CryptCloud$^+$: Secure and Expressive Data Access Control for Cloud Storage

Jianting Ning, Zhenfu Cao, *Senior Member, IEEE*, Xiaolei Dong, Kaitai Liang, *Member, IEEE*, Lifei Wei, and Kim-Kwang Raymond Choo, *Senior Member, IEEE*

Abstract—Secure cloud storage, which is an emerging cloud service, is designed to protect the confidentiality of outsourced data but also to provide flexible data access for cloud users whose data is out of physical control. Ciphertext-Policy Attribute-Based Encryption (CP-ABE) is regarded as one of the most promising techniques that may be leveraged to secure the guarantee of the service. However, the use of CP-ABE may yield an inevitable security breach which is known as the misuse of access credential (i.e. decryption rights), due to the intrinsic "all-or-nothing" decryption feature of CP-ABE. In this paper, we investigate the two main cases of access credential misuse: one is on the semi-trusted authority side, and the other is on the side of cloud user. To mitigate the misuse, we propose the first accountable authority and revocable CP-ABE based cloud storage system with white-box traceability and auditing, referred to as CryptCloud$^+$. We also present the security analysis and further demonstrate the utility of our system via experiments.

Index Terms—Secure Cloud Storage, Ciphertext-Policy Attribute-Based Encryption, Access Credentials Misuse, Traceability and Revocation, Auditing.

1 Introduction

THE prevalence of cloud computing may indirectly incur vulnerability to the confidentiality of outsourced data and the privacy of cloud users. A particular challenge here is on how to guarantee that only authorized users can gain access to the data, which has been outsourced to cloud, at anywhere and anytime [3]. One naive solution is to employ encryption technique on the data prior to uploading to cloud. However, the solution limits further data sharing and processing. This is so because a data owner needs to download the encrypted data from cloud and further re-encrypt them for sharing (suppose the data owner has no local copies of the data). A fine-grained access control over encrypted data is desirable in the context of cloud computing [51].

Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [15] may be an effective solution to guarantee the confidentiality of data and provide fine-grained access control here. In a CP-ABE based cloud storage system, for example, organizations (e.g., a university such as the University of Texas at San Antonio) and individuals (e.g., students, faculty members and visiting scholars of the university) can first specify access policy over attributes of a potential cloud user. Authorized cloud users then are granted access credentials (i.e., decryption keys) corresponding to their attribute sets (e.g., student role, faculty member role, or visitor role), which can be used to obtain access to the outsourced data. As a robust one-to-many encryption mechanism, CP-ABE offers a reliable method to protect data stored in cloud, but also enables fine-grained access control over the data.

Generally speaking, the existing CP-ABE based cloud storage systems fail to consider the case where access credential is misused. For instance, a university deploys a CP-ABE based cloud storage system to outsource encrypted student data to cloud under some access policies that are compliant with the relevant data sharing and privacy legislation (e.g., the federal Family Educational Rights and Privacy Act (FERPA) and Health Insurance Portability and Accountability Act of 1992 (HIPAA)). The official in charge at the organization (e.g. university's security manager) initializes the system parameters and issues access credentials for all users (e.g., students, faculty members, and visiting scholars). Each employee is assigned with several attributes (e.g., "administrator", "senior manager", "financial officer", "tenured faculty", "tenure-track faculty", "non tenure-track faculty", "instructors", "adjunct", "visitor", and/or "students"). Only the employees with attributes satisfying the decryption policy of the outsourced data are able to gain access to the student data stored in cloud (e.g. student admission materials).

As we may have known, the leakage of any sensitive student information stored in cloud could result in a range of consequences for the organization and individuals (e.g., litigation, loss of competitive advantage, and criminal charges). The CP-ABE may help us prevent security breach from outside attackers. But when an insider of the organization is suspected to commit the "crimes" related to the redistribution of decryption rights and the circulation of student information in plain format for illicit financial gains, how could we conclusively determine that the insider

- J. Ning is with the Department of Computer Science, National University of Singapore, Singapore. E-mail: jtning88@gmail.com
- Z. Cao and X. Dong are with Shanghai Key Lab for Trustworthy Computing, East China Normal University, Shanghai, China. E-mail: {zfcao,dongxiaolei}@sei.ecnu.edu.cn
- K. Liang is with the Department of Computer Science, University of Surrey, U.K. E-mail: k.liang@surrey.ac.uk
- L. Wei is with the School of Information Technology, Shanghai Ocean University, Shanghai, China. E-mail: lfwei@shou.edu.cn
- K.-K.R. Choo is with the Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249, USA. E-mail: raymond.choo@fulbrightmail.org

This is an extended version of the conference paper [35], with more than 50% new content.

is guilty? Is it also possible for us to revoke the compromised access privileges?

In addition to the above questions, we have one more which is related to key generation authority. A cloud user's access credential (i.e., decryption key) is usually issued by a semi-trusted authority based on the attributes the user possesses. How could we guarantee that this particular authority will not (re-)distribute the generated access credentials to others? For example, the organization security official leaks a lecturer Alice's key to an outsider Bob (who is not the employee of the university). One potential answer to the question is to employ multiple authorities. Nevertheless, this incurs additional cost in communication and infrastructure deployment and meanwhile, the problem of malicious collusion among authorities remains. Therefore, we posit that adopting an accountable authority approach to mitigate the access credential escrow problem is the preferred strategy.

Seeking to mitigate access credential misuse, we propose CryptCloud$^+$, an accountable authority and revocable CP-ABE based cloud storage system with white-box traceability and auditing. To the best of our knowledge, this is the first practical solution to secure fine-grained access control over encrypted data in cloud. Specifically, in our work, we first present a CP-ABE based cloud storage framework. Using this (generic) framework, we propose two accountable authority and revocable CP-ABE systems (with white-box traceability and auditing) that are fully secure in the standard model, referred to as ATER-CP-ABE and ATIR-CP-ABE, respectively. Based on the two systems, we present the construction of CryptCloud$^+$ that provides the following features.

1) *Traceability of malicious cloud users*. Users who leak their access credentials can be traced and identified.
2) *Accountable authority*. A semi-trusted authority, who (without proper authorization) generates and further distributes access credentials to unauthorized user(s), can be identified. This allows further actions to be undertaken (e.g. criminal investigation or civil litigation for damages and breach of contract).
3) *Auditing*. An auditor can determine if a (suspected) cloud user is guilty in leaking his/her access credential.
4) *"Almost" zero storage requirement for tracing*. We use a Paillier-like encryption as an extractable commitment in tracing malicious cloud users and more practically, we do not need to maintain an identity table of users for tracing (unlike the approach used in [27]).
5) *Malicious cloud users revocation*. Access credentials for individual traced and further determined to be "compromised" can be revoked. We design two mechanisms to revoke the "traitor(s)" effectively. The ATER-CP-ABE provides an explicitly revocation mechanism where a revocation list is specified explicitly into the algorithm **Encrypt**, while the ATIR-CP-ABE offers an implicitly revocation where the encryption does not need to know the revocation list but a key update operation is required periodically.

This paper extends our earlier work (a conference version in [35]), as follows.

1) We present a formal framework model of the proposed system, designed for practical cloud storage system deployment.
2) We address a weakness in the auditing procedure of the conference version. Specifically, a malicious user may change t_{id} of his secret key in the conference version, and the auditing procedure will fail in this case. As a mitigation, we revise the key generation algorithm and add an audit list to detect if the t_{id} is changed.
3) We enhance the functionality of the construction (w.r.t. AAT-CP-ABE) proposed in the conference version and further present two enhanced constructions, namely ATER-CP-ABE and ATIR-CP-ABE. These constructions allow us to effectively revoke the malicious users explicitly or implicitly. We also present the new definitions, technique and related materials of ATER-CP-ABE and ATIR-CP-ABE.
4) Based on the new ATER-CP-ABE and ATIR-CP-ABE, we present CryptCloud$^+$ which is an effective and practical solution for secure cloud storage.
5) We provide general extensions (of our system) on the *large universe*, the *multi-use*, and the *prime-order setting* cases, so that the solution introduced in this paper is more scalable in real-world applications.
6) We comprehensively evaluate the efficiency of the proposed ATER-CP-ABE and ATIR-CP-ABE via experiments.

Organization. In Section 2, we will present related work and describe our underlying approach. Section 3 outlines our framework model and design goal. Section 4 presents the background knowledge. In Sections 5 and 6, we define ATER-CP-ABE and ATIR-CP-ABE, prior to presenting their constructions and security analysis in Sections 7 and 8. Section 9 presents the proposed CryptCloud$^+$, a comparative summary, and evaluations. Potential extensions to our work are discussed in Section 10. Finally, Section 11 concludes the paper.

2 Related Work and Our Approach

2.1 Related Work

Cloud storage explores new applications of data storage, so that data owner does take full responsibility of data management "in local" no more [43]. However, due to the separation of data ownership and data access in cloud setting [24], the management of data, software, physical machines and platforms need to be delegated to cloud service providers, so that data owner only maintains little control on virtual machines [2], [46].

To protect the confidentiality of cloud data, many cloud-based fine-grained access control systems have been introduced in the literature [1], [20], [21], [25], [44], [47]. Searchable encryption enables secure search over ciphertexts by using the pre-defined keywords [12]. The data audit and deduplication enables users to check the integrity of the outsourced data [53] and to remove storage redundancy [48].

