electronics

Article

Attribute-Based Encryption Scheme with *k*-Out-of-*n* Oblivious Transfer

Hao Zhang *, Yue Zhao, Jintao Meng, Xue Wang and Kaijun Wu

Science and Technology on Communication Security Laboratory, Chengdu 610041, China;
zhaoy3058@cetcsc.com (Y.Z.); mengjt16313@cetcsc.com (J.M.); wangx12205@cetcsc.com (X.W.);
wukj2350@cetcsc.com (K.W.)
* Correspondence: zhangh12239@cetcsc.com

Abstract: Attribute-based encryption enables users to flexibly exchange and share files with others. In these schemes, users utilize their own attributes to acquire public-private key pairs from the key generation center. However, achieving this for users who wish to keep their attributes private poses a challenge. To address this contradiction, we propose an original scheme that combines ciphertext policy attribute-based encryption with a *k*-out-of-*n* oblivious transfer protocol. This scheme allows the distribution of corresponding public-private key pairs to users without the key generation center needing to obtain specific user attributes. Furthermore, it ensures the privacy of the key generation center. Security analysis demonstrates that the scheme is secure in the random oracle model. Our performance comparison and experimental results indicate that the scheme is both flexible and efficient.

Keywords: attribute-based encryption; privacy preservation; oblivious transfer; key escrow

Citation: Zhang, H.; Zhao, Y.; Meng, J.; Wang, X.; Wu, K. Attribute-Based Encryption Scheme with *k*-Out-of-*n* Oblivious Transfer. *Electronics* **2023**, *12*, 4502. https://doi.org/10.3390/electronics12214502

Academic Editor: Aryya Gangopadhyay

Received: 24 August 2023
Revised: 24 October 2023
Accepted: 31 October 2023
Published: 1 November 2023

1. Introduction

With the increasing application of cloud computing, a vast amount of data is stored in the cloud for processing. However, a significant portion of this data is sensitive and requires encryption to ensure its security. Traditional symmetric encryption methods, when applied to share encrypted files with multiple parties, necessitate the use of different symmetric keys for each encryption and decryption instance. This lack of flexibility in sharing and the complexity of key management pose challenges.

Attribute-based encryption (ABE) [1] emerges as a flexible solution that supports one-to-many encryption, providing an effective means to address the aforementioned challenges. The fundamental concept of ABE involves associating ciphertext and keys with attribute sets and access structures. ABE is generally categorized into Key Policy Attribute-Based Encryption (KP-ABE) [2] and Ciphertext Policy Attribute-Based Encryption (CP-ABE) [3]. KP-ABE involves the user's key incorporating an access structure (access policy), and the ciphertext aligns with a series of attribute sets. The user can correctly decrypt the ciphertext only if the attribute set of the ciphertext satisfies the access structure (access policy) of the user key. This approach is suitable for static scenarios where users are the principal entities and only specific ciphertexts matching their access policies can be decrypted. On the other hand, CP-ABE associates the user's key with a set of attributes, and the ciphertext contains an access structure (access policy). The user can correctly decrypt the ciphertext only if their attribute set aligns with the access structure (access policy) of the ciphertext. This design is more applicable to real-world scenarios where each user acquires keys from the Key Generation Center (KGC) based on their own attributes. Subsequently, the data owner encrypts the data with an access structure (access policy).

Therefore, CP-ABE is highly compatible with cloud computing. When implementing CP-ABE in cloud computing scenarios, users in possession of data can define an access structure (access policy) for the encrypted data. Only unique users whose attributes satisfy the access structure (access policy) can accurately decrypt the ciphertext. Consequently, there is no need for the user to replicate the encryption of the data when sharing it. This not only eliminates redundancy but also enhances flexibility for data owners in sharing data, thanks to the customizable access structure (access policy) settings.

While CP-ABE presents an algorithm based on public key cryptography capable of achieving precise access control functions, it encounters key escrow issues in practical applications. In traditional CP-ABE [4–6], users transmit their attribute sets to the KGC, which then generates the corresponding private key based on the user's attributes. Subsequently, the user encrypts and shares files using this private key. It is evident in this process that KGC gains knowledge of the specific attributes of the user. In real-world usage scenarios, KGC acts as an honest but curious entity. The attributes involved are personal and private information for users who are understandably reluctant to disclose it. Consequently, users express concerns about the potential compromise of their privacy. To address this issue, some solutions currently implemented involve concealing the access structure [7–9]. However, this hidden access structure primarily addresses privacy protection against malicious access by unauthorized users. Another approach is the joint generation of private keys by multiple KGCs [10,11]. While this solution prevents attributes from being exclusively known by a single KGC, it does not entirely resolve the problem of user privacy exposure to any KGC. The pursuit of a robust solution to diminish the risk of privacy leakage in CP-ABE continues to be a formidable challenge.

Addressing the aforementioned concerns, we propose that the k-out-of-n oblivious transfer protocol emerge as a potent solution. Fundamentally, the KGC maintains a set comprising n attributes. Users are allowed to selectively choose k attributes (where $k < n$) that resonate with their individual sets from these n attributes. Following this, users encrypt the chosen k attributes and convey them to the KGC. Consequently, the KGC, leveraging these k attributes, formulates the corresponding private key and allocates it to the users. A crucial aspect of this procedure is the KGC's lack of awareness concerning the specific attributes chosen by the users, ensuring that the particulars used in the private key's generation remain concealed. This method significantly bolsters user privacy safeguards in cloud computing contexts. Thus, the application of the k-out-of-n oblivious transfer protocol is elucidated as a proficient approach, augmenting privacy safeguards while preserving the intrinsic functionality of attribute-based encryption.

2. Related Work

Originally, ABE was limited to executing threshold operations, and its policy expression lacked the necessary versatility. Subsequently, researchers proposed ABE mechanisms based on ciphertext policy and key policy. These advancements broadened the scope of attribute operations and facilitated the implementation of flexible access control policies.

In CP-ABE, the user's key is identified by an attribute set, and the ciphertext is associated with the access structure. Before data is encrypted, the data owner is aware of the type of user permitted to access it. In the majority of CP-ABE scenarios, the access structure is made public. To protect the privacy of the data owner's private attributes contained in the access structure, various research works on hiding access structures have been proposed. These works are primarily categorized into two groups: CP-ABE schemes that partially hide the access structure and CP-ABE schemes that fully hide the access structure. Kapadia [12] proposed a CP-ABE scheme capable of hiding the access policy. This method achieves policy hiding by re-encrypting the ciphertext for each user, introducing an online semi-trusted server. However, this method makes the server the bottleneck of the entire system in terms of efficiency and security.

To enhance the access structure's flexibility in access control capabilities, Xu [13] utilized the tree access structure to implement a CP-ABE scheme capable of hiding the access policy. This scheme not only protects policies but also offers flexible access control capabilities. Zhang [14] introduced a CP-ABE scheme supporting partially hidden access structures (PHAS). Since attribute values are concealed in the ciphertext, users cannot directly judge the equivalence between their attributes and those in the access structure. They designed a DeJudge algorithm that uses linear algebra operations and LSSS monotonicity to help users calculate attributes, determining whether the set satisfies the access structure. However, a limitation is that the DeJudge algorithm imposes a significant computational burden on users. Chase [15] considered a distributed ABE scheme using the multi-authority model to address key escrow issues. They resolved challenges by involving multiple attribute authorities in the key generation process. However, the scheme's performance is influenced by the number of attributes, and its access structure has limited expressiveness, supporting only AND gates, restricting data owners' ability to formulate access policies. Zhao [16] designed a scheme combining multiple attribute authorities and a central authority structure. In this scheme, each attribute authority controls a distinct attribute set and sends the attribute private key to the user. To enhance performance, their scheme employs online/offline encryption to improve online computing efficiency. It is evident that existing approaches for hiding access policies often involve increased computing overhead or the incorporation of outsourced computing servers in the calculation process.

In addition to hiding the access structure, some ABE solutions achieve privacy protection through user key tracking. Liu [17] proposed a CP-ABE scheme equipped with black-box traceability. In this scheme, the user's key accompanies all supersets of the attribute set, making it identifiable to multiple users for decryption. Subsequently, ABE with black-box traceability [18–20] has seen ongoing research on efficient tracking and revocation. Sethi [21] introduced a multi-authority CP-ABE scheme with white-box traceability, policy updates, and outsourced decryption. This scheme supports distributed authority management and accommodates monotonic access structures.

