition 1. *(Access Structure [5]). Let $L_1, \cdots, L_n$ be a set of entities. M is a subset of $2^{\{L_1,\cdots,L_n\}}$. On condition that $\forall E, F$: if $E \in M$ and $E \subseteq F$, then $F \in M$ holds, M is monotonic. The access structure is such a collection of the nonempty subsets in $L_1, \cdots, L_n \backslash \varnothing$. The component set of M are defined as authorized sets used to describe authorized users, and others are unauthorized sets.*

C. Linear Secret Sharing Schemes (LSSS)

Definition 2. *(LSSS [17]). LSSS is the expression of access structure over the attribute universe U_a. It is defined as (A, ρ) including a $x \times y$ share-generating matrix A over Z_p and a map ρ from each row A_x of A into an attribute in U_a. Given a secret value s and a vector $\vartheta = (s, \theta_2, \cdots, \theta_y)^T$, where $\theta_2, \cdots, \theta_y \in_R Z_p$, then $\xi_x = A_x \cdot \vartheta$ is a share of s. Moreover, for any authorized set $L \in (A, \rho)$ and $I = \{k | \rho(k) \in L \bigwedge k \in [x]\}$, there exists a constance set $\{\omega_k\}_{k\in I}$ satisfies $\sum_{k\in I} \omega_k A_k = (1, 0, \cdots, 0)$ and thus $\sum_{k\in I} \omega_k \xi_x = s$.*

D. Cryptographic Background and Complexity Assumptions

Definition 3. *(Bilinear Maps [13]): Assuming X_1 and X_2 are two multiplicative cyclic groups of prime order p and f is a generator of group X_1. $\hat{e}$: $X_1 \times X_1 \to X_2$ is considered to be a bilinear map if it satisfies: 1)*

1) *Bilinear: $\hat{e}(f^x, h^y) = \hat{e}(f, h)^{xy}$, $\forall x, y \in Z_p, h \in X_1$.*
2) *Non-degenerate: $\hat{e}(f, f) \neq 1$.*
3) *Computable: $\hat{e}$ is feasible to be computed.*

Definition 4. *(q-DPBDHE2 assumption [39]). Given $T = (X_1, p, \hat{e}, f, f^a, \{f^{b^i}\}_{i\in[2q],i\neq q+1}, \{f^{c_j b^i}\}_{(i,j)\in[2q,q],i\neq q+1}, \{f^{a/c_i}\}_{i\in[q]}, \{f^{ab^i c_j/c_{j'}}\}_{(i,j,j')\in[q+1,q,q],j\neq j'})$, we can define the q-DPBDHE2 problem as differentiating $\hat{e}(f, f)^{ab^{q+1}}$ from $Z \in_R X_2$. If no Probabilistic Polynomial Time (PPT) algorithm can solve the q-DPBDHE2 problem with a non-neglighble advantage, q-DPBDHE2 assumption holds.*

Definition 5. *(l-SDH assumption [25]). Assuming a $(l + 1)$-tuple $(h, h^x, h^{x^2}, \cdots, h^{x^l})$, the l-SDH problem is to output a pair $(c, h^{1/1+c}) \in Z_p \times G_x$. If $|P_r[\mathcal{A}(h, h^x, h^{x^2}, \cdots, h^{x^l}) = (c, h^{1/x+c})]| \geq \varepsilon$ for an algorithm $\mathcal{A}$ randomly choosing $x \in Z_p$, then the advantage for an algorithm solving the problem is ε.*

If no probabilistic polynomial time (PPT) algortihm $\mathcal{A}$ can solve the l-SDH problem with the advantage at least ε, then the l-SDH assumption holds.

Definition 6. *(Computational Diffie-Hellman (CDH) assumption [28]). Given the muliplicative group G^* of prime order p with its generator $g_0 \in G^*$ and $g^x, g^y \in_R G^*$, where $x, y \in_R Z_p$, the CDH problem is to output $g^{xy} \in G^*$. If it can be solved with negligible advantage, then CDH assumption holds.*

IV. Problem Formulation

In this section, we present the system model and threat model of our proposal first. At the end, we give its formal system definition and security model.

A. System and Threat Model

In our system, as shown in Fig.2, we introduce seven different entities: Smart Logistics Cloud (SLC), Central Authority (CA), Attribute Authorities (AA), Public Auditing Server (PAS), Fog Nodes (FN), Data Owner (DO) and Data User (DU).

- SLC provides users with centralized data outsourcing and sharing services as well as unlimited storage and computational resources. It also supports cloud encryption and decryption for multiple cloud scalability and key escrow resistance.
- CA is in charge of system initialization by generating global public parameters and master keys for the whole system. It also provides a part of secret key generation for users to resist key escrow.
- AA is a key entity that takes charges of user attributes management and decryption distribution. There are multiple AAs in the system, each of which manages a disparate set of attributes in the system to address single point failure and key escrow problems.
- PAS is responsible for data integrity auditing by interaction with SLC in the challenge and response mode to protect the outsourced s-Logistics data from being violated by corrupted cloud servers.
- FN is an entity that is deployed in the merge of SLC and can supply outsourced decryption service for users to achieve short time-delay in data access and eliminate high computation in smart devices of s-Logistics.
- DO collects important information from devices of Smart Logistics System and uploads the massive s-Logistics data to SLC through FN which are encrypted with designated access policy for data confidentiality and fine-grained access control. Moreover, DO can prepare offline ciphertext components while devices are accessing the power source and generate final ciphertext online after the data is given to save computational cost.
- DU access the shared ciphertexts of s-Logistics data from SLC and downloads the transformed ciphertexts through FN if he is authorized. DU can further output plaintext of s-Logistics data he needs after verification. Besides, the traitor of DU that illegally leaks his decryption key for benefit will be traced by anyone of the system.

In our system of LUTPDS, PAS and DO are regarded as the fully trusted entities while the FN is considered to be untrusted which is curious about the transmitted ciphertext and may intentionally leak or modify the content of data when executing outsourced decryption. CA, AAs and SLC are regarded as semi-honest entity which prefers to perform honestly but may disclose the private data in an "honest-but-curious" manner. CA and AAs are in charge of key generation for user and bring about the risk of key escrow by collusion to master the whole secret keys and decryption keys of users in order to illegally access the outsourced data. SLC maintains large amounts of user data and has the ability to eavesdrop shared s-Logistics data or even violates the data integrity by accident or on purpose. Besides, malicious DU may be unauthorized to access the sensitive data, whereas can also try

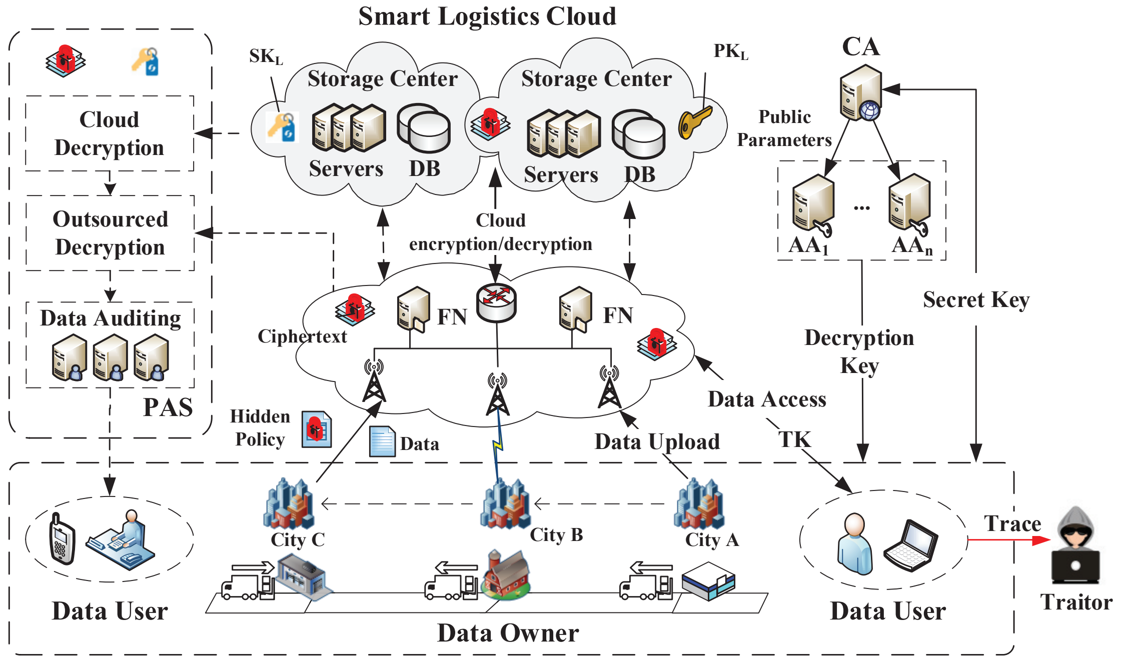


Fig. 2: The system model of LUTPDS for s-Logistics

to break the data security and privacy of DO. In this paper, we declare that the SLC and AAs cannot collude with each other, which is reasonable in real-world applications.

B. System Definition

We present the formal definition of our proposal that consists of several algorithms. In system initialization phase, $Setup_{Global}$ algorithm is executed by CA to initiate the system and $Setup_{AA}$ algorithm is used to set up each AA while $Setup_{SLC}$ algorithm is run for SLC initialization. In user enrollment phase, $UKeyGen$ algorithm is utilized to generate user key pair by CA and $DKeyGen$ algorithm is executed by AA to distribute decryption key for users while $TKeyGen$ is run by users for transformation key generation. In data encryption phase, $Encrypt_{off}$ and $Encrypt_{on}$ algorithms execute together for efficient data encryption. In data decryption phase, FN leverages $Transform$ algorithm for outsourced decryption. After successful verification by PAS with $Audit$ algorithm, DU can use $Decrypt$ algorithm to recover data plaintext. In the whole system, any entity can achieve public user tracing with $Trace$ algorithm.

Fig.3 shows the general workflow of LUTPDS which includes different phases of system initialization, user enrollment, data encryption, data decryption and user tracing, as well as the algorithms defined in System Definition IV-B.

C. Security Model

The security requirements of LUTPDS need the static security, traceability and data integrity security which are defined as follows.

