sensors

MDPI

Article

A Secure and Lightweight Fine-Grained Data Sharing Scheme for Mobile Cloud Computing

Haifeng Li [1], Caihui Lan [2,*], Xingbing Fu [3,4,5], Caifen Wang [6], Fagen Li [7] and He Guo [1]

1 School of Software, Dalian University of Technology, Dalian 116024, China; lihaifeng8848@mail.dlut.edu.cn (H.L.); guohe@dlut.edu.cn (H.G.)
2 School of Electronic and Information Engineering, Lanzhou City University, Lanzhou 730070, China
3 Guangxi Key Laboratory of Cryptography and Information Security, Guilin 541004, China; uestcfuxb@126.com
4 School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China
5 Guangdong Provincial Key Laboratory of Information Security Technology, Guangzhou 510275, China
6 College of Big Data and Internet, Shenzhen Technology University, Shenzhen 518118, China; wangcaifen@sztu.edu.cn
7 School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China; fagenli@uestc.edu.cn
* Correspondence: lanzhourm@163.com

Received: 9 July 2020; Accepted: 18 August 2020; Published: 21 August 2020

Abstract: With the explosion of various mobile devices and the tremendous advancement in cloud computing technology, mobile devices have been seamlessly integrated with the premium powerful cloud computing known as an innovation paradigm named Mobile Cloud Computing (MCC) to facilitate the mobile users in storing, computing and sharing their data with others. Meanwhile, Attribute Based Encryption (ABE) has been envisioned as one of the most promising cryptographic primitives for providing secure and flexible fine-grained "one to many" access control, particularly in large scale distributed system with unknown participators. However, most existing ABE schemes are not suitable for MCC because they involve expensive pairing operations which pose a formidable challenge for resource-constrained mobile devices, thus greatly delaying the widespread popularity of MCC. To this end, in this paper, we propose a secure and lightweight fine-grained data sharing scheme (SLFG-DSS) for a mobile cloud computing scenario to outsource the majority of time-consuming operations from the resource-constrained mobile devices to the resource-rich cloud servers. Different from the current schemes, our novel scheme can enjoy the following promising merits simultaneously: (1) Supporting verifiable outsourced decryption, i.e., the mobile user can ensure the validity of the transformed ciphertext returned from the cloud server; (2) resisting decryption key exposure, i.e., our proposed scheme can outsource decryption for intensive computing tasks during the decryption phase without revealing the user's data or decryption key; (3) achieving a CCA security level; thus, our novel scheme can be applied to the scenarios with higher security level requirement. The concrete security proof and performance analysis illustrate that our novel scheme is proven secure and suitable for the mobile cloud computing environment.

Keywords: attribute-based encryption (ABE); mobile cloud computing (MCC); verifiability; outsourced decryption; CCA-secure

1. Introduction

With the tremendous development of distributed computing technology and virtualization technology, cloud computing has gained popularity in various fields such as scientific research, economic finance, medical treatment, education and entertainment. In order to relieve the local storage

burden and enjoy the great benefits provided by cloud computing, such as powerful computing power and conveniently ubiquitous access, more and more individuals and organizations are willing to outsource their local data to the remote cloud server for storing, maintaining, manipulating and sharing with others [1]. Meanwhile, as the modern electronic technique and wireless communication technology have gained impressive progress in the past years, mobile devices (e.g., smart mobile, tablet, smart sensor, PDAs) as convenient handheld communication tools have become increasingly popular and more promising than ever before due to their mobility, convenience and availability. With the popularization of various mobile applications, all kinds of mobile devices have been observed in various domains, such as mobile commerce [2], mobile learning [3], mobile health monitoring [4], and so on. It is well recognized that, although mobile devices can provide conveniently handheld ubiquitous access anytime and anywhere, they are also restrained by their relatively weaker computing power, lower battery power and smaller storage space. The above-mentioned weaknesses of mobile devices greatly hinder the realistic application with respect to the applications of intensive computational tasks and massive storage demands. Nevertheless, cloud computing paradigm can offer an unimaginable infinite storage space and tremendous computing resource. Thus, a natural idea is combining the merits of mobile devices and the advantage of cloud computing to create a new paradigm, whereby the cloud computing is responsible for performing the heavy computing-intensive tasks and storing massive data of mobile user's as well as preserving the all the merits of mobile devices. Fortunately, in recent years, a seamless convergence has been observed between cloud computing and mobile device in various aspects and known as Mobile Cloud Computing (MCC). While the single widely accepted clear definition of MCC has still remained disputed and different scholar researches give different definitions of MCC from different prospective [5–7], based on the results of the research [8–10], in this work, we mainly refer to MCC as the well accepted fact that MCC can facilitate mobile users with the complicated data computing and unlimited storage services in the cloud server, thus, the mobile devices do not necessarily to equip with powerful configuration (such as, high CPU speed and large memory capacity) without sacrificing any desirable properties of the mobile devices. In short, mobile devices can seamlessly integrate with the premium powerful cloud computing.

Meanwhile, attribute based encryption has been envisioned as one of the most promising cryptographic primitives for providing secure and flexible fine-grained "one to many" access control, particularly in large scale distributed system with unknown participators. However, most of existing ABE schemes are not suitable for MCC because they involve expensive pairing operations which pose a formidable challenge for the resource-constrained mobile devices, thus, greatly delaying the widespread popularity of MCC.

In addition, due to size-limitation, it is impractical to solely depend on improving the hardware technique level to design top premium mobile devices with unlimited storage and computational power as same as the personal computer (PC). Therefore, it is essential to devise external devices or resources with partial or large support for the resource-constrained mobile devices to perform computationally intensive task. Fortunately, the emergence of mobile cloud computing can meet the demand of computation and storage resource for mobile device. The MCC is the integration of cloud computing and mobile computing, which can offer rich storage and computational resources over wireless networks to mobile users. In mobile cloud computing architecture, the heavy computation task and the massive data which have been previously done inside the mobile devices are being outsourced to the remote cloud server, accordingly, the physical control over the data of mobile user has been deprived. Thus, the mobile user may worry about whether their data is secure and privacy of outsourced sensitive data can be well preserved. User's concerns about the security and privacy of their outsourced data are the primary barriers that hinder mobile cloud computing from widespread adoption by enormous potential mobile users to a large extent.

1.1. Related Work

During the past decades, the cryptographic researchers have developed many new cryptographic primitives. Among them, ABE is envisioned as one of the most attractive cryptographic primitive since it can provide secure and flexible fine-grained "one to many" access control, especially in large scale distributed system with unknown participators. The ABE is derived from the identity-based encryption mechanism [11], in which the identity information can be uniquely identified as the public key for encryption. In 2005, Sahai and Waters [12] first designed an identity encryption scheme based on biological characteristic information, called fuzzy identity-based encryption scheme (Fuzzy-IBE). The Fuzzy-IBE scheme can be regarded as the "prototype" of ABE. In 2006, Goyal, Sahai and Waters et al. [13] introduced the concept of attributes, and expanded fuzzy identity-based encryption to ABE. In their scheme, the user identity information is generalized to attributes relevant to user identity, and the private key and ciphertext are associated with a set of attributes, and the user will be able to decrypt the ciphertext if and only if the ciphertext attributes and the secret key attributes match each other to certain threshold. According to how the secret key and ciphertext are embedded with the access policy, the ABE-based schemes can be mainly divided into two types: Key-Policy Attribute-Based Encryption (KP-ABE) scheme [13] and Ciphertext-Policy Attribute-Based Encryption (CP-ABE) scheme [14]. For KP-ABE scheme, the user's private keys are integrated with an access policy and the encrypted data are associated with a set of attributes. However, for CP-ABE scheme, the roles of an attribute set and an access structure switch to the opposite side. That is, in CP-ABE scheme, user's secret key is embedded with a set of attributes and encrypted data are associated with an access structure. By taking the advantages of ABE, many scholar researchers and industrial engineers have devised a number of novel schemes for securely sharing data in distributed systems such as cloud computing [15–24].

