symmetry

MDPI

Article

Practical, Provably Secure, and Black-Box Traceable CP-ABE for Cryptographic Cloud Storage

Huidong Qiao [1,2], Haihe Ba [1], Huaizhe Zhou [1], Zhiying Wang [1,*], Jiangchun Ren [1] and Ying Hu [2]

[1] College of Computer, National University of Defense Technology, Changsha 410073, China; qiaohuidong13@nudt.edu.cn (H.Q.); haiheba@nudt.edu.cn (H.B.); huaizhezhou@nudt.edu.cn (H.Z.); jcren@nudt.edu.cn (J.R.)

[2] College of Computer and Communication, Hunan Institute of Engineering, Xiangtan 411100, China; Huying1983@hnie.edu.cn

* Correspondence: zywang@nudt.edu.cn

Received: 12 September 2018; Accepted: 10 October 2018; Published: 11 October 2018

Abstract: Cryptographic cloud storage (CCS) is a secure architecture built in the upper layer of a public cloud infrastructure. In the CCS system, a user can define and manage the access control of the data by himself without the help of cloud storage service provider. The ciphertext-policy attribute-based encryption (CP-ABE) is considered as the critical technology to implement such access control. However, there still exists a large security obstacle to the implementation of CP-ABE in CCS. That is, how to identify the malicious cloud user who illegally shares his private keys with others or applies his keys to construct a decryption device/black-box, and provides the decryption service. Although several CP-ABE schemes with black-box traceability have been proposed to address the problem, most of them are not practical in CCS systems, due to the absence of scalability and expensive computation cost, especially the cost of tracing. Thus, we present a new black-box traceable CP-ABE scheme that is scalable and high efficient. To achieve a much better performance, our work is designed on the prime order bilinear groups that results in a great improvement in the efficiency of group operations, and the cost of tracing is reduced greatly to $O(N)$ or $O(1)$, where N is the number of users of a system. Furthermore, our scheme is proved secure in a selective standard model. To the best of our knowledge, this work is the first such practical and provably secure CP-ABE scheme for CCS, which is black-box traceable.

Keywords: CP-ABE; cryptographic cloud storage; black-box traceability; provable security

1. Introduction

Public cloud storage enables the users to store their huge data in a professional storage service platform with a relatively cheap cost. Additionally, the service could be handy and flexible to satisfy the changing needs of the customers. Meanwhile, a user can access the data anywhere if he connects to the Internet. In a public cloud, the security issues are always critical. Researches [1–7] about confidentiality, integrity, availability, auditability and so on are proposed to address various security problems. However, if a cloud service provider (CSP) is suspicious, the security problems cloud become much more complicated. Unlike in the distributed system such as P2P grid, whose trust model is usually constructed in a special model, in a traditional centralized cloud storage system, users have to trust the cloud service provider (CSP) completely, because the data stored in cloud is completely under the control of CSP. But in fact, CSP is usually a semi-trusted party (honest but curious), sometimes even a suspicious party. This adds the security concerns about data leaking and abusing, and it becomes the major obstacle for the users who try to move their data to a public cloud.

Distinct from normal cloud storage, Kamara and Lauter [5] proposed a new architecture referred as cryptographic cloud storage (CCS). In this CCS system, the data owners (DO) instead of CSP are in charge of the management of data security. CSP only provides the storage service in CCS, because CSP should usually be considered as a semi-trusted party.

Among the operations of security management, the access control is a critical function which offers the data availability (such as data sharing). To enforce the data access control, before uploading data to the cloud, a DO encrypts the data and makes that the authorized users can decrypt the data in a certain way. In [5], CP-ABE schemes are suggested to be used to realize this access control.

Unlike traditional public key encryptions which are used to perform one-to-one encryption, attribute-based encryption (ABE) [8] is a powerful tool to fulfill the requirement of one-to-many encryption. As a variant of ABE, the concept of CP-ABE is clarified by Goyal et al. [9], and Bethencourt et al. [10] proposed the first CP-ABE scheme which realized an expressive and fine-grained access control over the encrypted data. In a CP-ABE system, a set of attributes S is allocated to each user, and a DO can specify the access policy of a data by encrypting the data with a corresponding access structure. Then only the data user, whose attribute set S satisfies the access policy, can decrypt the ciphertext. For example, Alice wants to share a message with all accountants in the company and the IT engineers of New York branch. In a CP-ABE system, Alice can encrypt the message under the policy "accountant **OR** (IT engineer **AND** New York branch)", then publish the encrypted message to the cloud. All users can download the message, but only the user who possesses the attribute {accountant} or the attributes {IT engineer, New York branch} can recover the message because their attribute sets satisfy the access control policy. Obviously, CP-ABE can provide a role-based, fine-grained, and expressive access control based on the special encryption method, moreover, the management of access control is transferred from the CSP to the data owners. The data stored in a public CCS can be shared securely, even if the CSP may be suspicious. Thus, CP-ABE is regarded as an ideal technique for access control, especially in the cases of cryptographic cloud storage and similar scenarios.

However, to implement a CP-ABE scheme in CCS, there is still an important security issue need to be solved in CP-ABE, that is, how to effectively identify the malicious user who illegally shares his access privilege with others. Suppose that a content service provider rents cryptographic cloud storage to provide service, and allocates a private key for attribute set Sa = {Los Angeles area, San Francisco area, e-book, video} to Alice, while Bob is assigned a key for attribute set Sb = {San Francisco area, e-book, video}. In most CP-ABE systems, both of them are capable of generating a decryption key corresponding to the attributes set {San Francisco area, e-book} for others. Meanwhile, it is difficult to identify whether it is Alice or Bob who performs the key sharing. Moreover, Alice or Bob can also build a decryption device/black-box with his/her key embedded in this device, and provide decryption service to others. As a result, if it is not possible to identify who has performed the illegal privilege sharing, then a malicious user could make profits or even compete with the content service provider.

Because the access policies are role-based and the users' privileges are embodied in the attribute keys, it becomes a common problem for CP-ABE systems. In general, the problem may include two factors: (1) Leaking private keys to unauthorized users, and (2) constructing a decryption device/black-box to share privileges with others. To address the problem, a CP-ABE system needs to support traceability. Corresponding to the description above, Liu et al. [11] clarified the concept of traceability and pointed out that there are two levels of the traceability. Level one is referred as *white-box* traceability. This means that using a well-formed decryption key the tracing algorithm can identify the key owner. It implies the ability to trace a new key which is created from the malicious user's original key. Level two is referred as *black-box* traceability. This means the tracing algorithm can identify the builder of a decryption black-box/device $\mathcal{D}$, even if the decryption algorithm and the decryption key are unknown to the tracing algorithm. Obviously, tracing a black-box is much more difficult. However, the black-box traceability implies white-box one, thus, the black-box traceability usually makes more sense to security.

Since Liu et al., presented the first black-box traceable CP-ABE scheme in [11], there are several general black-box traceable schemes [12–15] that have been proposed. However, the computation of these schemes in [11–14] could be costly, especially for the tracing algorithm that needs to run the encryption for $O(N^3)$ times (N denotes the number of the users in system) to trace a black-box. Hence, it is almost impossible to apply these schemes in a system of medium size (e.g., N is more than 1000) or larger size. In addition, because the ciphertext size or public key size is sub-linear or linear in N, these schemes are also unscalable. It means that they are very difficult to be implemented in practice. A practical black-box traceable construction, which is scalable and the tracing algorithm runs in $O(N)$ time, is presented in [15]. To our best knowledge, it is the most efficient black-box traceable scheme. However, instead of the standard model, a generic group model [16], which is an artificial model based on the assumption that for performing any group operations the adversary has to access an oracle, is applied to prove the security of the scheme. This is considered as a major secure limitation of the scheme.

Our contribution. On the basis of the analysis above, we believe it is necessary to realize the black-box traceability in CCS. But we can see that most black-box traceable CP-ABE schemes are impractical due to the costly computation or the absence of scalability or the inefficiency of the tracing method. Although, the scheme of [15] firstly provides a practical scheme of black-box traceable CP-ABE, it is not that secure because its security proof is based on the generic group model instead of a stand model. Motivated by seeking a practical and secure CP-ABE scheme for the efficient implementation of access control in CCS, we design a new black-box traceable scheme. The following features make the scheme to be truly practical for the CCS system.

1. **High practicability**: To solve the problem of unacceptable computation cost, and to make the scheme to be practical for most CCS system, our scheme is constructed on the prime order bilinear groups instead of the composite order bilinear groups. This sharply cuts down the computation costs of group operations in CP-ABE system. Unlike the most of prior black-box traceable schemes, our scheme is scalable and significantly more efficient. More importantly, in our construction, the cost of tracing algorithm is $O(N)$ or even $O(1)$. Therefore, the scheme could be very practical in a variety of applications.
2. **Provable security**: The scheme is proved to be secure under a selective standard model while acquires a high efficiency similar to the scheme of [15].