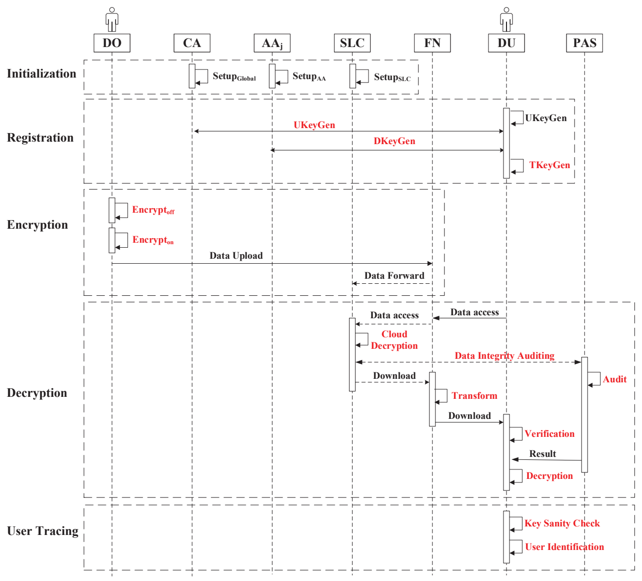


Fig. 3: The workflow of LUTPDS

1) Static security: The static security of our proposal with multiple authority allows that a set of authorities to be corrupted chosen by adversary. The static security is defined as following with interactions of an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$:

- *Setup*. The challenger $\mathcal{C}$ runs $Setup_{Global}$ to initiate the system and generates the global public key GPR which is sent to the adversary $\mathcal{A}$.
- *Phase 1*. The adversary $\mathcal{A}$ chooses a set of corrupted authorities $AA_C \subset U_A$ and generates the public keys of these authorities by itself. Then, it issues following queries:

  - It selects the authorities that are not corrupted as $AA_N \subset U_A$ for their authority public keys.
  - It chooses a set of valid users $\{GID_i\}_{i\in[m]}$ for their public and secret key pairs.
  - It issues a set of transformation key queries with global identity and attribute set pairs $\{(GID_i, S_i)\}_{i\in[n]}$, where $S_i \subset U_a$ and $f_1(S_i) \bigcap AA_C = \emptyset$.
- *Challenge*. The adversary $\mathcal{A}$ submits two equal-length messages M_0, M_1 and a challenging access policy (A, ρ) to $\mathcal{C}$. Note that the S_i in *Phase 1* and the attribute set S_{AA_C} mastered by corrupted authorities does not satisfy (A, ρ). The challenger $\mathcal{C}$ responses the queries of $\mathcal{A}$ with authority public keys $\{APK_j\}_{j\in AA_N}$ and user decryption keys $\{DK_{GID_i,S_i}\}_{i\in n}$ with corresponding transformation keys. It also outputs the corresponding ciphertext CT by randomly picks a bit $b \in \{0,1\}$ and encrypting M_b with encryption algorithm.
- *Guess*. The adversary $\mathcal{A}$ outputs the guess b_0 of b. If $b_0 = b$, it wins the game and the advantage is defined as below:
$$Adv_{\mathcal{A}} = |\Pr[b_0 = b] - \frac{1}{2}|.$$

Definition 7. *LUTPDS has static security against statically corrupted authorities in random oracle model if no PPT adversary can break the static security game with a non-negligible advantage.*

2) Traceability: The traceability of our scheme is modeled through a security game between adversary $\mathcal{A}$ and challenger $\mathcal{C}$ as follows:

Init: $\mathcal{C}$ executes $Setup_{Global}, Setup_{AA}$ algorithm to initiates the system and sends the global public key GPR and authority public key $\{APK_j\}$ to $\mathcal{A}$.

Key Query: $\mathcal{A}$ conducts n queries for decryption key generation with (GID_1, S_1),
$\cdots, (GID_n, S_n)$, where GID_i and S_i are the global identity and attribute set, respectively. Then, $\mathcal{C}$ runs $DKeyGen$ to derive decryption keys for $\mathcal{A}$

Key Forgery: $\mathcal{A}$ outputs a decryption key DK^*.

If $\mathcal{A}$ can win the game, then for a well-formed decryption key DK^*, $Trace(GPR, DK^*, \{APK_j\}) \notin \{null, GID_1, \cdots, GID_n\}$. Thus, the advantage of $\mathcal{A}$ in winning the game is:
$$Adv_{\mathcal{A}} = P_r[Trace(GPR, DK^*, \{APK_j\}) \notin \{null, \\ GID_1, \cdots, GID_n\}].$$

Definition 8. *LUTPDS has traceability if no PPT adversary can break the above game with a non-negligible advantage.*

3) Data Auditing Security: Here, we give the formal definition of data auditing security game between a PPT adversary $\mathcal{A}$ acting as a corrupted CA and a challenger $\mathcal{C}$.

- *Setup*: The challenger $\mathcal{C}$ initiates the system by executing $Setup_{Global}$ to generate global public key GPR which are sent to $\mathcal{A}$.
- *Phase 1*: The adversary $\mathcal{A}$ submits following queries to $\mathcal{C}$.
  - H_0 *query*: $\mathcal{A}$ issues this query for H_0 to $\mathcal{C}$.
  - H_2 *query*: $\mathcal{A}$ issues this query for H_2 to $\mathcal{C}$.
  - SK_1 *query*: $\mathcal{A}$ issues this query with global identity GID for $SK_{GID,1}$ to $\mathcal{C}$.
  - SK_2 *query*: $\mathcal{A}$ issues this query with global identity GID for $SK_{GID,2}$ to $\mathcal{C}$.
  - *S query*: $\mathcal{A}$ issues this query with file F for signature generation to $\mathcal{C}$ which then executes $UKeyGen$ and $Encrypt_{on}$ algorithms to generates the signatures for F. At the end, $\mathcal{C}$ returns the signatures back to $\mathcal{A}$.
- *Challenge*: $\mathcal{C}$ is in the role of auditor and sends the challenge request $R_c = \{k, v_k\}_{k\in[n]}$ to $\mathcal{A}$ as a prover for data possession proof.
- *Forge*: $\mathcal{A}$ sends the data posession proof R_p under the challenge request R_c. $\mathcal{A}$ breaks the security only if R_p can pass the auditing of $\mathcal{C}$ with non-negligible probability.

Definition 9. *LUTPDS secure in data auditing against a corrupted CA if no PPT adversary can break the data auditing game with non-negligible advantage.*

V. The Proposed Scheme for Fog-based s-Logistics

In this section, we describe first the overview of the techniques utilized in LUTPDS for s-Logistics data access control together with its workflow and then present its concrete construction.

A. Overview of techniques in LUTPDS

To address the problems of data sharing in Cloud-Fog-based s-Logistics, we adopt the setting of Fog-based IoT architecture for short time-delay in service provision. Considering data security, we devise the LUTPDS scheme based on CP-ABE with multi-authority and verifiable outsourced decryption to realize efficient data sharing in large scale s-Logistics systems. In our proposal, to avoid user collision, each user in such a distributed architecture is bound up with a Global Identity (GID) [21]. To resist key abuse, any malicious user who aims to gain illegal interest by leaking their decryption key to unauthorized outsiders will be revealed by white-box tracing mechanism. As a solution for key escrow problem with corrupted CA and AAs, we leverage the certificateless mechanism and multi-authority CP-ABE to prevent user secret key and decryption keys from being mastered by a central authority and single attribute authority, respectively. However, the authorities in multi-authority CP-ABE schemes may collude to recover the information of user decryption keys. To this end, we design the cloud-side encryption and decryption to further resist key escrow problem, which can also support the setting of multiple cloud for scalability. Moreover, to reduce the overhead in data encryption, we introduce online/offline technique. Furthermore, to achieve user privacy preserving in access policy embedded in ciphertexts and attribute set in transformed keys, we leverage the one-way anonymous key agreement protocol [27] to hide the cleartext in attributes of access policies and user attribute sets. To resist corrupt cloud servers in SLC that may compromise the integrity of outsourced data and the key escrow problem, we integrate the certificateless public data integrity auditing and cloud-assisted encryption into our data access control scheme.

B. Concrete Construction

1) System Initialization: CA generates system public parameters and master keys for the whole system. Each attribute authority and SLC also create their public and secret key pair for initialization.

- $Setup_{Global}(\kappa) \rightarrow GPR$: Given the security parameter κ, CA executes following initiation steps:
  - CA creates a big primes p of κ bits and generates a bilinear group $(G^*, G_T^*, p, g, \hat{e})$, where G^*, G_T^* are two multiplicative cyclic groups with order p, g is a generator of G^* and $\hat{e}: G^* \times G^* \rightarrow G_T^*$ is a bilinear map. It then calculates $Y = \hat{e}(g, g)$.
  - CA defines two functions $f_1 : U_a \rightarrow U_A$ from attribute universe U_a to authority universe U_A and $f_2 : [l] \rightarrow U_A$ that maps a row of the matrix in access policy to attribute authority. Then, it picks $k_c \in_R Z_p$ as the secret key of CA and computes its public key $PK_c = g_0^{k_c}$.
  - CA also chooses hash functions $H : \{0,1\}^* \rightarrow G_T^*, H_0 : \{0,1\}^* \rightarrow G^*, H_1 : U_a \rightarrow G^*, H_2 : \{0,1\}^* \rightarrow Z_p$ and a pair of symmetric encryption algorithm (Enc_s, Dec_s) with an element $g_0, u \in_R G^*$.
  - Finally, CA publishes the global public key as $GPR = \{G^*, G_T^*, \hat{e}, g, g_0, u, p, Y, (Enc_s, Dec_s),$ $f_1, f_2, H, H_0, H_1, H_2, PK_c\}$.
- $Setup_{AA}(GPR) \rightarrow \{APK_i, ASK_i\}$: Each attribute authority $AA_i \in U_A$ randomly selects $\alpha_i, \beta_i, a_i, b_i \in Z_p$. Then, it computes $\hat{e}(g,g)^{\alpha_j}$. Finally, it publishes its public key $APK_i = \{\hat{e}(g,g)^{\alpha_i}, g^{\beta_i}, g^{a_i}, g^{b_i}\}$ and keeps their master key $ASK_i = \{\alpha_i, \beta_i, a_i, b_i, \theta_i\}$ privately.
- $AttHidden(GPR, S, \{ASK_j\}) \rightarrow S^{'}$: On inputting an attribute set S and the secret keys $\{ASK_j\}$ of authorities corresponding to attribute $att_i \in S$, AA computes the hidden attribute set $S^{'}$ as $att_i^{'} = \hat{e}(g^{a_j}, H_0(att_i)^{\theta_j})$, where $j = f_1(att_i)$, that is, att_i is managed by authority AA_j.
- $Setup_{SLC}(GPR, ID_L)$: On inputting the identity ID_L of each storage center in SLC, the algorithm initiates the SLC by generating the secret key $SK_L = k_l \in_R Z_p$ and computing the public key $PK_L = g_0^{k_l}$ of the distributed storage center in SLC.