Preserving user privacy is of utmost importance, especially in sensitive application contexts like electronic health records and personal data sharing. In these situations, safeguarding the confidentiality of user attributes is imperative to mitigate the risks associated with unauthorized disclosures. In instances where the KGC acts as an honest-but-curious entity, existing methods fall short of achieving optimal outcomes—they are proficient at safeguarding against post-leakage tracking but ineffective at preempting the leakage of user privacy. Therefore, our primary focus is to investigate strategies that prevent the leakage of users' privacy to the KGC during the key generation phase, particularly when the KGC operates as an honest-but-curious entity. This approach is also aimed at safeguarding the KGC from malicious users who might attempt to traverse the entire attribute set controlled by the KGC through continuous registration and access, thereby ensuring the privacy of both parties.

Through our research, we have discovered that the oblivious transfer protocol presents a promising approach to addressing this issue. Oblivious transfer (OT) is a vital cryptographic protocol fundamental in the realm of secure multi-party computations, serving as a cornerstone for enhancing privacy and security across various cryptographic endeavors. In an oblivious transfer, two primary entities are involved: a sender possessing certain information and a receiver who wishes to acquire a segment of this information. Unique in its operation, the protocol allows the receiver to select a specific piece of information from the sender without revealing the choice. This attribute ensures the sanctity of the receiver's privacy, maintaining the confidentiality of the selected information segment. Below, we will introduce the development of oblivious transfer protocols.

Oblivious transfer is frequently employed as a crucial primitive in the design of security protocols. The OT primitive was proposed by Rabin [22]. In this scheme, the receiver can successfully decrypt the information sent by the sender with a probability of 1/2. After that, they even proposed a new 1-out-of-2 OT (OT_1^2). In this scheme, the sender sends two encrypted messages to the receiver, and the receiver can only choose one of them to successfully decrypt. Brassard [23] designed a 1-out-of-n OT (OT_1^n) based on the former, and the receiver can choose one of the n messages from the sender to decrypt. Tzeng [24] improved the efficiency of the OT_1^n by combining distributed ideas and secret sharing techniques. Moreover, k-out-n OT (OT_k^n) is a further extension of OT_1^n, where

$k < n$. Naor proposed the OT_k^n protocol [25] for the first time by using PRF. Under the premise of semi-honest receivers, the scheme mainly guarantees system security through onerous computation and communication expenses. In order to solve the above problems of high computational overhead and high communication costs, Chu [26] proposed a k-out-n OT protocol, but it does not really solve the problem or minimize these costs. Tzeng [27] proposed a OT_k^n protocol that uses two different ROMs under the Computational Diffie-Hellman Problem (CDH) assumption to keep the system secure in the presence of malicious receivers.

Compared to the solutions previously discussed, hidden access structures can mitigate attribute leakage due to unauthorized user access, but they fail to shield user privacy from the KGC. Additionally, schemes involving multiple authorization centers are susceptible to collusion attacks, providing only partial attribute concealment from the KGC without completely obscuring individual attributes. In the context of key tracking solutions, their effectiveness is predominantly in post-event accountability, falling short of proactive user privacy protection. In contrast, our proposed method leverages the oblivious transfer protocol, safeguarding user attributes during the key generation phase. This approach ensures that an honest-but-curious KGC remains unaware of the specific attributes associated with a user's private key during the key distribution process. The evolution of OT has inspired us, leading us to consider the utilization of OT_k^n for generating private keys in ABE. In simple terms, the KGC possesses n attributes. When a user requests a private key from the KGC, the user selects k attributes from the KGC. Consequently, the KGC remains unaware of the specific k attributes selected by the user, thereby achieving privacy protection for the user.

3. Preliminaries

3.1. k-Out-of-n Oblivious Transfer

The k-out-of-n oblivious transfer [25] is defined as follows: In this protocol, the sender and the receiver are generally involved. The sender is in charge of n messages $\{m_0, m_1, \cdots, m_n\}$, and the receiver has a set of k numbers $\{r_1, r_2, \cdots, r_k\} \subset \{0, 1, \cdots, n-1\}$. When the oblivious transfer protocol is completed, the receiver only holds k messages $\{m_{r_1}, m_{r_2}, \cdots, m_{r_k}\}$ without knowing anything about m_ξ, where $\xi \in \{0, 1, \cdots, n-1\}$ and $\xi \notin \{r_1, r_2, \cdots, r_k\}$, while the sender knows nothing about the message chosen by the receiver $\{r_1, r_2, \cdots, r_k\}$.

3.2. Bilinear Pairings

Bilinear mapping means a map is defined as follows: $\mathbb{G}_0$ and $\mathbb{G}_1$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}_0$ and e be a bilinear map, e: $\mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$. The bilinear map e has the following properties:

(a) Bilinearity: For all $g_1, g_2 \in \mathbb{G}_0$ and $u, v \in \mathbb{Z}_p$ has $e(g_1^u, g_2^v) = e(g_1, g_2)^{uv}$.

(b) Non-degeneracy: Given a generator g of the group $\mathbb{G}_0$, $e(g, g) \neq 1$.

3.3. Access Structure

Use a tree $\mathbb{T}$ to represent the access structure. All non-leaf nodes in the tree are represented as a threshold gate, determined by its child node and a threshold value. Define num_x as the number of children of a node x and V_x as threshold value of the node x, where $0 < V_x \leq num_x$. $V_x = 1$ it is an AND gate, case $V_x = num_x$. Let $ATT = \{att_1, att_2, \cdots, att_n\}$ be an attribute set; all leaf nodes x are represented as an attribute $att_i \in ATT$ and its threshold value $V_x = 1$. We denote $par(x)$ to represent the parent of the node x. We can define a function $att(x)$, where a node x is a leaf node associated with an attribute. Furthermore, we define the order between the children of each node in the $\mathbb{T}$ by labeling the children of each node starting from 1. The function $index(x)$ replies to the amount associated with the node x, and the value of $index(x)$ is particularly assigned to the node in $\mathbb{T}$ for a specified key.

4. Our Construction

In this area, we give the details of how to construct our scheme. We begin by explaining the system model and introducing its main algorithms and functions. Afterwards, we provide a description of our attribute-based encryption scheme with OT_k^n protocol. Finally, we will discuss the security analysis and experimentation of this scheme.

4.1. Notions

The notions utilized in this paper are enumerated in Table 1.

Table 1. Notions and definitions for our scheme.

Notions	Definition
λ	a security parameter
MPK	the system's public parameters
MSK	the master private parameters
M	a message
$\mathbb{T}$	the tree access structure
$\mathbb{A}$	the set of leaf nodes
V_x	the threshold value of the node
f_x	a polynomial equation
CT	a ciphertext with access structure $\mathbb{T}$
S	a user's attributes
PK	a user's public key
SK	a user's private key
$\mathbb{G}$	a cyclic additive group of prime order ℓ
$\mathbb{G}_T$	a multiplicative group
e: $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$	the bilinear map
g	a generator of $\mathbb{G}$
ATT	the whole attribute set
att_i	the ith attribute of ATT
n	the number of whole attributes in the system
k	the number of users' attributes

4.2. System Model

This paper proposes an attribute-based encryption with an oblivious transfer protocol, which mainly includes four parts: KGC, cloud storage server, data owner, and user. This scheme alleviates the problem of attribute privacy protection between the KGC and the user through the OT_k^n protocol. The system model of CP-ABE with OT_k^n is shown in Figure 1. The scheme proposed in this paper includes the following four stages:

Setup $(1^\lambda) \rightarrow (MPK,\ MSK)$: The Setup algorithm is run by KGC. Input the security parameter λ, and the algorithm outputs the system public parameter MPK and master private key MSK. KGC publicizes MPK and keeps MSK secret.

Encrypt $(MPK, M, \mathbb{T}) \rightarrow CT$: The encryption algorithm is run by the data owner. Input a system public parameter MPK, a message M, and the tree access structure $\mathbb{T}$, and the algorithm outputs ciphertext CT.

KeyGen $(MPK, S) \rightarrow SK$: The KeyGen algorithm is run by KGC and the user. Input a system public parameter MPK and the user's attributes S, and the algorithm outputs the user's private key SK.

Decrypt $(CT, SK) \to M$: The decryption algorithm is run by the user. Input ciphertext CT and the user's private key SK, and the algorithm outputs the message M.