Cloud storage is also regarded as a perfect combination with Internet of Things (IoT) [8], [16], [54]. This is because the cloud may provide considerable storage and computational resources for the devices of IoT (e.g., in e-health networks [45], [55] and vehicular DTN networks [56]) which are usually resource restrained. However, this combination yields security and privacy challenges.

In the context of Attribute-Based Encryption (ABE), Sahai and Waters [41] initially introduce the notion of ABE, which is subsequently formalized by Goyal *et al.* [15]. Specifically, Goyal *et al.* define Key-Policy Attribute-Based Encryption (KP-ABE) and Ciphertext-Policy Attribute-Based Encryption (CP-ABE). Since then, a range of ABE schemes have been proposed in the literature [9], [18], [19], [31], [37], [42]. While these schemes are designed to achieve better efficiency, expressiveness and security, they do not address traceability and revocation issues.

Li *et al.* introduce the notion of accountable CP-ABE [23] to prevent unauthorized key distribution among colluded users. In a later work [22], a user accountable multi-authority CP-ABE system is proposed. Liu *et al.* also proposed white-box [27] and black-box [26] traceability [1] CP-ABE systems supporting policy expressiveness in any monotone access structures. Ning *et al.* [30], [32], [34], [36] propose several practical CP-ABE systems with white-box traceability and black-box traceability. Deng *et al.* [11] provide a tracing mechanism of CP-ABE to find the leaked access credentials in cloud storage system.

A number of attribute revocation solutions for CP-ABE systems have also been proposed in the literature, such as [52]. Sahai *et al.* [40] define the problem of revocable storage and provide a fully secure construction for ABE based on ciphertext delegation. Yang *et al.* [49] propose a revocable multi-authority CP-ABE system that achieves both forward and backward security. More recently, Yang *et al.* [50] propose an attribute updating method to achieve the dynamic change on attribute (such as revoking previous attribute and re-granting previously revoked attribute).

However, the aforementioned research works do not consider the misbehavior of key generation authority, the feasibility of auditing, and the revocation (of misbehaver). These are the problems that we target to address in this paper.

2.2 Our Approach

An overview of the approach we use to realize the traceability of malicious cloud users, accountable authority, auditing and malicious cloud users revocation is briefly introduced below (please see Sections 7 and 8 for more technical details).

As previously discussed, to trace malicious cloud users leaking access credentials, we use a Paillier-like encryption [38] as an extractable commitment to achieve white-box traceability. Specifically, the extractable commitment allows us to commit the identity of a user when he/she requests for access credential. The commitment is regarded as a part of the credential. Due to the hiding and binding technique of the Paillier-like extractable commitment, a user cannot reveal and further "modify" the identity which is "encoded" in the credential.

The algorithm **Trace** allows us to use a trapdoor for the commitment to recover the user's identity from the corresponding credential. We remark that the access credential needs to perform an *access credential sanity check* (i.e., using the *key sanity check* algorithm) prior to the tracing step. The access credential sanity check is a deterministic algorithm [13], [14], which is used to determine if the credential is *well-formed* during decryption. Leveraging the commitment, we will no need to maintain an identity table, which is unlike the approach introduced in [27]. This allows us to "reduce" additional storage cost for tracing.

In order to achieve accountable authority, an access credential is jointly determined by both the authority and the corresponding user. This prevents the authority from having "absolute" control over the credential. The user is allowed to obtain the credential uac (according to his/her attributes and identity) from the authority by using a secure access credential generation protocol. But the authority does not know which access credential the user obtains. If the authority (re-)distributes the credential $u\tilde{a}c$ belonging to the registered user (with access credential uac) without any permission of the user, with all but a negligible probability, $u\tilde{a}c$ will differ from uac that the user holds. The access credential pair $(uac, u\tilde{a}c)$ will form a cryptographic proof of the misbehavior of the authority. We note that the similar technique also can be used to enable an auditor to determine if a user accused of credential leak is guilty. We assume that the auditor must be fair and credible (e.g., an external KPMG or PwC).

We provide two effective revocation mechanisms to revoke the malicious users explicitly or implicitly, inspired by [4], [29]. For explicit revocation, we specify a revocation list RL explicitly into the algorithm **Encrypt**. During the execution of the algorithm **KeyGen**, the master secret key α is split into two parts: one for access control and the other for revocation. For malicious users who are in RL, they will fail to decrypt any new ciphertext as the sub-master secret key corresponding to revocation part cannot be canceled out in decryption. For implicit revocation, the **Encrypt** operation does not need to know the revocation list. Instead, an algorithm **KeyUpdate** periodically issues the update key for all non-revoked users. We employ a (random secret) first degree polynomial (i.e., $f(w) = \theta w + \alpha$) and $f(1), f(t)$ to share the master secret key α between the secret key and the update key, where $f(1)$ is used for access control and $f(t)$ is for revocation. For malicious users who are in RL, since they cannot obtain the update keys, they cannot decrypt any new ciphertext. The property of revocability is achieved by combining the traceability and the revocation mechanisms described above. Specifically, the traceability mechanism guarantees that once a user is identified malicious (i.e. leaking credential), his/her identity will be placed in a revocation list. By using the explicit and implicit revocation techniques we introduced with the revocation list, we make sure that any "new" ciphertext cannot be decrypted by the "revoked" users.

1. Traceability can be broadly categorized into white-box traceability and black-box traceability [33]. White-box traceability can identify "who leaks the decryption privilege" from a leaked key, while black-box traceability can be used to trace "who is responsible for" building a decryption device with the corresponding key.

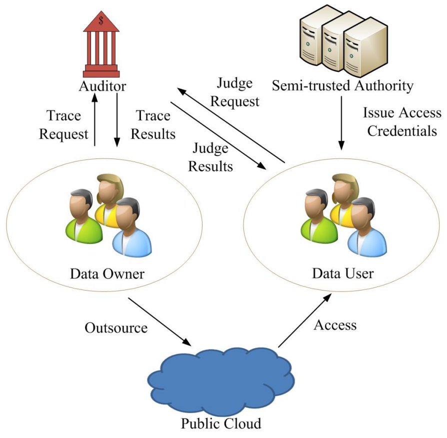


Fig. 1 CP-ABE based cloud storage system

3 Framework Model and Design Goal

Fig. 1 describes our CP-ABE based cloud storage system, with the following key entities:

- Data owners (DOs) encrypt their data under the relevant access policies prior to outsourcing the (encrypted) data to a public cloud (PC).
- PC stores the outsourced (encrypted) data from DOs and handles data access requests from data users (DUs)
- Authorized DUs are able to access (e.g. download and decrypt) the outsourced data.
- Semi-trusted authority (AT) generates system parameters and issues access credentials (i.e., decryption keys) to DUs.
- Auditor (AU) is trusted by other entities, takes charge of audit and revoke procedures, and returns the trace and audit results to DOs and DUs.

The PC is honest-but-curious in the sense that it may curiously gather more information about the outsourced (encrypted) data but will not deviate from the specification (i.e. correctly executing tasks assigned by DOs). AT is semi-trusted in the sense that it may (re-)distribute access credentials to those who are unauthorized but generate system parameters (to be shared with AU) honestly. A fully trusted AU keeps a copy of the system parameters shared by AT. DOs encrypt their data to prevent unauthorized access. Authorized DUs may intentionally leak their access credentials, such as selling credentials to a third-party. In practice, access credentials are likely to attract potential buyers (in black market), and the system traitors (selling the credentials) may never have been caught. For simplicity, we assume DOs could determine that their outsourced data had been abnormally accessed, and the trace procedure could further access the leaked access credentials. Our goal is to propose an accountable authority and revocable CryptCloud with white-box traceability and auditing to achieve the following requirements:

1) Security guarantees should be provided - protecting the confidentiality of the data and the flexibility of access control over encrypted data;
2) Computation should be cost-effective - minimizing the computation cost spent on trace and revocability; and
3) Audit, trace and revoke procedures should be efficient - shortening the time in catching a system betrayer.

4 Background

4.1 Preliminaries

We define $[l] = \{1, 2, ..., l\}$ to be $l \in \mathbb{N}$ and $[0, l] = [l] \cup \{0\}$, for $s \overset{R}{\leftarrow} S$, s is picked randomly from S.