2) User Enrollment: CA distributes the part of secret key with which each user obtains his complete secret key. Each AA creates the decryption key for users after receiving their enrollment requestments by which users can generate their transformation keys for outsourced decryption in Fog.

- $UKeyGen(GPR, GID) \rightarrow \{PK_{GID}, SK_{GID}\}$: Given the global identity GID of a user, CA selects $k_{GID} \in_R Z_p$ as a part of secret key $SK_{GID,1} = k_{GID}$ of the user GID and computes corresponding public key $PK_{GID} = g_0^{k_{GID}}$. Then, after receiving the enrollment request of user GID, CA computes $SK_{GID,2} = k_c \cdot H_2(GID)$ as another part of secret key of the user GID. CA sends $SK_{GID,2}$ to the user GID through secure channel. Finally, the user gets the secret key $SK_{GID} = \{SK_{GID,1}, SK_{GID,2}\}$ and publishes the public key PK_{GID}.
- $DKeyGen(GPR, GID, S_{GID}, \{ASK_j\}) \rightarrow DK_{GID,S}$: Given the user global identity GID and his attribute set S_{GID} as well as the master key $\{ASK_j\}$ of attribute authorities that each attribute $att_i \in S_{GID}$ belongs to (different attribute in S_{GID} may belong to the same authority), each corresponding authority AA_j $(j = H(att_i))$ executes the following steps:
  - For each $att_i \in S_{GID}$, the corresponding authority AA_j chooses $\mu_{GID}, \nu_{GID} \in_R Z_p$ and generates the secret attribute key components $DK_{GID,i}$ for user GID according to attribute set S_{GID}, where

$$\begin{aligned}
&D_{GID,i,1} = g^{\overline{a_j+GID+b_j\mu_{GID}}^{\alpha_j}} \cdot \\
&\qquad H_0(GID)^{\overline{a_j+GID+b_j\mu_{GID}}^{\beta_j}} H_1(att_i)^{\nu_{GID}}, \\
&D_{GID,2} = GID, D_{GID,i,3} = \mu_{GID}, \\
&D_{GID,i,4} = g^{\nu_{GID}}, D_{GID,i,5} = g^{(a_j+b_j\mu_{GID})\nu_{GID}}
\end{aligned}$$

  - Finally, the user GID gets the decryption key $DK_{GID,S} = \{S, \{DK_{GID,i}\}_{att_i \in S_{GID}}\}$ via secure channel.
- $TKeyGen(GPR, DK_{GID,S}, SK_{GID}) \rightarrow TK_{GID,S}$: Given the decryption key $DK_{GID,S}$ and the secret key SK_{GID} of user GID, each user generates the transformation key $TK_{GID,S}$ for outsourced decryption by following steps:
  - It computes the transformation key components as:

$$\begin{aligned}
&K_0 = H_0(D_{GID,2})^{SK_{GID,1}}, \\
&\forall att_i \in S : \\
&\qquad K_{GID,i,1} = (D_{GID,i,1})^{SK_{GID,1}}, \\
&\qquad K_{GID,2} = D_{GID,2}, \\
&\qquad K_{GID,i,3} = D_{GID,i,3}, \\
&\qquad K_{GID,i,4} = (D_{GID,i,4})^{SK_{GID,1}}, \\
&\qquad K_{GID,i,5} = (D_{GID,i,5})^{SK_{GID,1}}
\end{aligned}$$

  - Then, each user generates the hidden attribute set $S^{'}$ according to S by interacting with the authorities that in charge of each $att_i \in S$ and executes the algorithm $AttHidden(GPR, S, \{ASK_j\})$ to obtain $S^{'}$. Thus, no other parties can distinguish each attribute of the user.
  - Finally, the user generates his transformation key $TK_{GID,S} = \{GID, S^{'}, K_0, \{K_{GID,i,1}, K_{GID,i,2},$ $K_{GID,i,3}, K_{GID,i,4}, K_{GID,i,5}\}_{i \in S}\}$.

3) Data Encryption: DO encrypts the collected s-Logistics records based on a designated hidden access policy and outsources the final ciphertext to SLC for data sharing through FN.

- $Encrypt_{off}(GPR, SK_{GID}) \rightarrow IT$: DO randomly picks the secret value $s \in Z_p$ and symmetric key $k_s \in_R Z_p$ of Enc_s to compute $C_0 = k_s \cdot \hat{e}(PK_L^{SK_{GID,1}}, H_0(ID_L)) \cdot$

$\hat{e}(g,g)^s$ and $C_v = H(\hat{e}(g,g)^s)$, where PK_L, ID_L are the public key and identity of the storage center in SLC. Then, DO chooses $\lambda_x^{'}, \theta_x, \vartheta_x^{'} \in_R Z_p$ and calculates $C_{x,1}^{'} = \hat{e}(g,g)^{\lambda_x^{'}}, C_{x,2}^{'} = g^{\theta_x}, C_{x,3}^{'} = g^{\vartheta_x^{'}}$. Finally, the algorithm constucts the intermediate ciphertext pool $IT = \{IT_0, IT_1\}$, where $IT_0 = \{k_s, s, C_0, C_v\}$ and $IT_1 = \{\lambda_x^{'}, \theta_x, \vartheta_x^{'}, C_{x,1}^{'}, C_{x,2}^{'}, C_{x,3}^{'}\}$.

- $Encrypt_{on}(GPR, SK_{GID}, IT, F, \mathcal{A}, \{PK_j\}) \rightarrow CT$: Given the intermediate ciphertext pool IT, the data file F with a designated LSSS access policy (A, ρ) and public keys of corresponding authorities, DO executes as following:
  - First of all, DO chooses a random tuple $\{k_s, s, C_0, C_v\}$ from IT_0. For each block $f_i \in F$ (suppose F has n blocks), it computes the encrypted file as $C_s = \{C_i\}_{f_i \in F}$, where $C_i = Enc_s(k_s, f_i)$ with symmetric encryption algorithm Enc_s in GPK.
  - Then, for LSSS access policy $\mathcal{A} = (A, \rho)$, where A is a $l \times m$ share-generating matrix and ρ is the corresponding map from each row A_x of A to a certain attribute $att_x \in U_a$ (that is, $\rho(A_x) = att_x$, for each $x \in [l]$), we can get the index $j = f_1(att_x)$ of the corresponding attribute authority AA_j in charge of the attribute att_x. The algorithm selects l random tuples from IT_1 as $\{\lambda_x^{'}, \theta_x, \vartheta_x^{'}, C_{x,1}^{'}, C_{x,2}^{'}, C_{x,3}^{'}\}_{x \in [l]}$.
  - DO constructs two random vectors $\gamma = \{s, \gamma_2, \cdots, \gamma_m\}$ and $\gamma^{'} = \{0, \gamma_2^{'}, \cdots, \gamma_m^{'}\}$, where $\gamma_i, \gamma_i^{'} \in_R Z_p$ for $i \in [m]$. Thus, it can get $\lambda_x = A_x \cdot \gamma$ which is the share component of s for each attribute $\rho(x)$ corresponding to x-th row A_x, and $\vartheta_x = A_x \cdot \gamma^{'}$ which is the share component of 0 for each attribute of access policy. Moreover, the algorithm computes the ciphertext components according to $\mathcal{A}$ for each $x \in [l]$ as below:

$$\begin{aligned}
&C_{x,1} = C_{x,1}^{'} \cdot \hat{e}(g,g)^{\alpha_{f_2(x)}\theta_x}, C_{x,2} = C_{x,2}^{'},\\
&C_{x,3} = g^{\beta_{f_2(x)}\theta_x} C_{x,3}^{'}, C_{x,4} = H_1(\rho(x))^{\theta_x},\\
&C_{x,5} = g^{a_{f_2(x)}\theta_x}, C_{x,6} = g^{b_{f_2(x)}\theta_x},\\
&C_{x,7} = \lambda_x - \lambda_x^{'}, C_{x,8} = \vartheta_x - \vartheta_x^{'}
\end{aligned}$$

  - Next, DO calculates the fully hidden access policy by hiding the attribute set S_a related to the LSSS policy $\mathcal{A}$. To this end, DO interacts with corresponding authorities that charge the attributes in S_a with algorithm $AttHidden(GPR, A_a, \{ASK_j\})$ to get the fully hidden attribute set $\mathcal{S}_a$ and construct the fully hidden access policy $\mathcal{A}^{'}$.
  - Further, DO computes the signature for each encrypted file block $C_i \in C_s$ as $\sigma_i = H_0(ID_f||i)^{SK_{GID,1}} u^{SK_{GID,2} H_2(C_i)}$, where $SK_{GID,1}, SK_{GID,2}$ is the secret key of DO. Thus, the algorithm obtains the signature $\sigma_s = \{\sigma_i\}_{C_i \in C_s}$. Then, it calculates the file tag as $\zeta = ID_f||GID||PK_o||Sig(ID_f||GID||PK_o)$, where Sig is a signature algorithm, ID_f is the identity of the file and GID and PK_o are the identity and the public key of DO.
  - Finally, DO outputs the ciphertext $CT = \{\zeta, \mathcal{A}^{'}, C_s, \sigma_s, C_v, C_0, \{C_{x,1}, C_{x,2}, C_{x,3}, C_{x,4}, C_{x,5}, C_{x,6}, C_{x,7}, C_{x,8}\}_{x \in [l]}\}$ which is uploaded to SLC. Before adopting the ciphertext, SLC verifies its correctness by checking the signature of file tag ζ and the equation $\hat{e}(\sigma_i, g_0) \overset{?}{=} \hat{a}(H_0(ID_f||i), PK_o)\hat{e}(u^{H_2(GID)H_2(C_i)}, PK_c)$, where PK_o is the public key of DO. If the ciphertext CT is valid, SLC accepts it and finishes data outsourcing.

4) Data Decryption: DU accesses the shared ciphertexts in SLC through FN. Then, the storage center of SLC partially decrypts the ciphertexts and offloads decryption to FN as well as data integrity auditing to PAS. With the transformation key and the ciphertexts, FN transforms the ciphertexts into transformed ciphertexts which are used to recover the plaintexts by DU with their secret key.