Despite the ABE primitive is very powerful and promising, it still suffers from an efficiency weakness due to the fact that the traditional ABE based schemes involve many expensive pairing operations. It would become a significant challenge for mobile users since their local resources are limited, especially in battery life, storage capacity and computing power. In order to handle the efficiency problem of the local ABE schemes, in USENIX 2011, Green et al. [25] proposed an ABE based scheme with outsourced decryption by introducing an external cloud server (i.e., a proxy server) and using a transformation key from the users to significantly simplify the ciphertext in the cloud server side. In their new paradigm, the cloud server performed the majority expensive pairing operations while only left a small amount of lightweight computation task for local users, thus, greatly relieving the computation cost of ABE in the user side. However, the creative work of Green et al. [25] can accelerate the decryption procedure for local users by using untrusted servers, which may bring about a new security loophole that is how to guarantee the correctness of the transformation ciphertext since the aided decryption server is not fully trust. The semi-trust cloud server maybe deliberately return a false transformed ciphertext to the local user for saving its resource to gain more profits or other reasons, which will result in the user obtains the incorrect decryption results. In order to ensure the correctness for the transformed ciphertext, Lai et al. [26] developed an outsourcing attribute-based encryption with checkability to check the correctness of the transformed ciphertext returned by the untrusted cloud server, which can not only offload the heavy intensive computing tasks for local users, but also guarantee the correctness of the outsourcing decryption. It is no doubt that checkability is tremendous progress for outsourcing ABE schemes. Subsequently, many ABE based outsourcing schemes with various security properties have been proposed so far [27–37].

1.2. Motivation and Contribution

In order to attract the considerable potential mobile users not hesitate to adopt the mobile cloud computing any longer, the cloud service provider should tackle these security and privacy issues to provide a completely secure and ease environment. Apparently, to eliminate the user's concerns on their outsourced data, a natural method is to encrypt the sensitive data before outsourcing them to

the cloud servers; however, the encrypted data will bring about new issues. For instance, retrieval or sharing the encrypted data is greatly difficult. Traditional access control techniques are almost for plaintext, while in the mobile cloud computing environment, the privacy data and sensitive data of mobile user are stored in the cloud server in the form of ciphertext. Therefore, in this circumstance, the distributed access control technique over ciphertext plays a key role. In general, there are two non-interactive access control techniques adopting in the cloud computing. One is Fully Homomorphic Encryption (FHE) [38–40]. The proposals [41–43] adopt the fully homomorphic encryption technique, which can fully support the addition, and multiplication homomorphic operations and can be directly manipulated over the encrypted data without revealing any sensitive information. It seems an ideal access control technique especially suitable for the cloud computing setting. However, this technique is merely a theoretical approach and is currently inefficient and impractical in real-world application because this solution is required a huge expensive computational overheads. The other distributed access control technique over ciphertext is based on ABE. ABE mechanism is a secure and flexible fine-grained access control, especially applicable for large scale distributed system with unknown participators. While CP-ABE scheme can provide fine-grained access control and is applicable for cloud computing, adopting a CP-ABE scheme directly into a mobile cloud computing that may yield some open issues. One of the most severe drawbacks of the current ABE schemes is that the inherent low efficiency problem, which becomes more severely in the storage limited and computation resource-constrained mobile devices. For example, most current ABE schemes mainly adopt the bilinear maps which will bring about huge expensive pairing operations of ABE schemes. Moreover, the high computation costs during decryption procedure grows linearly along with the number of attributes involved in the access policy. It is a serious challenge for resource-limited mobile users to perform these time-consuming pairing operations; and thus, creating a bottleneck for efficiency of mobile cloud computing. This will greatly impede the widespread adoption of mobile cloud computing. As a response to this problem, many cryptographic scholars have constructed multiple attribute-based encryption with outsourced decryption schemes to reduce the heavy computation costs in mobile devices, such as [25,26,29,44]. While these outsourced attribute-based encryption works have made great process to improve the efficiency of mobile cloud computing, the bottleneck of the efficiency of mobile cloud computing is not fully addressed. For instance, scheme [25] only supports outsourcing the expensive pairing operations to the cloud server, but it does not consider whether the cloud sever returns the corrected transformed ciphertext. It is essential to ensure the validity of the transformed ciphertext from the cloud server because it is untrusted entity and it may cheat the user with forged ciphertext for some evil purpose or just for economic benefits. Therefore, it is necessary to check the validity of the returned results from the cloud server. Later, some scholars investigated the checkability on the returned outsourcing computation results and designed verifiable outsourcing decryption schemes, such as [26]. However, these schemes only support the chosen-plaintext attack(CPA) security level which will limits their application to some extent, and cannot be used in environment for the higher demand for security. As the mobile cloud computing gain an increasing popularity, it is of most urgent to address this realistic problem for improving the performance of the mobile cloud computing. In this work, by adopting the transformation key technique [25], we propose a secure and lightweight fine-grained data sharing scheme for mobile cloud computing scenario, which can provide verifiable outsourcing decryption for intensive computing task during decryption phase to the cloud server without revealing the user's data or decryption key. Moreover, our proposal can achieve security against the chosen-ciphertext attacks (CCA) and thus can be used to the circumstance with higher security level requirements, such as medical data sharing. For instance, a doctor can access and diagnose the Personal Health Record (PHR) of the patients with a mobile device (such as mobile phone, tablet) conveniently by outsourcing the heavy computation operations to the MCC.

1.3. Paper Organization

The remainder of the paper is organized as follows. In Section 2, we discuss some relevant preliminaries used in our paper, such as bilinear pairing, complexity assumption, access tree. In Section 3, we present the problem statement, including the system model, the definition of our novel scheme and the security model. In Section 4, we present the detailed construction of our new scheme. In Section 5, we give the security analysis under the random oracle model. In Section 6, we conduct the concrete performance evaluation and compare the efficiency with other state-of-the-art schemes in terms of functionality, theoretical analysis and experimental simulation. Finally, we draw the conclusion of the whole paper in Section 7.

2. Preliminaries

2.1. Bilinear Pairing

Definition 1 (Bilinear Pairing)**.** *Let G_1 and G_T be two multiplicative cyclic groups with the equal prime order p and assume g is a generator of G_1, the map $e : G_1 \times G_1 \rightarrow G_T$ is defined as a bilinear map if and only if the following three properties are hold.*

(1) Bilinearity. For all elements $g_1, g_2 \in G_1$,$e({g_1}^a, {g_2}^b) = e(g_1, g_2)^{ab}$, where $a, b \in Z_p$ are two random numbers and Z_p is a finite field.
(2) Non-degeneracy. There exists elements $g_1, g_2 \in G_1$ such that $e(g_1, g_2) \neq 1_{G_T}$, where 1 is the identity element of G_T.
(3) Computability. For all elements $g_1, g_2 \in G_1$, there exists an efficient algorithm to compute $e(g_1, g_2)$.

2.2. Complex Assumption

Definition 2 (DBDH Problem)**.** *The Decision Bilinear Diffie–Hellman (DBDH) problem is defined as follows: Given G_1 and G_T be two multiplicative cyclic groups with the equal prime order p, g is a generator of G_1, $e : G_1 \times G_1 \rightarrow G_T$ is a bilinear map, Z_p is a finite field, consider the two following probability distributions: $\mathcal{D}_1 = \mathcal{D}(g, g^a, g^b, g^c, e(g, g)^{abc})$, where a,b,c are randomly chosen in Z_p; and $\mathcal{D}_2 = \mathcal{D}(g, g^a, g^b, g^c, R)$, where a,b,c are randomly chosen in Z_p, and R is randomly chosen in G_T, there is no Probabilistic Polynomial Time(PPT) algorithm can distinguish the two probability distributions: $\mathcal{D}_1$ and $\mathcal{D}_2$ with non-negligible probability so far.*

More formally, the advantage of a distinguisher against the DBDH assumption is defined to be :

$$Adv(\mathcal{D}) = |P_{a,b,c \in_R Z_p, R \in_R G_T}[1 \leftarrow \mathcal{D}_2] - P_{a,b,c \in_R Z_p}[1 \leftarrow \mathcal{D}_1]|.$$

2.3. Access Tree

An access tree is used to describe an access structure. To facilitate the description, we define some notations as follows.

T: This represents an access tree representing the access structure.

x: This represents a node in the access tree T, which can be categorized into two types: Leaf node and non-leaf node (interior node). Each non-leaf interior node is represented a threshold gate, such as "AND" or "OR" threshold gate while each leaf node is associated with an attribute.

num_x: This represents the number of children of the node x.

k_x: This represents the threshold value of node x, where $0 \leq k_x \leq num_x$. If $k_x = 1$ and x is an interior node, it means that the threshold is an "OR" gate. If $k_x = num_x$ and x is an interior node, it means that the threshold is an "AND" gate. In particular, the threshold value of each leaf node x is defined as $k_x = 1$.