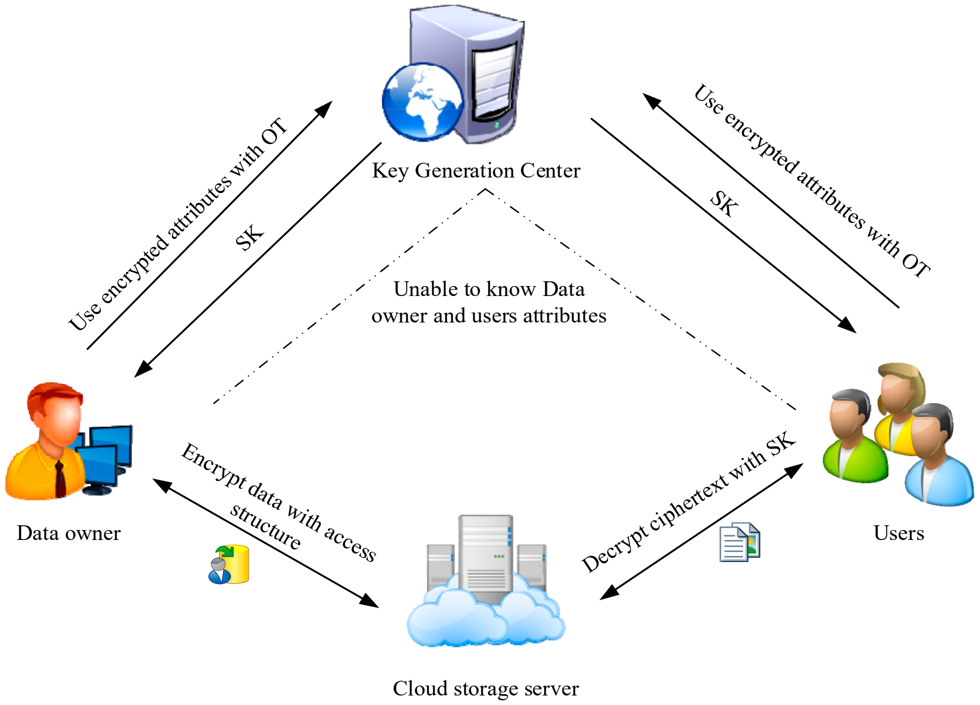

Key Generation Center
Use encrypted attributes with OT
SK
Use encrypted attributes with OT
SK
Unable to know Data owner and users attributes
Data owner
Users
Encrypt data with access structure
Decrypt ciphertext with SK
Cloud storage server

Figure 1. System model of CP-ABE with k-out-of-n oblivious transfer.

4.3. Proposed Scheme

Setup $(1^{\lambda}) \to (MPK,\ MSK)$: The setup algorithm is run by KGC. Let $\mathbb{G}$ be a cyclic additive group of prime order ℓ, and let g be a generator of $\mathbb{G}$. In addition, let e: $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ denote the bilinear map, and let $\mathbb{G}_T = e(g, g)$ be a multiplicative group. Taking as input a security parameter λ and an attribute set $ATT = \{att_1, att_2, \cdots, att_n\}$ and supposing the attribute att_1 is mapped to index i for all $i = 1, 2, \cdots, n$. We will employ the hash functions H: $\{0, 1\}^* \to \{0, 1\}^l$, $H_1 : \{0, 1\}^* \times \mathbb{G} \to \mathbb{Z}_q/\ell$, $H_2 : \{0, 1\}^* \to \mathbb{G}$, $H_3 : \mathbb{G} \to \{0, 1\}^l$ that we would model as a random oracle. The construction is as follows:

(1) Choose a random $s \in \mathbb{Z}_q$ and calculate $P_{pub} = g^s$. Pick two random exponents $a, b \in \mathbb{Z}_q$, and compute $P = g^b$.

(2) The public parameters are published as (MPK, MSK), where $MPK = (\mathbb{G}, \mathbb{G}_T, l, g, P_{pub}, P, e(g, g)^a, H, H_1, H_2, H_3)$ and $MSK = (s, b, g^a)$.

Encrypt $(MPK, M, \mathbb{T}) \to CT$: The algorithm of encryption is run by the data owner. Our encryption is based on the Bethencourt approach [2]. It uses the tree access structure $\mathbb{T}$ to encrypt the message M. The details are as follows:

(1) Choose a polynomial f_x for each node or leaf x in the tree $\mathbb{T}$. For each node x in the $\mathbb{T}$, set the degree D_x of the polynomial f_x to be one less than the threshold value V_x of that node, that is, $D_x = V_x - 1$. We use a top-down approach to pick these polynomials, and it begins with the root node R. First, the algorithm randomly chooses a $b \in \mathbb{Z}_q$ and initializes $f_R(0) = c$. Then, it chooses D_R other points of the polynomial f_R randomly to define it entirely. For another node x, it sets $f_x(0) = f_{Par(x)}(index(x))$ and chooses D_R other points randomly to entirely define f_x.

(2) On input the set of leaf nodes $\mathbb{A} \subseteq \mathbb{T}$, then compute $\forall i \in \mathbb{A} : C_{1,i} = g^{f_i(0)}$, $C_{2,i} = H_2(att(i))^{f_i(0)}$, $C_3 = P^c$. Inputting a message M, compute $C = Me(g, g)^{ac}$.

The data owner outputs ciphertext $CT = (\mathbb{T}, C, C_{1,i}, C_{2,i}, C_3)$.

KeyGen$(MPK, S) \to SK$. The algorithm for key generation is run by the user and KGC. Users give the attribute $S \subseteq ATT$ to KGC, and KGC outputs the key associated with that S.

To prevent KGC from learning the key consistent with a set of attributes S, we combine the idea of k-out-of-n oblivious transfer protocol. The details are as follows:

(1) KGC uses its ID_{kgc} to compute $OT.PK_{kgc} = H_2\left(ID_{kgc}\right)$, $OT.SK_{kgc} = sH_2\left(ID_{kgc}\right)$. Then, on input user ID, compute $OT.PK_{ID} = H_2(ID)$, $OT.SK_{ID} = sH_2(ID)$ reply for the user.

(2) The user sets γ_j to denote the number of these attributes S according to $ATT = \{att_1, att_2, \cdots, att_n\}$, and randomly chooses $\alpha, \beta \in \mathbb{Z}_q$. Then computes $K = \alpha\beta OT.PK_{ID}$, $K_j = H(\gamma_j)^{\beta} OT.SK_{ID}$, where $j = 1, 2, \cdots, k$ and $\gamma_j \in \{1, 2, \cdots, n\}$. Afterwards, the user randomly chooses $\eta \in \mathbb{Z}_q$ and computes $\rho_{id} = H_3(ID, K, K_1, K_2, \cdots, K_k)$. Then the user computes a signature $\sigma_{id} = (U_{id}, V_{id})$, where $U_{id} = \eta OT.PK_{ID}$, $V_{id} = (\eta + h_{id})OT.SK_{ID}$, and $h_{id} = H_1(\rho_{id}, U_{id})$. Finally, user output $M_{id} = \{ID, K, K_1, K_2, \cdots, K_k, \sigma_{id}\}$.

(3) When KGC receives the M_{id}, it first computes $\varpi_{id} = H_3(ID_r, K, K_1, K_2, \cdots, K_k)$, verifying $(P, V_{id}) = e\left(P_{pub}, U_{id} + \varpi_{id} OT.PK_{ID}\right)$. If it is false, output $\perp$; otherwise, randomly chooses $\varphi \in \mathbb{Z}_q$ and computes $\mathrm{A}_1 = \varphi K_1, \mathrm{A}_2 = \varphi K_2, \cdots, \mathrm{A}_k = \varphi K_k$.

(4) Afterwards, KGC randomly chooses a $r \in \mathbb{Z}_p$, and then randomly chooses $r_n \in \mathbb{Z}_p$ for each attribute $ATT = \{att_1, att_2, \cdots, att_n\}$. Then it computes the key message as $M_{key} = (SK_1, M.SK_{2,n})$, where $SK_1 = g^{(a+r)/b}$, $M.SK_{2,n} = e\left(H(\psi_n)K, OT.SK_{kgc}\right)^{\varphi} \oplus (D_{1,n}, D_{2,n})$. Moreover, $D_{1,j} = g^r \cdot H_2(n)^{r_n}$, $D_{2,j} = g^{r_n}$, and ψ_n is a number from 1 to n in order.