2. Related Work

Based on the works of the ABE system, Goyal proposed two variants of ABE: Key-police ABE (KP-ABE) and CP-ABE. In a KP-ABE system, the message is encrypted under an attribute set, and an access formula over attributes is assigned to the user's private key. On the contrary, in a CP-ABE system, an attribute set is assigned to a user's private key, and a message is encrypted under an access formula over attributes. In both systems, the user can decrypt the message when the attribute set satisfies the access formula. Bethencourt, Sahai, and Waters (BSW) [10] proposed the first expressive CP-ABE scheme. BSW scheme is highly efficient for computation and its access policy can be fine-grained and expressive. But BSW system has the limitation that it is not proved secure under a standard model. To make up the deficiency, in the subsequent works, various CP-ABE schemes are proposed [17–30]. Most of these schemes managed to achieve the provable security under a standard model, some of the works also improve the performance or provide extra security such as access structure hiding, user revocation, or multi-authority.

However, as mentioned in the previous section, the lack of traceability could be a great obstacle to the utilization of CP-ABE schemes. To support traceability and prevent the abuse of private keys, Jin Li et al. [31] proposed an *accountable* CP-ABE scheme. In the key generation of this scheme, some user-specific information is embedded in the user's private keys, hence, a malicious user can be identified with the leaked key. However, in their scheme, to trace a malicious user, his original private key must be obtained. Thus, using his private key, a malicious user can choose to construct

a decryption black-box with his private key hidden in it, and sells it to others. So, it is not a full traceability. Liu et al. [11] clarified the concept of traceability and categorized it into two kinds of traceability: White-box and black-box traceability. Actually, accountable CP-ABE schemes can be classified as white-box traceable schemes, as well as the schemes of [6,32–36]. The white-box traceability is not a full traceability, because the decryption black-box is still untraceable in these systems.

To solve the issue, Liu firstly presented a black-box traceable CP-ABE scheme in [11]. Their construction is inspired by the works of Boneh and Waters [37], which is a traceable broadcast encryption scheme. Furthermore, Liu et al. [11,12] classified the decryption black-boxes into *key-like* black-box and *policy-specific* black-box. A *key-like* decryption black-box $\mathcal{D}$ is associated with an attribute set $S_{\mathcal{D}}$. $\mathcal{D}$ can decrypt the ciphertext if the access policy of the ciphertext can be satisfied by $S_{\mathcal{D}}$. A decryption black-box $\mathcal{D}$ is referred as *policy-specific*, if it is associated with an access policy $A_{\mathcal{D}}$. $\mathcal{D}$ can decrypt the ciphertext with access policy $A_{\mathcal{D}}$. Key-like black-box has a stronger decryption ability than policy-specific black-box, but tracing a policy-specific black-box is more difficult. Liu et al., proved that if a policy-specific black-box is traceable in a CP-ABE scheme, the key-like black-box also must be traceable. Therefore, the works of [12] focused on how to trace a policy-specific black-box. As the first two black-box traceable scheme, these schemes are constructed on the composite order group which makes group operations to be very costly. Besides that, the length of ciphertext and the public key size are sub-linear in the number of system users, so the system is unscalable. For improving the efficiency of ciphertext, Ning et al. [13] presented a scheme with a short ciphertext whose size is linear in the size of access policy rather than the number of system users. Their scheme also owns the traceability against policy-specific black-box.

However, the computation costs of these mentioned black-box traceable schemes are expensive, because all of them are constructed on the composite order bilinear groups, which results in great losses in computation efficiency and makes it impractical in many settings. Thus, Liu et al. [38] proposed a new black-box traceable scheme based on prime order groups. After that, to provide full security and achieve better efficiency, Liu et al. [14] presented another scheme on prime order groups by making use of dual pairing vector spaces (DVPS), which is developed by Lewko [39]. But the tracing algorithm of all these schemes could be costly, because the encryption has to be performed for $O(N^3)$ times in tracing (N denotes the number of users). Since the cost of encryption and decryption are both relatively heavy, it may be impractical in a system of medium size or larger size to trace a black-box. In addition, all these schemes are unscalable due to that the sizes of public key and the ciphertext are sub-linear in N.

Hence, a lightweight black-box traceable scheme which is scalable is proposed in [15], and it is practical in a large system because the overhead of tracing algorithm is $O(N)$. The work of [15] is derived from the construction of BSW scheme and achieves a similar efficiency. However, the scheme has the same security limitation as BSW scheme, that is, it is just proved secure in the generic group model. Thus, its security is considered to be weak because the security proof is not based on the standard complexity assumptions.

Therefore, we are motivated to design a novel black-box traceable scheme which is proved to be secure in a standard model while keeps the advantages of [15]. In fact, we adopt a relatively simple construction based on the prime order groups, so as to lower the cost of group operations largely. At the same time, we design a high efficient tracing mechanism and by using it we can trace a malicious user in $O(N)$ or $O(1)$ time. In our construction, the ciphertext size, the public key length, and the private key length are all independent of the number of the users, this makes the scheme to be scalable. Meanwhile, we design the subtle construction to realize security under decisional q-parallel Bilinear Diffie-Hellman Exponent assumption (decisional q-parallel BDHE) [30]. To the best of our knowledge, the work of this paper is the first practical black-box traceable CP-ABE scheme, that is provably secure in a standard model. It overcomes the deficiencies of prior black-box traceable schemes, and enables the secure implementation of access control in CCS.

3. Background

In this section, we first present the background information for access structures and LSSS. Next, we give the formal definitions for bilinear groups and decisional q-parallel BDHE. Then, we present the security models for conventional CP-ABE schemes and compulsory traceability.

3.1. Access Structures

The formal definition of the access structure can be found in [40]. In this context, an access structure $\mathbb{A}$ is a collection that consists of the authorized sets of attributes. The sets in $\mathbb{A}$ are referred as the authorized sets, otherwise, they are referred as unauthorized sets. Note that the access structures are restricted to be monotone in this paper. In fact, to construct the general access structure, we can create the not of an attribute as another attribute. However, it may double the size of the attribute universe.

3.2. Linear Secret-Sharing Schemes (LSSS)

In this work, we use LSSS to construct the access structure of a ciphertext according to the access policy. Here we give an informal description of LSSS, the formal definition is presented in [30,40]. Suppose that Π is a LSSS over attributes confirming to the formal definition. Then, there must exist a share-generating matrix A according to the LSSS, and for each row A_i of A, function ρ maps it to an attribute $\rho(i)$. Assuming that for an access structure $\mathbb{A}$ we have a LSSS Π, and the corresponding share-generating matrix A is a $m \times n$ matrix. Given a sharing vector $v = (s, r_2, ..., r_n)$, where $r_2, ..., r_n \in \mathbb{Z}_p$ are chosen at random and $s \in \mathbb{Z}_p$ is a secret to be shared, we can get Av as the vector consists of m shares of the secret s, therefore, a share $\lambda_i = (Av)_i$ belongs to the attribute $\rho(i)$. For an attribute set S, let the set $I_S = \{i | \rho(i) \in S\}$. Suppose that there is always the "target" vector (1,0,0,...,0) in the span of rows I_S for all sets $S \in \mathbb{A}$. Then the matrix A has the linear reconstruction property, that is, we can find the constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I_S}$ for S in polynomial time such that, for any shares $\{\lambda_i\}_{i \in I_S}$ of a secret s, there is $\sum_{i \in I_S} \omega_i \lambda_i = s$.

Note that for an unauthorized set S, the target vector should not be in the span of rows I_S. Therefore, it can be proved that for an unauthorized set S there must exist a vector $w = (w_1, ..., w_n)$ such that $w_1 = -1$ and $A_i \cdot w = 0$ for all $i \in I_S$.

3.3. Bilinear Groups and Complexity Assumption

Our constructions will be based on the efficient bilinear group operations, thus, we need to make a brief review of the bilinear groups and the definition of the complexity assumption.

3.3.1. Bilinear Groups

Assuming $\mathcal{G}$ denotes a bilinear group generator. Taking a security parameter λ, $\mathcal{G}$ produces $(p, g, \mathbb{G}, \mathbb{G}_T, e)$ as the output. $\mathbb{G}$ and $\mathbb{G}_T$ are two multiplicative cyclic groups of prime order p, where p is a big prime, and g is a generator of $\mathbb{G}$. e denotes a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, and it has the following properties:

1. Computability: For all $u, v \in \mathbb{G}$, $e(u, v)$ is computable.
2. Non-degeneracy: $e(g, g) \neq 1$.
3. Bilinearity: For any $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, there is $e(u^a, v^b) = e(u, v)^{ab}$.