$parent(x)$: The function $parent(x)$ is used to return the parent of the node x in the access tree.

$index(x)$: The function $index(x)$ is used to return a unique number associated with the node x, where the number is uniquely assigned to x in a certain manner.

$att(x)$: The function $att(x)$ is used to return an attribute associated with the leaf node x in the access tree.

T_x: This represents the sub-tree for T rooted at the node x in the access tree.

If an attribute set S matches the sub-access-tree T_x, it represents as $T_x(S) = 1$. $T_x(S)$ outputs 1 if and only if the following conditions are satisfied:

(1) If x is a leaf node, $T_x(S) = 1$ if and only if $att(x) \in S$.

(2) If x is an interior node, each child $T_z(S)$ of node x is individually computed in a recursive way. $T_x(S) = 1$ if and only if at least k_x children return 1.

3. Problem Statement

In this section, we will discuss the system model and definition of our efficient and lightweight fine-grained data sharing scheme for mobile cloud computing.

3.1. System Model

As illustrated in Figure 1, the system framework and interaction among its elements of outsourced ABE scheme for the mobile cloud computing are presented, which consists of five types of entities: Namely, Key Generation Center (KGC), Cloud Service Provider (CSP), Mobile Cloud Computing (MCC), Data Owners (DO) and Data Users (DU).

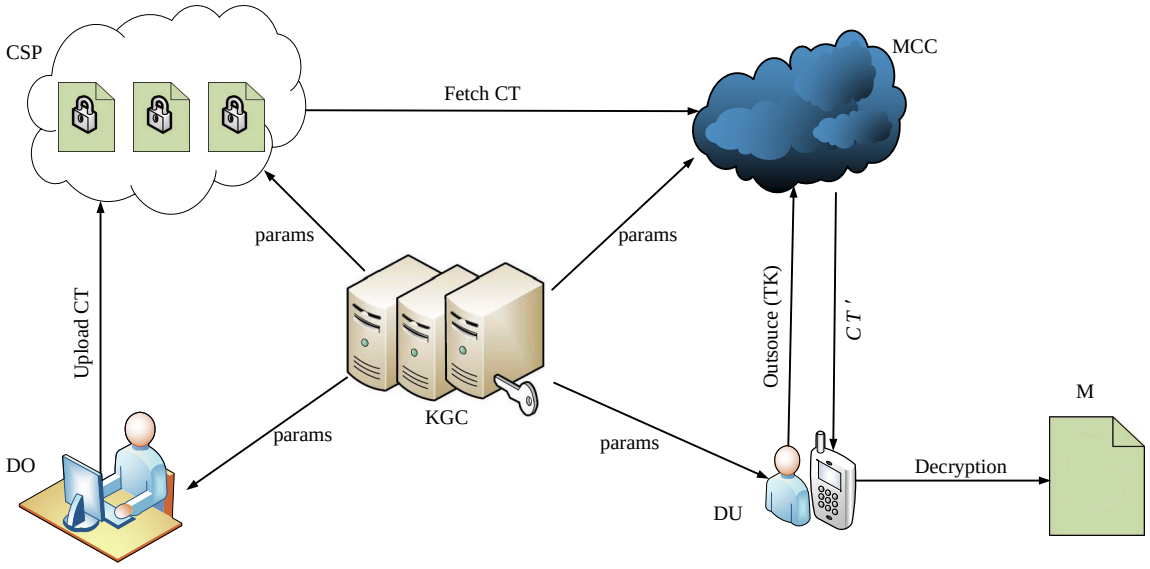

Figure 1. System model of our scheme.

Briefly speaking, the KGC is responsible for generating and distributing the private and public key pairs for other entities in the system. The CSP is a semi-trusted entity which takes charge of storing the data for DO. DO produces message and encrypts it into ciphertext CT, then uploads it to the CSP for storage or sharing it with others. DO also determines who is permitted to access and decrypt the ciphertext CT. The MCC can seamlessly integrate with the mobile devices and facilitate the mobile devices to process data by virtue of its seemingly infinite storage and powerful computing ability. The DU in this paper mainly refers to the users equipped with the resource-constrained mobile devices, but he/she can efficiently process data with the help of MCC.

3.2. Definition of Our Scheme

In this subsection, we provide the detailed definition of our outsourced ABE scheme for the mobile cloud computing, which is composed of the following seven algorithms.

(1) Setup(k,U). This algorithm is performed by KGC to initialize the system. Taking the security parameter k and attribute universe U as input, it outputs the public parameters *params*, and the master secrete key mk.
(2) Extract(*params*,mk,S). This algorithm is performed by KGC. Taking the public parameters *params*, the master secrete key mk and an attribute set S of DU, it extracts the private key for the DU related to the attribute set S.
(3) Encrypt(*params*,M,T). This algorithm is performed by DO. Taking the public parameters *params*, the plain message M, and access policy T as input, it outputs the ciphertext CT.
(4) Decrypt(*params*,CT,sk_S, M). This algorithm is performed by the DU without using the MCC. Taking the public parameters *params*, the ciphertext CT and the key set sk_S as input, it outputs the message M.
(5) $GenTK_{out}$(*params*,sk_S,TK). This algorithm is performed by the DU. Taking the public parameters *params* and the key set sk_S as input, it outputs transformation key TK.
(6) $Transform_{out}$(*params*, CT, TK,CT'). This algorithm is performed by the MCC. Taking the public parameters *params*, the ciphertext CT and the transformation key TK , it outputs transformed ciphertext CT', which has partially decrypted by the MCC.
(7) $Decrypt_{out}$(*params*,CT',M). This algorithm is performed by the DU. Taking the public parameters *params* and the transformed ciphertext CT', it outputs the message M.

3.3. Security Model

Similar to most existing works, the MCC is assumed as a semi-trust entity, which means that the MCC is an "honest but curious" entity. To be specific, on one hand, the MCC is faithfully performing each operation of the assigned protocol and returns the correct results ; on the other hand, the MCC may be curious about the encrypted data contents and may try to learn or infer the sensitive information of the underlying plaintext of the encrypted data by virtue of its powerful computation ability. Based on the security model of Lai et al. [26], in this section, we define a security model for our SLFG-DSS to specify the capabilities and possible actions of the attacker by a game involved two participants: The challenger $\mathscr{C}$ and the attacker $\mathscr{A}$. The interactive process between them can be expressed as following steps.

Setup. The attacker $\mathscr{A}$declares a challenging access structure T^*(T^* including l leaf nodes and the corresponding attributes is $w_{*1}, w_{*2}, \cdots, w_{*l}$, respectively). The challenger $\mathscr{C}$ runs $Setup$ algorithm with the security parameter k and attribute universe U to output the master secrete key mk and the system public parameters *params*, then keeps the master secrete key mk privately and sends the system public parameters *params* to the attacker $\mathscr{A}$.

Phase 1. The attacker $\mathscr{A}$ adaptively issues the following queries:

Key extraction query. The attacker $\mathscr{A}$ adaptively chooses an attribute set $S = \{S_1, S_2, \cdots, S_n\}$ to launch the private key query; the challenger $\mathscr{C}$ returns the corresponding key set sk_S.

Decryption query. Given a ciphertext CT_* and an attribute set S, the challenger $\mathscr{C}$ performs the decryption algorithm Decrypt (CT,sk_S) and returns the decrypted result M.

$Decrypt_{out}$ query. Given certain attribute set S, ciphertext CT and CT', the challenger $\mathscr{C}$ runs $Decrypt_{out}$ algorithm and returns the decryption results M.

Challenge. The attacker $\mathscr{A}$ submits two messages M_0, M_1 with equal length and one access structure T^*, the challenger $\mathscr{C}$ randomly selects $b \in \{0, 1\}$, then the attacker $\mathscr{A}$ returns the challenge ciphertext CT_* to the attacker $\mathscr{A}$.

Phase 2. The attacker $\mathscr{A}$ continues to adaptively initiate the inquiries in phase 1 with the following two restrictions:

(1) $\mathscr{A}$ cannot issue the private key query that the selected attribute set satisfy the access structure T^*.
(2) $\mathscr{A}$ cannot make the decryption query over CT_*.

Guess. At the end of the game, the attacker $\mathscr{A}$ outputs the guess result $b' \in \{0, 1\}$ of b and the attacker $\mathscr{A}$ succeeds in the game if and only if $b' = b$.