(5) KGC randomly chooses $\xi \in \mathbb{Z}_p$ and computes $\rho_{kgc} = H_3\left(A_1, A_2, \cdots, A_k, M_{key}\right)$ and outputs signature $\sigma_{kgc} = \left(U_{kgc}, V_{kgc}\right)$, where $U_{kgc} = \xi \cdot OT.PK_{kgc}$, $V_{kgc} = \left(\xi + h_{kgc}\right) OT.SK_{kgc}$ and $h_{kgc} = H_1\left(\rho_{kgc}, U_{kgc}\right)$. Finally, KGC outputs $M_{kgc} = \{ID_{kgc}, A_1, A_2, \cdots, A_k, M_{key}, \sigma_{kgc}\}$.

(6) When the user receives the M_{kgc}, it first computes $\varpi_{kgc} = H_3(ID_{kgc}, A_1, A_2, \cdots, A_k, M_{key})$, verifying $e\left(P, V_{kgc}\right) = e\left(P_{pub}, U_{kgc} + \varpi_{kgc} OT.PK_{kgc}\right)$. If it is false, output $\perp$; otherwise, compute $SK_{2,t} = M.SK_{2,n} \oplus e\left(A_t, OT.PK_{kgc}\right)^{\alpha}, t \in \{1, 2, \cdots, k\}$. Finally, the user obtains $SK = (SK_1, SK_{2,t})$.

Decrypt $(CT, SK) \rightarrow M$: The decryption procedure is run by the user. We define the following recursive algorithm:

(1) If the node x is a leaf node, we can let $w = att(x)$ and define it as follows: If $w \in ATT = \{att_1, att_2, \cdots, att_n\}$, the user executes the recursive algorithm $Dec(CT, SK, x) = \frac{e\left(D_{1,j}, C_{1,i}\right)}{e\left(D_{2,j}, C_{2,i}\right)}$, otherwise $Dec(CT, SK, x)$ output $\perp$.

(2) If the node x is a non-leaf node, for all nodes ω that are children of x, it calls $Dec(CT, SK, \omega)$ and stores the output as F_ω. Let ATT_x be an arbitrary k_x-sized set of child nodes ω such that $F_\omega \neq \perp$. If no such set exists, then the node was not satisfied, and the function returned $\perp$. Otherwise, the user computes:

$$
\begin{aligned}
F_\omega &= \prod_{\omega \in ATT_x} F_\omega^{\Delta_{i,ATT'x}(0)} \\
&= \prod_{\omega \in ATT_x} \left(e(g,g)^{r \cdot f_\omega(0)}\right)^{\Delta_{i,ATT'x}(0)} \\
&= \prod_{\omega \in ATT_x} \left(e(g,g)^{r \cdot f_x(i)}\right)^{\Delta_{i,ATT'x}(0)} \\
&= e(g,g)^{r \cdot f_x(0)}, \textit{ where } ATT'_x = \{index(\omega) : \omega \in ATT_x\},\ i = \mathrm{index}(\omega)
\end{aligned}
$$

(3) If the tree is satisfied by $ATT = \{att_1, att_2, \cdots, att_n\}$, user set $A = Dec(CT, SK, r) = e(g,g)^{rf_R(0)} = e(g,g)^{rc}$, and computes $C/(e(C_3, SK_1)/A)$. If it is false, output $\perp$; otherwise, output M as the decryption of the ciphertext.

Figure 2 is a schematic diagram of the algorithm steps and data flow of our solution.

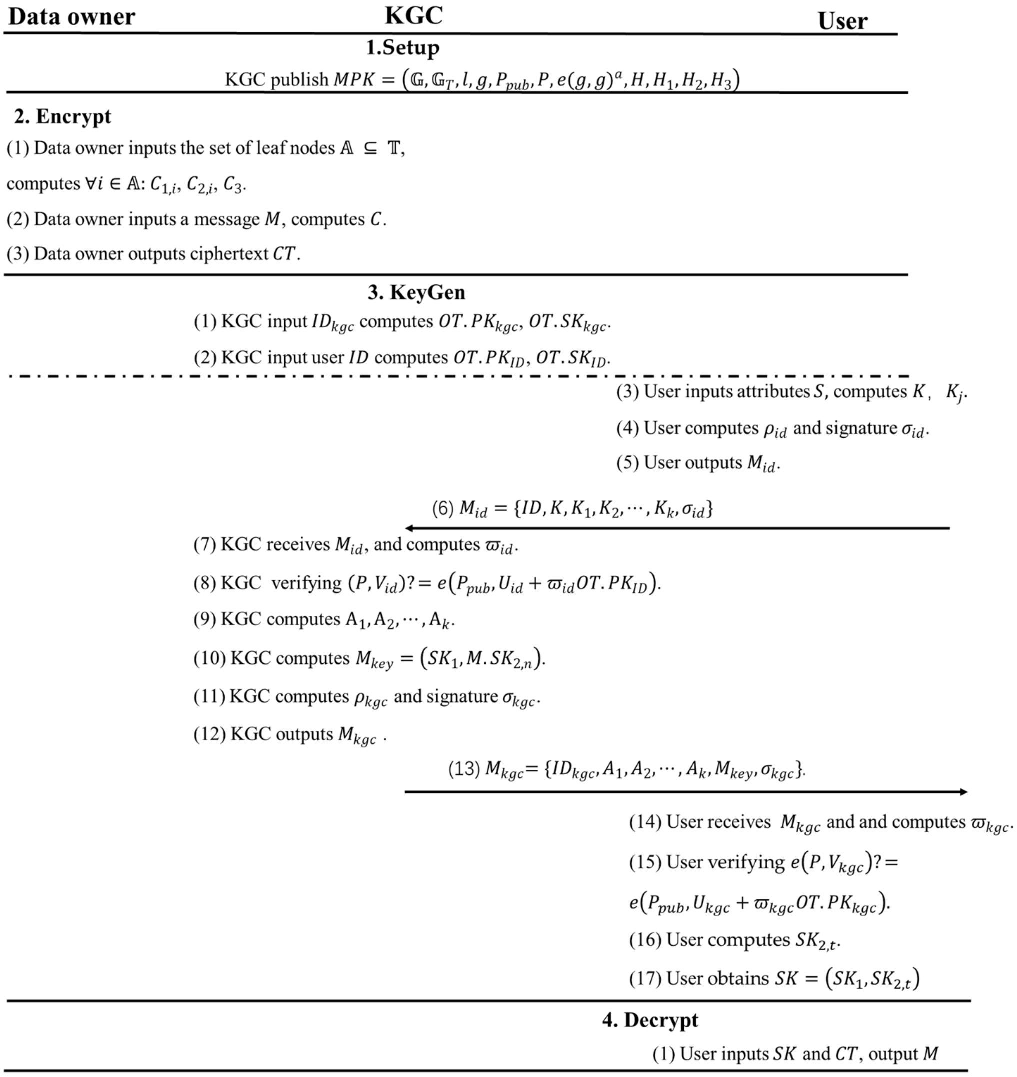

Data owner
KGC
User
1.Setup
KGC publish $MPK = (\mathbb{G}, \mathbb{G}_T, l, g, P_{pub}, P, e(g,g)^a, H, H_1, H_2, H_3)$
2. Encrypt
(1) Data owner inputs the set of leaf nodes $\mathbb{A} \subseteq \mathbb{T}$,
computes $\forall i \in \mathbb{A}$: $C_{1,i}$, $C_{2,i}$, C_3.
(2) Data owner inputs a message M, computes C.
(3) Data owner outputs ciphertext CT.
3. KeyGen
(1) KGC input ID_{kgc} computes $OT.PK_{kgc}$, $OT.SK_{kgc}$.
(2) KGC input user ID computes $OT.PK_{ID}$, $OT.SK_{ID}$.
(3) User inputs attributes S, computes K, K_j.
(4) User computes ρ_{id} and signature σ_{id}.
(5) User outputs M_{id}.
(6) $M_{id} = \{ID, K, K_1, K_2, \cdots, K_k, \sigma_{id}\}$
(7) KGC receives M_{id}, and computes ϖ_{id}.
(8) KGC verifying $(P, V_{id})? = e(P_{pub}, U_{id} + \varpi_{id} OT.PK_{ID})$.
(9) KGC computes $A_1, A_2, \cdots, A_k$.
(10) KGC computes $M_{key} = (SK_1, M.SK_{2,n})$.
(11) KGC computes ρ_{kgc} and signature σ_{kgc}.
(12) KGC outputs M_{kgc} .
(13) M_{kgc}= $\{ID_{kgc}, A_1, A_2, \cdots, A_k, M_{key}, \sigma_{kgc}\}$.
(14) User receives M_{kgc} and and computes ϖ_{kgc}.
(15) User verifying $e(P, V_{kgc})? =$
$e(P_{pub}, U_{kgc} + \varpi_{kgc} OT.PK_{kgc})$.
(16) User computes $SK_{2,t}$.
(17) User obtains $SK = (SK_1, SK_{2,t})$
4. Decrypt
(1) User inputs SK and CT, output M

Figure 2. Our scheme flow chart.