Definition 1. *(Access Structure [5]) : We denote a collection (respectively, monotone collection) $\mathbb{A} \subseteq 2^S$ of non-empty sets of attributes to be an access structure (respectively, monotone access structure) on S, where S is the attribute universe. A collection $\mathbb{A} \subseteq 2^S$ is monotone if $\forall B, C \in \mathbb{A} : if\ B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. The sets in $\mathbb{A}$ are the authorized sets, and the sets not in $\mathbb{A}$ are the unauthorized sets.*

Definition 2. *(Linear Secret-Sharing Scheme (LSSS) [5]). Let S and p be the attribute universe and a prime, respectively. A secret-sharing scheme $\prod$ with domain of secrets $\mathbb{Z}_p$ realizing access structure on S is linear (over $\mathbb{Z}_p$) if (1) The shares of a secret $s \in \mathbb{Z}_p$ for each attribute form a vector over $\mathbb{Z}_p$; (2) For each access structure $\mathbb{A}$ on S, there exists a matrix M with l rows and n columns known as the share-generating matrix for $\prod$. For $i = 1, ..., l$, we define a function ρ labels row i of M with attribute $\rho(i)$ from S. More details can be found in [5], [34]. When we consider the column vector $\vec{v} = (s, r_2, ..., r_{n'})$, where $s \in \mathbb{Z}_p$ is the secret to be shared and $r_2, ..., r_{n'} \in \mathbb{Z}_p$ are randomly chosen. Then $M\vec{v} \in \mathbb{Z}_p^{l \times 1}$ is the vector of l shares of the secret s according to $\prod$. The share $(M\vec{v})_j$ "belongs" to attribute $\rho(j)$, where $j \in [l]$.*

We will now describe the composite order bilinear groups. Let $\mathcal{G}$ be a group generator, which takes a security parameter λ as input and outputs a description of a bilinear group G. We define the output of $\mathcal{G}$ to be $(p_1, p_2, p_3, G, G_T, e)$, where p_1, p_2, p_3 are distinct primes, G and G_T are cyclic groups of order $N = p_1 p_2 p_3$, and $e : G \times G \to G_T$ is a map such that: (1) Bilinearity: $\forall u, v \in G$, $a, b \in \mathbb{Z}_N$, we have $e(u^a, v^b) = e(u, v)^{ab}$; and (2) Non-degeneracy: $\exists g \in G$ such that $e(g, g)$ has order N in G_T. More details are available from [19].

4.2 Complexity Assumptions

Assumption 1. *(Subgroup Decision Problem for 3 Primes): [18] Given a group generator $\mathcal{G}$, define the following distribution: $\mathbb{G} = (N = p_1 p_2 p_3, G, G_T, e) \overset{R}{\leftarrow} \mathcal{G}$, $g \overset{R}{\leftarrow} G_{p_1}, X_3 \overset{R}{\leftarrow} G_{p_3}$, $D = (\mathbb{G}, g, X_3)$, $T_1 \overset{R}{\leftarrow} G_{p_1 p_2}, T_2 \overset{R}{\leftarrow} G_{p_1}$.*

The advantage of $\mathcal{A}$ in breaking this assumption is defined as: $Adv1_{\mathcal{G},\mathcal{A}}(\lambda) = |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|$. We say that $\mathcal{G}$ satisfies Assumption 1 if $Adv1_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any probabilistic polynomial-time (PPT) algorithm $\mathcal{A}$.

Assumption 2. *[18] Given a group generator $\mathcal{G}$, define the following distribution: $\mathbb{G} = (N = p_1 p_2 p_3, G, G_T, e) \overset{R}{\leftarrow} \mathcal{G}$, $g, X_1 \overset{R}{\leftarrow} G_{p_1}, X_2, Y_2 \overset{R}{\leftarrow} G_{p_2}, X_3, Y_3 \overset{R}{\leftarrow} G_{p_3}$ $D = (\mathbb{G}, g, X_1 X_2, X_3, Y_2 Y_3)$, $T_1 \overset{R}{\leftarrow} G, T_2 \overset{R}{\leftarrow} G_{p_1 p_3}$.*

The advantage of $\mathcal{A}$ in breaking this assumption is defined as: $Adv2_{\mathcal{G},\mathcal{A}}(\lambda) = |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|$. We say that $\mathcal{G}$ satisfies Assumption 2 if $Adv2_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any PPT algorithm $\mathcal{A}$.

Assumption 3. *[18] Given a group generator $\mathcal{G}$, define the following distribution: $\mathbb{G} = (N = p_1p_2p_3, G, G_T, e) \xleftarrow{R} \mathcal{G}, \alpha, s \xleftarrow{R} \mathbb{Z}_N$, $g \xleftarrow{R} G_{p_1}, X_2, Y_2, Z_2 \xleftarrow{R} G_{p_2}, X_3 \xleftarrow{R} G_{p_3}$ $D = (\mathbb{G}, g, g^{\alpha}X_2, X_3, g^sY_2, Z_2)$, $T_1 = e(g,g)^{\alpha s}, T_2 \xleftarrow{R} G_T$.*

The advantage of $\mathcal{A}$ in breaking this assumption is defined as: $Adv3_{\mathcal{G},\mathcal{A}}(\lambda) = |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|$. We say that $\mathcal{G}$ satisfies Assumption 3 if $Adv3_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any PPT algorithm $\mathcal{A}$.

Assumption 4. *(l-SDH assumption [7], [13]) : Let G be a bilinear group of prime order p and g be a generator of G, the l-Strong Diffie-Hellman (l-SDH) problem in G is defined as follows: given a $(l+1)$-tuple $(g, g^x, g^{x^2}, ..., g^{x^l})$ as inputs, output a pair $(c, g^{1/(c+x)}) \in \mathbb{Z}_p \times G$. An algorithm $\mathcal{A}$ has advantage ϵ in solving l-SDH in $\mathbb{G}$ if $Pr[\mathcal{A}(g, g^x, g^{x^2}, ..., g^{x^l}) = (c, g^{1/(c+x)})] \geq \epsilon$, where the probability is over the random choice of x in $\mathbb{Z}_p^*$ and the random bits consumed by $\mathcal{A}$.*

We say that the (l, t, ϵ)-SDH assumption holds in G if no t-time algorithm has advantage at least in solving the l-SDH problem in G.

4.3 Zero-knowledge Proof of Knowledge of Discrete Log

Informally, the discrete log protocol's zero-knowledge proof of knowledge (ZK-POK) allows a prover to demonstrate a proof (to a verifier) that it has the discrete log t of a given group element T. Such a protocol has the following properties: zero-knowledge (i.e. proving that a simulator S can construct the view of a verifier in the protocol without being given the witness as the input) and proof of knowledge property (i.e. proving that a knowledge-extractor Ext can interact with the prover to extract the witness via rewinding technique) [6], [13].

4.4 Terminologies for Binary Tree

Let $\mathcal{L} = \{1, ..., f\}, \mathcal{D}$ be the sets of leaves and nodes for a complete binary tree, respectively. For a leaf $l \in \mathcal{L}$, $\mathbf{Path}(l) \subset \mathcal{D}$ denotes the set of all nodes on the path from node l to the root (including l and the root). For $RL \subseteq \mathcal{L}$, we define $\mathbf{Cover}(RL) \subset \mathcal{D}$ as follows: (1) Mark all nodes in $\mathbf{Path}(l)$ for all $l \in RL$. (2) Set $\mathbf{Cover}(RL)$ as the set of all unmarked children of the marked nodes. It can be shown to be the minimal set that contains no node in $\mathbf{Path}(l)$ for $l \in RL$ but includes at least one node in $\mathbf{Path}(l)$ for $l \notin RL$. It is known that $|\mathbf{Cover}(RL)| \leq |RL|(log(f/|R|) + 1)$ [4], [29].

5 The Model of ATER-CP-ABE

5.1 Definition

An Accountable Authority and Explicitly Revocable CP-ABE with White-Box Traceability and Auditing (ATER-CP-ABE) is a CP-ABE scheme that is able to hold the misbehaving authority accountable, to trace malicious user by given decryption key, to determine whether the suspect is guilty, and to explicitly revoke malicious user. We revise the algorithms **Setup**, **Encrypt** and **Decrypt** presented in the conference version [35] by including a revocation list to achieve the revocation of malicious user explicitly. We will now describe our ATER-CP-ABE scheme, which consists of the following algorithms:

- **Setup**$(\lambda, \mathcal{U}) \rightarrow (pp, msk)$: On input a security parameter λ and the attribute universe description $\mathcal{U}$, it outputs the public parameters pp and the master secret key msk. It also initializes an empty revocation list RL.
- **KeyGen**$(pp, msk, id, S) \rightarrow sk_{id,S}$: This is an interactive protocol between AT and a user U. Common inputs to both AT and U are pp and a set of attributes S for a user with identity id. The private input to AT is msk. In additional, AT and U may use a sequence of random coin tosses as private input. At the end of the protocol execution, U is issued a secret key $sk_{id,S}$ corresponding to id and S.
- **Encrypt**$(pp, m, \mathbb{A}, RL) \rightarrow ct$: On input pp, a plaintext message m, an access structure $\mathbb{A}$ over the universe of attributes, and a revocation list RL, it outputs a ciphertext ct.
- **Decrypt**$(pp, sk_{id,S}, ct) \rightarrow m$ or $\perp$: On input pp, a secret key $sk_{id,S}$, and a ciphertext ct, it outputs the plaintext m if the attribute set S of sk satisfies the access structure of ct and $id \notin RL$. Otherwise, it outputs $\perp$.
- **KeySanityCheck**$(pp, sk) \rightarrow 1$ or 0: On input pp and a secret key sk, it outputs 1 if sk passes the key sanity check. Otherwise, it outputs 0. The key sanity check is a deterministic algorithm [13], [14], which is used to guarantee that the secret key is well-formed in the decryption process.
- **Trace**$(pp, msk, sk) \rightarrow id$ or $\top$: On input pp, msk and a secret key sk, it first checks whether sk is *well-formed* in order to further determine whether sk needs to be traced. A secret key sk defined as *well-formed* if **KeySanityCheck**$(pp, sk) \rightarrow 1$. For a well-formed sk, it extracts the identity from sk. It then outputs an identity with which the sk associates, and places it in the revocation list RL. Otherwise, it outputs a symbol $\top$ indicating that sk does not need to be traced.
- **Audit**$(pp, sk_{id}, sk^*_{id}) \rightarrow guilty$ or $innocent$: This is an interactive protocol between U and AU to determine whether a user is $guilty$ or $innocent$.