3.3.2. Complexity Assumption

In our scheme, the security is based on the decisional q-parallel BDHE assumption [30]. Suppose that q is a positive integer and $(p, g, \mathbb{G}, \mathbb{G}_T, e)$ is produced by a bilinear group generator $\mathcal{G}$. The elements $a, s, b_1, b_2, ..., b_q \in \mathbb{Z}_p$ are chosen at uniform random. Given $Y =$

$$g, g^s, g^a, ..., g^{a^q}, g^{a^{q+2}}, ..., g^{a^{2q}}$$

$$\forall 1 \le j \le q \quad g^{s \cdot b_j}, g^{a/b_j}, ..., g^{a^q/b_j}, g^{a^{q+2}/b_j}, ..., g^{a^{2q}/b_j}$$
$$\forall 1 \le i, j \le q, i \ne j \quad g^{a \cdot s \cdot b_i/b_j}, ..., g^{a^q \cdot s \cdot b_i/b_j}$$

it should be difficult to distinguish between $e(g,g)^{(a^{q+1})s}$ and a random element $R \in \mathbb{G}_T$.

To solve the assumption, an algorithm ζ outputs a guess $gs \in \{0,1\}$, the advantage of ζ in breaking this assumption can be defined as:

$$Adv^q_{\mathcal{G},\zeta} = \left| Pr[\zeta(Y, E = e(g,g)^{(a^{q+1})s}) = 0] - Pr[\zeta(Y, E = R) = 0] \right|$$

where R is a random element of $\mathbb{G}_T$. If for any probabilistic polynomial-time (PPT) algorithm ζ, $Adv^q_{\mathcal{G},\zeta}$ is always negligible, then $\mathcal{G}$ satisfies the decisional q-parallel BDHE assumption.

3.4. CP-ABE Definition and Security Model

A conventional CP-ABE system usually includes this four algorithms: Setup, Encrypt, KeyGen, and Decrypt.

`Setup`$(U, \lambda) \to (PK, MK)$. The setup algorithm is used to set up the system parameters. It takes as inputs the attribute universe U and the security parameter λ and outputs a master secret key MK and the public parameter PK.

`KeyGen`$(S, MK) \to (SK)$. This algorithm generates a private key SK according to the user's attribute set S by applying the master key MK.

`Encrypt`$(M, \mathbb{A}, PK) \to (CT)$. `Encrypt` algorithm encrypts a message M under an access structure $\mathbb{A}$ by using the public parameter PK. It outputs a ciphertext CT that can be decrypted by the user whose attribute set satisfies $\mathbb{A}$.

`Decrypt`$(CT, PK, SK) \to (M)$. This algorithm decrypts a ciphertext CT, that contains the access structure $\mathbb{A}$, with the private key SK and the public parameter PK. If the attribute set of SK satisfies $\mathbb{A}$, it can correctly decrypt CT and outputs M, or it outputs $\perp$.

3.4.1. Selective Security Model for CP-ABE

Here we present the formal definition of the security model for CP-ABE system. It is typically described by a semantic security game that is played by an adversary $\mathcal{A}$ and a challenger.

Init. $\mathcal{A}$ selects an access structure $\mathbb{A}^*$ and submits it to the challenger.

Setup. The `Setup` algorithm is performed by the challenger to produce the public parameter PK, then challenger gives it to $\mathcal{A}$.

Phase 1. For each $1 \le i \le q'$, $\mathcal{A}$ queries the challenger for private key according to the attribute set S_i, and the challenger responds with key SK_{S_i} . Note that none of $S_1, ..., S_{q'}$ satisfies $\mathbb{A}^*$.

Challenge. $\mathcal{A}$ chooses two messages M_0, M_1 of equal length and gives them to the challenger. A random coin $b \in \{0,1\}$ is flipped by the challenger. Then the challenger encrypts M_b under the access structure $\mathbb{A}^*$, and gives the ciphertext to $\mathcal{A}$.

Phase 2. For each $(q'+1) \le i \le q$, $\mathcal{A}$ queries the challenger for private key according to the attribute set S_i, and the challenger responds with key SK_{S_i} . Note that none of $S_{q'+1}, ..., S_q$ satisfies $\mathbb{A}^*$.

Guess. $\mathcal{A}$ finally outputs a guess $b' \in \{0,1\}$.

If $b' = b$, we say that $\mathcal{A}$ wins the game. In this game, we define the advantage of $\mathcal{A}$ as $Pr[b' = b] - 1/2$.

Definition 1. *A CP-ABE scheme is selectively chosen-plaintext attack (CPA) secure, if for any PPT adversary $\mathcal{A}$, the advantage is always negligible in the above game.*

3.4.2. Traceability for CP-ABE

We focus on the tracing for key-like decryption black-box in this paper, because we can trace the policy-specific black-box with a similar method in our construction. Different from the decryption black-box in [11–14,38], which is described as a probabilistic device, in our context we adopt a relative simple concept of decryption black-box introduced in [15]. That is, a decryption black-box $\mathcal{D}$, which is associated with the attribute set $S_{\mathcal{D}}$, can decrypt CT and outputs the correct message M, if the access structure of CT can be satisfied by $S_{\mathcal{D}}$, otherwise, it outputs $\perp$. It also leads to a simpler collusion problem. For example, if there are two malicious users, it is shown in [15] that the decryption ability of a decryption black-box, which is built by them together, is equal to the decryption ability of the two black-boxes they build, respectively. Therefore, we can consider the collusion as putting the collusive malicious users' independent black-box together. So we put emphasis on tracing a black-box built by an adversary alone.

We always need to interact with a black-box, when tracing it. As well as in all existing black-box traceable schemes, in our work, we should send some special ciphertext to the black-box, and try to identify the black-box builder by using the information returned from the black-box. Hence, we need to design another encryption algorithm. This algorithm should produce the special ciphertext such that the decryption results can be applied to identify the black-box builder. We define the algorithm as follows:

> $\texttt{Encrypt}_{Trace}(M, \mathbb{A}, PK) \rightarrow (TCT, trap)$. The algorithm applies the public parameter PK to encrypt the message M under an access structure $\mathbb{A}$. It outputs a tracing ciphertext TCT which can be "decrypted" by the user whose attributes satisfy $\mathbb{A}$. When a decryption result is returned, $trap$ will be used to search for the private key. Note that TCT is called as tracing ciphertext in this paper.

3.4.3. Security Model for Compulsory Traceability

To track a black-box, we need to analyze the correct decryption result of the tracing ciphertext. But if an adversary has the ability to distinguish between the normal ciphertext and the tracing ciphertext, he may output an incorrect decryption results for a tracing ciphertext to frustrate tracing, and keeps to decrypt the normal ciphertext correctly. Thus, we shall make sure that for any adversary the tracing ciphertext is indistinguishable from the normal ciphertext. In this case, the traceability is referred to as compulsory traceability.

The compulsory traceability property is formally defined in [15]. It is described as follows by a security game played by an adversary $\mathcal{A}$ and a challenger. The game is designed with the intuition such that a user, even if he has the access right to the ciphertext, is not able to determine whether a ciphertext is a tracing one or a normal one.

Setup. The $\texttt{Setup}$ algorithm is performed by the challenger to produce the public parameter PK, then challenger gives it to $\mathcal{A}$.

Phase 1. For each $1 \leq i \leq q'$, $\mathcal{A}$ queries the challenger for the private key according to attribute set S_i, and the challenger responds with key SK_{S_i}.

Challenge. $\mathcal{A}$ selects a access structure $\mathbb{A}^*$, then submits it to the challenger. The challenger chooses a message $M \in \mathbb{G}_T$ at random, and a random coin $b \in \{0,1\}$ is flipped. When $b = 0$ the challenger runs $\texttt{Encrypt}(PK, M, \mathbb{A}^*) \rightarrow (CT_0)$ and outputs (CT_0), or he runs $\texttt{Encrypt}_{Trace}(PK, M, \mathbb{A}^*) \rightarrow (CT_1, trap)$ and outputs (CT_1). Next he gives CT_b to $\mathcal{A}$.

Phase 2. For each $(q'+1) \leq i \leq q$, $\mathcal{A}$ queries the challenger for private key according to the attribute set S_i, and the challenger responds with key SK_{S_i} .

Guess. $\mathcal{A}$ finally outputs a guess $b' \in \{0,1\}$.