The advantage of the attacker $\mathscr{A}$ to win the game is defined as:

$$Adv^{IND-SLFG-DSS-CCA2}(A) = 2\Pr[b' = b] - 1.$$

4. Our Concrete Construction

In this section, the concrete construction of our new scheme will be presented in detail as below.

(1) Setup(*k*,*U*). G_1, G_T are two bilinear cyclic groups with order q ($\geq 2^k$), $e : G_1 \times G_1 \to G_T$ is a bilinear pair of the two groups, all attributes set is W. Lagrange coefficients $L_{i,U}(x) = \prod_{j \in U, j \neq i} \frac{x-j}{i-j}$, U is the number set in Z_q. Select random number g, g_2 in G_1, select $a, \beta \in Z_q^*$ randomly, calculate $g_1 = g^a$, $g_3 = g_2^{a^{-1}\beta}$; select four anti-collision hash functions $H_0 : Z_q^* \to G_1$, $H_1 : G_T \to \{0,1\}^k$, $H_2 : \{0,1\}^* \to G_1$ and $H_3 : G_T \to Z_q^*$. Publish public parameters $params = \{G_1, G_T, H_0, H_1, H_2, H_3, g, g_1, g_2, g_3, e\}$, and keep the master secrete key $mk = \{\alpha, \beta, g_2^a\}$ secretly.

(2) Extract(*params*,*mk*,*S*). Given a user's attribute set S=$\{w_1, w_2, \cdots, w_n\}$, the KGC generates a set of keys for the mobile user: $sk_S = \{(g_2^{\alpha+d}({g_1}^{w_1}h_1)^{u_1}, g^{u_1}), \cdots, (g_2^{\alpha+d}({g_1}^{w_n}h_n)^{u_n}, g^{u_n}), {g_1}^{\beta^{-1}d}\}$, where $u_1, u_2, \cdots, u_n, d$ are n+1 random numbers in Z_q^*, $h_i = H_0(w_i)$ $i = 1, 2, \cdots, n$.

(3) Encrypt(*params*,*M*,*T*). On input message m, access tree T (including l leaf nodes and the corresponding attributes are $w_{*1}, w_{*2}, \cdots, w_{*l}$, respectively), return the ciphertext CT by the following steps.

① Select $r \in Z_q^*$ randomly, and for each node x, select a polynomial q_x with the degree $d_x = k_x - 1$ in top-down manner, and for root node R of the tree, set $q_R(0) = r$. Otherwise, for non-root node x, set $q_x(0) = q_{p(x)}(index(x))$, where the $p(x)$ represents the parent node of x, $index(x)$ return an unique number associated with the node x, which is uniquely assigned to x in a certain manner.
② Calculate $h_{*j} = H_0(w_{*j})$, $C_1 = g_3^r$, $C_2 = g^h$ and $C_{*j} = (C_{*j}^1, C_{*j}^2) = \{g^{q_x(0)}, ({g_1}^{w_{*j}}h_{*j})^{q_x(0)}\}$ for each leaf node x, which is corresponding to a certain attribute in the access tree T.
③ Calculate $K = H_1(e(g_1, g_2)^r)$, $C_0 = M \oplus K$, $h = H_3(e(g_1, g_2)^r, M)$.
④ Calculate $\sigma = H_2(T, C_0, C_1, C_{*1}, C_{*2}, \cdots, C_{*l}, g^h)^h$.
⑤ Output $CT = (T, C_0, C_1, C_2, C_{*1}, C_{*2}, \cdots, C_{*l}, \sigma)$.

(4) Decrypt(*params*,*CT*,sk_S, *M*). Once receiving ciphertext CT, the DU determines whether the attribute set S is satisfied with the access structure T. If it is not satisfied with the access structure, the DU returns $\perp$. Otherwise, the DU can obtain the message M by decrypting the ciphertext CT as follows.

① Define a recursive algorithm $Dec(CT, sk_S, x)$. On input the ciphertext CT, the key set sk_S associated with the attribute set S and a node x in the tree T. Denote $attr(x)$ as the true attribute associated with leaf node x. The specific decryption process by computing as follows:

For x is the leaf node, the decryption results are returned according to the following two conditions.

(a) If $i = attr(x) \in S$, return

$$\begin{aligned} Dec(CT, sk_S, x) &= \frac{e(C_{*x}^1, g_2^{\alpha+d}({g_1}^i h_i)^{u_i})}{e(C_{*x}^2, g^{u_i})} \\ &= \frac{e(g^{q_x(0)}, g_2^{\alpha+d}({g_1}^i h_i)^{u_i})}{e(({g_1}^i h_i)^{q_x(0)}, g^{u_i})} \\ &= e(g_1, g_2)^{q_x(0)} e(g, g_2^d)^{q_x(0)}. \end{aligned}$$

(b) If $i = attr(x) \notin S$, return $Dec(CT, sk_S, x)$=$\perp$.

② For x is a non-leaf node, if all child nodes z of node x, the number of nodes which meet $Dec(CT, sk_S, z) \neq \perp$ is less than the threshold k_x, return$Dec(CT, sk_S, x)$=$\perp$. Otherwise, randomly select k_x child nodes which meet $Dec(CT, sk_S, z) \neq \perp$ to form a set ${S_x}'$, and denote as $S_x = \{i = index(z) | z \in S'_x\}$, then proceed as follows.

$$Dec(CT, sk_S, x) = \prod_{z \in {S_x}'} Dec(CT, sk_S, z)^{L_{i,S_x}(0)}$$
$$= e(g_1, g_2)^{q_x(0)} e(g, {g_2}^d)^{q_x(0)}.$$

(a) Call the $Dec(CT, sk_S, R)$ algorithm, where R is the root node of access tree. We can get $temp' = e(g_1, g_2)^r e(g, {g_2}^d)^r$, then we can further obtain

$$temp = \frac{temp'}{e(g_1^{\beta^{-1}d}, g_3^r)} = e(g_1, g_2)^r.$$

(b) Calculate $K = H_1(temp), M = C_0 \oplus K, h = H_3(temp, M)$.
(c) Set $H = H_2(T, C_0, C_1, C_{*1}, C_{*2}, \cdots, C_{*l}, C_2)$, if the following two equations $C_2 = g^h$, $e(\sigma, g) = e(H, C_2)$ hold, output the message M; otherwise, output $\perp$.

(5) $GenTK_{out}$($params$,sk_S,TK). The DU can generate the transformation key as follows. He/She firstly selects a random number $t \in Z_q^*$, and computes the transformation key TK as below.

$$TK = \{{sk_S}', g^t\} = \{(g_2^{t(\alpha+d)}(g^{w_1}h_1)^{tu_1}, g^{tu_1}), \cdots, (g_2^{t(\alpha+d)}(g^{w_n}h_n)^{tu_n}, g^{tu_n}), {g_1}^{t\beta^{-1}d}, g^t\}.$$

By adopting the transformation key technique [25], a complicated ciphertext of DO can be transformed to another simple form ciphertext by the MCC acting as a semi-trusted proxy. There exist leakage risks of the decryption key sk_S of DU. However, in our scheme, during the transformation key generation procedure, DU adopts the random mask technique through random number t to prevent the cloud server and other attacker from obtaining the decryption key sk_S of DU. Moreover, t is treated as index, thus, the attacker cannot obtain the value of t based on the classical DLP problem.

(6) $Transform_{out}$($params$, CT, TK,CT'). This algorithm is performed between the DU and the MCC.

Once receiving the ciphertext CT and transformation key TK, the MCC determines whether the user's attribute set S matches the access structure T. If it does not match the access structure, the MCC returns $\perp$. Otherwise, the MCC can transform the ciphertext CT by transformation key TK as follows.

① Call $Dec(CT, {sk_S}', R)$ algorithm to calculate

$$Tmp_1 = \frac{Dec(CT, sk'_S, R)}{e(g_1^{t\beta^{-1}d}, g_3^r)} = e(g_1, g_2)^{tr}.$$

② Calculate $Tmp_2 = e(\sigma, g^t), Tmp_3 = e(H, C_2)$.
③ Return $CT' = (Tmp_1, Tmp_2, Tmp_3, C_0, C_2)$.

The transformed ciphertext CT' involve five components, which has partially decrypted by the MCC. To be specific, it contains 3 elements in G_T, 1 element in G_1 and k bits random string, thus, the length of the transformed ciphertext CT'($3|G_T| + |G_1| + k$) is constant, which is independent with the number of the attributes.