5.2 Security

The ATER-CP-ABE scheme is secure if the following three requirements are satisfied.

1) It must satisfy the standard semantic security notion for CP-ABE, namely: ciphertext indistinguishability under chosen plaintext attacks (IND-CPA).
2) It is intractable for the authority to create a decryption key sk such that the algorithm **Trace** (taking sk as input) outputs an identity id and the algorithm **Audit** (taking id as input) decides that the corresponding user is guilty.

3) It is infeasible for a user to create a decryption key such that the algorithm **Audit** indicates that the user is innocent.

To show if a scheme achieves the aforementioned security requirements, we define the following games.

The IND-CPA game. The IND-CPA game for ATER-CP-ABE is similar to that of CP-ABE [19] but with the exception that every key query is accompanied by an explicit identity and the attacker $\mathcal{A}$ declares a revocation list in the **Challenge** phase. The game works as follows.

- **Setup**: The challenger runs **Setup**$(\lambda, \mathcal{U})$, and further sends the public parameters pp to $\mathcal{A}$.
- **Query Phase 1**: $\mathcal{A}$ adaptively queries the challenger for secret keys corresponding to the sets of attribute $\{(id_i, S_i)\}_{i \in Q_1}$. For each (id_i, S_i), the challenger calls **KeyGen**$(pp, msk, id_i, S_i) \rightarrow sk_{id_i,S_i}$ and sends sk_{id_i,S_i} to $\mathcal{A}$.
- **Challenge**: $\mathcal{A}$ declares two equal length messages m_0, m_1, an access structure $\mathbb{A}^*$ and a revocation list RL^*. Note that $\mathbb{A}^*$ cannot be satisfied by any queried attribute sets $\{(id_i, S_i)\}_{i \in Q_1}$. The challenge flips a random coin $\delta \in \{0, 1\}$ and calls **Encrypt**$(pp, m_\delta, \mathbb{A}^*, RL^*) \rightarrow ct$. It sends ct to $\mathcal{A}$.
- **Query Phase 2**: $\mathcal{A}$ adaptively queries the challenger for the secret keys corresponding to sets of attribute $\{(id_i, S_i)\}_{i \in [Q_1+1, Q]}$ but with the restriction that none of these satisfies $\mathbb{A}^*$. For each (id_i, S_i), the challenger calls **KeyGen**$(pp, msk, id_i, S_i) \rightarrow sk_{id_i,S_i}$ and sends sk_{id_i,S_i} to $\mathcal{A}$.
- **Guess**: $\mathcal{A}$ outputs a guess $\delta' \in \{0, 1\}$ for δ.

The advantage of $\mathcal{A}$ in this game is defined as $Adv = |\Pr[\delta' = \delta] - 1/2|$.

Definition 3. *The ATER-CP-ABE is IND-CPA secure if all PPT $\mathcal{A}$ have only a negligible advantage in the above game.*

The Dishonest-Authority game. The intuition behind this game is that an adversarial authority may attempt to create a decryption key that will frame a user. It is defined by a game between a challenger and an attacker $\mathcal{A}$.

- **Setup**: $\mathcal{A}$ (acting as a malicious authority) generates public parameters pp, and sends pp, a user's (id, S) to the challenger. The challenger runs a sanity check on pp and (id, S) aborts if the check fails.
- **Key Generation**: $\mathcal{A}$ and the challenger engage in the key generation protocol **KeyGen** to generate a decryption key sk_{id} corresponding to the user's id and S. The challenger obtains the decryption key sk_{id} as input and runs a sanity check to ensure that the key is well-formed. It aborts if the check fails.
- **Output**: $\mathcal{A}$ outputs a decryption key $sk^*_{id^*}$ and wins if **Trace**$(pp, msk, sk^*_{id^*}) \rightarrow id^*$, and **Audit**$(pp, sk_{id}, sk^*_{id^*}) \rightarrow guilty$.

The advantage of $\mathcal{A}$ in this game is defined as $Adv = |\Pr[\mathcal{A}\ succeeds]|$, where the probability is taken over the random coins of **Trace**, **Audit**, $\mathcal{A}$ and the challenger.

Definition 4. *The ATER-CP-ABE is Dishonest-Authority secure if all PPT $\mathcal{A}$ have only a negligible advantage in the above game.*

The Dishonest-User game. The intuition behind this game is that a malicious user may create a new decryption key that will frame the authority. It is defined by a game between a challenger and an attacker $\mathcal{A}$.

- **Setup**: The challenger runs **Setup**$(\lambda, \mathcal{U})$, and sends the public parameters pp to $\mathcal{A}$.
- **Key Query**: $\mathcal{A}$ submits the sets of attribute $\{(id_i, S_i)\}_{i \in q}$ to request the corresponding decryption keys. The challenger calls **KeyGen**$(pp, msk, id_i, S_i) \rightarrow sk_{id_i,S_i}$ and returns sk_{id_i,S_i} to $\mathcal{A}$.
- **Key Forgery**: $\mathcal{A}$ outputs a decryption key sk^*. If $\{$**Trace**$(pp, msk, sk^*) \neq \top$ and **Trace**$(pp, msk, sk^*) \notin \{id_1, ..., id_q\}\}$ *or* $\{$**Trace**$(pp, msk, sk^*) = id$ and **Audit**$(pp, sk_{id}, sk^*) \rightarrow innocent\}$, $\mathcal{A}$ wins the game.

The advantage of $\mathcal{A}$ in this game is defined as $Adv = |\Pr[\mathcal{A}\ succeeds]|$, where the probability is taken over the random coins of **Trace**, **Audit**, $\mathcal{A}$ and the challenger.

Definition 5. *The ATER-CP-ABE is fully traceable if all PPT $\mathcal{A}$ have only a negligible advantage in the above game.*

The Key Sanity Check game. As of [36], the Key Sanity Check game for ATER-CP-ABE is defined by the following game between an attacker and a simulator. On input a security parameter λ, a simulator invokes an attacker $\mathcal{A}$ on λ. $\mathcal{A}$ returns the public parameters pp, a ciphertext ct and two different secret keys $sk_{id,S}$ and $\tilde{sk}_{id,S}$ corresponding to the same set of attribute S for a user with identity id. $\mathcal{A}$ wins the game if

(1) **KeySanityCheck**$(pp, sk_{id,S}) \rightarrow 1$.
(2) **KeySanityCheck**$(pp, \tilde{sk}_{id,S}) \rightarrow 1$.
(3) **Decrypt**$(pp, sk_{id,S}, ct) \neq \perp$.
(4) **Decrypt**$(pp, \tilde{sk}_{id,S}, ct) \neq \perp$.
(5) **Decrypt**$(pp, sk_{id,S}, ct) \neq$ **Decrypt**$(pp, \tilde{sk}_{id,S}, ct)$.

The advantage of $\mathcal{A}$ in the above game is defined as $\Pr[\mathcal{A}\ wins]$. The intuition of "Key Sanity Check" is captured by combining the notion given in the above game and **KeySanityCheck** and **Decrypt** (defined in this section) [36].

6 The Model of ATIR-CP-ABE

6.1 Definition

An Accountable Authority and Implicitly Revocable CP-ABE with White-Box Traceability and Auditing (ATIR-CP-ABE) is almost the same with the ATER-CP-ABE scheme, except that it revokes malicious users implicitly. Compared to the conference version [35], we here modify **Setup** by adding a revocation list, **Encrypt** by adding a present time attribute, and add **KeyUpdate** to achieve the revocation of malicious users implicitly. The algorithms of the ATIR-CP-ABE are almost the same with that of the ATER-CP-ABE, excepting for the additional **KeyUpdate** algorithm and the modified **Encrypt** algorithm shown as following:

- **KeyUpdate**$(pp, msk, x, RL) \rightarrow sk_{x,RL}$: On input pp, msk, a present time attribute x and a revocation list RL, the algorithm outputs the update key $sk_{x,RL}$ for time period x and sends it to all non-revoked users.

- **Encrypt**$(pp, m, \mathbb{A}, x) \rightarrow ct$: On input pp, a plaintext message m, an access structure $\mathbb{A}$ over the universe of attributes and a present time attribute x, it outputs a ciphertext ct.

6.2 Security

The security requirements of ATIR-CP-ABE is the same with that of ATER-CP-ABE. Similarly, we need to define four security games, namely: IND-CPA, Dishonest-Authority, Dishonest-User and Key Sanity Check security games.

The IND-CPA game for ATIR-CP-ABE is similar to that of ATER-CP-ABE, with the exception that the adversary does not declare the revocation list during the **Challenge** phase. The Dishonest-Authority, Dishonest-User and Key Sanity Check security games of the ATIR-CP-ABE is the same as that of ATER-CP-ABE (see Section 5.2), respectively.