Note that $\mathbb{A}^*$ can be satisfied by at least one of the attribute set of $S_1, ..., S_q$.

If $b' = b$, we say that $\mathcal{A}$ wins the game. In this game, we define the advantage of $\mathcal{A}$ as $Pr[b' = b] - 1/2$.

Definition 2. *A black-box traceable CP-ABE scheme is compulsory traceable, if for any PPT adversary $\mathcal{A}$, the advantage is always negligible in the above game.*

4. Our Construction

Here, we present the concrete construction of our system. We also present a performance analysis and compare our scheme with the recent works. From the analysis, we show that our scheme is valuable for practical application in CCS due to the efficient computation and the scalability of the system.

4.1. Concrete Construction

In our construction, we take use of the LSSS access matrix as our access structure. Note that the notation $[m]$ denotes, for example, the set $\{1, 2, ..., m\}$.

$\texttt{Setup}(U, \lambda) \rightarrow (PK, MK)$. Taking as inputs the attribute universe U and the security parameter λ (λ determines the size of p), this algorithm chooses a bilinear group generator to produce $(p, g, \mathbb{G}, \mathbb{G}_T, e)$. Then, the algorithm randomly chooses exponents $\alpha, \beta, a \in \mathbb{Z}_p$ and the group elements $\{h_x, f_x \in \mathbb{G}\}_{x \in U}$. It publishes the public parameter as:

$$PK = (\mathbb{G}, \mathbb{G}_T, g, g^a, h = g^\beta, e(g,g)^\alpha, \{h_x, f_x\}_{x \in U})$$

and secretly keeps the master key $MK = (\beta, g^\alpha)$.
$\texttt{KeyGen}(MK, S) \rightarrow (SK, ID_{SK})$. This algorithm generates a private keys SK according to the user's attribute set S by applying the master key MK. It selects the exponents $r, \{r_j\}_{j \in S} \in \mathbb{Z}_p$ at random. Then, it computes and outputs the key SK:

$$(D = g^{(\alpha + ar)/\beta}, \forall j \in S : D_j = g^r \cdot f_j^{r_j}, D_j^{'} = g^{r_j}, D_j^{''} = h_j^r, D_j^{'''} = h_j^{r_j})$$

and records $ID_{SK} = e(g^a, g^r)$ in a list LID.
$\texttt{Encrypt}(PK, M, (A, \rho)) \rightarrow (CT)$. $\texttt{Encrypt}$ algorithm encrypts a message M under (A, ρ) by using the public parameter PK. A is an $m \times n$ LSSS matrix according to the access policy, and each row A_i of A can be mapped to an attribute $\rho(i)$. This algorithm chooses the elements of vector $v = (s, v_2, ..., v_n) \in \mathbb{Z}_p^n$ at random. Then, for each row A_i of A, it randomly chooses $z_i, t_i \in \mathbb{Z}_p$ and calculates $u_i = A_i \cdot v$. Finally, it outputs ciphertext CT as

$$((A, \rho), C = Me(g,g)^{\alpha s}, \widetilde{C} = h^s, \forall i \in [m] : C_i = g^{au_i} h_{\rho(i)}^{z_i}, C_i^{'} = f_{\rho(i)}^{au_i} h_{\rho(i)}^{t_i}, C_i^{''} = g^{z_i}, C_i^{'''} = f_{\rho(i)}^{z_i}, C_i^{''''} = g^{t_i}).$$

$\texttt{Decrypt}(PK, SK, CT) \rightarrow (M)$. The algorithm decrypts a ciphertext CT, that contains the access structure (A, ρ), with the private key SK and the public parameter PK. If the access policy of (A, ρ) can be satisfied by an attribute set $I \subseteq S$ (S is the attribute set associated with SK), the algorithm computes the constants $w_i \in \mathbb{Z}_p$ such that $\sum_{\rho(i) \in I} w_i A_i = \{1, 0, ..., 0\}$, then computes

$$\frac{e(D, \widetilde{C})}{\prod_{\rho(i) \in I} \left(\frac{e(D_{\rho(i)}, C_i) e(D_{\rho(i)}^{'''}, C_i^{''''})}{e(D_{\rho(i)}^{'}, C_i^{'}) e(D_{\rho(i)}^{''}, C_i^{''}) e(D_{\rho(i)}^{'''}, C_i^{'''})} \right)^{w_i}} = e(g,g)^{\alpha s}$$

The algorithm continues to compute $M = C / e(g,g)^{\alpha s}$ to finish the decryption. If the attribute set according to SK cannot satisfy (A, ρ), it outputs $\perp$.

4.2. Traceability

When tracing a decryption black-box, we need to send the black-box a tracing ciphertext and analyze the decryption result to identify the keys embedded in the black-box, then expose the owner of

the keys. We now describe our $\texttt{Encrypt}_{Trace}$ algorithm, which is used to generate the tracing ciphertext, as follows:

$\texttt{Encrypt}_{Trace}(PK, M, (A, \rho)) \rightarrow (TCT, trap)$. The algorithm encrypts a message M under (A, ρ) by using the public parameter PK. This algorithm takes almost the same steps as algorithm $\texttt{Encrypt}(PK, M, (A, \rho)) \rightarrow (CT)$. The only difference is that it chooses tow random elements s and s' in $\mathbb{Z}_p$, and forms the sharing vector as $v = (s', v_2, ..., v_n)$ to calculate $u_i = A_i \cdot v$. Then, it computes the ciphertext TCT as

$$((A, \rho), C = Me(g, g)^{\alpha s}, \widetilde{C} = h^s, \forall i \in [m] : C_i = g^{au_i} h_{\rho(i)}^{z_i}, C_i^{'} = f_{\rho(i)}^{au_i} h_{\rho(i)}^{t_i}, C_i^{''} = g^{z_i}, C_i^{'''} = f_{\rho(i)}^{z_i}, C_i^{''''} = g^{t_i})$$

and the $trap$ is $trap = s' - s$.

In a tracing ciphertext TCT, $\{u_i\}_{i \in [m]}$ are the shares of s' rather than s. So following this way, a black-box $\mathcal{D}$ can be tracked. Firstly, we choose an access structure (A, ρ) that can be satisfied by $S_{\mathcal{D}}$ of $\mathcal{D}$, and select a message $M \in \mathbb{G}_T$ at random. Secondly, we run algorithm $\texttt{Encrypt}_{Trace}(PK, M, (A, \rho)) \rightarrow (TCT, trap)$, and send TCT to $\mathcal{D}$ while keep the $trap$. If $\mathcal{D}$ correctly performs the decryption algorithm $\texttt{Decrypt}$ as

$$\frac{e(D, \widetilde{C})}{\prod_{\rho(i) \in S_{\mathcal{D}}} \left(\frac{e(D_{\rho(i)}, C_j) e(D_{\rho(i)}^{'''}, C_i^{''''})}{e(D_{\rho(i)}^{'}, C_i^{'}) e(D_{\rho(i)}^{''}, C_i^{''}) e(D_{\rho(i)}^{'''}, C_i^{'''})} \right)^{w_i}} = e(g, g)^{\alpha s + ar(s - s')}$$

$$M' = C / e(g, g)^{\alpha s + ar(s - s')} = Me(g, g)^{ar(s' - s)}$$

and returns M', we can compute $W = M'/M = e(g, g)^{ar(s' - s)}$. For every ID_{SK} in LID, we compute $(ID_{SK})^{trap}$ and compare it to W, until they are equal. Then, we have this ID_{SK} as the identity of the decryption key of $\mathcal{D}$. Therefore, if during the key generation the users'ID are recorded corresponding to their private keys, we can expose the builder of $\mathcal{D}$.

Note that in order to make sure that $M' = Me(g, g)^{ar(s' - s)}$ is always a valid message, we assume that group $\mathbb{G}_T$ is the message space. So $\mathcal{D}$ cannot distinguish between the normal ciphertext and the tracing ciphertext by determining whether M' is valid or not.

Efficient tracing. We can trace a black-box efficiently by setting the parameter $s' = s + 1$ rather than randomly choosing s'. In this case, $trap$ is always set to 1. Thus, for the search in LID, we can directly compare W to ID_{SK}, and make the search quite efficient.

4.3. Performance Analysis

4.3.1. Theoretical Analysis of Performance

In general, we can measure the normal performance of a CP-ABE system in terms of computation costs of decryption and encryption, scalability, private key size, ciphertext size and so on. In Table 1, we provide a brief performance comparison of some related works.

In Table 1, m denotes the size of the access policy (the number of LSSS matrix rows), N is the number of system users, $|I|$ is the size of the attribute set involved in decryption, and $|S|$ is the size of the user's attribute set. The encryption costs are measured by the number of times that exponentiation computation is performed, and the decryption costs are measured by the number of times that pairing computation is performed.