- $Transform(GPR, CT, TK_{GID,S}, PK_o, ID_L, SK_L) \rightarrow CT_t$: Given the transformation key $TK_{GID,S}$ of the DU GID and the cipertext CT with the public key PK_o of DO as well as the identity ID_L and secret key SK_L of the storage center, FN conducts the following procedures:
  - The storage center of SLC computes $C_0^{'} = C_0/\hat{e}(PK_o^{SK_L}, H_0(ID_L))$ and gets the fully hidden access policy $\mathcal{A}^{'} = \{A, \rho\}$ from CT with its hidden attribute set $\mathcal{A}_a$. Then, from the transformation key $TK_{GID,S}$, FN can fetch the hidden user attribute set $S^{'}$. Thus, it can rebuild a index set $\mathcal{I} \subset [l]$ of the rows in A by calculating $\mathcal{I} = \mathcal{S}_a \bigcap S^{'}$.
  - Then, FN finds a set of constances $\{\omega_x \in Z_p\}$ which makes the equation $\sum_{x \in \mathcal{I}} \omega_x A_x = (1, 0, \cdots, 0)$ holds if the DU GID is authorized. For each row $x \in \mathcal{I}$ of access policy, the algorithm computes:

$$\begin{aligned}
&Q_x = C_{x,1} \cdot Y^{C_{x,7}},\\
&R_x = \frac{\hat{e}(K_0, C_{x,3} \cdot g^{C_{x,8}})}{\hat{e}(K_{GID,\rho(x),1}, C_{x,2}^{K_{GID,2}} C_{x,5} C_{x,6}^{K_{GID,\rho(x),3}})}\\
&\quad \cdot \hat{e}(K_{GID,\rho(x),4}^{K_{GID,\rho(x),2}} K_{GID,\rho(x),5}, C_{x,4})
\end{aligned}$$

  - Next, according to the constances $\{\omega_x\}$, SLC computes the following equation:

$$Q = \prod_{x \in \mathcal{I}} (Q_x)^{\omega_x}, R = \prod_{x \in \mathcal{I}} (R_x)^{\omega_x}$$

  - Finally, FN outputs the transformed ciphertext $CT_t = \{\zeta, C_s, \sigma_s, C_v, C_0^{'}, Q, R\}$, in which the $CT_t^{'} = \{\zeta, C_s, \sigma_s\}$ are sent to PAS for integrity auditing and $CT_t^{''} = \{\zeta, C_0, C_v, Q, R\}$ are sent to DU for decryption.
- $Audit(GPR, CT_t^{'}) \rightarrow C_s/null$: Given the encrypted file blocks $C_s = \{C_1, \cdots, C_n\}$ and its corresponding signature set $\sigma_s = \{\sigma_1, \cdots, \sigma_n\}$, PAS interacts with SLC for data integrity auditing as follows:

  - PAS checks the validity of file tag ζ by signature verification algorithm and gets the corresponding file identifier ID_f and the DO's identity GID with public key PK_o.
  - PAS then selects $v_k \in_R Z_p$ for each index $k \in I$, where $I \subset [n]$, and builds a challenge request $R_c = \{k, v_k\}$ which is sent to SLC.
  - SLC generates corresponding proof by computing $A = \sum_{k\in I} v_k H_2(C_k)$ and $B = \prod_{k\in I} \sigma_k^{v_k}$. It then builds a proof response as $R_p = \{A, B\}$ which is sent back to PAS.
  - PAS verifies the proof R_p by following equations:

$$\hat{e}(B, g_0) \overset{?}{=} \hat{e}(\prod_{k\in I} H_0(ID_f||k)^{v_k}, PK_o)\cdot \hat{e}(u^{A\cdot H_2(GID)}, PK_c)$$

  If the above equation holds, PAS sends ζ, C_s to DU for decryption; otherwise, it abandons the ciphertexts and outputs $null$.

- $Decrypt(GPR, C_s, CT_t^{''}, SK_{GID}) \rightarrow F$: Given the transformed ciphertext (C_s, C_v, C_0, R) and the secret key SK_{GID} of the user GID, DU first calculates $T = Q \cdot R^{1/SK_{GID,1}}$. If the equation $H(T) = C_v$ holds, the outsourced decryption is correct and DU calculates as following:

$$k_s^* = C_0/T, f_i = Dec_s(k_s^*, C_i)$$

in which it gets the symmetric key k_s^* and recovers the plaintext f_i of the file ID_f with symmetric decryption algorithm Dec_s and $C_i \in C_s$.

5) User Tracing: The global identity associated with malicious users of leaked decryption keys will be exposed by embedding the exact identities into the decryption keys of users.

- $Trace(GPR, DK_{GID,S}, \{APK_j\}) \rightarrow GID/null$: Given the leaked decryption key $DK_{GID,S}$, the global public key GPR and the public key of corresponding authority public keys $\{APK_j\}$, any entity of the system first check if $DK_{GID,S}$ is in the form of $DK_{GID,S} = \{S, \{DK_{GID,i}\}_{att_i \in S_{GID}}\}$ and $DK_{GID,i} = \{D_{GID,i,1}, D_{GID,i,2}, D_{GID,i,3}, D_{GID,i,4}, D_{GID,i,5}\}$. Then, it runs the following check as:
Key Sanity Check: $\exists att_i \in S$, s.t.

$$D_{GID,i,1}, D_{GID,i,4}, D_{GID,i,5} \in G^*,$$
$$D_{GID,i,2}, D_{GID,i,3} \in Z_p,$$
$$\hat{e}(g, D_{GID,i,5}) = \hat{e}(D_{GID,i,4}, g^{a_j} \cdot (g^{b_j})^{D_{GID,i,3}}),$$
$$\hat{e}(D_{GID,i,1}, g^{a_j} g^{D_{GID,i,2}} (g^{b_j})^{D_{GID,i,3}}) = \hat{e}(g,g)^{a_j}\cdot \hat{e}(H_0(D_{GID,i,2}), g^{\beta_j})\hat{e}(H_1(\rho(i)), D_{GID,i,4}^{D_{GID,i,2}} D_{GID,i,5})$$

where $j = f_1(att_i)$. The algorithm outputs the identity of the malicious user GID if its decryption key $DK_{GID,S}$ passes the key sanity check. Otherwise, it outputs $null$.

VI. Security and Performance Analysis

In this section, we show the detailed analysis of our proposal from the view of security and performance.

A. Security

This section presents detailed security proof for our LUTPDS in terms of static security of indistinguishability against chosen plaintext attack (IND-CPA), traceability and data auditing security.

1) Static Security:

Theorem 1. *If the RW scheme in [39] is static secure against statically corrupted authorities in random oracle model, then our proposed LUTPDS is static secure on condition that q-DPBDHE2 assumption holds in the same model.*

Proof. Suppose a PPT adversary $\mathcal{A}$ that can win the static security game of our LUTPDS with advantage ϵ, then we can build a simulator $\mathcal{B}$ to break the RW scheme with same advantage. Let the challenger of RW scheme is $\mathcal{C}$.

- *Setup*. The challenger $\mathcal{C}$ runs $GlobalSetup$ of RW scheme to initiate the system and generates the global public key $GP = \{p, G^*, g, H_0, H_1, U_a, U_A, f_1, f_2\}$ which is sent to the simulator $\mathcal{B}$. The simulator $\mathcal{B}$ selects a random element $g_0 \in G^*$, random values $k_c, k_l \in_R Z_p$ and sends $GPR = \{G^*, \hat{e}, g, g_0, p, Y = \hat{e}(g,g), f_1, f_2, H, H_0, H_1, PK_c = g_0^{k_c}\}$ as well as cloud public key $PK_L = g_0^{k_l}$ to the adversary $\mathcal{A}$.
- *Phase 1*. The adversary $\mathcal{A}$ chooses a set of corrupted authorities $AA_C \subset U_A$ and generates the public key of these authorities by itself as $\{PK_C\}$ which is sent to $\mathcal{B}$. Then, it issues following queries to $\mathcal{B}$:
  - It selects the authorities that are not corrupted as $AA_N \subset U_A$ for their authority public keys.
  - It chooses a set of valid users $\{GID_i\}_{i\in[m]}$ for their public and secret key pairs.
  - It issues a set of decryption queries with global identity and attribute set pairs $\{(GID_i, S_i)\}_{i\in[n]}$, where $S_i \subset U_a$ and $f_1(S_i) \bigcap AA_C = \emptyset$.
- *Challenge*. The adversary $\mathcal{A}$ submits two equal-length messages M_0, M_1 and a challenging access policy (A, ρ) to $\mathcal{C}$. Note that the S_i in *Phase 1* and the attribute set S_{AA_C} mastered by corrupted authorities does not satisfy (A, ρ). Then, the simulator $\mathcal{B}$ acts as below:
  - It submits the public keys $\{APK_C\}$ of corrupted authorities and the queries for the public keys of non-corrupted authorities AA_N to $\mathcal{C}$. $\mathcal{B}$ also forwards the global identity and attribute set pairs $\{(GID_i, S_i)\}_{i\in[n]}$ to $\mathcal{C}$ for corresponding decryption keys. In addition, $\mathcal{B}$ sends M_b and the challenging access policy (A, ρ) to $\mathcal{C}$ for corresponding ciphertext, where the bit $b \in_R \{0, 1\}$.
  - After receiving the authority public keys $\{APK_j\}_{j\in AA_N}$, the decryption keys $\{DK^*_{GID_i,S_i} = \{D^*_{GID_i,k} = g^{\alpha_j} H_0(GID_i)^{\beta_j} H_1(att_k)^{\mu_{GID_i}}, D'_{GID_i,k} = g^{\mu_{GID_i}}\}_{att_k \in S_i}\}$ and the challenging ciphertext $CT^* = \{C_0^* = M_b \cdot \hat{e}(g,g)^s, \{C^*_{x,1} = \hat{e}(g,g)^{\lambda_x} \hat{e}(g,g)^{\alpha_{f_2(x)}\theta_x}, C^*_{x,2} = g^{\theta_x}, C^*_{x,3} = g^{\beta_{f_2(x)}\theta_x} g^{\vartheta_x}, C^*_{x,4} = H_1(\rho(x))^{\theta_x}\}\}$, the simmulator $\mathcal{B}$ selects $\hat{\mu}_{GID_i} \in_R Z_p$ and generates the secret key as $SK_{GID_i,1} = k_{GID_i} \in_R Z_p$, the decryption key D_{GID_i,S_i} as:

$$D_{GID_i,k,1} = (D^*_{GID_i,k})^{\frac{1}{a_j+GID_i+b_j\hat{\mu}_{GID_i}}},$$
$$D_{GID_i,2} = GID_i, D_{GID_i,i,3} = \hat{\mu}_{GID_i},$$
$$D_{GID_i,k,4} = (D'_{GID_i,k})^{\frac{1}{a_j+GID_i+b_j\hat{\mu}_{GID_i}}},$$
$$D_{GID_i,k,5} = D_{GID_i,k,4}^{a_j+b_jD_{GID_i,i,3}}$$

and corresponding transformation key TK_{GID_i,S_i} as :

$$K_0 = H_0(D_{GID_i,2})^{SK_{GID_i,1}},$$
$$\forall att_k \in S_i :$$
$$K_{GID_i,k,1} = (D_{GID_i,k,1})^{SK_{GID_i,1}},$$
$$K_{GID_i,2} = D_{GID_i,2}, K_{GID_i,k,3} = D_{GID_i,k,3},$$
$$K_{GID_i,k,4} = (D_{GID_i,k,4})^{SK_{GID_i,1}},$$
$$K_{GID_i,k,5} = (D_{GID_i,k,5})^{SK_{GID_i,1}}$$