(7) $Decrypt_{out}$($params$,CT',t). After received the transformed ciphertext CT', the DU calculates $temp = {Tmp}^{t^{-1}}$, $K = H_1(temp), M = C_0 \oplus K$, $h = H_3(temp, M)$. If the following two equations $C_2 = g^h$ and $Tmp_2 = {Tmp_3}^t$ hold, outputs message M, otherwise, outputs $\perp$.

Apparently, it can be observed that the DU can check the validity of the transformed ciphertext CT'.

5. Security Analysis

Under the security model defined in Section 3.3, in this section, we prove that the proposed SLFG-DSS scheme is secure against the IND-SLFG-DSS-CCA2 with respect to Theorem 1.

Theorem 1. *If the Decisional Bilinear Diffie-Hellman Problem (DBDHP) is difficult in* (G_1, G_T)*, the proposed scheme is sure against the IND-SLFG-DSS-CCA2 under the random oracle model.*

Proof. Given a random instance (g, g^a, g^b, R), the goal of challenger $\mathscr{C}$ is to decide whether R is equal to $e(g,g)^{a^2b}$.

Setup. At the beginning of the game, the challenger $\mathscr{C}$ defines four hash functions and system parameters according to the given instance and returns the results to the adversary $\mathscr{A}$ as follows.

$H_0(w)$=$g_1{}^{-w}g^{r_w}$, where $r_w \in Z_q^*$ is random number.

$H_1(R)$=K, where $K \in \{0,1\}^k$ is a random string.

$H_2(str) = X$, where $X \in G_1$ is a random number.

$H_3(R) = z$, where $z \in Z_q^*$ is a random number.

Challenger $\mathscr{C}$ randomly selects $r, v, t \in Z_q^*$, and sends the system parameters $params = (G_1, G_T, H_0, H_1, H_2, H_3, g, g_1 = g^a, g_2 = g^{ar}, g_3 = g^{vr}, g^t, e)$ to the attacker $\mathscr{A}$. The attacker $\mathscr{A}$ can only get the hash function value through hash query. Here, the master secrete key mk =$\{a, \beta = av, g^{a^2r}\}$.

Phase 1. The attacker $\mathscr{A}$ adaptively launches the following queries:

Key extraction query. The attacker $\mathscr{A}$ adaptively selects an attribute set $S = \{S_1, S_2, \cdots, S_n\}$ to issue the private key query; the challenger $\mathscr{C}$ selects n+1 random numbers: $u_1, u_2, \cdots, u_n, d'$, and returns the corresponding key set

$$sk_S = \{(g^{d'}(g_1{}^{w_1}h_1)^{u_1} = g_2{}^{a+d}(g_1{}^{w_1}h_1)^{u_1}, g^{u_1}), \cdots, (g^{d'}(g_1{}^{w_n}h_n)^{u_n} = g_2{}^{a+d}(g_1{}^{w_n}h_n)^{u_n}, g^{u_n}), g^{v^{-1}d'}g_1{}^{-v^{-1}} = g_1{}^{\beta^{-1}d}\},$$

where $h_i = H_0(w_i)$ $i = 1, 2, \cdots, n$, $d = d' - a$.

Decryption query. Given a ciphertext CT_*, the challenger $\mathscr{C}$ firstly performs the key extraction query for attribute set S to get sk_S, then, executes the decryption algorithm Decrypt (CT,sk_S) and returns the decrypted result M.

Note that if the attribute set of the ciphertext satisfies the access tree of the challenged ciphertext, the challenger $\mathscr{C}$ just obtains the key set sk_S by the key extraction query, but does not return it to the attacker.

$Decrypt_{out}$ query. In this query stage, when the attacker $\mathscr{A}$ queries transform information over a certain attribute set S, challenger $\mathscr{C}$ obtains the key set sk_S through the key extraction query, then calculates ${sk_S}^t$. TK =$\{(g_2{}^{ta+td}(g_1{}^{w_1}h_1)^{tu_1}, g^{tu_1}), \cdots, g_2{}^{ta+td}(g_1{}^{w_n}h_n)^{tu_n}, g^{tu_n}), g_1{}^{t\beta^{-1}d}\}$. Once receiving the decryption query over ciphertext CT' from the attacker, the challenger $\mathscr{C}$ decrypts CT' using t and returns the decryption results by performing $Decrypt_{out}$ algorithm.

Challenge. The attacker $\mathscr{A}$ sends two messages with equal length M_0, M_1 and one access structure T^*(T^* involves l leaf nodes and the corresponding attributes are $w_{*1}, w_{*2}, \cdots, w_{*l}$, respectively), the challenger $\mathscr{C}$ randomly selects $b \in \{0,1\}$, then the attacker $\mathscr{A}$ returns the challenge ciphertext CT_* as follows.

(1) Query hash function $H_0(w_{*i})$ $i = 1, 2, \cdots, l_*$, obtain $r_{w_{*i}}$.
(2) Denote *root* is the root node of T^*, set $q_{root}(0) = r_*$. According to the step 1 in encryption algorithm, calculate $q_x(0)$ for each leaf node x.
(3) Calculate $C_{*i} = \{g^{bq_x(0)}, g^{br_{w_{*i}}q_x(0)}\}$ for each x which is corresponding to an attribute in the access tree T^*.

(4) Calculate $h = H_3(R^{br_*}, m)$, $C_{0*} = M \oplus H_1(R^{br_*})$, $C_{1*} = g_3^{br_*}$ and $\sigma_* = H_2(T^*, C_{0^*}, C_{1^*}, C_{*1}, C_{*2}, \cdots, C_{*l_*}, g^h)^h$.
(5) Return $CT_* = (T^*, C_{0^*}, C_{1^*}, C_{*1}, C_{*2}, \cdots, C_{*l_*}, \sigma_*)$.

Phase 2. The attacker $\mathscr{A}$ continues to adaptively initiate the inquiries in phase 1 with the following two restrictions:

(1) $\mathscr{A}$ cannot issue the private key query that the selected attribute set satisfy the access structure T^*.
(2) $\mathscr{A}$ cannot make the decryption query over CT_*.

Guess. At the end of the game, the attacker $\mathscr{A}$ outputs the guess result $b' \in \{0,1\}$ of b and the attacker $\mathscr{A}$ succeeds in the game if and only if $b' = b$, then challenger $\mathscr{C}$ outputs 1, otherwise, outputs 0.

Obviously, $temp_* = e(g_2^a, g^{br_*})$ can be calculated according to the decryption algorithm. Therefore, if $R = e(g,g)^{a^2b}$, then CT_* is the legitimate ciphertext of m_b, i.e., the challenger $\mathscr{C}$ can decide whether R is equal to $e(g,g)^{a^2b}$ according to the reply of the attacker $\mathscr{A}$.

In addition, there is no failure case during the simulation. Therefore, if the attacker $\mathscr{A}$ can broke the scheme with non-negligible probability ε, it means that the challenger $\mathscr{C}$ can decide whether R is equal to $e(g,g)^{a^2b}$ with non-negligible probability. This is a paradox because it is well accepted that the DBDHP problem is intractable. □

6. Performance Evaluation

In this section, we first conduct the functionality comparison of our novel scheme with the schemes [14,26,29,45]. Subsequently, we evaluate and compare the efficiency among them from both theoretical analysis and experimental simulation aspects.

6.1. Functionality Comparison

In this subsection, we conduct the functionality comparison between our new scheme and several other schemes [14,26,29,45] in terms of the functionalities of outsourcing decryption, verifiability and CCA security. The comparison results are listed in Table 1. As shown in Table 1, it can be observed that scheme [14] is the standardized ABE schemes and cannot support all of the three properties. Scheme [29] can realize outsourcing decryption but not support the properties of verifiability and CCA security. Scheme [26,45] can achieve the security property of outsourcing decryption and verifiability, but fail to support CCA security. Only our proposed novel scheme supports all of the three properties simultaneously, i.e., outsourcing decryption, verifiability, and CCA security.

Based on the above analysis, we could safely draw a conclusion that our novel scheme has better security level than others and can be applicable to the circumstance with higher security level demand.

Table 1. Comparisons of functionalities.