5. Security Analysis

In this area, we analyze the security of this protocol and prove that the protocol can achieve KGC's privacy and the user's privacy protection.

Security Assumptions. For our attribute-based encryption scheme with the OT_k^n protocol against malicious users, we will use two hardness problem assumptions: one is the Decisional Diffie-Hellman (DDH) problem, and the other is the Chosen-Target Computational Diffie-Hellman (CT-CDH) problem.

Assumption 1. *Decisional Diffie-Hellman assumption. Let $p = 2q + 1$, where p and q are two primes, and let $\mathbb{G}_p$ be the subgroup of $\mathbb{Z}_q$ with order p. The following two distribution ensembles are computationally indistinguishable:*

$Y_1 = \left\{\left(g, g^a, g^b, g^{ab}\right)\right\}_{\mathbb{G}_p}$, where g is a generator of $\mathbb{G}_p$ and randomly chooses a, $b \in \mathbb{Z}_q$.

$Y_2 = \left\{ \left(g, g^a, g^b, g^c \right) \right\}_{\mathbb{G}_p}$, where g is a generator of $\mathbb{G}_p$ and randomly chooses a, b, $c \in \mathbb{Z}_q$.

Assumption 2. *Chosen-Target Computational Diffie-Hellman assumption. Let $\mathbb{G}_p$ be a group of prime order q, g be a generator of $\mathbb{G}_p$, and randomly choose $x \in \mathbb{Z}_q$. Let $H_1 : \{0,\ 1\}^* \to \mathbb{G}_p$ be a cryptographic hash function. The adversary A is given input (q, g, g^x, H_1) and two oracles: the target oracle $T_G(\cdot)$ that returns a random element $w_i \in \mathbb{G}_p$ at the i-th query and the helper oracle $H_G(\cdot)$ that returns $(\cdot)^*$. Let q_T and q_H be the number of queries A made to the target oracle and helper oracle, respectively. The probability that A outputs k pairs $((v_1, j_1), (v_2, j_2), \cdots, (v_k, j_k))$, where $v_i = \left(w_{j_i}\right)^x$ for $i \in \{1,\ 2,\ \cdots,\ k\}$, $q_H \le k \le q_T$, is negligible.*

Theorem 1. *The proposed protocol can realize the protection of users' privacy.*

Proof. During the key distribution process, the user selected the number of some attributes $\gamma_j \in \{1,\ 2,\ \cdots,\ n\}$ from the all attribute collection $ATT = \{att_1,\ att_2,\ \cdots,\ att_n\}$. First, the user hashed and randomized γ_j by H and β, and output $K_j = H(\gamma_j)^{\beta} OT.SK_{ID}$. We maintain that the choice γ_j can only be known by the user themselves and not by anyone else. Due to the computational difficulty of the DDH problem, even if the adversary has the ability to obtain the user's private key $OT.SK_{ID}$, they would still be unable to obtain $H(\gamma_j)^{\ \beta}$ from K_j. In other words, it is impossible for adversary A to determine γ_j as they are unable to compute $H(\gamma_j)^{\ \beta}$ and therefore cannot obtain any information about it. Let $\mathcal{A} = \left\{ (\gamma_j, \beta) \in \mathbb{Z}_q * \mathbb{Z}_n \;\middle|\; H(\gamma_j)^{\ \beta} OT.SK_{ID} = K_j \right\}$; that is to say, all the possible pairs (β, γ_j) satisfying the equation $H(\gamma_j)^{\ \beta} OT.SK_{ID} = K_j$ together constitute $\mathcal{A}$. Given a value K_j and a fixed value of $OT.SK_{ID}$, there exists only one unique value of $H(\gamma_j)^{\ \beta}$ that satisfies the equation. From the definition of a hash function, we know that if a specific value of $H(\gamma_j)^{\ \beta}$ is given, then it is possible to uniquely determine the corresponding value of γ_j and β. There is a one-to-one correspondence between γ_j and β. Given this fact, we can observe that there are $n(\beta, \gamma_j)$ pairs in $\mathcal{A}$, with the dimension of γ_j being n. Specifically, $\Pr[\gamma_j | K_j] = \Pr[\gamma_j] = 1/n$; this means that, upon seeing a particular K_j, there is no way to reveal the user's choice γ_j other than guesswork. Therefore, the proposed protocol has the ability to protect users' privacy. □

Theorem 2. *The proposed protocol can realize the protection of KGC's privacy.*

Proof. We can prove that under Arguments 1 and 2, it is computationally impossible for the malicious user U^* to obtain the $(k+1)$th message. Specifically, for argument (1), U^* should pursue the scheme steps to generate the values of K and $kK_j s$; on the contrary, U^* fails to get the k selected messages that it intended. In arguments (2), we will prove that U^* cannot obtain the $(k+1)$th messages other than his choice, because when he tries to obtain the $(k+1)$th messages, he is actually solving the difficult problem of the CT-CDH problem. □

Argument 1. *U^*must comply with the scheme to calculate the values of $K(= \alpha\beta \cdot OT.PK^*_{ID})$ and $K_j\left(= H(\gamma_j)^{\beta} \cdot OT.SK^*_{ID}\right)$, for $j = 1$ to k; if not, U^* cannot receive the k messages that it has chosen.*

Next, we will discuss in detail three cases: (a) U^* fakes K but makes K_j honest; (b) U^* counterfeits K_j but honestly generates K; and (c) U^* forges the values of K and K_j.

(a) U^* fakes K but makes K_j honestly. Suppose U^* is dishonest in calculating K, but honestly calculating K_j as given in the scheme. Let us suppose the U^* computes $K_j = H(\gamma_j)^{\beta} \cdot OT.SK^*_{ID}$ and chooses an $X \in \mathbb{G}$ at random to replace K. Then, the KGC will compute $\mathrm{A}_k = \varphi K_k$, $M.SK_{2,n} = e\left(H(\psi_n)X, OT.SK_{kgc}\right)^{\varphi} \bigoplus (D_{1,n}, D_{2,n})$ and return them to U^*. In consequence, U^* is unable to decrypt $M.SK_{2,n}$ to receive the k messages since

$e\left(A_t, OT.PK_{kgc}\right)^{\alpha}$ is certainly not equal to $e\left(H(\psi_n)X, OT.SK_{kgc}\right)^{\varphi}$. For obtaining the k messages, the U^* can only compute $e\left(H(i)X, OT.SK_{kgc}\right)^{\varphi}$ equal to $\left(A_t, OT.PK_{kgc}\right)^{\alpha}$ by obtaining KGC's private key $OT.SK_{kgc}$ and one-time secrecy φ. However, this is computationally infeasible because extracting φ from A_k is a DDH problem.

(b) U^* fakes sK_j but forms K honestly. Suppose U^* is dishonest in calculating sK_j, and honestly generates K as given in the scheme. Let us suppose, the U^* computes $K = \alpha\beta \cdot OT.PK^*_{ID}$, and chooses $X_j \in \mathbb{G}$ at random to replace K. Then, the KGC will compute $\mathrm{A}_k = \varphi K_k$, $M.SK_{2,n} = e\left(H(i)K, OT.SK_{kgc}\right)^{\varphi} \bigoplus (D_{1,n}, D_{2,n})$, for $i = 1$ to n, and return them to U^*. In consequence, U^* unable decrypt $M.SK_{2,n}$ since $e(A_t, OT.PK_{kgc})^{\alpha}$= $e\left(\varphi X_j, OT.PK_{kgc}\right)^{\alpha}$ is certainly not equal to $e\left(H(i)K, OT.SK_{kgc}\right)^{\varphi}$. For obtaining the k messages, the U^* can only compute $e(H(i)K, OT.SK_{kgc})^{\varphi} (= e(H(i)\alpha\beta OT.PK^*_{ID}, OT.SK_{kgc})^{\varphi})$ equal to $\left(A_t, OT.PK_{kgc}\right)^{\alpha}$ by obtaining KGC's private key $OT.SK_{kgc}$ and one-time secrecy φ. However, this is computationally infeasible because extracting φ from A_k is a DDH problem.