For those pairs $\{(GID_i, S_i)\}_{m\le i\le n}$, the simulator $\mathcal{B}$ acts its own key generation algorithms.

– It then computes the challenging ciphertext as $CT = \{C_0 = C_0^* \cdot \hat{e}(PK_L^{SK_{GID_i,1}}, H_0(ID_L)), \{ C_{x,1} = C^*_{x,1}, C_{x,2} = C^*_{x,2}, C_{x,3} = C^*_{x,3}, C_{x,4} = C^*_{x,4}, C_{x,5} = C_{x,2}^{a_{f_2(x)}}, C_{x,6} = C_{x,2}^{b_{f_2(x)}}\}_{x\in[l]}\}$ and sends the authority public keys $\{APK_j\}_{j\in AA_N}$, transformation keys $\{TK_{GID_i,S_i}\}$ and challenging ciphertext CT to $\mathcal{A}$.

- *Guess*. The adversary $\mathcal{A}$ outputs the guess b_0 of b and thus $\mathcal{B}$ outputs the same guess to $\mathcal{C}$.

Therefore, if $\mathcal{A}$ can win the static security of our LUTPDS with advantage ϵ, the simulator $\mathcal{B}$ can break RW scheme with the same advantage. As a result of this reduction, if q-DPBDHE2 assumption holds, RW is static secure and so does oour LUTPDS. □

2) Traceability: In this part, we show the reduction of the traceability of our LUTPDS scheme to Boneh-Boyen scheme in [51].

Theorem 2. *If the Boneh-Boyen scheme is secure against strong existential forgery, then our LUTPDS scheme is fully traceable if l-SDH assumption holds.*

Proof. If there exists a PPT adversary $\mathcal{A}$ that can break the traceability game with non-negligible advantage ε, then we can construct a simulation $\mathcal{B}$ that have the ability to break the security of Boneh-Boyen (BB) scheme under adaptive chosen message attack with advantage ε. Suppose $\mathcal{C}$ is the challenger of BB scheme and G^*, G_T^* are two bilinear groups of prime order p with a bilinear map $\hat{e} : G^* \times G^* \to G_T^*$ and a generator $g \in G^*$.

Init. $\mathcal{C}$ initiates BB scheme and sends its public key $\{G^*, G_T^*, p, g, g^{a_j}, g^{b_j}\}$ to $\mathcal{B}$ which initiates the whole system by creating the global public key $GPR = \{G^*, G_T^*, p, g, Y = \hat{e}(g, g), f_1, f_2, H_0, H_1\}$ and the attribute authorities by picking $\alpha_j, \beta_j \in_R Z_p$ to generate the authority public key as $APK_j = \{\hat{e}(g, g)^{\alpha_j}, g^{\beta_j}, g^{a_j}, g^{b_j}\}$, where $j \in U_A$. $\mathcal{B}$ sends the GPR and $\{APK_j\}$ to adversary $\mathcal{A}$.

Key Query. $\mathcal{A}$ submits the global identity and attribute set pair $\{(GID_i, S_i)\}\{i \in [n]\}$ to $\mathcal{B}$ for the corresponding decryption key query. To respond the adversary $\mathcal{A}$, $\mathcal{B}$ creates two empty lists L_1, L_2 and executes following steps:

- O_{H_0}: If a tuple $(GID, r_{GID}, g^{r_{GID}}) \in L_1$, $\mathcal{B}$ responds $g^{r_{GID}}$, else, it responds $g^{r'_{GID}}$ by selecting $r'_{GID} \in_R Z_p$ and adding the tuple $(GID, r'_{GID}, g^{r'_{GID}})$ to L_1.
- O_{H_1}: If a tuple $(att_i, r_i, g^{r_i}) \in L_2$, $\mathcal{B}$ responds g^{r_i}, else, it responds $g^{r'_i}$ by selecting $r'_i \in_R Z_p$ and adding the tuple $(att_i, r'_i, g^{r'_i})$ to L_2.

As to (GID_i, S_i), for each $att_k \in S_i$ and $j = f_1(att_k) \in U_a$, $\mathcal{B}$ sends (GID_i, j) to $\mathcal{C}$ to acquire the signature $(\mu, \pi_i = g^{\frac{1}{a_j+GID_i+b_j\mu}})$, where $\mu \in_R Z_p$. Then, $\mathcal{B}$ picks $\nu \in_R Z_p$ and calculates

$$D_{GID_i,k,1} = \pi_i^{\alpha_j+\beta_j r_{GID_i}} g^{r_i\nu}$$
$$= g^{\frac{\alpha_j}{a_j+GID_i+b_j\mu}} H_0(GID_i)^{\frac{\beta_j}{a_j+GID_i+b_j\mu}} H_1(att_k)^{\nu},$$
$$D_{GID_i,2} = GID_i, D_{GID_i,k,3} = \mu,$$
$$D_{GID_i,k,4} = g^{\nu}, D_{GID_i,k,5} = g^{(a_j+b_j\mu)\nu}$$

and sets $D_{GID_i,S_i} = \{\{D_{GID_i,k,1}, D_{GID_i,k,3}, D_{GID_i,k,4}, D_{GID_i,k,5}\}_{att_k\in S_i}, D_{GID_i,2},\}$ which is sent to $\mathcal{A}$.

Key Forgery. $\mathcal{A}$ submits a decryption key DK^* to $\mathcal{B}$. If the forged decryptionkey DK^* passes the **Key Sanity Check** and $Trace(GPR, DK^*, \{APK_j\}) \notin \{null, GID_1, \cdots, GID_n\}$, the following equations holds that

$$\hat{e}(g, D_{GID,k,5}) = \hat{e}(D_{GID,k,4}, g^{a_j} \cdot (g^{b_j})^{D_{GID,k,3}}), \quad (1)$$
$$\hat{e}(D_{GID,k,1}, g^{a_j} g^{D_{GID,k,2}} (g^{b_j})^{D_{GID,k,3}}) =$$
$$\hat{e}(g, g)^{\alpha_j} \cdot \hat{e}(H_0(D_{GID,2}), g^{\beta_j}) \cdot$$
$$\hat{e}(H_1(\rho(i)), D_{GID,k,4}^{D_{GID,2}} D_{GID,i,5}) \quad (2)$$

After querying the random oracles O_{H_0}, O_{H_1}, $\mathcal{B}$ gets the tuple $(GID, r_{GID}, g^{r_{GID}})$ from L_1 and (k, r_k, g^{r_k}) from L_2. Let $D_{GID,k,4} = g^{\theta}$, according to Eq.1, $D_{GID,k,5} = g^{(a_j+b_jD_{GID,k,3})\theta}$ and according to Eq.2, $\mathcal{B}$ can get

$$D_{GID,k,1} = g^{\frac{\alpha_j+\beta_j r_{GID}+r_k\theta(a_j+GID+b_jD_{GID,k,3})}{a_j+GID+b_jD_{GID,k,3}}}$$
$$= g^{\frac{\alpha_j+\beta_j r_{GID}}{a_j+GID+b_jD_{GID,k,3}}} D_{GID,k,4}^{r_k}$$

Thus, $\mathcal{B}$ can obtain $\pi = (\frac{D_{GID,k,1}}{D_{GID,k,4}^{r_k}})^{\frac{1}{\alpha_j+\beta_j r_{GID}}} = g^{\frac{1}{a_j+GID+b_jD_{GID,k,3}}}$ and the pair $(D_{GID,k,3}, \pi)$ which is the signature of $D_{GID,2}$. As $Trace(GPR, DK^*, \{APK_j\}) \notin \{null, GID_1, \cdots, GID_n\}$, $D_{GID,2} \notin \{null, GID_1, \cdots, GID_n\}$. We know that $\mathcal{B}$ can construct a signature on $DK_{GID,2}$ without querying for $\mathcal{C}$ and breaks the BB scheme with the same advantage ε as $\mathcal{A}$.

If l-SDH assumption holds, BB scheme is strong existential forgery secure under adaptive chosen message attack and thus, our scheme is fully traceable. □

3) Data Auditing Security:

Theorem 3. *If no PPT adversary can win the data auditing security game with non-negligible advantage, LUTPDS is secure in data auditing against a corrupted CA on condition that the CDH assumption holds.*

Proof. If any PPT adersary $\mathcal{A}$ can win the integrity game with non-negligible advantage ϵ, then the challenger $\mathcal{B}$ of CDH instance (g_0, g_0^x, g_0^y) can break the CDH assumption with identical advantage.