Scheme	Outsourcing	Verifiability	CCA Security
[14]	No	No	No
[29]	Yes	No	No
[26]	Yes	Yes	No
[45]	Yes	Yes	No
Ours	Yes	Yes	Yes

6.2. Performance Analysis

In this subsection, we conduct performance analysis and comparison between our proposed scheme and the other existing scheme [14,26,29,45] from both theoretical analysis and experimental simulation aspects.

Table 2 summarizes the comparison of computation complexity of our novel scheme with the several other schemes [14,26,29,45] which demonstrates the theoretical numerical analysis results.

To facilitate expression, let T_{pair}, T_{G_1}, T_{G_T} and T_H denote the computational cost of one bilinear pairing operation, one group operation (including exponentiation, multiplication) in G_1, one group (including exponentiation, multiplication) operation in G_T, one general one-way hash function operation, respectively.

In Encrypt phase, the DO in reference [14] requires to perform $2N_1 + 1$ group operations in G_1, 2 group operations in G_T, and N_1 general one-way hash function operations, respectively, Therefore, the computation cost is $2T_{G_T} + (2N_1 + 1)T_{G_1} + N_1T_H$. The DO in reference [29] requires to perform $2 + 4N_1$ group operations in G_1 and 2 group operations in G_T, respectively, Therefore, the computation cost is $(2 + 4N_1)T_{G_1} + 2T_{G_T}$. The DO in reference [26] requires to perform $6 + 8N_1$ group operations in G_1, 4 group operations in G_T, and 2 general one-way hash function operations, respectively. Therefore, the computation cost is $(6 + 8N_1)T_{G_1} + 4T_{G_T} + 2T_H$. The DO in reference [45] requires to perform $2N_1$ group operations in G_1, and $2 + N_1$ group operations in G_T, respectively, Therefore, the computation cost is $2N_1T_{G_1} + (2 + N_1)T_{G_T}$. The DO in our proposed scheme requires to perform $4N_1 + 3$ group operations in G_1, 1 group operations in G_T, and $N_1 + 3$ general one-way hash function operations, respectively. Therefore, the computation cost is $(4N_1 + 3)T_{G_1} + T_{G_T} + (N_1 + 3)T_H$.

In Decrypt phase, the DU in reference [14] needs to execute $2N_2 + 1$ bilinear pairing operations and $2\sum_{z\in T} d_z + 2$ group operations in G_T, respectively, Therefore, the computation cost is $(2N_2 + 1)T_{pair} + (2\sum_{z\in T} d_z + 2)T_{G_T}$. Reference [29] does not provide the Decrypt algorithm, therefore, the computation cost for Decrypt phase is denoted as *None*. The DU in reference [26] needs to execute $4N_2 + 2$ bilinear pairing operations, 4 group operations in G_1, and $4N_2 + 4$ group operations in G_T, respectively. Therefore, the computation cost is $(4N_2 + 2)T_{pair} + 4T_{G_1} + (4N_2 + 4)T_{G_T}$. Reference [45] does not provide the Decrypt algorithm, therefore, the computation cost for Decrypt phase is denoted as *None*. The DU in our proposed scheme needs to execute $2N_2 + 3$ bilinear pairing operations, 1 group operations in G_1, $2\sum_{z\in T} d_z + 1$ group operations in G_T, and 3 general one-way hash function operations, respectively. Therefore, the computation cost is $(2N_2 + 3)T_{pair} + T_{G_1} + (2\sum_{z\in T} d_z + 1)T_{G_T} + 3T_H$.

In Transform phase, reference [14] does not provide the Transform algorithm, therefore, the computation cost for Transform phase is denoted as *None*. The DU in reference [29] needs to carry out $3N_2 + 2$ bilinear pairing operations, and $4N_2 + 2$ group operations in G_T, respectively, Therefore, the computation cost is $(3N_2 + 2)T_{pair} + (4N_2 + 2)T_{G_T}$. The DU in reference [26] needs to carry out $4N_2 + 2$ bilinear pairing operations, and $4N_2 + 2$ group operations in G_T, respectively, Therefore, the computation cost is $(4N_2 + 2)T_{pair} + (4N_2 + 2)T_{G_T}$. The DU in reference [45] needs to carry out $4N_2$ bilinear pairing operations, $3N_2$ group operations in G_1, $5N_2$ group operations in G_T, and 1 general one-way hash function operations, respectively. Therefore, the computation cost is $4N_2T_{pair} + 3N_2T_{G_1} + 5N_2T_{G_T} + 1T_H$. The DU in our proposed scheme needs to carry out $2N_2 + 3$ bilinear pairing operations, $2\sum_{z\in T} d_z + 1$ group operations in G_T, and 1 general one-way hash function operations, respectively. Therefore, the computation cost is $(2N_2 + 3)T_{pair} + (2\sum_{z\in T} d_z + 1)T_{G_T} + T_H$.

In $Decrypt_{out}$ phase, reference [14] does not provide the $Decrypt_{out}$ algorithm, therefore, the computation cost for $Decrypt_{out}$ phase is denoted as *None*. The DU in reference [29] has to run 2 group operations in G_T. Therefore, the computation cost is $2T_{G_T}$. The DU in reference [26] has to run 4 group operations in G_1, 4 group operations in G_T, and 2 general one-way hash function operations, respectively, Therefore, the computation cost is $4T_{G_1} + 4T_{G_T} + 2T_H$. The DU in reference [45] has to run 2 group operations in G_T. Therefore, the computation cost is $2T_{G_T}$. The DU in our proposed scheme has to run 1 group operation in G_1, 2 group operations in G_T, and 2 general one-way hash function operations, respectively, Therefore, the computation cost is $1T_{G_1} + 2T_{G_T} + 2T_H$.

Table 2 illustrates that computation complexity for the operation of encryption, decryption and transformation. Here, we mainly focus on the comparison of decryption time. The decryption time of scheme [14] is roughly proportional to the number of attributes, especially with respect to the expensive pairing operations, while it keeps constant in schemes [26,29,45] and ours since they have migrated the heavy operations in the decryption phase to the cloud server and only remain several lightweight operations. However, schemes [29] fails to support the properties of verifiability. Careful observation will further reveal that our scheme slightly outperforms Lai et al.'s scheme [26] because our proposed scheme reduce three element operations in G_1, two element operations in G_T, and Fan et al.'s scheme [45] slightly outperforms ours because Fan et al.'s scheme reduce one element operation in G_1, and two hash operations, however, their scheme is CPA-secure, while ours is CCA2-secure.

Table 2. Comparison of computation complexity.

Scheme	Encrypt	Decrypt	Transform	$Decrypt_{out}$
[14]	$(2N_1+1)T_{G_1}+2T_{G_T}+N_1T_H$	$(2N_2+1)T_{pair}+(2\sum_{z\in T}d_z+2)T_{G_T}$	*None*	*None*
[29]	$(2+4N_1)T_{G_1}+2T_{G_T}$	*None*	$(3N_2+2)T_{pair}+(4N_2+2)T_{G_T}$	$2T_{G_T}$
[26]	$(6+8N_1)T_{G_1}+4T_{G_T}+2T_H$	$(4N_2+2)T_{pair}+4T_{G_1}+(4N_2+4)T_{G_T}$	$(4N_2+2)T_{pair}+(4N_2+2)T_{G_T}$	$4T_{G_1}+4T_{G_T}+2T_H$
[45]	$2N_1T_{G_1}+(2+N_1)T_{G_T}$	*None*	$4N_2T_{pair}+3N_2T_{G_1}+5N_2T_{G_T}+1T_H$	$2T_{G_T}$
Ours	$(4N_1+3)T_{G_1}+T_{G_T}+(N_1+3)T_H$	$(2N_2+3)T_{pair}+T_{G_1}+(2\sum_{z\in T}d_z+1)T_{G_T}+3T_H$	$(2N_2+3)T_{pair}+(2\sum_{z\in T}d_z+1)T_{G_T}+1T_H$	$1T_{G_1}+2T_{G_T}+2T_H$

In order to validate theoretical analysis of the efficiency of our proposed scheme, we also conduct the experiment simulation by using a rapidly prototyping development tool Charm 0.43 [46] with Python programming language, and the system platform is Ubuntu 16.04 64-bit operating system with the Intel(R)Core (TM) 4130 CPU @3.40GHz, 4GB RAM. In addition, each experiment simulation is conducted 30 times, and the mean of the experiment results is taken as the final result, which are intuitively illustrated in Figures 2 and 3, respectively.