(c) U^* fakes both the values of K and K_j. Let us suppose the U^* chooses $X \in \mathbb{G}$ at random to replace K and fakes K_j as $H(\gamma_j)X$. Under the assumption, the value of $\mathrm{A}_k = \varphi K_k = \varphi H(\gamma_j)X$ is calculated by the sender as well as the ciphertexts $M.SK_{2,n} = e\left(H(\gamma_j)X, OT.SK_{kgc}\right)^{\varphi} \bigoplus (D_{1,n}, D_{2,n})$ for $j = 1$ to k. Although U^* is aware of the value of $\varphi H(\gamma_j)X$ (because it is exactly equal to the A_k obtained from KGC), it still cannot calculate $e\left(\varphi H(\gamma_j)X, OT.SK_{kgc}\right)$ in the absence of knowledge of $OT.SK_{kgc}$. According to the above description, we know that when K is X and K_k is $H(\gamma_j)X$, U^* cannot get $M.SK_{2,n}$. In addition, U^* probably sets K_k as $H(\gamma_j)Y$, where $Y(\neq X)$ is a random value in $\mathbb{G}$. In conclusion, under the violation of calculating the values of K and K_k, U^* was unable to acquire the k chosen messages.

Argument 2. *If U^* accompanies the scheme truthfully to get k messages, though it wants to process the $(k+1)$th message, afterwards it would confront the tough CT-CDH problem with the assumption of a random oracle.*

The U^* intends to get messages means U^* would possess the awareness of $e(H(i)K, OT.SK_{kgc})^{\varphi}\left(= e\left(A_t, OT.PK_{kgc}\right)^{\alpha}\right)$, in fact, according to argument (1), an honest user U should have knowledge of k values, where $e\left(H(i)K, OT.SK_{kgc}\right)^{\varphi}$, for $i = 1$ to n, whereas $e\left(H(i)K, OT.SK_{kgc}\right)^{\varphi} = e\left(A_t, OT.PK_{kgc}\right)^{\alpha}$, for $t = \gamma_j$ and j = 1 to k. Let suppose $y^{(i)} \in \mathbb{G}_T$ and $e\left(H(i)K, OT.SK_{kgc}\right)^{\varphi} = y^{(i)}$. In consonance with argument (1), for acquiring the k selected message, U^* is unable to modify the structures of $K\left(= \alpha\beta \cdot OT.PK^*_{ID}\right)$ and $K_k = H(\gamma_j)^{\beta} \cdot OT.SK^*_{ID}$. In these conditions $y^{(i)}$ can only be decomposed into $y^{(i)} = e(H(i)\alpha\beta \cdot OT.PK^*_{ID}, OT.SK_{kgc})^{\varphi}$ = $e\left(\alpha\beta H(i) \cdot OT.SK^*_{ID}, OT.PK_{kgc}\right)^{\varphi}$ since $OT.SK_{kgc} = s \cdot OT.PK_{kgc}$ and $OT.SK^*_{ID} = s \cdot OT.PK^*_{ID}$. Furthermore, under the assumption of random oracle and the fact that U^* is able to learn the $\alpha, \beta, OT.SK^*_{ID}$ and $OT.PK_{kgc}$, $y^{(i)}$ could be expressed as $(g_i)^{\varphi}$, where g_i = $e\left(\alpha\beta H(i) \cdot OT.SK^*_{ID}, OT.PK_{kgc}\right)$ and $\varphi \in \mathbb{G}_T$ is a random element. Thereafter, the malicious U^* actually encounters the determination of the $(k+1)$th pair $\left(\gamma_{k+1}, (g_{\gamma_{k+1}})^{\varphi}\right)$ with the awareness of k pairs of $(\gamma_1, (g_{\gamma_1})^{\varphi})$, $(\gamma_2, (g_{\gamma_2})^{\varphi})$, $\cdots$, $\left(\gamma_k, (g_{\gamma_k})^{\varphi}\right)$, where $\left(g_{\gamma_j}\right)^{\varphi} = e\left(A_t, OT.PK_{kgc}\right)^{\alpha}$, but without the awareness of KGC's secrecy φ (because it is DDH difficult problem for calculating φ from $A_t (= \varphi K_k)$. Consequently, the user was unable to get the $(k+1)$th message.

In accordance with Arguments 1 and 2, we have proven Theorem 2 that our scheme is able to realize the protection of KGC's privacy.

6. Experiment and Evaluation

In this part, we will verify the effectiveness of this scheme with respect to theoretical examination and experimental verification.

Theoretical examination: To be fair, we only consider the adopt tree structure CP-ABE scheme. Table 2 shows the comparison of the properties between the schemes. From Table 2, we can know that our solution is aimed at protecting the user's attribute privacy from being known by KGC under the condition that KGC is honest and curious. At the same time, our solution does not require multiple authorization centers or additional outsourced calculations. In Tables 3 and 4, we conduct theoretical analysis from two aspects of computing overhead and storage overhead for the preferred scheme and our scheme. The storage overhead is mainly for the amount of PK, SK, and CT, and the computing overhead is basically for the time cost of KeyGen, encryption, and decryption. The PK refers to the size of the user's public key. The SK means the size of the user's private key. The CT means the size of the ciphertext. Expand in detail; suppose the access structure $\mathbb{T}$ contains k-level nodes. Let $|\mathbb{T}_R|$ and $|\mathbb{T}_i|$ express the complete amount of the leaf nodes in $\mathbb{T}$ as well as in the subtree rooted at level node V_x in $\mathbb{T}$ individually. The $|\mathbb{G}|$ and $|\mathbb{G}_T|$ mean the length of one element in $\mathbb{G}$, $\mathbb{G}_T$; the $|S|$ means the groups of attributes; and the n means the number of attributes. The E_G, E_T means an exponentiation operation time expense in $\mathbb{G}$, $\mathbb{G}_T$; the P means a pairing computation time expense.

Table 2. Properties comparison.

Scheme	Type of Hiding	Multiple Authority	KGC Model	User Attributes Protection
Efficient CP-ABE [28]	Partially hidden	N	honest	N
Multiauthority CP-ABE [29]	Partially hidden	Y	honest	N
Privacy-preserving and efficient CP-ABE [30]	Fully hidden	N	honest and curious	N
Our Scheme	Fully hidden	N	honest and curious	Y

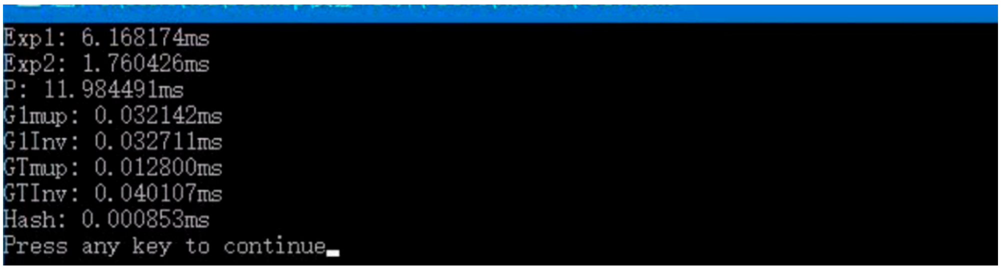
Table 3. Storage overhead comparison of different schemes.

Scheme	PK	SK	CT
Efficient CP-ABE [28]	$6\|\mathbb{G}\|+\|\mathbb{G}_T\|$	$(2\|S\|+1)\|\mathbb{G}\|$	$(2\|\mathbb{T}_R\|+n)\|\mathbb{G}\|+\|\mathbb{G}_T\|$
Multiauthority CP-ABE [29]	$7\|\mathbb{G}\|+\|\mathbb{G}_T\|$	$(n+\|S\|)\|\mathbb{G}\|$	$(2n+2\|\mathbb{T}_R\|)$ $\|\mathbb{G}\|+2\|\mathbb{G}_T\|$
Privacy-preserving and efficient CP-ABE [30]	$7\|\mathbb{G}\|+2\|\mathbb{G}_T\|$	$(\|S\|+1)\|\mathbb{G}\|$	$(2n+1+2\|\mathbb{T}_R\|)\|\mathbb{G}\|$ $+\|\mathbb{G}_T\|$
Our Scheme	$8\|\mathbb{G}\|+2\|\mathbb{G}_T\|$	$(2n+\|S\|+1)\|\mathbb{G}\|$ $+n\|\mathbb{G}_T\|$	$(2n+\|\mathbb{T}_R\|+1)\|\mathbb{G}\|$ $+\|\mathbb{G}_T\|$

Experimental verification: In order to verify the results of our above theoretical analysis, based on the PBC library [31], we simulated and implemented the schemes in [28–30] and our system, respectively. Specifically, we experimented on our MacBook Air, whose CPU has an Intel Core i5 (1.1 GHz), 8 GB of RAM, and runs Ventura 13.3. For the purpose of the 80-bit security level target, our scheme adopted the super-singular curve $y^2 = x^3 + x$ over a 512-bit finite field to design a 160-bit elliptic curve group to simulate running these schemes. In these figures, the units of computation cost are milliseconds, while the total of the execution times of all algorithms is considered the total execution time. The experimental verification is conducted using the PBC library to implement the cryptographic computation code. The experimental process mainly entails implementing the cryptographic formulas and computations involved in the discussed schemes through the C program. Figure 3 below depicts the actual computation time derived from running the code on our computer. We precisely conducted the experimental verification of our scheme and the selected comparative schemes on the same platform and library.