- *Setup*: The challenger $\mathcal{B}$ initiates the system by generating the global public key $GPR = \{G^*, g_0, PK_c = g_0^x, H_0, H_2, u = g_0^y g^a\}$, where $a \in_R Z_p$. Then $\mathcal{C}$ sents GPR to $\mathcal{A}$.
- *Phase 1*: The adversary $\mathcal{A}$ submits following queries adaptively to $\mathcal{B}$ which creates several empty lists $L_0, L_2, L_{k1}, L_{k2}, L_s$ for response generation.
  - H_0 *query*: $\mathcal{A}$ issues this query with GID for H_0 to $\mathcal{C}$ which responds with (GID, h_0) if it exists in L_0, otherwise, it selects $h_0 \in_R G^*$ and adds (GID, h_0) into L_0.
  - H_2 *query*: $\mathcal{A}$ issues this query with ID_f for H_2 to $\mathcal{C}$ which then returns (ID_f, h_2) back to $\mathcal{A}$ is it exists in L_2, or else, it picks $h_2 \in_R Z_p$ and adds (ID_f, h_2) into L_2.
  - SK_1 *query*: $\mathcal{A}$ issues this query with global identity GID for $SK_{GID,1}$ to $\mathcal{C}$ which responds with the tuple (GID, y) if it exists in L_{k1}, otherwise, it chooses a random value $y \in Z_p$ and adds (GID, y) to L_{k1}.
  - SK_2 *query*: $\mathcal{A}$ issues this query with global identity GID for $SK_{GID,2}$ to $\mathcal{C}$ which in turn sends the $\mathcal{A}$ with a tuple (GID, z) if it exists in L_{k1}, otherwise, it chooses a random value $z \in Z_p$ and adds (GID, z) to L_{k2}.
  - *S query*: $\mathcal{A}$ issues this query with (GID, C_k) for signature generation of the encrypted file block C_k to $\mathcal{C}$ which then programs the above queries to get hash values h_0, h_2 of H_0 and H_2 as well as partial secret key $SK_{GID,2} = z$ and set $PK_{GID} = g_0^y, PK_c = g_0^x$. Finally, $\mathcal{C}$ obtains the public key and user secret key SK_{GID} for signature generation with $Encrypt_{on}$ algorithm.
- *Challenge*: $\mathcal{C}$ is in the role of auditor and sends the challenge request $R_c = \{k, v_k\}_{k \in [n]}$ to $\mathcal{A}$ acting as a prover for data possession proof for file ID_f with its ciphertexts $C_s = \{C_i\}_{f_i \in F}$. For a honest prover, it will generates the correct proof response (A, B) under the challenge request R_c which can pass the auditing as:
$$\hat{e}(B, g_0) = \hat{e}(\prod_{k \in I} h_0^{v_k}, PK_o) \cdot \hat{e}(u^{A \cdot h_2}, PK_c)$$
- *Forge*: $\mathcal{A}$ sends a data posession proof $R_p^{'} = (A^{'}, B^{'}) \neq (A, B)$ under the challenge request R_c. which can also pass the auditing as :
$$\hat{e}(B^{'}, g_0) = \hat{e}(\prod_{k \in I} h_0^{v_k}, PK_o) \cdot \hat{e}(u^{A^{'} \cdot h_2}, PK_c)$$

Thus, we have the following equation:

$$\begin{aligned}\hat{e}(B^{'}/B, g_0) &= \hat{e}(u^{\triangle A \cdot h_2}, PK_c) \\ &= \hat{e}((g_0^y g^a)^{\triangle A \cdot h_2}, PK_c) \\ &= \hat{e}(g^{a \triangle A \cdot z}, g_0) \cdot \hat{e}(g_0^{xy}, g_0)^{\triangle A \cdot h_2}\end{aligned}$$

where $\triangle A = A^{'} - A$,. Therefore, $g_0^{xy} = (B^{'} \cdot B^{-1} \cdot g^{-a \triangle A \cdot z})^{\frac{1}{\triangle A \cdot h_2}} \mod p$ and $\mathcal{C}$ can acquire the solution for CDH problem if $\triangle A \cdot h_2 \neq 0 \mod p$ with probability $1 - \frac{1}{p}$. which is in contradiction to CDH assumption. As a result, if $\mathcal{A}$ can win the game then $\mathcal{C}$ can break the CDH assumption with non-negligible probability.

□

B. Performance

Here, we theoretically analyze the computation and storage complexity of our LUTPDS and evaluate its actual performance with experiments implementation by comparing with several existing state-of-the-art schemes in [21], [27].

1) Theoretical Analysis: In this section, we thoroughly analysis the complexity of our scheme in theory with comparison to the state-of-the-art schemes [21], [27] in detail from the aspects of storage complexity including Public Parameters Size (PP Size), Secret User Key Size (SKey Size), Transformation Key Size (TKey Size), Ciphertext Size (CT Size) and computation complexity containing Key Generation, Encryption, User Decryption, Outsourced Decryption. We first stipulate some denotions:

- E_1, E_2: exponentiation in G^* and G_T^*.
- P: computation of the pairing function $\hat{e}$.
- $l, |I|$: the rows and complexity of access policy.
- $|S|, |U_a|$: the size of user attribute set S and attribute universe U_a.
- $|G^*|, |G_T^*|$: the length of elements in group G^* and G_T^*.
- $|Z_p|$: the length of element in Z_p.
- $|A|, |F|$: the number of authorities and file blocks.

The comparison result is summarized in Table III and Table IV.

Table III shows the computation complexity comparison in the aforementioned schemes. With respect to the key generation cost, our scheme and [27] cost more than [21] as a result of attribute hiding, which incurs a slight extra cost of $|S|P$ to AAs. Our scheme needs more $(2|S| - 3|A|)E_1$ than [27], thus, the growth rate in our scheme is more although the key generation cost may be less when $|S|$ is small. As for encryption cost, the growth rate to the complexity of access policy l is the least in our scheme, thus the encryption cost in our scheme is growing slowest of all and when $|F|$ is small, its encryption cost is also the least. Due to the outsourced decryption, the cost for out decryption in our scheme is more than that of [27] as a result of traceability, while the user decryption cost in our scheme is the least and that of [27] costs more $(|A| - 1)E_2$ than our scheme.

In regard to the storage complexity comparison shown in Table IV, the authority public key cost in our scheme and [21] are both costant and far less than [27] which costs a linear

TABLE III: Computation Complexity Comparison

Scheme	KeyGen	Encryption	User Decryption	Out Decryption
Scheme [21]	$5\|S\|E_1$	$6lE_1 + (2l+1)E_2$	$3\|I\|E_1 + \|I\|E_2 + 3\|I\|P$	$--$
Scheme [27]	$(3\|S\| + 4\|A\|)E_1 + \|S\|P$	$(5l+2)E_1 + (l+1)E_2$	$\|A\|E_2$	$2\|I\|E_2 + (4\|I\| + \|A\|)P$
LUTPDS	$(5\|S\| + \|A\|)E_1 + \|S\|P$	$(4l + 2\|F\|)E_1 + lE_2$	E_2	$4\|I\|E_1 + 3\|I\|E_2 + 3\|I\|P$

TABLE IV: Storage Complexity Comparison

Scheme	PP Size	TKey Size	SKey Size	CT Size
Scheme [21]	$3\|G^*\| + \|G_T^*\|$	$--$	$3\|S\|\|G^*\| + 2\|S\|\|Z_p\|$	$5l\|G^*\| + (l+1)\|G_T^*\|$
Scheme [27]	$(\|U_a\|+1)\|G^*\|$ $+(\|U_a\|+1)\|G_T^*\|$	$(2\|S\| + 2\|A\|)\|G^*\|$	$\|A\|\|Z_p\|$	$(3l+1)\|G^*\| + (3l+1)\|G_T^*\| + \|Z_p\|$
LUTPDS	$3\|G^*\| + \|G_T^*\|$	$(3\|S\|+1)\|G^*\| + 2\|S\|\|Z_p\|$	$2\|Z_p\|$	$(\|F\| + 5l)\|G^*\| + (l+1)\|G_T^*\|$ $+(2l+1)\|Z_p\|$

storage complexity for authority public key with the size of attribute universe. Thus, the flexibility in [27] is less than the others. Moreover, the storage cost for transformation key in our scheme is slightly $|S||Z_p|$ more than that of [27] due to the extra cost for traceability, but the cost for user secret key in our scheme is the least, which saves much storage burden for end devices. With respect to the size of ciphertext, our scheme introduces some extra cost for online/offline encryption and the signatures used in data integrity auditing, which supports higher security level and efficiency in encryption.

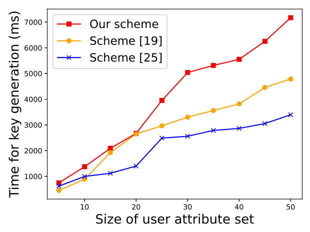
2) Performance Evaluation: To precisely evaluate the performance of our scheme, we implement our scheme and the schemes in [21], [27] and summarize the comparison of their actual computation and storage cost in Fig.4 with extensive experimental simulations.

In our experiments, we implement and develop these schemes using Java Programming Language with the Java Pairing-Based Cryptography library (JPBC) [52] which is the wrapper of the Stanford Pairing-Based Cryptography Library (PBC) and can support pairing, exponential, addition, multiplication and inversion operations in finit field and groups. In our implementations, we use the Type A pairing defined over a 160-bit elliptic curve group over 512-bit finite field, that is, the supersingular elliptic curve $E(F_p) : y^2 = x^3 + x$ with embedding degree 2, where p is a 512-bit Solinas prime. Moreover, we employ two servers equipped with Windows10 system on Intel Core i5 CPU 2.13 GHz and 8.00-GB RAM to act as multiple clouds and two laptops with Intel Core i5 CPU and 6.00-GB RAM to act as the Fog Node and end device. In addition, to assess the storage cost of user secret key and transformation key as well as the time cost in key generation, we set the size of user attribute set in a range of $|S| = \{5, 50\}$ while in the storage cost of ciphertext and time cost for encryption and decryption, the complexity of access policy is set as $l = |I| = \{10, 50\}$. For the evaluation of authority public key size, the size of attribute universe is set as $|U_a|$.