Figure 2 depicts the performance comparison of outsourced decryption with non-outsourced decryption for our scheme. We can easily find that our scheme with the outsourced decryption method is significantly superior to our scheme without the outsourced decryption method. Apparently, as depicted in Figure 2, the decryption time of the non-outsourced decryption method grows rapidly with the number of attributes, while that of outsourced decryption is nearly constant time at a quite low level. The reason for this we have discussed in previous section.

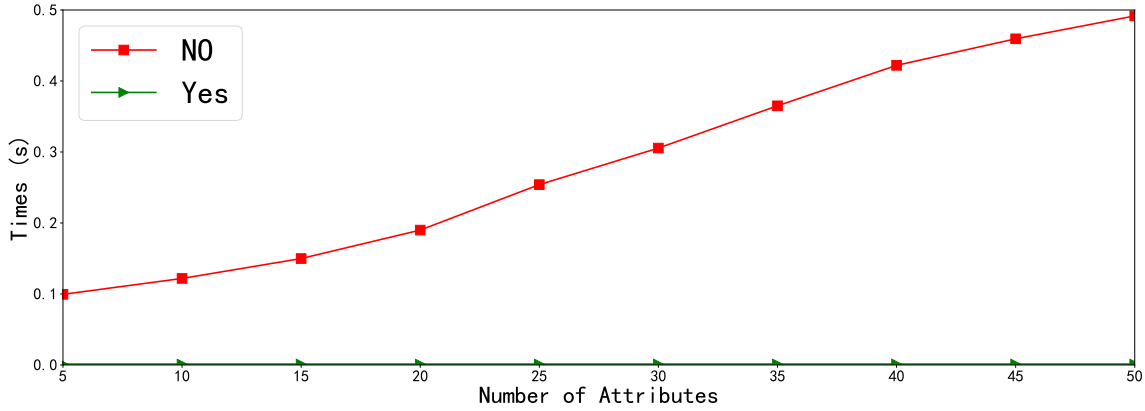

Figure 2. Comparison of outsourcing and non-outsourcing of our scheme, where "Yes" represents our scheme with outsourcing and "No" represents our scheme without outsourcing.

Figure 3 shows the comparison of outsourced decryption time between our proposal, Lai et al.'s scheme [26], and Fan et al.'s scheme [45]. It is clearly observed that outsourced decryption time of our proposal and Lai et al.'s scheme [26] are roughly similar, but a closer look will find that our scheme slightly outperforms Lai et al.'s scheme [26], which confirms the theoretical analysis discussed in

previous subsection. It is also can be seen that Fan et al.'s scheme [45] slightly outperforms than ours, however, their scheme only realizes CPA-security while ours achieves CCA2-security. It is worth and meaningful for mobile uses to enhance the security level from CPA to CCA2 at the very little cost. It can be safely concluded that our proposed novel SLFG-DSS scheme can achieve higher security level and high performance.

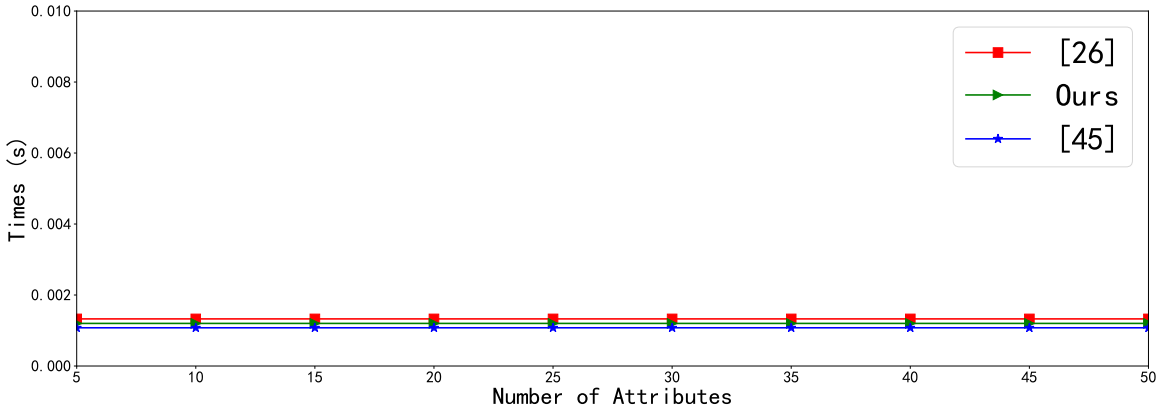

Figure 3. Comparison of outsourced decryption time.

7. Conclusions

In this paper, we investigate the efficiency bottleneck of using the ABE scheme to achieve fine-grained data sharing in mobile cloud computing and propose a secure and lightweight fine-grained data sharing scheme for mobile cloud computing which simultaneously supports the following desired security properties: (1) Checkability; (2) resisting decryption key exposure; (3) achieving CCA security level. In our novel schemes, by utilizing the transformation key technique, we outsource the most time-consuming pairing operations of the ABE scheme, which previously executed on the mobile device side, and only leaves a slight number of inexpensive operations. The concrete security proof and performance analysis demonstrate that our novel scheme is secure and practical for mobile cloud computing.

Considering the tremendous progress of quantum computers, our future work will be focused on developing some post-quantum-secure outsourced attribute-based encryption schemes from lattice to resist against quantum computer attacks in the near future.

Author Contributions: Conceptualization, H.L. and C.L.; methodology, H.L. and C.L.; software, H.L. and C.L.; validation, H.L., C.L. and X.F.; security analysis, H.L. and C.L.; resources, H.G.; writing—original draft preparation, H.L.; writing—review and editing, H.L. and X.F.; visualization, X.F.; supervision, C.W., F.L. and H.G.; funding acquisition, H.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by National Natural Science Foundation of China under Grant No. 61602080, No. 61602084; Zhejiang Provincial Natural Science Foundation of China under Grant No. LY19F020045; Guangxi Key Laboratory of Cryptography and Information Security under Grant No. GCIS201718; the Opening Project of Guangdong Provincial Key Laboratory of Information Security Technology under Grant No. 2017B030314131-05; the Department of Education of Zhejiang Province of China under Grant No. Y201636547; and the Key Research Project of Zhejiang Province under Grant No. 2017C01062.

Acknowledgments: The authors are very grateful to the anonymous referees for their valuable comments and constructive suggestions to improve the quality of our paper.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ABE	Attribute-Based Encryption
KP-ABE	Key-Policy Attribute-Based Encryption
CP-ABE	Ciphertext-Policy Attribute-Based Encryption
KGC	Key Generation Center
CSP	Cloud Service Provider
MCC	Mobile Cloud Computing
DO	Data Owners
DU	Data Users
TK	Transformation Key
CT	Ciphertext
CPA	Chosen-Plaintext Attack
CCA	Chosen-Ciphertext Attack
PHR	Personal Health Record
DBDH	Decisional Bilinear Diffie-Hellman
PPT	Probabilistic Polynomial Time