Table 4. Computation efficiency comparison of different schemes.

Scheme	KeyGen	Encryption	Decryption
Efficient CP-ABE [28]	$(3n+2)E_G$	$(n+2\|\mathbb{T}_R\|)E_G$+E_T	$(n+2\|\mathbb{T}_R\|)$ P+$\|\mathbb{T}_R\|E_T$
Multiauthority CP-ABE [29]	$(n+2)$ E_G	$(4n+2+2\|\mathbb{T}_R\|)E_G$ $+2P+2E_T$	$(4n+2)P$+ 2 $\|\mathbb{T}_R\|E_T$
Privacy-preserving and efficient CP-ABE [30]	$(2n+1)$ E_G	$(2n+\|\mathbb{T}_R\|+1)E_G$ $+$ E_T2P	$(2n$ $+\|\mathbb{T}_R\|+1)P$ $+nE_T$
Our Scheme	$(2n+1)E_G+nP$ $+$ nE_T	$(2n+\|\mathbb{T}_R\|+1)E_G$ $+E_T+P$	$(2n$ $+\|\mathbb{T}_R\|)P$ $+(2n+\|\mathbb{T}_R\|)E_T$



```
Exp1: 6.168174ms
Exp2: 1.760426ms
P: 11.984491ms
G1mup: 0.032142ms
G1Inv: 0.032711ms
GTmup: 0.012800ms
GTInv: 0.040107ms
Hash: 0.000853ms
Press any key to continue
```

Figure 3. Cryptography computation runtime overhead.

Figure 4 mainly presents the trend of computing time for key generation as the attribute increases. Due to the oblivious transfer protocol, our scheme has additional overhead in the key generation stage, but the added overhead is still acceptable. Figure 5 shows the computation time required for encryption as the attribute increases. In the encryption stage, our overhead is basically the same as other tree structures in the CP-ABE scheme. Figure 6 demonstrates the relationship between the computation time of decrypting overhead and the number of attributes. As with other schemes, the computational overhead in the decryption stage increases with the number of attributes. It has been proven by experiments that the addition of the *k*-out-of-*n* oblivious transfer protocol will not significantly affect the performance of the scheme under the condition of protecting user privacy.

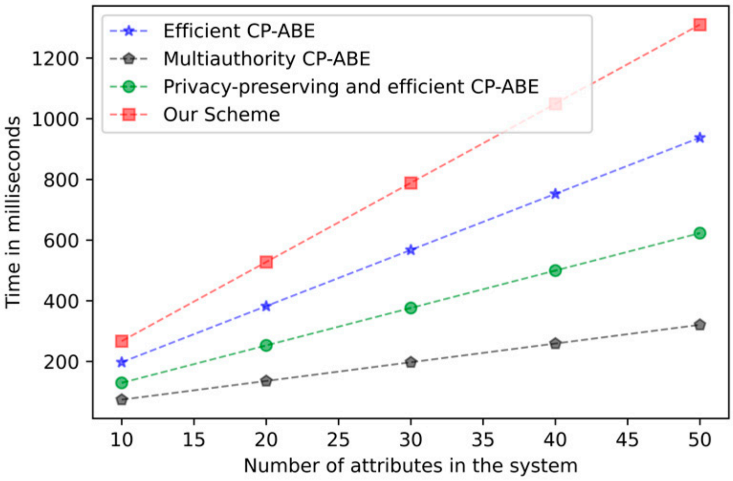


Figure 4. Key generation computation time comparison.

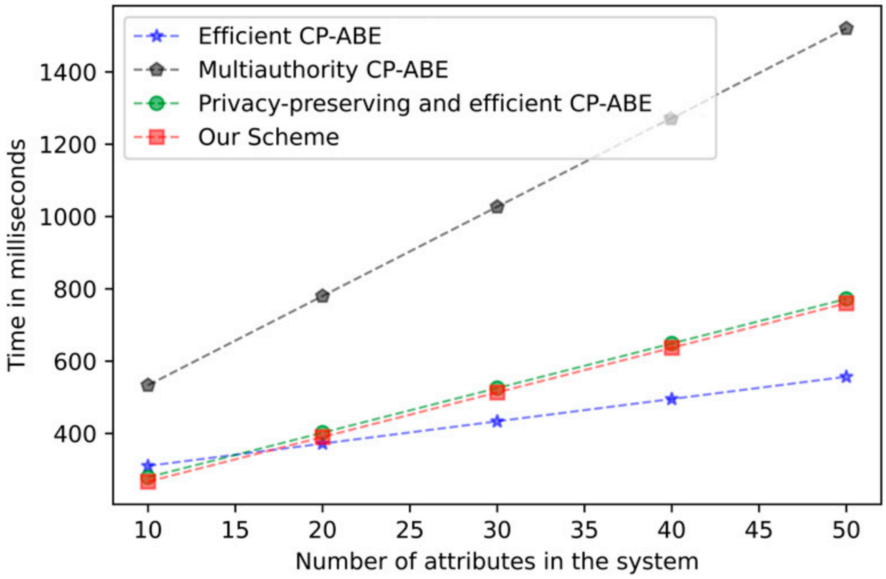


Figure 5. Encryption computation time comparison.

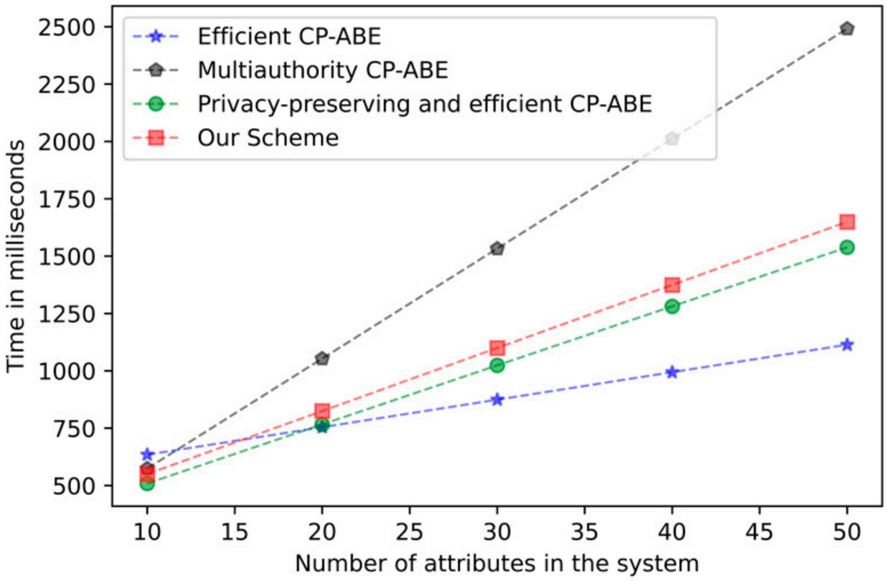


Figure 6. Decryption computation time comparison.