7 ATER-CP-ABE

7.1 Construction

- **Setup**$(\lambda, \mathcal{U}) \rightarrow (pp, msk)$: The algorithm calls the group generator $\mathcal{G}$ with λ as input and obtains a bilinear group G of order $N = p_1p_2p_3$ (i.e. 3 distinct primes), G_{p_i} the subgroup of order p_i in G, and g, g_3 the generator of the subgroup G_{p_1}, G_{p_3}, respectively. It chooses $\alpha, a, \kappa, \mu \in \mathbb{Z}_N$ and $v \in G_{p_1}$ randomly. For each attribute $i \in \mathcal{U}$, the algorithm chooses a random $u_i \in \mathbb{Z}_N$. Also, it chooses two random primes p and q for which it holds $p \neq q$, $|p| = |q|$ and $gcd(pq, (p-1)(q-1)) = 1$, where $n = pq$, $\pi = lcm(p-1, q-1)$, $Q = \pi^{-1} \mod n$ and $g_1 = 1 + n$. In addition, we define the set of users in the system $\mathbb{U}$ as the set of leaves in the complete binary tree $\mathcal{L}$ (i.e. each user is defined as a leaf in $\mathcal{L}$). We assume that $\mathcal{D} \subseteq \mathbb{Z}_n^*$. It initializes a revocation list RL and an audit list maintained by AU. Let h denotes the maximum of $|$**Cover**$(RL)|$ for all $RL \subseteq \mathbb{U}$. It chooses random $\{h_i\}_{i\in[0,h]} \in \mathbb{Z}_N$. The public parameters are set to $pp = (N, n, g_1, v, g, g^a, g^\kappa, g^\mu, e(g,g)^\alpha, \{\mathcal{U}_i = g^{u_i}\}_{i\in\mathcal{U}}, \{\mathcal{H}_i = g^{h_i}\}_{i\in[0,h]})$. The master secret key is set to $msk = (p, q, \alpha, a, \kappa, g_3)$.
- **KeyGen**$(pp, msk, id, S) \rightarrow sk_{id,S}$: Both AT and U (with the identity id [2]) interact in the key generation protocol as follows.
  1. U chooses $t \in \mathbb{Z}_N$ randomly and computes $R_U = g^t$. It then sends g^t, id and a set of attributes S to AT. It further runs an interactive ZK-POK of the discrete log of R_U with respect to g with AT.
  2. AT checks whether the ZK-POK is valid. If the check fails, then AT aborts the interaction. Otherwise, it chooses random $c \in \mathbb{Z}_N$, $r \in \mathbb{Z}_n^*$, $\{r_d\}_{d\in\textbf{Path}(id)} \in \mathbb{Z}_N$, $R, R_0, R_0', \{R_i\}_{i\in S}, \{R_{d,1}, R_{d,2}\}_{d\in\textbf{Path}(id)} \in G_{p_3}$ and α_1, α_2 subject to the constraint that $\alpha = \alpha_1 + \alpha_2$. It then computes the primary secret key sk_{pri} as follows:
$$\langle S, \overline{T} = g_1^{id} r^n \mod n^2, \overline{K} = g^{\frac{\alpha_1}{a+\overline{T}}} (g^t)^{\frac{\kappa}{a+\overline{T}}} v^c R,$$
$$\overline{L} = g^c R_0, \overline{L'} = g^{ac} R_0', \{\overline{K_i} = \mathcal{U}_i^{(a+\overline{T})c} R_i\}_{i\in S},$$
$$\{\overline{D}_{d,1} = g^{\alpha_2} (\prod_{i=0}^{h} \mathcal{H}_i^{d^i})^{r_d} R_{d,1}, \overline{D}_{d,2} = g^{r_d} R_{d,2}\}_{d\in\textbf{Path}(id)}\rangle.$$
It sends the tuples (c, sk_{pri}) and (id, c, g^t) to U and AU, respectively. AU adds the tuple (id, c, g^t) in the audit list.
  3. U checks whether the following equalities hold:
(1) $e(\overline{L'}, g) = e(\overline{L}, g^a) = e(g^a, (g)^c)$.
(2) $\exists d \in$ **Path**(id) $s.t.$ $e(\overline{K}, g^a g^{\overline{T}}) = \frac{e(g,g)^\alpha e(\overline{L'}(\overline{L})^{\overline{T}}, v) e(R_U, g^\kappa) e(\overline{D}_{d,2}, \prod_{i=0}^{h} \mathcal{H}_i^{d^i})}{e(\overline{D}_{d,1}, g)}$.
(3) $\forall i \in S$ $s.t.$ $e(\mathcal{U}_i, \overline{L'}(\overline{L})^{\overline{T}}) = e(\overline{K_i}, g)$.
If the equalities do not hold, then U aborts the interaction. Otherwise, U computes $t_{id} = \frac{t}{c}$, and sets his/her decryption key $sk_{id,S}$ as:
$$\langle S, K = \overline{K}(g^\mu)^{t_{id}}, T = \overline{T}, L = \overline{L}, L' = \overline{L'}, R_U, t_{id},$$
$$\{K_i = \overline{K_i}\}_{i\in S}, \{D_{d,1} = \overline{D}_{d,1}, D_{d,2} = \overline{D}_{d,2}\}_{d\in\textbf{Path}(id)}\rangle.$$
- **Encrypt**$(pp, m, (A, \rho), RL) \rightarrow ct$: The algorithm chooses $\overrightarrow{y} = (s, y_2, ..., y_{n'})^\perp \in \mathbb{Z}_N^{n'\times 1}$ randomly, where s is the random secret to be shared among the shares. It chooses $r_j \in \mathbb{Z}_N$ for each row A_j of A randomly. The ciphertext ct is set as:
$$\langle C = m \cdot e(g,g)^{\alpha s}, C_0 = g^s, C_1 = (g^a)^s, C_2 = (g^\kappa)^s,$$
$$C_3 = (g^\mu)^s, \{C_{j,1} = v^{A_j \overrightarrow{y}} \mathcal{U}_{\rho(j)}^{-r_j}, C_{j,2} = g^{r_j}\}_{j\in[l]},$$
$$\{C_{d,3} = (\prod_{i=0}^{h} \mathcal{H}_i^{d^i})^s\}_{d\in\textbf{Cover}(RL)}, (A, \rho)\rangle.$$
- **Decrypt**$(pp, sk_{id,S}, ct) \rightarrow m$ or $\perp$: The algorithm outputs $\perp$ if the attribute set S cannot satisfy the access structure (A, ρ) of ct. Otherwise, it computes constants $\omega_j \in \mathbb{Z}_N$ such that $\sum_{\rho(j)\in S} \omega_j A_j = (1, 0, ..., 0)$. It then computes:
$$D = e((C_0)^T C_1, K)(e(C_2, R_u)e(C_3, (g^T g^a)^{t_{id}}))^{-1},$$
$$E = \Pi_{\rho(j)\in S}(e(C_{j,1}, (L)^T L')e(C_{j,2}, K_{\rho(j)}))^{\omega_j}.$$
Since $id \notin RL$, it finds a node d such that **Cover**$(RL) \cap$ **Path**(id), and computes
$$F = \frac{D}{E} \cdot \frac{e(D_{d,1}, C_0)}{e(D_{d,2}, C_{d,3})} = e(g,g)^{\alpha s}, m = C/F.$$
- **KeySanityCheck**$(pp, sk) \rightarrow 1$ or 0: The secret key sk passes the key sanity check if
  (1) The secret key sk is in the form of $(S, K, T, L, L', R_U, \{K_i\}_{i\in S}, t_{id}, \{D_{d,1}, D_{d,2}\}_{d\in\textbf{Path}(id)})$ and $K, L, L', \{K_i\}_{i\in S}, R_U, \{D_{d,1}, D_{d,2}\}_{d\in\textbf{Path}(id)} \in G, T \in \mathbb{Z}_{n^2}^*$.
  (2) $e(L', g) = e(L, g^a)$.
  (3) $\exists d \in$ **Path**(id) $s.t.$ $e(K, g^a g^T) = \frac{e(g,g)^\alpha e(L'(L)^T, v) e(R_U, g^\kappa) e((g^a g^T)^{t_{id}}, g^\mu)}{e(D_{d,1}, g) e(D_{d,2}, \prod_{i=0}^{h} \mathcal{H}_i^{d^i})^{-1}}$.
  (4) $\forall i \in S$ $s.t.$ $e(\mathcal{U}_i, L'(L)^T) = e(K_i, g)$.

  If sk passes the key sanity check, then the algorithm outputs 1; otherwise, it outputs 0.

2. We assume that the identity id is an element in $\mathbb{Z}_n^*$. One can extend it to arbitrary identities in $\{0,1\}^*$ by using a collision-resistant hash $H : \{0,1\}^* \rightarrow \mathbb{Z}_n^*$.

- **Trace**$(pp, msk, sk) \rightarrow id$ or $\top$: If **KeySanityCheck**$(pp, sk) \rightarrow 0$, then it outputs $\top$. Otherwise, the algorithm extracts the identity id from $T = g_1^{id} r^n \mod n^2$ in sk. Note that $Q = \pi^{-1} \mod n$ and $T^{\pi Q} = g_1^{id \cdot \pi Q} \cdot r^{n \cdot \pi Q} = g_1^{id} = 1 + id \cdot n \mod n^2$. Thus, it recovers $id = \frac{((T)^{\pi Q} \mod n^2) - 1}{n} \mod n$, outputs the identity id, and places it in RL.
- **Audit**$(pp, sk_{id}, sk_{id}^*) \rightarrow guilty$ or $innocent$: Suppose a user U (with identity id and secret key sk_{id}) is identified as a malicious user by the system (through the traced key sk_{id}^*), but claims to be innocent and framed by the system. U will interact with AU via the following protocol.