Table 1. Performance Comparison with related works.

	Ciphertext Size	Private Key Size	Enc Cost	Dec Cost	Scalability	Order of Groups
[11]	$2m + 17\sqrt{N}$	$\|S\| + 4$	$3m + 22\sqrt{N}$	$2\|I\| + 10$	×	Composite
[13]	$2m + 5$	$\|S\| + 6 + O(N)$	$3m + 5$	$2\|I\| + 6$	×	Composite
[14]	$6m + 46\sqrt{N} + 2$	$6\|S\| + 12$	$6m + 61\sqrt{N} + 3$	$6\|I\| + 30$	×	Prime
[15]	$4m + 2$	$4\|S\| + 1$	$5m + 2$	$4\|I\| + 1$	√	Prime
This work	$5m + 2$	$4\|S\| + 1$	$7m + 2$	$5\|I\| + 1$	√	Prime

Computation efficiency. In fact, our scheme is much simpler than most prior black-box traceable schemes, especially the schemes of [11–13] which are constructed on composite order bilinear groups. In general, the order n of a composite order elliptic curve group must be at least 1024 bits, in order to make sure n is infeasible to factor. Meanwhile, a prime order elliptic curve group whose size is 160 bits can provide an equivalent level of security [41]. Thus, the group operations on composite order bilinear groups, especially pairing and exponentiations computations, are very costly. For example, the cost of a Tate pairing operation on 1024 bits composite order elliptic curve is about 50 times the cost of the same pairing operation on a prime order curve with comparably security [42]. Hence, we manage to construct our system based on the prime order bilinear groups. Although the scheme of [14] is also constructed on the prime order bilinear groups, it is clear in Table 1 that its computation cost is much higher than our scheme.

Scalability. In most prior black-box traceable CP-ABE schemes, the public key size or ciphertext length or private key length is dependent of N, so the schemes are not scalable. Therefore, we have to reset the total system, when a new user joins in. This makes these schemes impractical in many applications. By contrast, our construction is scalable, due to that the size of the public key, the ciphertext length, and the private key length are all independent of N.

Tracing efficiency. As mentioned in Section 2, when there are relatively more users in a system, tracing could be very costly in most black-box traceable CP-ABE systems. To trace a black-box in [11–14], we needs to perform the tracing step for $N+1$ times, and for each time we has to run the encryption algorithm for $8\lambda(N/\epsilon)^2$ times, where $\epsilon \leq 1$ and λ is the security parameter. In our system, one just needs to perform the algorithm $\texttt{Encrypt}_{Trace}$ for only one time to trace a black-box, and $\texttt{Encrypt}_{Trace}$ has the equal cost to the normal encryption algorithm $\texttt{Encrypt}$. In addition, in our scheme one has to search the LID to find the malicious user, but it is a light operation and the cost is at most $O(N)$. Actually, by making use of *efficient tracing*, the searching cost is almost negligible in contrast to the encryption. In this case, if the cost is measured in terms of the heavy operations as exponentiations and pairing computations, the cost of tracing is $O(1)$.

Security. In Table 2, we also provide a brief security comparison with some related works. We can see in Table 2, all the schemes are CPA secure or selectively CPA secure. But different from [15], our scheme is based on complexity assumption, thus, it owns the security comparable to the schemes in [11,14,30]. In addition, by comparing the performance, it is obvious that the scheme of [15] and the scheme of this paper are the only two practical black-box traceable schemes that can be implemented in CCS. However, the scheme of [15] is only proved secure in a generic group model, while this work is proved secure in a standard model under the non-interactive assumption. Thus, this scheme is more suitable for the implementation of the access control in CCS.

Table 2. Security Comparison with related works.

	Assumptions	Type of Security	Traceability
[30]	Decisional q-parallel BDHE Assumption	selective CPA secure	no
[11]	Assumption 1 in [29], General Subgroup Decision Assumption, 3-Party Diffie-Hellman Assumption, q-Parallel BDHE Assumption	CPA secure	black-box
[14]	Decisional Linear Assumption, Decisional 3-Party Diffie-Hellman Assumption, q-Parallel BDHE Assumption	CPA secure	black-box
[15]	Generic Group	CPA secure	black-box
This work	Decisional q-parallel BDHE Assumption	selective CPA secure	black-box

4.3.2. Performance Measurements

Obviously, the system performance of CP-ABE mainly depends on the encryption and decryption algorithm, thus, we implement our system in the experiments and present the measurements results of the two algorithms. So far the scheme of Qiao et al. [15] is the most efficient black-box traceable CP-ABE scheme, thus, we compare the measurement results of our work with the implementation of [15].

In the experiments, the access structure is set as a single AND gate. In the decryption tests, this guarantees the uniformity and avoids the different outcomes caused by the different decryption keys. We use an elliptic curve group, the representations of which are 512 bits long, and the size of which is 160 bits long. The schemes are implemented by using the Java Pairing Based Cryptography (JPBC) library [43]. We run all the experiments on the same PC, the CPU of which is an Intel Core i7-3520M.

As shown in Table 1, the performance of the encryption of [15] should be better than the encryption of this works. Actually, it is shown in Figure 1, the encryption algorithm of this work is more efficient than that of [15]. This is because there are $2m$ more hash operations need to be performed in the encryption of [15]. Note that m denotes the number of attributes (the size of the access structure). By testing, we have that the average cost of the hash operation into group $\mathbb{G}$ is 26 milliseconds, while the average cost of exponentiation in group $\mathbb{G}$ is 11 milliseconds. Therefore, although there are $2m$ more exponentiation operations in this work, we achieve a better performance than [15] for the encryption algorithm.

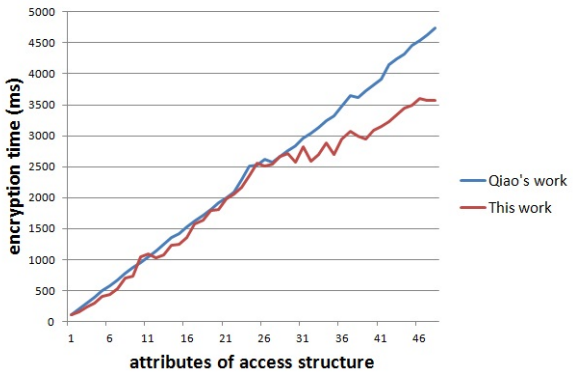

Figure 1. Encryption time.

It is easy to see in Figure 2 that the decryption algorithm of [15] achieves a better performance than that of this paper because there are $|I|$ more paring operations in the encryption algorithm of this work. The measurement results coincide with the theoretical analysis in Table 1.

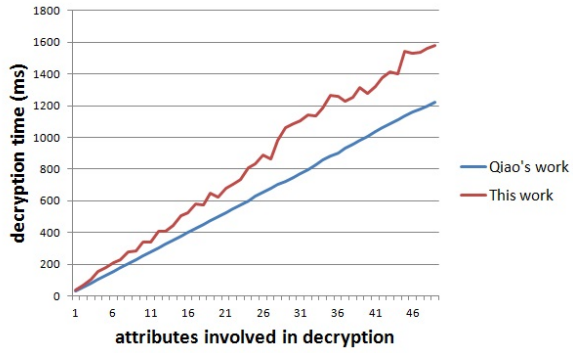

Figure 2. Decryption time.

In summary, the performance of this work is comparable to [15], and the security is improved because this work is provably secure in a standard model rather than a generic model.

5. Security Proof

The proof of Theorem 1 shows that our CP-ABE scheme is selective CPA secure, and the proof of Theorem 2 shows that our system is compulsory traceable.

Theorem 1. *Under the decisional q-parallel BDHE assumption, a polynomial time adversary can never selectively break our scheme with a challenge matrix A of size $m \times n$, where $m, n \leq q$.*

Proof of Theorem 1. We assume that an adversary $\mathcal{A}$ can break our CP-ABE scheme with non-negligible advantage $Adv_{\mathcal{A}}$ in the selective security game. We show that this adversary can be used to construct a simulator ζ, and ζ can be used to break the decisional q-parallel BDHE assumption with a non-negligible probability. So it leads to a contradiction if the decisional q-parallel BDHE assumption holds, therefore, completes the proof.

Suppose that a decisional q-parallel BDHE challenge $(q, p, \mathbb{G}, \mathbb{G}_T, e, g, Y, E)$ is given to the simulator.

Init. The adversary $\mathcal{A}$ chooses a LSSS access structure (A, ρ) with matrix A of size $m \times n$, where $m, n \leq q$, and gives it to the simulator.