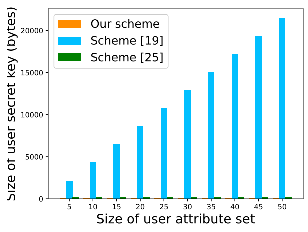
To analyze the time and storage cost in key generation, Fig.4(a) shows the key generation time cost comparison between our LUTPDS and [21], [27]. We notice that due to the introduction of user tracing mechanism and attribute hiding algorithm, the time cost in our scheme is slightly more than that of the other two schemes as it brings only two extra exponentiation operations in decryption key generation and does not pose much burden on AAs. Moreover, Fig.4(b) depicts the user secret key storage cost comparison of the above schemes. As our scheme and [27] employ the outsourced decryption, they greatly reduce the storage cost for user secret key in smart devices of s-Logistics, while that of [21] is linear to the increase of the size of user attribute set $|S|$. Our scheme cost the least storage of all for user secret key as it is not affected by the number of authorities $|A|$. Furthermore, Fig.4(c) shows the comparison of the storage cost for transformation key between our scheme and [27] as the scheme in [21] does not need a transformation key. When the size of user attribute set $|S|$ is small, our scheme cost less storage and when $|S|$ grows it will exceed the cost in [27] which seldomly appears in real-world applications. In addition, our scheme and [21] cost much less storage for authority public key than that of [27] as the latter does not support large attribute universe, which is shown in Fig.4(d).

To assess the time and storage cost in encryption, from Figs.4(e) and 4(f), we can analyze the encryption time cost in the above schemes in diverse setting of the file number $|F|$ to assess its effect in encryption cost. It is obvious that the time cost in the three schemes are linear to the complexity of access policy l in encryption. The encryption time cost in [21] is the most. For the sake of online/offline technique, the growth rate in our scheme is smaller than the others, thus, although the scheme [27] costs less than our scheme when l is small, it exceeds ours when $l > 20$ and $|F|$ is set as a general 5. If $|F|$ is set with a big value, the time cost for encryption in our scheme is more than that of [27], which is not the general case in real-world applications. Similarly, we assess the ciphertext storage cost in Figs.4(g) and 4(h) with the same setting as in encryption time evaluation. We observe that the storage cost for ciphertext in all schemes are proportion to the complexity of access policy l and that of our scheme cost slightly more than other two as it brings about extra storage cost for signatures used in data integrity auditing.

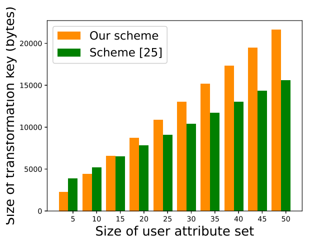
To evaluate the time cost in outsourced and user decryption, in Figs.4(i) and 4(j) which depicts the outsourced decryption time cost in fog side under different settings of authority number $|A| = [5, 10]$, as the scheme [21] fails to support outsourced decryption, we make a comparison between our scheme and [27]. We observe that the time cost in both



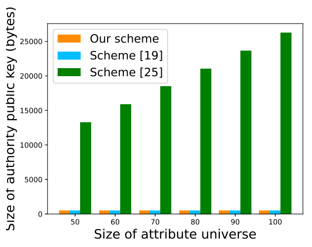
(a) Key generation time cost



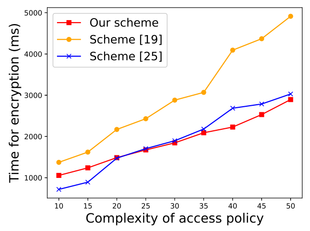
(b) User key storage cost



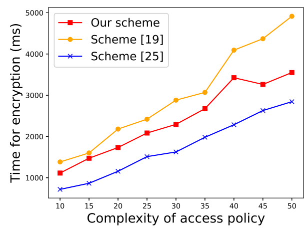
(c) Transformation key storage cost



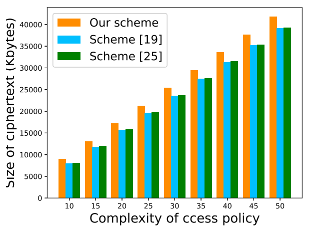
(d) Global public key storage cost



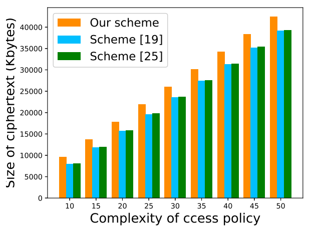
(e) Encryption time when $|F| = 5$



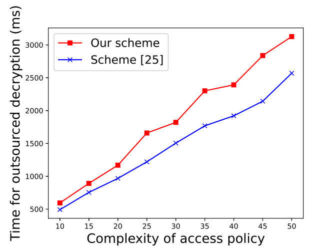
(f) Encryption time when $|F| = 10$



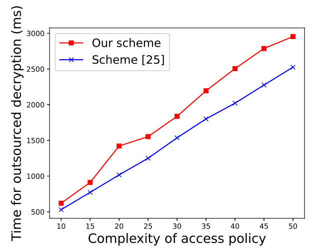
(g) Ciphertext size when $|F| = 5$



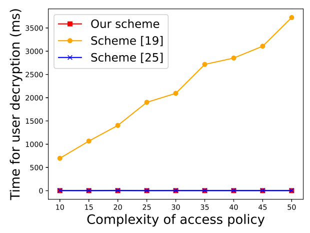
(h) Ciphertext size when $|F| = 10$



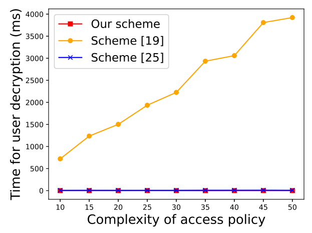
(i) OutDec time when $|A| = 5$



(j) OutDec time when $|A| = 10$



(k) Decryption time when $|A| = 5$



(l) Decryption time when $|A| = 10$

Fig. 4: Comparison of the actual efficiency.

schemes are linear to the increase of the complexity of access policy $|I|$ in outsourced decryption. Our scheme costs slightly more time than [27] which is also affected by the number of authorities $|A|$. In terms of the end devices, , Figs.4(k) and 4(l) shows the comparison of the time cost for user decryption. The time cost in [21] is proportional to the growing of $|I|$ while that of the other two are nearly constant and far less. Specifically, due to the outsourced decryption, our scheme and [27] achieves smaller time cost for decryption, in which the latter is also affected by $|A|$, thus, our LUTPDS costs the least time in user decryption.

To summarize, the results of our experimental simulations indicate that our LUTPDS is flexible and versatile. It is also efficient in terms of the encryption cost, user decryption cost, user key cost with far smaller and constant cost for public. Besides, it is the only scheme that supports fine-grained data access control and data integrity auditing as well as scalability, user privacy-preserving, key escrow and abuse resistance and large universe simultaneously. Therefore, LUTPDS scheme is more suitable for data sharing in Fog-Based s-Logistics systems with resource-constraint smart devices.

VII. Conclusion

In this paper, we propose a scalable Large Universe and Traceable Privacy-preserving Data Sharing (LUTPDS) scheme for Fog-based s-Logistics over multiple clouds that can support data confidentiality, fine-grained access control and data integrity auditing at the same time by devising a novel CP-ABE scheme. It can achieve scalability, key-escrow and abuse resistance, large attribute universe and data integrity compared with existing schemes. Due to the online/offline and verifiable outsourced decryption techniques, LUTPDS can eliminate the high computation cost in CP-ABE and resist untrusted Fog Nodes in outsourced decryption with an authenticated code. Benefited from certificateless public data integrity auditing mechanism, the data integrity can be guaranteed with the assistance of a trusted auditor in the setting of untrusted cloud servers. Moreover, the security analysis and thorough performance evaluation show that our proposal is secure and practicable for Fog-based s-Logistics systems with resource-restraint IoT devices compared with existing schemes.

In our future work, we would dedicate into the efficient attribute and user revocation in data sharing scheme for Fog-Based IoT environment.

Acknowledgment

This work is supported by Natural Science Foundation of Inner Mongolia, China, 2020 (No. 2020LH06007), Innovation Fund of Inner Mongolia University of Science and Technology, China (No.2019QDL-B51) and Inner Mongolia Major science and technology projects: artificial intelligence application technology and product research, development Application Research and demonstration in modern pastures (2019ZD025).

REFERENCES

[1] X. Tang, "Research on smart logistics model based on internet of things technology," *IEEE Access*, vol. 8, pp. 151 150–151 159, 2020.

[2] T. Gregor, M. Krajcovic, and D. Wiecek, "Smart connected logistics," *Procedia Engineering*, vol. 192, pp. 265–270, 2017.

[3] M. Humayun, N. Jhanjhi, B. Hamid, and G. Ahmed, "Emerging smart logistics and transportation using iot and blockchain," *IEEE Internet of Things Magazine*, vol. 3, no. 2, pp. 58–62, 2020.

[4] Y. Ding, M. Jin, S. Li, and D. Feng, "Smart logistics based on the internet of things technology: an overview," *International Journal of Logistics Research and Applications*, pp. 1–23, 2020.

[5] J. Zhang, T. Li, M. S. Obaidat, C. Lin, and J. Ma, "Enabling efficient data sharing with auditable user revocation for iov systems," *IEEE Systems Journal*, 2021.

[6] Y. Guan, J. Shao, G. Wei, and M. Xie, "Data security and privacy in fog computing," *IEEE Network*, vol. 32, no. 5, pp. 106–111, 2018.

[7] K.-P. Yu, L. Tan, M. Aloqaily, H. Yang, and Y. Jararweh, "Blockchain-enhanced data sharing with traceable and direct revocation in iiot," *IEEE transactions on industrial informatics*, 2021.

[8] T. Shi, H. Ma, G. Chen, and S. Hartmann, "Location-aware and budget-constrained service deployment for composite applications in multi-cloud environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 8, pp. 1954–1969, 2020.

[9] C. Feng, K. Yu, A. K. Bashir, Y. D. Al-Otaibi, Y. Lu, S. Chen, and D. Zhang, "Efficient and secure data sharing for 5g flying drones: a blockchain-enabled approach," *IEEE Network*, vol. 35, no. 1, pp. 130–137, 2021.

[10] K. Fan, Q. Pan, K. Zhang, Y. Bai, S. Sun, H. Li, and Y. Yang, "A secure and verifiable data sharing scheme based on blockchain in vehicular social networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5826–5835, 2020.

[11] N. Lu, Y. Zhang, W. Shi, S. Kumari, and K.-K. R. Choo, "A secure and scalable data integrity auditing scheme based on hyperledger fabric," *Computers & Security*, vol. 92, p. 101741, 2020.

[12] J. Li, Y. Zhang, X. Chen, and Y. Xiang, "Secure attribute-based data sharing for resource-limited users in cloud computing," *Computers & Security*, vol. 72, pp. 1–12, 2018.

[13] J. Zhang, J. Ma, T. Li, and Q. Jiang, "Efficient hierarchical and time-sensitive data sharing with user revocation in mobile crowdsensing," *Security and Communication Networks*, vol. 2021, 2021.