  (1) U sends its secret key sk_{id} and identity id to AU. If **KeySanityCheck**$(pp, sk_{id}) \rightarrow 0$, then AU aborts; otherwise, proceeds to (2).
  (2) AU searches id in its audit list: if id cannot be found, then AU aborts. Otherwise, proceeds to (3).
  (3) AU obtains the tuple (id, c, g^t) from its audit list and computes $g_{au} = g^{\frac{t}{c}}$. It checks whether $g_{au} = g^{t_{id}}$ holds. If not, then it indicates that U has changed its t_{id}, and the algorithm outputs $guilty$. Otherwise, proceeds to (4).
  (4) U checks if $t_{id} = t_{id}^*$ holds. If not, then it outputs $innocent$ and U is removed from the revocation list RL. Otherwise, it outputs $guilty$ and this implies that sk_{id}^* is leaked by the particular U.

7.2 Security Analysis

(1) IND-CPA Security.

Since the construction of the ATER-CP-ABE is based on the CP-ABE scheme [18], we reduce the IND-CPA security proof of our construction to that of [18]. We denote by Σ_{cp}, Σ_{atercp} the CP-ABE in [18] and the ATER-CP-ABE, respectively. The security model of Σ_{cp} in [18] is similar to the IND-CPA security model of Σ_{atercp} in Section 5.2, except that every key query is accompanied by an identity and the decryption key is jointly determined by a user and the authority.

Lemma 1. *[18] If Assumptions 1, 2, and 3 hold, then Σ_{cp} is secure.*

Lemma 2. *If Σ_{cp} in [18] is secure, then Σ_{atercp} is secure in the IND-CPA security game of Section 5.2.*

Proof. The proof of this lemma is almost identical to that of Lemma 2 [35], except that the **Setup** phase generates additional parameters for revocation and the **Encrypt** and **KeyGen** algorithms generate additional ciphertext and key part for revocation respectively.

Theorem 1. *If Assumptions 1, 2, and 3 hold, then Σ_{atercp} is IND-CPA secure.*

Proof. It follows directly from Lemma 1 and Lemma 2.

(2) Dishonest-Authority Security.

Theorem 2. *If computing the discrete log is hard in G_{p_1}, then the advantage of an adversary in the DishonestAuthority game is negligible for ATER-CP-ABE.*

Proof. The proof of this theorem is almost identical to that of Theorem 2 [35], except that the **Setup** and **KeyGen** algorithms generate additional parameters and key part for revocation, respectively.

(3) Dishonest-User Security.

Theorem 3. *If q-SDH assumption and Assumption 2 hold, then ATER-CP-ABE is DishonestUser secure provided that $q' < q$.*

Proof. The proof of this theorem is almost identical to that of Theorem 3 [35], except that the **Setup** and **KeyGen** algorithms generate additional parameters and key part for revocation, respectively.

(4) Key Sanity Check Proof.

Theorem 4. *The advantage of an attacker in the key sanity check game is negligible for the ATER-CP-ABE scheme.*

Proof. The proof of this theorem is almost identical to that of Theorem 4 [35], except that the **Setup** phase generates additional parameters for revocation, the **KeyGen** algorithm generates additional key part for revocation, and the **Decrypt** algorithm takes additional operation for revocation.

8 ATIR-CP-ABE

8.1 Construction

- **Setup**$(\lambda, \mathcal{U}) \rightarrow (pp, msk)$: The algorithm operates exactly as in the ATER-CP-ABE, excepting for the following differences. It additionally chooses $h \in \mathbb{Z}_N$ randomly. Let $\mathcal{T}$ be the universe of time periods and we assume that $\mathcal{T} \subseteq \mathbb{Z}_N^* \setminus \{1, 2\}$. It chooses a random $\theta \in \mathbb{Z}_N^*$ and defines $f(w) = \theta w + \alpha$. The public parameters are set to $pp = (N, n, g_1, v, g, g^a, g^\kappa, g^\mu, e(g,g)^\alpha, \{\mathcal{U}_i = g^{u_i}\}_{i \in \mathcal{U}}, \mathcal{H} = g^h)$, and the master secret key is set to $msk = (p, q, \alpha, a, \kappa, g_3, \theta)$.
- **KeyGen**$(pp, msk, id, S) \rightarrow sk_{id,S}$: The authority AT and a user U (with the identity id [3]) interact in the key generation protocol as follows.
  1. This step operates exactly as in the ATER-CP-ABE.
  2. AT checks whether the ZK-POK is valid. If the check fails, then AT aborts the interaction. Otherwise, it chooses a random $c, r_d \in \mathbb{Z}_N$, random $r \in \mathbb{Z}_n^*$, random elements $R, R_0, R'_0, \{R_i\}_{i \in S}, R_{d,1}, R_{d,2} \in G_{p_3}$. Then, it computes the primary secret key sk_{pri} as:

$$\langle S, \overline{T} = g_1^{id} r^n \mod n^2, \overline{K} = g^{\frac{f(1)}{a+\overline{T}}} (g^t)^{\frac{\kappa}{a+\overline{T}}} v^c R,$$
$$\overline{L} = g^c R_0,\ \overline{L'} = g^{ac} R'_0, \{\overline{K_i} = \mathcal{U}_i^{(a+\overline{T})c} R_i\}_{i \in S},$$
$$\overline{D}_{d,1} = g^{f(2)} \mathcal{H}^{r_d} R_{d,1}, \overline{D}_{d,2} = g^{r_d} R_{d,2} \rangle.$$

  It sends tuple (c, sk_{pri}) to U. Meanwhile, it sends tuple (id, c, g^t) to AU. AU puts tuple (id, c, g^t) in its audit list.
  3. U checks whether the following equalities hold:

3. We assume that the identity id is an element in $\mathbb{Z}_n^*$.

(1) $e(\overline{L'}, g) = e(\overline{L}, g^a) = e(g^a, (g)^c)$.

(2) $e(g,g)^\alpha = (\frac{e(\overline{K}, g^a g^{\overline{T}})}{e(\overline{L'}(\overline{L})^{\overline{T}}, v)e(R_U, g^\kappa)})^2 \cdot (\frac{e(\overline{D}_{d,1}, g)}{e(\overline{D}_{d,2}, \mathcal{H})})^{-1}$.

(3) $\forall i \in S \; s.t. \; e(\mathcal{U}_i, \overline{L'}(\overline{L})^{\overline{T}}) = e(\overline{K_i}, g)$.

If not, then U aborts the interaction. Otherwise, U computes $t_{id} = \frac{t}{c}$, sets the decryption key $sk_{id,S}$ as:

$$\langle S, K = \overline{K}(g^\mu)^{t_{id}}, T = \overline{T}, L = \overline{L}, L' = \overline{L'}, R_U,$$
$$t_{id}, \{K_i = \overline{K_i}\}_{i \in S}, D_{d,1} = \overline{D}_{d,1}, D_{d,2} = \overline{D}_{d,2}\rangle.$$

- **KeyUpdate**$(pp, msk, x, RL) \rightarrow sk_{x,RL}$: On input pp, msk, a present time attribute x and the revocation list RL, AT chooses random $r_{d'} \in \mathbb{Z}_N$, $R_{d',1}, R_{d',2} \in G_{p_3}$. Then, it computes the update key $sk_{x,RL}$ for time period x as:

$$\langle x, D_{d',1} = g^{f(x)}(\mathcal{H}^x)^{r_{d'}} R_{d',1}, D_{d',2} = g^{r_{d'}} R_{d',2}\rangle.$$

It then sends $sk_{x,RL}$ to all unrevoked users (according to RL).

- **Encrypt**$(pp, m, (A, \rho), x) \rightarrow ct$: The algorithm chooses $\overrightarrow{y} = (s, y_2, ..., y_{n'})^\perp \in \mathbb{Z}_N^{n' \times 1}$ randomly, where s is the random secret to be shared among the shares. It then chooses $r_j \in \mathbb{Z}_N$ for each row A_j of A randomly. The ciphertext ct is set as:

$$\langle C = m \cdot e(g,g)^{\alpha s}, C_0 = g^s, C_1 = (g^a)^s,$$
$$C_2 = (g^\kappa)^s, C_3 = (g^\mu)^s, C_4 = (\mathcal{H}^x)^s,$$
$$\{C_{j,1} = v^{A_j \overrightarrow{y}} \mathcal{U}_{\rho(j)}^{-r_j}, C_{j,2} = g^{r_j}\}_{j \in [l]}, (A, \rho)\rangle.$$

- **Decrypt**$(pp, sk_{id,S}, ct) \rightarrow m$ or $\perp$: The algorithm outputs $\perp$ if the attribute set S cannot satisfy the access structure (A, ρ) of ct. Otherwise, it computes constants $\omega_j \in \mathbb{Z}_N$ such that $\sum_{\rho(j) \in S} \omega_j A_j = (1, 0, ..., 0)$. It then computes:

$$D = e((C_0)^T C_1, K)(e(C_2, R_u)e(C_3, (g^T g^a)^{t_{id}}))^{-1},$$
$$E = \Pi_{\rho(j) \in S}(e(C_{j,1}, (L)^T L')e(C_{j,2}, K_{\rho(j)}))^{\omega_j},$$

For unrevoked users who hold the update key $(D_{d',1}, D_{d',2})$, it computes

$$F = (\frac{D}{E})^{\frac{x}{x-1}} \cdot (\frac{e(D_{d',1}, C_0)}{e(D_{d',2}, C_4)})^{\frac{1}{1-x}} = e(g,g)^{\alpha s}, m = C/F.$$

- **KeySanityCheck**$(pp, sk) \rightarrow 1$ or 0 : The secret key sk passes the key sanity check if

(1) The secret key sk is in the form of $(S, K, T, L, L', R_U, t_{id}, \{K_i\}_{i \in S}, D_{d,1}, D_{d,2}, D_{d',1}, D_{d',2}, x)$ and $K, L, L', R_U, \{K_i\}_{i \in S}, D_{d,1}, D_{d,2}, D_{d',1}, D_{d',2} \in G, T \in \mathbb{Z}_{n^2}^*, x \in \mathbb{Z}_N^* \setminus \{1, 2\}$.