Setup. The simulator chooses exponents $\alpha', \beta, \{c_x, d_x\}_{x \in U} \in \mathbb{Z}_p$ at random, and set $e(g,g)^{\alpha} = e(g^a, g^{a^q}) e(g,g)^{a'}$, that means we have $\alpha = \alpha' + a^{q+1}$. For each $x \in U$, X denotes the set $X = \{i : \rho(i) = x\}$ (i is the index of the row in A), and the simulator computes f_x and h_x as:

$$h_x = g^{c_x} \prod_{i \in X} g^{\sum_{k=1}^{n} a^k A_{i,k}/b_i}, f_x = g^{d_x}$$

Note that if $X = \emptyset$, then $h_x = g^{c_x}$. It publishes the public parameter as:

$$PK = (\mathbb{G}, \mathbb{G}_T, g, g^a, h = g^{\beta}, e(g,g)^{\alpha}, \{h_x, f_x\}_{x \in U})$$

Phase 1. For each private key query, the simulator responds to the query as follows. Suppose the adversary $\mathcal{A}$ gives the simulator a key query for an attribute set S that does not satisfy the access structure (A, ρ), and let the set $I_S = \{i | \rho(i) \in S\}$. Then, the simulator chooses $r' \in \mathbb{Z}_p$ at random, and continues to find a vector $w = (w_1, ..., w_n)$, such that $w_1 = -1$ and $w \cdot A_i = 0$ for all $i \in I_S$. This vector must exist, due to the property of LSSS which is discussed in Section 3. Then the simulator implicitly set $r = r' + \sum_{l=1}^{n} w_l a^{q-(l-1)}$ by letting

$$g^r = g^{r'} \prod_{l=1}^{n} (g^{a^{q-l+1}})^{w_l}, D = g^{\alpha'/\beta} g^{ar'/\beta} \prod_{l=2}^{n} (g^{a^{q-l+2}/\beta})^{w_l}$$

Thus, for each $j \in S$, the simulator randomly chooses $\{r_j\}_{j \in S} \in \mathbb{Z}_p$ and computes:

$$D_j = g^r f_j^{r_j}, D_j^{'} = g^{r_j}, D_j^{'''} = h_j^{r_j}$$

$$D_j^{''} = g^{rc_j} \prod_{i \in J} \prod_{k=1}^{n} \left(g^{r' a^k A_{i,k}/b_i} \prod_{l=1, l \neq k}^{n} (g^{a^{q+k-l+1} A_{i,k}/b_i})^{w_l} \right)$$

J denotes the set $J = \{i : \rho(i) = j\}$. Note that all the terms of the form $g^{a^{q+1} A_{i,k}/b_k}$, which cannot be simulated, are canceled because $wA_i = 0$.

Challenge. $\mathcal{A}$ gives the simulator two message M_0, M_1. The simulator flips a random coin $b \in \{0, 1\}$, and computes

$$C = M_b e(g^s, g^{\alpha'}) E, \widetilde{C} = (g^s)^{\beta}$$

To generate the rest of ciphertext, the simulator randomly chooses $y_2', ..., y_n'$ and $\{z_i', t_i'\}_{i \in [m]} \in \mathbb{Z}_p$. It implicitly constructs the secret sharing vector $v = (s, y_2' + sa, y_3' + sa^2, ..., y_n' + sa^{n-1})$ and implicitly

sets $z_i = z_i' - sb_i, t_i = t_i' + d_{\rho(i)} z_i$. For each row A_i of A, let the set $I^* = \{j : \rho(j) = \rho(i) \quad and \quad j \neq i\}$. Thus, the simulator produces the ciphertext as

$$C_i = h_{\rho(i)}^{z_i'} g^{-(sb_i)c_{\rho(i)}} g^{a(\sum_{k=2}^{n} A_{i,k} y_k')} \prod_{j \in I^*} (g^{\sum_{k=1}^{n} -(a^k A_{j,k} s b_i / b_j)})$$

$$C_i^{'} = h_{\rho(i)}^{t_i'} (C_i)^{d_{\rho(i)}}, C_i^{''} = g^{z_i'} g^{-sb_i}, C_i^{'''} = (C_i^{''})^{d_{\rho(i)}}, C_i^{''''} = g^{t_i'} C_i^{'''}$$

and gives the whole ciphertext to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ queries the simulator for private keys, and the response of simulator is the same as that in Phase 1.

Guess. $\mathcal{A}$ outputs a guess b'. When $b = b'$ the simulator outputs 0 to indicate that $E = e(g,g)^{a^{q+1}s}$, otherwise the output is 1. When $E = e(g,g)^{a^{q+1}s}$, the simulation is perfect, hence, we have

$$Pr[\zeta(Y, E = e(g,g)^{a^{q+1}s}) = 0] = Adv_{\mathcal{A}} + \frac{1}{2}$$

If E is a random element R of $\mathbb{G}_T$, b will be independent from $\mathcal{A}$'s view. In this case, we have $Pr[\zeta(Y, E = R) = 0] = \frac{1}{2}$. Thus, ζ can break the decisional q-parallel BDHE assumption with a non-negligible advantage:

$$Pr[\zeta(Y, E = e(g,g)^{a^{q+1}s}) = 0] - Pr[\zeta(Y, E = R) = 0] = Adv_{\mathcal{A}}$$

So we complete the proof. □

By using the generic group model, we show that this scheme is compulsory traceable if the adversary acts generically on the groups.

The generic bilinear group model [10]. Suppose that there are two random encodings ψ_0, ψ_1 of group $\mathbb{Z}_p$, which are injective maps $\psi_0, \psi_1 : \mathbb{Z}_p \to \{0,1\}^m$, where $m > 3log(p)$. Let the group $\mathbb{G} = \{\psi_0(x) : x \in \mathbb{Z}_p\}$ and $\mathbb{G}_T = \{\psi_1(x) : x \in \mathbb{Z}_p\}$. We give oracles for the computation of the group operation on $\mathbb{G}, \mathbb{G}_T$ and bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. $\mathbb{G}$ is referred as the generic bilinear group.

Theorem 2. *Let $\psi_0, \psi_1, \mathbb{G}, \mathbb{G}_T$ be defined as the generic bilinear group model. Suppose that the adversary $\mathcal{A}$ receives no more than q group elements from its quest to the oracles and its interaction with the challenger. Then, in the security game of compulsory traceability, the advantage of $\mathcal{A}$ is $O(q^2/p)$.*

Proof of Theorem 2. This theorem indicates that if group order p is large enough, $\mathcal{A}$ has a negligible advantage in playing the security game for compulsory traceability. We proceed the proof by introducing some notations. For encodings ψ_0, ψ_1, we let $g = \psi_0(1)$, and g^x denotes $\psi_0(x)$, and $e(g,g)^y$ denotes $\psi_1(y)$.

At Setup time, the simulator randomly chooses exponents $a, \alpha, \beta, \{c_x, d_x\}_{x \in U} \in \mathbb{Z}_p$. It publishes the public parameter as:

$$PK = (\mathbb{G}, \mathbb{G}_T, g, g^a, h = g^\beta, e(g,g)^\alpha, \{h_x = g^{c_x}, f_x = g^{d_x}\}_{x \in U})$$

and keeps the master key $MK = (\beta, g^\alpha)$.

In Phase 1, For the k'th private decryption key query according to the attribute set S_k, the simulator randomly chooses $r^{(k)}, \{r_j^{(k)}\}_{j \in S_k} \in \mathbb{Z}_p$. Then, it computes and outputs a private key:

$$D = g^{(\alpha + ar^{(k)})/\beta}, \forall j \in S_k : D_j = g^{r^{(k)}} \cdot f_j^{r_j^{(k)}}, D_j^{'} = g^{r_j^{(k)}}, D_j^{''} = h_j^{r^{(k)}}, D_j^{'''} = h_j^{r_j^{(k)}}$$

as the response.

In a Challenge, $\mathcal{A}$ gives the simulator an access structure (A, ρ) with matrix A of size $m \times n$, and the simulator performs as follows. It firstly chooses a message $M \in \mathbb{G}_T$ and an element $s \in \mathbb{Z}_p$ both at random. Then, it flips a coin $b \in \{0, 1\}$. When $b = 0$ the simulator sets $s' = s$, otherwise it randomly chooses $s' \in \mathbb{Z}_p$. Next, the simulator constructs the vector $v = (s', v_2, ..., v_n)$, where $v_2, ..., v_n \in \mathbb{Z}_p$ are randomly chosen. For each row A_i of A, it continues to choose $z_i, t_i \in \mathbb{Z}_p$ at random, and calculates $u_i = A_i \cdot v$. Finally, the simulator outputs ciphertext CT as

$$((A, \rho), C = Me(g, g)^{\alpha s}, \widetilde{C} = h^s, \forall i \in [m] : C_i = g^{au_i} h_{\rho(i)}^{z_i}, C_i^{'} = f_{\rho(i)}^{au_i} h_{\rho(i)}^{t_i}, C_i^{''} = g^{z_i}, C_i^{'''} = f_{\rho(i)}^{z_i}, C_i^{''''} = g^{t_i})$$

The simulator gives the ciphertext to $\mathcal{A}$.