References

1. Mell, P.; Grance, T. The NIST Definition of Cloud Computing. 2011. Available online: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf (accessed on 20 August 2020.)
2. Wu, J.H.; Wang, S.C. What drives mobile commerce?: An empirical evaluation of the revised technology acceptance model. *Inf. Manag.* **2005**, *42*, 719–729. [CrossRef]
3. Nugraha, A.; Supangkat, S.H.; Nugroho, D. Goesmart: Social media education in cloud computing. In Proceedings of the 2012 International Conference on Cloud Computing and Social Networking (ICCCSN), West Java, Indonesia, 26–27 April 2012; pp. 1–6.
4. De, D.; Mukherjee, A. Femto-cloud based secure and economic distributed diagnosis and home health care system. *J. Med. Imaging Health Inform.* **2015**, *5*, 435–447. [CrossRef]
5. Dinh, H.T.; Lee, C.; Niyato, D.; Wang, P. A survey of mobile cloud computing: architecture, applications, and approaches. *Wirel. Commun. Mob. Comput.* **2013**, *13*, 1587–1611. [CrossRef]
6. Fernando, N.; Loke, S.W.; Rahayu, W. Mobile cloud computing: A survey. *Future Gener. Comput. Syst.* **2013**, *29*, 84–106. [CrossRef]
7. Rahimi, M.R.; Ren, J.; Liu, C.H.; Vasilakos, A.V.; Venkatasubramanian, N. Mobile cloud computing: A survey, state of art and future directions. *Mob. Networks Appl.* **2014**, *19*, 133–143. [CrossRef]
8. Liu, F.; Shu, P.; Jin, H.; Ding, L.; Yu, J.; Niu, D.; Li, B. Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications. *IEEE Wirel. Commun.* **2013**, *20*, 14–22.
9. Abolfazli, S.; Sanaei, Z.; Ahmed, E.; Gani, A.; Buyya, R. Cloud-based augmentation for mobile devices: motivation, taxonomies, and open challenges. *IEEE Commun. Surv. Tutor.* **2013**, *16*, 337–368. [CrossRef]
10. Othman, M.; Madani, S.A.; Khan, S.U. A survey of mobile cloud computing application models. *IEEE Commun. Surv. Tutor.* **2013**, *16*, 393–413.
11. Shamir, A. Identity-based cryptosystems and signature schemes. In *Workshop on the Theory and Application of Cryptographic Techniques*; Springer: Berlin/Heidelberg, Germany, 1984; pp. 47–53.
12. Sahai, A.; Waters, B. Fuzzy identity-based encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 457–473.
13. Goyal, V.; Pandey, O.; Sahai, A.; Waters, B. Attribute-based encryption for fine-grained access control of encrypted data. In Proceedings of the 13th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 30 October–3 November 2006; pp. 89–98.
14. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-policy attribute-based encryption. In Proceedings of the 2007 IEEE symposium on security and privacy (SP'07), Berkeley, CA, USA, 20–23 May 2007; pp. 321–334.
15. Wang, H.; He, D.; Shen, J.; Zheng, Z.; Zhao, C.; Zhao, M. Verifiable outsourced ciphertext-policy attribute-based encryption in cloud computing. *Soft Comput.* **2017**, *21*, 7325–7335. [CrossRef]
16. Li, J.; Li, X.; Wang, L.; He, D.; Ahmad, H.; Niu, X. Fuzzy encryption in cloud computation: efficient verifiable outsourced attribute-based encryption. *Soft Comput.* **2018**, *22*, 707–714. [CrossRef]

17. Chase, M.; Chow, S.S. Improving privacy and security in multi-authority attribute-based encryption. In Proceedings of the 16th ACM Conference on Computer and Communications Security, Hong Kong, China, 7–11 June 2009; pp. 121–130.
18. Liang, K.; Au, M.H.; Liu, J.K.; Susilo, W.; Wong, D.S.; Yang, G.; Yu, Y.; Yang, A. A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing. *Future Gener. Comput. Syst.* **2015**, *52*, 95–108. [CrossRef]
19. Qian, H.; Li, J.; Zhang, Y.; Han, J. Privacy-preserving personal health record using multi-authority attribute-based encryption with revocation. *Int. J. Inf. Secur.* **2015**, *14*, 487–497. [CrossRef]
20. Li, J.; Yao, W.; Zhang, Y.; Qian, H.; Han, J. Flexible and fine-grained attribute-based data storage in cloud computing. *IEEE Trans. Serv. Comput.* **2016**, *10*, 785–796. [CrossRef]
21. Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *International Workshop on Public Key Cryptography*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 53–70.
22. Fan, C.I.; Huang, V.S.M.; Ruan, H.M. Arbitrary-state attribute-based encryption with dynamic membership. *IEEE Trans. Comput.* **2013**, *63*, 1951–1961. [CrossRef]
23. Fu, X.; Nie, X.; Wu, T.; Li, F. Large universe attribute based access control with efficient decryption in cloud storage system. *J. Syst. Softw.* **2018**, *135*, 157–164. [CrossRef]
24. Li, J.; Shi, Y.; Zhang, Y. Searchable ciphertext-policy attribute-based encryption with revocation in cloud storage. *Int. J. Commun. Syst.* **2017**, *30*, e2942. [CrossRef]
25. Green, M.; Hohenberger, S.; Waters, B. Outsourcing the decryption of abe ciphertexts. In *USENIX Security Symposium*; USENIX Association: San Francisco, CA, USA, 2011; pp. 1–16.
26. Lai, J.; Deng, R.H.; Guan, C.; Weng, J. Attribute-based encryption with verifiable outsourced decryption. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 1343–1354.
27. Li, J.; Wang, Y.; Zhang, Y.; Han, J. Full verifiability for outsourced decryption in attribute based encryption. *IEEE Trans. Serv. Comput.* **2017**, *13*, 478–487. [CrossRef]
28. Li, J.; Sha, F.; Zhang, Y.; Huang, X.; Shen, J. Verifiable outsourced decryption of attribute-based encryption with constant ciphertext length. *Secur. Commun. Netw.* **2017**, *2017*, 3596205. [CrossRef]
29. Wang, H.; Yang, B.; Wang, Y. Server aided ciphertext-policy attribute-based encryption. In Proceedings of the 2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops, Gwangju, Korea, 25–27 March 2015; pp. 440–444.
30. Zhang, R.; Ma, H.; Lu, Y. Fine-grained access control system based on fully outsourced attribute-based encryption. *J. Syst. Softw.* **2017**, *125*, 344–353. [CrossRef]
31. Liao, Y.; He, Y.; Li, F.; Jiang, S.; Zhou, S. Analysis of an ABE scheme with verifiable outsourced decryption. *Sensors* **2018**, *18*, 176. [CrossRef] [PubMed]
32. Qin, B.; Zhao, Q.; Zheng, D.; Cui, H. (Dual) server-aided revocable attribute-based encryption with decryption key exposure resistance. *Inf. Sci.* **2019**, *490*, 74–92. [CrossRef]
33. Ali, M.; Sadeghi, M.R.; Liu, X. Lightweight Fine-Grained Access Control for Wireless Body Area Networks. *Sensors* **2020**, *20*, 1088. [CrossRef] [PubMed]
34. Yang, Y.; Liu, X.; Deng, R.H.; Li, Y. Lightweight sharable and traceable secure mobile health system. *IEEE Trans. Dependable Secur. Comput.* **2020**, *17*, 78–91. [CrossRef]
35. Guo, R.; Li, X.; Zheng, D.; Zhang, Y. An attribute-based encryption scheme with multiple authorities on hierarchical personal health record in cloud. *J. Supercomput.* **2020**, *76*, 4884–4903. [CrossRef]
36. Kibiwott, K.P.; Zhao, Y.; Kogo, J.; Zhang, F. Verifiable fully outsourced attribute-based signcryption system for IoT eHealth big data in cloud computing. *Math. Biosci. Eng.* **2019**, *16*, 3561. [CrossRef]
37. Zhang, S.; Li, W.; Wen, Q.; Zhang, H.; Jin, Z. A Flexible KP-ABE Suit for Mobile User Realizing Decryption Outsourcing and Attribute Revocation. *Wirel. Pers. Commun.* **2020**, to be published. [CrossRef]
38. Gentry, C. Fully homomorphic encryption using ideal lattices. In Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing, Bethesda, MD, USA, 31 May–2 June 2009; pp. 169–178.
39. Gentry, C.; Halevi, S. Implementing gentry's fully-homomorphic encryption scheme. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 129–148.
40. Brakerski, Z.; Vaikuntanathan, V. Efficient fully homomorphic encryption from (standard) LWE. *SIAM J. Comput.* **2014**, *43*, 831–871. [CrossRef]

41. Gennaro, R.; Gentry, C.; Parno, B. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In *Annual Cryptology Conference*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 465–482.
42. Chung, K.M.; Kalai, Y.; Vadhan, S. Improved delegation of computation using fully homomorphic encryption. In *Annual Cryptology Conference*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 483–501.
43. Jin, F.; Zhu, Y.; Luo, X. Verifiable fully homomorphic encryption scheme. In Proceedings of the 2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet), Yichang, China, 21–23 April 2012; pp. 743–746.
44. Li, J.; Chen, X.; Li, J.; Jia, C.; Ma, J.; Lou, W. Fine-grained access control system based on outsourced attribute-based encryption. In *European Symposium on Research in Computer Security*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 592–609.
45. Fan, K.; Liu, T.; Zhang, K.; Li, H.; Yang, Y. A secure and efficient outsourced computation on data sharing scheme for privacy computing. *J. Parallel Distrib. Comput.* **2020**, *135*, 169–176. [CrossRef]
46. Akinyele, J.A.; Garman, C.; Miers, I.; Pagano, M.W.; Rushanan, M.; Green, M.; Rubin, A.D. Charm: A framework for rapidly prototyping cryptosystems. *J. Cryptogr. Eng.* **2013**, *3*, 111–128. [CrossRef]