[14] J. Zhang, J. Ma, Y. Yang, X. Liu, and N. N. Xiong, "Revocable and privacy-preserving decentralized data sharing framework for fog-assisted internet of things," *IEEE Internet of Things Journal*, 2021.

[15] J. Zhang, Y. Yang, X. Liu, and J. Ma, "An efficient blockchain-based hierarchical data sharing for healthcare internet of things," *IEEE Transactions on Industrial Informatics*, 2022.

[16] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *2007 IEEE symposium on security and privacy (SP'07)*. IEEE, 2007, pp. 321–334.

[17] J. Zhang, J. Ma, Z. Ma, N. Lu, and D. Wei, "Efficient hierarchical data access control for resource-limited users in cloud-based e-health," in *2019 International Conference on Networking and Network Applications (NaNA)*, 2019.

[18] S. Gao, G. Piao, J. Zhu, X. Ma, and J. Ma, "Trustaccess: A trustworthy secure ciphertext-policy and attribute hiding access control scheme based on blockchain," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5784–5798, 2020.

[19] P. Zeng, Z. Zhang, R. Lu, and K.-K. R. Choo, "Efficient policy-hiding and large universe attribute-based encryption with public traceability for internet of medical things," *IEEE Internet of Things Journal*, 2021.

[20] J. Zhao and P. Zeng, "Efficient and large universe ciphertext-policy attribute-based encryption with black-box traceability for ehealth," in *The International Conference on Cyber Security Intelligence and Analytics*. Springer, 2020, pp. 480–485.

[21] K. Zhang, H. Li, J. Ma, and X. Liu, "Efficient large-universe multi-authority ciphertext-policy attribute-based encryption with white-box traceability," *Science in China Series F: Information Sciences*, vol. 61, no. 3, p. 032102, 2018.

[22] S. Banerjee, S. Roy, V. Odelu, A. K. Das, S. Chattopadhyay, J. J. Rodrigues, and Y. Park, "Multi-authority cp-abe-based user access control scheme with constant-size key and ciphertext for iot deployment," *Journal of Information Security and Applications*, vol. 53, p. 102503, 2020.

[23] Y. Zhang, D. Zheng, and R. H. Deng, "Security and privacy in smart health: Efficient policy-hiding attribute-based access control," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2130–2145, 2018.

[24] H. Cui, R. H. Deng, J. Lai, X. Yi, and S. Nepal, "An efficient and expressive ciphertext-policy attribute-based encryption scheme with partially hidden access structures, revisited," *Computer Networks*, vol. 133, pp. 157–165, 2018.

[25] D. Han, N. Pan, and K.-C. Li, "A traceable and revocable ciphertext-policy attribute-based encryption scheme based on privacy protection," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[26] S. Wang, H. Wang, J. Li, H. Wang, J. Chaudhry, M. Alazab, and H. Song, "A fast cp-abe system for cyber-physical security and privacy in mobile healthcare network," *IEEE Transactions on Industry Applications*, vol. 56, no. 4, pp. 4467–4477, 2020.

[27] K. Fan, H. Xu, L. Gao, H. Li, and Y. Yang, "Efficient and privacy preserving access control scheme for fog-enabled iot," *Future Generation Computer Systems*, vol. 99, pp. 134–142, 2019.

[28] M. Cui, D. Han, J. Wang, K.-C. Li, and C.-C. Chan, "Arfv: An efficient shared data auditing scheme supporting revocation for fog-assisted vehicular ad-hoc networks," *IEEE Transactions on Vehicular Technology*, 2020.

[29] Y. Xu, C. Zhang, G. Wang, Z. Qin, and Q. Zeng, "A blockchain-enabled deduplicatable data auditing mechanism for network storage services," *IEEE Transactions on Emerging Topics in Computing*, 2020.

[30] K. Fan, Z. Bao, M. Liu, A. V. Vasilakos, and W. Shi, "Dredas: Decentralized, reliable and efficient remote outsourced data auditing scheme with blockchain smart contract for industrial iot," *Future Generation Computer Systems*, vol. 110, pp. 665–674, 2020.

[31] N. Garg, S. Bawa, and N. Kumar, "An efficient data integrity auditing protocol for cloud computing," *Future Generation Computer Systems*, vol. 109, pp. 306–316, 2020.

[32] C. Lee, Y. Lv, K. Ng, W. Ho, and K. Choy, "Design and application of internet of things-based warehouse management system for smart logistics," *International Journal of Production Research*, vol. 56, no. 8, pp. 2753–2768, 2018.

[33] J. Wen, L. He, and F. Zhu, "Swarm robotics control and communications: Imminent challenges for next generation smart logistics," *IEEE Communications Magazine*, vol. 56, no. 7, pp. 102–107, 2018.

[34] Y. Su and Q.-m. Fan, "The green vehicle routing problem from a smart logistics perspective," *IEEE Access*, vol. 8, pp. 839–846, 2019.

[35] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," pp. 568–588, 2011.

[36] K. Yang and X. Jia, "Expressive, efficient, and revocable data access control for multi-authority cloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1735–1744, 2014.

[37] S. J. De and S. Ruj, "Efficient decentralized attribute based access control for mobile clouds," *IEEE Transactions on Cloud Computing*, vol. 8, no. 1, pp. 124–137, 2020.

[38] Y. Rouselakis and B. Waters, "Practical constructions and new proof methods for large universe attribute-based encryption," pp. 463–474, 2013.

[39] Y. Rouselakis and B. Waters, "Efficient statically-secure large-universe multi-authority attribute-based encryption," in *International Conference on Financial Cryptography and Data Security*. Springer, 2015, pp. 315–332.

[40] S. Hohenberger and B. Waters, "Online/offline attribute-based encryption," pp. 293–310, 2014.

[41] P. Datta, R. Dutta, and S. Mukhopadhyay, "Fully secure online/offline predicate and attribute-based encryption," pp. 331–345, 2015.

[42] H. Ma, R. Zhang, Z. Wan, Y. Lu, and S. Lin, "Verifiable and exculpable outsourced attribute-based encryption for access control in cloud computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 6, pp. 679–692, 2017.

[43] Q. Xu, C. Tan, W. Zhu, Y. Xiao, Z. Fan, and F. Cheng, "Decentralized attribute-based conjunctive keyword search scheme with online/offline encryption and outsource decryption for cloud computing," *Future Generation Computer Systems*, vol. 97, pp. 306–326, 2019.

[44] M. Green, S. Hohenberger, B. Waters *et al.*, "Outsourcing the decryption of abe ciphertexts." in *USENIX Security Symposium*, vol. 2011, no. 3, 2011.

[45] J. Lai, R. H. Deng, C. Guan, and J. Weng, "Attribute-based encryption with verifiable outsourced decryption," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, pp. 1343–1354, 2013.

[46] H. Wang, D. He, and J. Han, "Vod-adac: Anonymous distributed fine-grained access control protocol with verifiable outsourced decryption in public cloud," *IEEE transactions on services computing*, vol. 13, no. 3, pp. 572–583, 2017.

[47] H. Cui, R. H. Deng, G. Wu, and J. Lai, "An efficient and expressive ciphertext-policy attribute-based encryption scheme with partially hidden access structures," vol. 10005, pp. 19–38, 2016.

[48] H. Cui, R. H. Deng, J. Lai, X. Yi, and S. Nepal, “An efficient and expressive ciphertext-policy attribute-based encryption scheme with partially hidden access structures, revisited,” *Computer Networks*, vol. 133, pp. 157–165, 2018.

[49] Z. Zhou, D. Huang, and Z. Wang, “Efficient privacy-preserving ciphertext-policy attribute based-encryption and broadcast encryption,” *IEEE Transactions on Computers*, vol. 64, no. 1, pp. 126–138, 2015.

[50] H. Zhong, W. Zhu, Y. Xu, and J. Cui, “Multi-authority attribute-based encryption access control scheme with policy hidden for cloud storage,” *Soft Computing*, vol. 22, no. 1, pp. 243–251, 2018.

[51] D. Boneh and X. Boyen, “Short signatures without random oracles and the sdh assumption in bilinear groups,” *Journal of cryptology*, vol. 21, no. 2, pp. 149–177, 2008.

[52] A. De Caro and V. Iovino, “jpbc: Java pairing based cryptography,” in *Proceedings of the 16th IEEE Symposium on Computers and Communications, ISCC 2011*, Kerkyra, Corfu, Greece, June 28 - July 1, 2011, pp. 850–855.

Yanbo Yang received the B.S. and Ph.D degrees in School of Telecommunications Engineering from Xidian University, China, in 2006 and 2014, respectively. Now He is lecture in Information Engineering School of Inner Mongolia University of science and technology, China. He is currently pursuing the fusion application of UAV/UGV, Blockchain technology, big data and machine learning in Industrial Internet area. Yanbo Yang is the corresponding author.

Jiawei Zhang received the B.S. and M.S. degrees in School of Telecommunications Engineering from Xidian University, China, in 2007 and 2010, respectively, and the Ph.D. degree in School of Computer Science and Technology in Xidian University, China, in 2021. He is currently a Lecturer with the School of Cyber Engineering, Xidian University. His current research interests include access control, data security, cloud and edge security, blockchain, cryptography and network security.

Ximeng Liu (S’ 13-M’ 16-SM’ 21) received the B.Sc. degree in electronic engineering from Xidian University, Xi’an, China, in 2010 and the Ph.D. degree in Cryptography from Xidian University, China, in 2015. Now he is the full professor in the College of Mathematics and Computer Science, Fuzhou University. He was a research fellow at the School of Information System, Singapore Management University, Singapore. He has published more than 250 papers on the topics of cloud security and big data security including papers in IEEE TC, IEEE TIFS, IEEE TDSC, IEEE TPDS, IEEE TKDE, IEEE IoT Journal, and so on. He awards “Minjiang Scholars” Distinguished Professor, “Qishan Scholars” in Fuzhou University, and ACM SIGSAC China Rising Star Award (2018). His research interests include cloud security, applied cryptography and big data security.

Jianfeng Ma received the B.S. degree in computer science from Shaanxi Normal University in 1982, and M. S. degree in computer science from Xidian University in 1992, and the Ph. D. degree in computer science from Xidian University in 1995. Currently he is the directer of Department of Cyber engineering and a professor in School of Cyber Engineering, Xidian University. He has published over 150 journal and conference papers. His research interests include information security, cryptography, and network security.