(2) $e(L', g) = e(L, g^a)$.

(3) $e(g,g)^\alpha = (\frac{e(K, g^a g^T)}{e(L'(L)^T, v)e(R_U, g^\kappa)e((g^a g^T)^{t_{id}}, g^\mu)})^2 \cdot (\frac{e(D_{d,1}, g)}{e(D_{d,2}, \mathcal{H})})^{-1}$.

(4) $\forall i \in S \; s.t. \; e(\mathcal{U}_i, L'(L)^T) = e(K_i, g)$.

If sk passes the key sanity check, then the algorithm outputs 1; otherwise, it outputs 0.

- **Trace**$(pp, msk, sk) \rightarrow id$ or $\top$: The algorithm operates exactly as in the ATER-CP-ABE.
- **Audit**$(pp, sk_{id}, sk_{id}^*) \rightarrow$ *guilty* or *innocent* : The algorithm operates exactly as in the ATER-CP-ABE.

8.2 Security Analysis

(1) IND-CPA Security

Theorem 5. *If Assumptions 1, 2, and 3 hold, then ATIR-CP-ABE is IND-CPA secure.*

Proof. The proof is almost the same with that of Theorem 1 in ATER-CP-ABE – see the proof of Theorem 1.

(2) DishonestAuthority Security.

Theorem 6. *If computing discrete log is hard in G_{p_1}, then the advantage of an adversary in the DishonestAuthority game is negligible for the ATIR-CP-ABE scheme.*

Proof. The proof is almost the same with that of Theorem 2 in ATER-CP-ABE – see the proof of Theorem 2.

(3) DishonestUser Security.

Theorem 7. *If q-SDH assumption and Assumption 2 hold, then ATIR-CP-ABE is DishonestUser secure provided that $q' < q$.*

Proof. The proof is almost the same with that of Theorem 3 in ATER-CP-ABE – see the proof of Theorem 3.

(4) Key Sanity Check Proof.

Theorem 8. *The advantage of an attacker in the key sanity check game is negligible for the ATIR-CP-ABE scheme.*

Proof. The proof is almost the same with that of Theorem 4 in ATER-CP-ABE – see the proof of Theorem 4.

9 The Proposed CryptCloud$^+$

Based on ATER-CP-ABE and ATIR-CP-ABE, we propose the CryptCloud$^+$. The system works as follows. AT first generates the system parameters to setup the system and shares the entire system parameters (including public and private parameters) with AU. It then publishes the public parameters. Also, AT generates access credentials (i.e. decryption keys) for DUs according to their identities and attributes.

DOs encrypt their data under access policies (which are chosen by themselves) and then outsource the encrypted data to PC. Any authorized DU is able to decrypt the outsourced ciphertexts to access to the underlying data. A DU is authorized if the set of attribute he/she possesses satisfies the access policy defined over the outsourced data.

At some point, a legitimate access credential may be sold online, such as in an underground forum. As long as the credential is located or when a DO sends a trace request (when he/she finds that his/her data have been modified or accessed by others), AU calls the trace procedure to identify the traitor(s). If there exists a DU being identified as the traitor but claims to be innocent, then AU may call the audit procedure to make a further judgment. The malicious traitor will be revoked explicitly or implicitly subsequent to the findings. Specifically, we let Σ and Σ' be the ATER-CP-ABE and the ATIR-CP-ABE schemes respectively. Let $\Lambda = \{\Sigma, \Sigma'\}$, and our CryptCloud$^+$ works as follows.

- **System Setup**: AT setups the system. It runs $(pp, msk) \leftarrow \Lambda.$**Setup**$(\lambda, \mathcal{U})$ to generate system public parameter pp and master secret key msk. It shares

pp and msk with AU, prior to publishing pp and keeping msk secret.

- **Cloud User Enrollment**: After the request of a DU to join the system has been approved, the DU is assigned an unique identity id and an attribute set S which describes the DU. AT generates a secret access credential uac according to the identity id and attribute set S for the DU as follows. It calls $sk_{id,S} \leftarrow \Lambda.\mathbf{KeyGen}(pp, msk, id, S)$, sets the DU's secret access credential as $uac_{id,S} = sk_{id,S}$ and sends $uac_{id,S}$ to the DU.
- **File Outsource**: A DO takes the following steps to outsource the data to the PC. The data is first encrypted under a symmetric encryption (e.g. AES) with a randomly chosen symmetric session key $m_{skey} \in G_T$, and the resulting ciphertext is ct. The DO then defines an access policy $\mathbb{A}$ (represented by an LSSS (A, ρ)) and encrypts the random chosen symmetric session key m_{skey} by calling $ct_{skey} \leftarrow \Lambda.\mathbf{Encrypt}(pp, m_{skey}, (A, \rho), \Phi)$ (where $\Phi = \{RL, x\}$). Finally, the outsourced file is formed as $ct_{skey}||ct$, where ct_{skey} and ct are the header and the mainbody of the file, respectively.
- **File Access**: When a DU requests an outsourced file, the PC returns the requested file $ct_{skey}||ct$ to the DU. The DU calls $m_{skey} \leftarrow \Lambda.\mathbf{Decrypt}(pp, sk_{id,S}, ct_{skey})$ (using his/her secret access credential $uac_{id,S} = sk_{id,S}$) to recover the symmetric session key m_{skey}. The DU uses m_{skey} to decrypt ct and obtains the mainbody of the file.
- **Access Credential Update**: If the underlying construction is ATIR-CP-ABE [4], then the system needs to include an additional access credential update procedure. It calls $sk_{x,RL} \leftarrow \Lambda.\mathbf{KeyUpdate}(pp, msk, x, RL)$, sets the update access credential for time period x as $uac_{x,RL} = sk_{x,RL}$, and sends $uac_{x,RL}$ to all unrevoked DUs.
- **Trace**: When AU finds a secret access credential uac is being sold online or receives a trace request from a DO, it runs $id \leftarrow \Lambda.\mathbf{Trace}(pp, msk, uac)$ to find out who the leaker is.
- **Audit**: When a DU with identity id is traced as the leaker but claims innocence, it sends an audit request along with his/her access credential uac_{id} to AU. Upon receiving the audit request, AU calls $guilty\ or\ innocent \leftarrow \Lambda.\mathbf{Audit}(pp, uac_{id}, uac^*_{id})$ to determine whether the (accused) user is indeed innocent, where uac^*_{id} is the leaked access credential.

Our CryptCloud$^+$ is designed for the context of secure cloud storage. A storage subscriber can be caught while he/she conducts some misbehaviors, for example, he/she "shares" the storage and decryption rights with other non-subscribers. Some unusual data access may get the attention of a cloud server. The uncommon events could be the case where the access number of some specific encrypted files is significantly increased, the data access time is changed suddenly (for instance the data usually is downloaded between 9 am - 9 pm, but now it happens at mid-night). These events could be archived in a log, where the log could store login time, IP address, and the corresponding encrypted files. While a "decryption" device is found, the server may use the algorithm **Trace** to track down the malicious insider (by using the log history as input) and further revoke his/her subscription. Note that more practical extensions of the CryptCloud$^+$ will be discussed in Section 10.

We state that our mechanism may also be explored in other real-world applications, such as PayTV and electronic Groupon. Since our approach requires almost no storage cost for misbehave tracing, it is fairly suitable for the applications with huge amount of service subscribers. For instance, while a Netflix subscriber shares his/her rights with a non-subscriber, some unusual events must be incurred, such as various login locations and IP addresses, and streaming in quite different categories. The local Netflix administrator may choose either the explicit or the implicit revocation mechanism to "kick" this malicious user out of system. We note that we will attempt to explore the proposed method in more applications in future.

4. Note that ATER-CP-ABE does not contain the **KeyUpdate** algorithm. Hence, there is no access credential update step if the underlying construction is ATER-CP-ABE.

TABLE 1 Comparative Summary [1]

	[23]	[22]	[27]	[26]	[34]	[35]	this work
T	×	×	√	√	√	√	√
AA	√	×	×	×	×	√	√
ST [2]	n	n	l	n	c	n	n
MA	×	×	√	√	√	√	√
PA	×	×	×	×	×	√	√
FS	×	×	√	√	×	√	√
SM	×	√	√	√	√	√	√
R	×	×	×	×	×	×	√

[1] T denotes traceability of leaker (malicious user), AA denotes accountable authority, ST denotes storage for tracing, MA denotes supporting any monotone access structures, PA denotes auditing, FS denotes fully secure, SM denotes standard model, and R denotes revocation.