In Phase 2, $\mathcal{A}$ queries the simulator for private keys, and the response of simulator is the same as that in Phase 1. Note that there exists at least one set S_k which satisfies the access structure (A, ρ).

Now, we continue the proof with the restrictions that (1) the order of $\mathbb{G}$ and $\mathbb{G}_T$ are p, and (2) $\mathcal{A}$ can only apply the values it receives from the oracles or the simulator to make queries. Therefore, an oracle query must be a rational function f of the variables $a, \alpha, \beta, \lambda, \{c_x, d_x\}_{x \in U}, (r^{(k)})'s, (r_j^{(k)})'s, \{z_i, u_i, t_i\}_{i \in [m]}$. To make it clear, all possible component terms of a query into $\mathbb{G}$ are enumerated in Table 3. Thus, we can obtain any component term of a query into $\mathbb{G}_T$ by multiplying two of the types in Table 3. In addition, $\mathcal{A}$ can also add the terms 1, α and $r^{(k)}$ to a query into group $\mathbb{G}_T$. In general, $\mathcal{A}$ can make queries into $\mathbb{G}_T$ that are the arbitrary linear combinations of those terms mentioned above.

Table 3. Possible query types in $\mathbb{G}$.

a	β	c_x	d_x	$(\alpha + ar^{(k)})/\beta$
$r^{(k)} + d_j r_j^{(k)}$	$r_j^{(k)}$	$c_j r^{(k)}$	$c_j r_j^{(k)}$	βs
$au_i + c_{\rho(i)} z_i$	$au_i d_{\rho(i)} + c_{\rho(i)} t_i$	z_i	$d_{\rho(i)} z_i$	t_i

In the next part of the proof, we can see that $\mathcal{A}$ cannot determine whether or not $s' = s$ because in this simulation his view is identically distributed to what his view could have been when the simulator sets $s' = s$.

Now, let us assume that the simulator sets $s' = s$ in the challenge. It is obvious that in this case the $\mathcal{A}$'s view will differ only if he can construct such two queries f and f' into group $\mathbb{G}_T$, that $f \neq f'$ but $f|_{(s'=s)} = f'|_{(s'=s)}$. Therefore, we should have that $f - f' = \xi(s - s')$, for some polynomial $\xi \neq 0$. It implies that $\mathcal{A}$ can make a pair of queries: $\xi s = f - f' + \xi s'$ and $\xi s' = f' - f + \xi s$, such that both of them contains the same polynomial ξ. Next, we will show that is impossible for $\mathcal{A}$ to create such a pair of queries in the game.

Let the nonempty set $T = \{k : S_k \quad satisfies \quad (A, \rho)\}$. Using a key SK_k, where $k \in T$, $\mathcal{A}$ can perform the computation:

$$\prod_{\rho(j) \in S_k} \left(\frac{e(D_{\rho(j)}, C_j) e(D_{\rho(j)}^{'''}, C_j^{''''})}{e(D_{\rho(j)}^{'}, C_j^{'}) e(D_{\rho(j)}^{''}, C_j^{''}) e(D_{\rho(j)}^{'''}, C_j^{'''})}\right)^{w_j} = e(g, g)^{ar^{(k)} s'}$$

by querying the oracle. Thus, $\mathcal{A}$ can get the query polynomial as the form of $\sum_{k \in T'} (\gamma_k a r^{(k)}) s'$, for some constants $\gamma_k \neq 0$ and some set $T' \subseteq T$. Obviously, $\sum_{k \in T'} (\gamma_k a r^{(k)}) s'$ is the only polynomial form which $\mathcal{A}$ can create to satisfy the form of $\xi s'$. So ξ must be the polynomial $\xi = \sum_{k \in T'} \gamma_k a r^{(k)}$. It means that $\mathcal{A}$ has to be able to create the query such that $\xi s = \sum_{k \in T'} (\gamma_k a r^{(k)}) s$. To create a query containing $ar^{(k)}s$, $\mathcal{A}$ has to pair βs with $(\alpha + ar^{(k)})/\beta$, and there is no other way for $\mathcal{A}$. Then, $\mathcal{A}$ can get the term $\alpha s + ar^{(k)}s$. However, none of the terms, that $\mathcal{A}$ has access to, can cancel the term αs. So, $\mathcal{A}$ cannot create such a pair of queries ξs and $\xi s'$, as well as f and f'.

Therefore, unless there happens to be an "unexpected collision", $\mathcal{A}$'s view will be identically distributed to his view in the case of $s' = s$. An unexpected collision means that two distinct functions $f \neq f'$ evaluate to the same value. At this point due to the random values of variables, the values

of f and f' may coincide, that is $f - f' = 0$. $f - f'$ is a non-zero polynomial, thus, the probability of $f - f' = 0$ is at most $O(1/p)$ [16]. In this game, $\mathcal{A}$ can receive no more than q elements. Hence, the probability of that such a collision happens is $O(q^2/p)$ by a union bound. Thus, the advantage of an adversary in playing the compulsory traceability security game is $O(q^2/p)$, and this complete the proof.

In fact, following the same way, we can prove that an adversary also cannot determine whether or not $s' = s + 1$, therefore, the efficient tracing is also compulsory traceable. □

6. Conclusions

We discussed the security problem of CP-ABE, which is referred to as traceability. We demonstrated that it is necessary to track the malicious users who abuse the privilege in cryptographic cloud storage system. However, we also argued that most prior CP-ABE schemes, which support the tracing of decryption black-box, are not practical for implementation in cryptographic cloud storage system, due to their inefficiency and absence of scalability. Recently, a practical black-box traceable scheme [15], which is derived from BSW scheme, has been proposed to address the problem. But it only provided a weak security proof based on the generic group model. Hence, we designed a novel CP-ABE scheme that is both practical and provably secure in a standard model. In this paper, we only described how to trace a key-like decryption black-box in our system. In fact, for the policy-specific black-box, we can provide a similar way to trace it in our system. Our scheme has a high efficiency that is comparable to the scheme of [15], while it is proved secure under the decisional q-parallel BDHE assumption. Besides, we proved our scheme is also compulsory traceable in the generic group model.

Author Contributions: H.Q. contributed to the design of the ideas, and the writing of the paper. H.B., H.Z., Z.W., J.R. and Y.H. proofread the paper.

Funding: The research is funded in part by the National Natural Science Foundation of China, under Grant No. 61303191 and No. 61402508. And it is also supported by the National High Technology Research and Development Program of China (863), under Grant No.2015AA016010.

Acknowledgments: We would like to thank the anonymous reviewers for their insightful suggestions on improving this paper.

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

Abbreviations

The following abbreviations are used in this manuscript:

CCS	Cryptographic cloud storage
CP-ABE	Ciphertext-policy attribute-based encryption
KP-ABE	Key-policy attribute-based encryption
DO	Data owner
CSP	Cloud service provider
LSSS	Linear Secret-Sharing Schemes
PPT	probabilistic polynomial-time
BDHE	Bilinear Diffie-Hellman Exponent

References

1. Yun, A.; Shi, C.; Kim, Y. On protecting integrity and confidentiality of cryptographic file system for outsourced storage. In Proceedings of the ACM Conference on Computer and Communications Security, Chicago, IL, USA, 9–13 November 2009; pp. 67–75.
2. Bowers, K.D.; Juels, A.; Oprea, A. HAIL: A high-availability and integrity layer for cloud storage. In Proceedings of the ACM Conference on Computer and Communications Security, Chicago, IL, USA, 9–13 November 2009; pp. 187–198.