7. Conclusions

Attribute-based encryption offers an ideal solution for flexible data sharing, yet the credibility of the KGC within the attribute encryption scheme is pivotal for user confidence. In this paper, we introduce an innovative scheme that combines a OT_k^n protocol with a CP-ABE scheme. During the computation of the user's public-private key pair, the KGC employs the OT_k^n protocol to hide the user's essential attributes, preventing the leakage of user privacy. Distinguished from other solutions, our approach primarily addresses the challenge of safeguarding user attribute privacy, assuming that the KGC operates as an honest curiosity model. This allows the KGC to generate private keys for users without knowledge of the specific attributes associated with each user. In the security analysis of this scheme, we specifically examined two situations: (1) the protection of the user's privacy; and (2) the protection of KGC's privacy. By utilizing the DDH and CT-CDH assumptions, we demonstrated that the scheme effectively safeguards user privacy from disclosure. Furthermore, we conducted a performance comparison of this scheme with other CP-ABE schemes of the same type. After incorporating the OT_k^n protocol, the computation time overhead for key generation, encryption, and decryption within the scheme did not experience a significant increase. Therefore, we are confident that this concept can provide substantial support for the wider adoption of attribute encryption in the future. While our current scheme is suitable for tree access structures, we recognize the flexibility and diversity of access control structures in attribute encryption. Our future research aims to develop a general method to utilize the oblivious transfer protocol with any access structure.

Author Contributions: Software, J.M.; Formal analysis, X.W.; Resources, K.W.; Writing—original draft, H.Z.; Writing—review & editing, Y.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by the National Natural Science Foundation of China under Grants U20B2049 and U20B2046 and in part by the Key Research and Development Project of Sichuan Province of China under Grant 2022YFG0172; the Sichuan Science and Technology Program under Grant 2023YFG0146; the Foundation of Science and Technology on Communication Security Laboratory under Grant 61421030209012105; and the Stability Program of Science and Technology on Communication Security Laboratory (2002).

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

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

References

1. Sahai, A.; Waters, B. Fuzzy identity-based encryption. In Proceedings of the Advances in Cryptology–EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, 22–26 May 2005; Proceedings 24. Springer: Berlin/Heidelberg, Germany, 2005; pp. 457–473.
2. 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.
3. 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.
4. Han, D.; Pan, N.; Li, K.C. A traceable and revocable ciphertext-policy attribute-based encryption scheme based on privacy protection. *IEEE Trans. Dependable Secur. Comput.* **2020**, *19*, 316–327. [CrossRef]
5. Cui, H.; Deng, R.H.; Qin, B.; Weng, J. Key regeneration-free ciphertext-policy attribute-based encryption and its application. *Inf. Sci.* **2020**, *517*, 217–229. [CrossRef]
6. Sowjanya, K.; Dasgupta, M. A ciphertext-policy Attribute based encryption scheme for wireless body area networks based on ECC. *J. Inf. Secur. Appl.* **2020**, *54*, 102559. [CrossRef]
7. Zhang, Z.; Zhang, J.; Yuan, Y.; Li, Z. An expressive fully policy-hidden ciphertext policy attribute-based encryption scheme with credible verification based on blockchain. *IEEE Internet Things J.* **2021**, *9*, 8681–8692. [CrossRef]
8. Hu, G.; Zhang, L.; Mu, Y.; Gao, X. An expressive "test-decrypt-verify" attribute-based encryption scheme with hidden policy for smart medical cloud. *IEEE Syst. J.* **2020**, *15*, 365–376. [CrossRef]
9. Zeng, P.; Zhang, Z.; Lu, R.; Choo, K.-K.R. Efficient policy-hiding and large universe attribute-based encryption with public traceability for internet of medical things. *IEEE Internet Things J.* **2021**, *8*, 10963–10972. [CrossRef]
10. Xie, M.; Ruan, Y.; Hong, H.; Shao, J. A CP-ABE scheme based on multi-authority in hybrid clouds for mobile devices. *Future Gener. Comput. Syst.* **2021**, *121*, 114–122. [CrossRef]
11. Miao, Y.; Deng, R.; Liu, X.; Choo, K.-K.R.; Wu, H.; Li, H. Multi-authority attribute-based keyword search over encrypted cloud data. *IEEE Trans. Dependable Secur. Comput.* **2019**, *18*, 1667–1680. [CrossRef]
12. Kapadia, A.; Tsangp, P.; Smiths, W. Attribute- based publishing with hidden credentials and hidden policies. In Proceedings of the Network and Distributed System Security Symposium, NDSS 2007, San Diego, CA, USA, 28 February–2 March 2007; pp. 179–192.
13. Xu, R.; Lang, B. A CP-ABE scheme with hidden policy and its application in cloud computing. *Int. J. Cloud Comput.* **2015**, *4*, 279–298. [CrossRef]
14. Zhang, W.; Zhang, Z.; Xiong, H.; Qin, Z. PHAS-HEKR-CP-ABE: Partially policy-hidden CP-ABE with highly efficient key revocation in cloud data sharing system. *J. Ambient. Intell. Humaniz. Comput.* **2022**, *13*, 613–627. [CrossRef]
15. Chase, M.; Chows, S.M. Improving privacy and security in multi-authority attribute-based encryption. In Proceedings of the ACM Conference on Computer and Communications Security, Chicago, IL, USA, 9–13 November 2009; pp. 121–130. [CrossRef]
16. Zhao, C.; Xu, L.; Li, J.; Fang, H.; Zhang, Y. Toward secure and privacy-preserving cloud data sharing: Online/offline multiauthority CP-ABE with hidden policy. *IEEE Syst. J.* **2022**, *16*, 4804–4815. [CrossRef]

17. Liu, Z.; Cao, Z.; Wong, D.S. Blackbox traceable CP-ABE: How to catch people leaking their keys by selling decryption devices on ebay. In Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, Berlin, Germany, 4–8 November 2013; pp. 475–486.
18. Luo, F.; Al-Kuwari, S. Generic construction of black-box traceable attribute-based encryption. *IEEE Trans. Cloud Comput.* **2021**, *11*, 942–955. [CrossRef]
19. He, X.; Li, L.; Peng, H. An enhanced traceable CP-ABE scheme against various types of privilege leakage in cloud storage. *J. Syst. Archit.* **2023**, *136*, 102833. [CrossRef]
20. Liu, Z.; Ding, Y.; Yuan, M.; Wang, B. Black-box accountable authority CP-ABE scheme for cloud-assisted e-health system. *IEEE Syst. J.* **2022**, *17*, 756–767. [CrossRef]
21. Sethi, K.; Pradhan, A.; Bera, P. Practical traceable multi-authority CP-ABE with outsourcing decryption and access policy updation. *J. Inf. Secur. Appl.* **2020**, *51*, 102435. [CrossRef]
22. Rabin, M.O. *How to Exchange Secrets with Oblivious Transfer*; Tech. Report. TR-81; Aiken Computation Lab, Harvard University: Cambridge, MA, USA, 1981.
23. Brassard, G.; Crepeau, C.; Robert, J.-M. All-or-nothing disclosure of secrets. In Proceedings of the International Conference on Advances in Cryptology (CRYPTO'86), Santa Barbara, CA, USA, 11–15 August 1986; Volume 263, pp. 234–238.
24. Tzeng, W.G. Efficient 1-out-n oblivious transfer schemes. In Proceedings of the 5th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2002, Paris, France, 12–14 February 2002; Springer: Berlin/Heidelberg, Germany, 2002; pp. 159–171.
25. Naor, M.; Pinkas, B. Oblivious transfer and polynomial evaluation. In Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, Atlanta, GA, USA, 1–4 May 1999; pp. 245–254.
26. Chu, C.-K.; Tzeng, W.-G. Eicient k-out-of-n Oblivious Transfer Schemes. *J. UCS* **2008**, *14*, 397–415.
27. Chu, C.K.; Tzeng, W.G. Efficient k-out-of-n oblivious transfer schemes with adaptive and non-adaptive queries. In Proceedings of the International Workshop on Public Key Cryptography, Les Diablerets, Switzerland, 23–26 January 2005; Springer: Berlin/Heidelberg, Germany, 2005; pp. 172–183.
28. Chen, N.; Li, J.; Zhang, Y.; Guo, Y. Efficient CP-ABE scheme with shared decryption in cloud storage. *IEEE Trans. Comput.* **2020**, *71*, 175–184. [CrossRef]
29. Das, S.; Namasudra, S. Multiauthority CP-ABE-based Access Control Model for IoT-enabled Healthcare Infrastructure. *IEEE Trans. Ind. Inform.* **2022**, *19*, 821–829. [CrossRef]
30. Zhou, Y.; Zheng, S.; Wang, L. Privacy-preserving and efficient public key encryption with keyword search based on CP-ABE in cloud. *Cryptography* **2020**, *4*, 28. [CrossRef]
31. The Pairing-Based Cryptography Library. 2006. Available online: https://crypto.stanford.edu/pbc/manual/ (accessed on 1 May 2023).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.