3. Wang, L.; Hayashi, T.; Kanamori, S.; Waseda, A.; Nojima, R.; Moriai, S. PRINCESS: A secure cloud file storage system for managing data with hierarchical levels of sensitivity. In Proceedings of the ACM Conference on Computer and Communications Security, Denver, CO, USA, 12–16 October 2015; pp. 1684–1686.
4. Tang, H.; Wu, J.; Cui, Y.; Weng, J.; Guan, C.; Ren, K. Enabling ciphertext deduplication for secure cloud storage and access control. In Proceedings of the 11th ACM Asia Conference on Computer and Communications Security, Xi'an, China, 30 May–3 June 2016; pp. 59–70.
5. Kamara, S.; Lauter, K. Cryptographic cloud storage. *Lect. Notes Comput. Sci.* **2010**, *6054*, 136–149.
6. Ning, J.; Dong, X.; Cao, Z.; Wei, L. Accountable authority ciphertext-policy attribute-based encryption with white-box traceability and public auditing in the cloud. *Lect. Notes Comput. Sci.* **2015**, *9327*, 270–289.
7. Ba, H.; Zhou, H.; Qiao, H.; Wang, Z.; Ren, J. RIM4J: An Architecture for Language-Supported Runtime Measurement against Malicious Bytecode in Cloud Computing. *Symmetry* **2018**, *10*, 253. [CrossRef]
8. Sahai, A.; Waters, B. Fuzzy identity-based encryption. *Lect. Notes Comput. Sci.* **2005**, *3494*, 457–473.
9. Goyal, V.; Pandey, O.; Sahai, A.; Waters, B. Attribute-based encryption for fine-grained access control of encrypted data. In Proceedings of the ACM Conference on Computer and Communications Security, Alexandria, WV, USA, 30 October–3 November 2006; pp. 89–98.
10. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-policy attribute-based encryption. In Proceedings os the IEEE Symposium on Security and Privacy, Berkeley, CA, USA, 20–23 May 2007; pp. 321–334.
11. 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 ACM Conference on Computer and Communications Security, Berlin, Germany, 4–8 November 2013; pp. 475–486.
12. Liu, Z.; Cao, Z.; Wong, D.S. Traceable CP-ABE: How to trace decryption devices found in the wild. *IEEE Trans. Inf. Forensics Secur.* **2015**, *10*, 55–68.
13. Ning, J.; Cao, Z.; Dong, X.; Gong, J.; Chen, J. Traceable CP-ABE with short ciphertexts: How to catch people selling decryption devices on ebay efficiently. *Lect. Notes Comput. Sci.* **2016**, *9879*, 551–569.
14. Liu, Z.; Wong, D.S. Traceable CP-ABE on prime order groups: Fully secure and fully collusion-resistant blackbox traceable. *Lect. Notes Comput. Sci.* **2016**, *9543*, 109–124.
15. Qiao, H.; Ren, J.; Wang, Z.; Ba, H.; Zhou, H. Compulsory traceable ciphertext-policy attribute-based encryption against privilege abuse in fog computing. *Future Gener. Comput. Syst.* **2018**, *88*, 107–116. [CrossRef]
16. Shoup, V. Lower bounds for discrete logarithms and related problems. *Lect. Notes Comput. Sci.* **1997**, *1233*, 256–266.
17. Jiang, Y.; Susilo, W.; Mu, Y.; Guo, F. Flexible ciphertext-policy attribute-based encryption supporting AND-gate and threshold with short ciphertexts. *Int. J. Inf. Secur.* **2018**, *17*, 463–475. [CrossRef]
18. Cui, H.; Deng, R.H.; Wu, G.; Lai, J. An efficient and expressive ciphertext-policy attribute-based encryption scheme with partially hidden access structures. *Lect. Notes Comput. Sci.* **2016**, *10005*, 19–38.
19. Balu, A.; Kuppusamy, K. An expressive and provably secure Ciphertext-Policy Attribute-Based Encryption. *Inf. Sci.* **2014**, *276*, 354–362. [CrossRef]
20. Lai, J.; Deng, R.H.; Li, Y. Fully secure cipertext-policy hiding CP-ABE. *Lect. Notes Comput. Sci.* **2011**, *6672*, 24–39.
21. Wei, J.H.; Liu, W.F.; Hu, X.X. Forward-secure ciphertext-policy attribute-based encryption scheme. *J. Commun.* **2014**, *35*, 38–45.
22. Li, Q.; Ma, J.; Li, R.; Xiong, J.; Liu, X. Provably secure unbounded multi-authority ciphertext-policy attribute-based encryption. *Secur. Commun. Netw.* **2015**, *8*, 4098–4109. [CrossRef]
23. Liang, X.; Cao, Z.; Lin, H.; Xing, D. Provably secure and efficient bounded ciphertext policy attribute based encryption. In Proceedings of the 4th International Symposium on ACM Symposium on Information, Computer and Communications Security, ASIACCS'09, Sydney, Australia, 10–12 March 2009; pp. 343–352.
24. Ibraimi, L.; Tang, Q.; Hartel, P.; Jonker, W. Efficient and provable secure ciphertext-policy attribute-based encryption schemes. *Lect. Notes Comput. Sci.* **2009**, *5451*, 1–12.
25. Doshi, N.; Jinwala, D.C. Fully secure ciphertext policy attribute-based encryption with constant length ciphertext and faster decryption. *Secur. Commun. Netw.* **2018**, *7*, 1988–2002. [CrossRef]
26. Emura, K.; Miyaji, A.; Nomura, A.; Omote, K.; Soshi, M. A ciphertext-policy attribute-based encryption scheme with constant ciphertext length. *Lect. Notes Comput. Sci.* **2009**, *5451*, 13–23.

27. Goyal, V.; Jain, A.; Pandey, O.; Sahai, A. Bounded ciphertext policy attribute based encryption. *Lect. Notes Comput. Sci.* **2008**, *5126*, 579–591.
28. Cheung, L.; Newport, C. Provably secure ciphertext policy ABE. In Proceedings of the ACM Conference on Computer and Communications Security, Alexandria, WV, USA, 29 October–2 November 2007; pp. 456–465.
29. Lewko, A.; Waters, B. New proof methods for attribute-based encryption: Achieving full security through selective techniques. *Lect. Notes Comput. Sci.* **2012**, *7417*, 180–198.
30. Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. *Lect. Notes Comput. Sci.* **2011**, *6571*, 53–70.
31. Li, J.; Ren, K.; Kim, K. A2BE: Accountable Attribute-Based Encryption for Abuse Free Access Control. *Iacr Cryptol. Eprint Arch.* **2009**, *2009*, 118.
32. Zhou, J.; Cao, Z.; Dong, X.; Lin, X. TR-MABE: White-box traceable and revocable multi-authority attribute-based encryption and its applications to multi-level privacy-preserving e-healthcare cloud computing systems. In Proceedings of the 2015 IEEE Conference on Computer Communications (INFOCOM), Hong Kong, China, 26 April–1 May 2015; Volume 26, pp. 2398–2406.
33. Zhang, K.; Li, H.; Ma, J.; Liu, X. Efficient large-universe multi-authority ciphertext-policy attribute-based encryption with white-box traceability. *Sci. China Inf. Sci.* **2018**, *61*, 32102. [CrossRef]
34. Ning, J.; Dong, X.; Cao, Z.; Wei, L.; Lin, X. White-box traceable ciphertext-policy attribute-based encryption supporting flexible attributes. *IEEE Trans. Inf. Forensics Secur.* **2015**, *10*, 1274–1288. [CrossRef]
35. Ning, J.; Cao, Z.; Dong, X.; Wei, L.; Lin, X. Large universe ciphertext-policy attribute-based encryption with white-box traceability. *Lect. Notes Comput. Sci.* **2014**, *8713*, 55–72.
36. Liu, Z.; Cao, Z.; Wong, D.S. White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 76–88.
37. Boneh, D.; Waters, B. A fully collusion resistant broadcast, trace, and revoke system. In Proceedings of the ACM Conference on Computer and Communications Security, Alexandria, WV, USA, 30 October–3 November 2006; pp. 211–220.
38. Liu, Z.; Wong, D.S. Practical attribute-based encryption: Traitor tracing, revocation and large universe. *Comput. J.* **2016**, *59*, 983–1004. [CrossRef]
39. Lewko, A. Tools for simulating features of composite order bilinear groups in the prime order setting. *Lect. Notes Comput. Sci.* **2012**, *7237*, 318–335.
40. Beimel, A. Secure Schemes for Secret Sharing and Key Distribution. Ph.D. Thesis, Israel Institute of Technology, Haifa, Israel, 1996.
41. Barker, E.B.; Barker, W.C.; Burr, W.E.; Smid, M.E. *Recommendation for Key Management—Part 1*; Special Publication 800-57; National Institute of Standards & Technology: Gaithersburg, MD, USA, 2007.
42. Freeman, D.M. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. *Lect. Notes . Comput. Sci.* **2010**, *6110*, 44–61.
43. De Caro, A.; Iovino, V. jPBC: Java pairing based cryptography. In Proceedings of the 16th IEEE Symposium on Computers and Communications (ISCC 2011), Kerkyra, Greece, 28 June–1 July 2011; pp. 850–855.