[2] We use n to denote the almost zero storage requirement for tracing, l for linear storage for tracing, and c for constant storage for tracing.

9.1 Comparative Summary

Table 1 shows a comparative summary between our proposed CP-ABE systems and related schemes, in terms of features (i.e. accountable authority, auditing, revocation, etc.) and performance. Fig. 2 illustrates the system storage overhead for traitor tracing of proposed CP-ABE systems, [27] (denoted as Com1) and [34] (denoted as Com2). It is clear that our proposed CP-ABE systems not only provides the accountable authority, auditing and revocation properties, but also requires almost a zero storage requirement for tracing. Thus, the proposed approach is suitable for practical deployment.

9.2 Evaluation

In this section, we evaluate the performance of the proposed systems presented in Sections 7 and 8. The experiments are performed on a laptop with the following specifications: Intel Core i5-5200U, 2.20 GHz, 4 GB memory, and Windows 7 operation system with Service Pack 1. We use the pairing-based cryptography library [28] with type A1 curve to realize the proposed systems. The programming language used is Java with JDK32-1.6.0 and JPBC-2.0.0 [10].

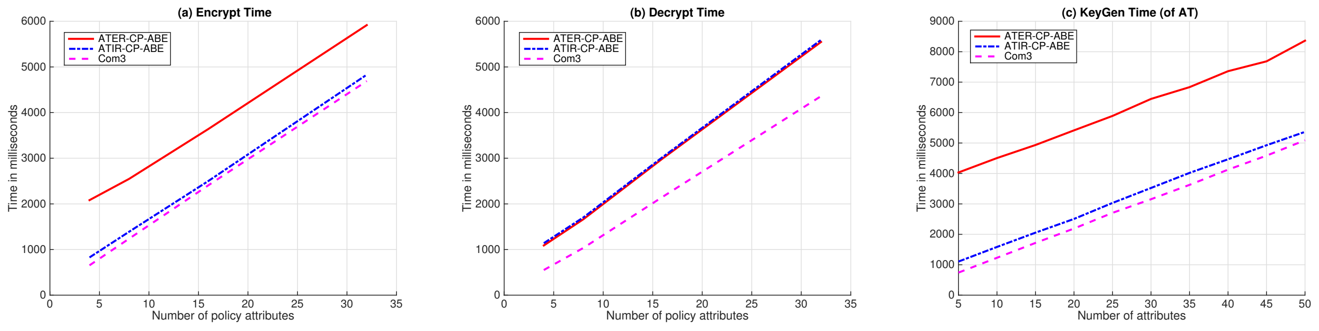


Fig. 3 Experimental results

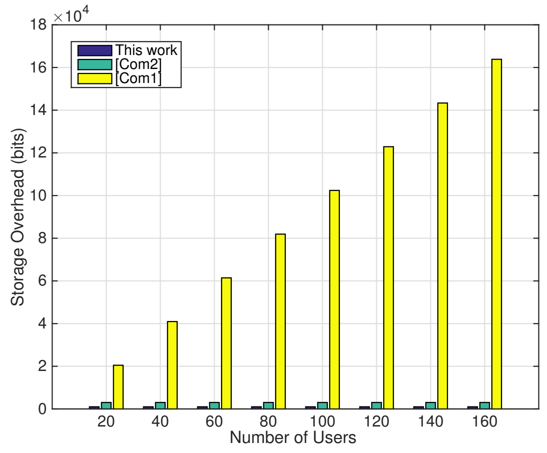


Fig. 2 System storage overhead for traitor tracing [1]

[1]We assume that the length of a group element g is 160 bits, the length of the random number c in [27] is 1024 bits, and the threshold value $\bar{t}$ in [34] is 10. We do not, however, take the public parameters into account.

In CP-ABE systems, the complexity of ciphertext policy impacts both the encryption time and the decryption time. To account for this, we generate ciphertext policies in the form of (S_1 and S_2 ... and S_l) to simulate the worst-case situation, where S_i is an attribute. We attempt to evaluate the efficiency of ATER-CP-ABE and ATIR-CP-ABE by comparing the total time taken during each stage with the original CP-ABE scheme in [18] (denoted as Com3), which does not consider the access credentials abuse issue and the revocation issue. As depicted in Fig. 3, we examine the time cost of executing individual stage (including the Encrypt Time, the Decrypt Time and the KeyGen Time (of AT)). Since we consider both access credential abuse issue and the revocation issue, it is not surprising to observe that our systems require more time. From Fig. 3(a), Fig. 3(b) and Fig. 3(c), we observe that our ATER-CP-ABE and ATIR-CP-ABE address both access credential abuse and revocation challenges without introducing significant overhead compared to the original CP-ABE scheme in [18]. We note that the technique to add the access credentials abuse issue and the revocation issue (as introduced in this paper) is a general construction and is also applicable to any other CP-ABE systems. In other words, it is possible to employ our technique into a more efficient CP-ABE to enhance the efficiency.

10 Extensions

10.1 Large Universe CryptCloud$^+$

In general, a CP-ABE may support either "small universe" and "large universe" size of attribute set. In the "small universe" construction, the size of attribute universe is polynomially bounded in the security parameter and attributes are fixed at system setup, and furthermore the size of public parameters grows linearly with the number of attributes. If the specified bound is not sufficiently large enough, then the number of attributes may not be sufficient for a large number of cloud users (i.e. when the number of users exceeds the threshold). When this happens, the entire system may need to be completely re-built. This can be an expensive and time consuimg exercise.

On the contrary, in the "large universe" construction, the size of the attribute universe is unbounded and the attributes do not need to be specified at system setup. The CP-ABE with "large universe" does scale well in the sense that the administrator of the system does not need to worry about choosing a particular bound of the attributes at the system setup phase [34], [36].

Both ATER-CP-ABE and ATIR-CP-ABE are designed for "small universe"; thus, we need to fix the attributes during setup and limit the number of the attributes. We can adapt the "layered" technique introduced in [39] to obtain "large universe" ATER-CP-ABE and ATIR-CP-ABE constructions, while the new large universe construction can only be selectively secure. Based on the underlying large universe ATER-CP-ABE and ATIR-CP-ABE, we can further obtain a large universe CryptCloud$^+$.

10.2 From One-Use to Multi-Use

Our ATER-CP-ABE and ATIR-CP-ABE are both *one-use* CP-ABE constructions. Note that the ρ in our systems is an injective function for each access policy associated to a ciphertext. During the row label of the share-generating matrix, the attributes are only used once. This type of construction is known as a one-use CP-ABE.

We can extend our ATER-CP-ABE and ATIR-CP-ABE to a *multi-use* system using the encoding technique described in [18]. Specifically, we take k copies of each attribute A instead of a single attribute. Then, we have new "attributes": $\{A : 1, ..., A : k\}$. Now, we can label a row of the access matrix $\mathbb{A}$ with $\{A : i\}$. The attribute can then be used multiple times. Based on the underlying *multi-use* ATER-CP-ABE and ATIR-CP-ABE, we can further obtain a *multi-use* CryptCloud$^+$. Note that the sizes of the public parameters and the access matrix will remain constant in size under this transformation; thus, our CryptCloud$^+$ is practical for commercial applications.

10.3 Prime-Order Group

The proposed ATER-CP-ABE and ATIR-CP-ABE are built on composite order group. Since security in composite order groups constructions typically relies on the hardness of factoring the group order, this requires the use of large group orders. However, this results in considerably slower pairing operations. Therefore, it is preferable to obtain the same functionality in prime-order groups [17].

We can extend our ATER-CP-ABE and ATIR-CP-ABE to prime-order setting using the techniques introduced in [17]. Specifically, we can adopt the dual pairing vector space framework to formulate an assumption in prime order groups that can be used to mimic the effect of the general subgroup decision assumption in composite order groups [17]. Based on the prime-order setting ATER-CP-ABE and ATIR-CP-ABE, we can further obtain a prime-order setting CryptCloud$^+$ that is more efficient than the composite-order setting one.

11 Conclusion and Future Work

In this work, we have addressed the challenge of credential leakage in CP-ABE based cloud storage system by designing an accountable authority and revocable CryptCloud which supports white-box traceability and auditing (referred to as CryptCloud$^+$). This is the first CP-ABE based cloud storage system that simultaneously supports white-box traceability, accountable authority, auditing and effective revocation. Specifically, CryptCloud$^+$ allows us to trace and revoke malicious cloud users (leaking credentials). Our approach can be also used in the case where the users' credentials are redistributed by the semi-trusted authority.

We note that we may need black-box traceability, which is a stronger notion (compared to white-box traceability), in CryptCloud. One of our future works is to consider the black-box traceability and auditing.

Furthermore, AU is assumed to be fully trusted in CryptCloud$^+$. However, in practice, it may not be the case. Is there any way to reduce trust from AU? Intuitively, one method is to employ multiple AUs. This is similar to the technique used in threshold schemes. But it will require additional communication and deployment cost and meanwhile, the problem of collusion among AUs remains. Another potential approach is to employ secure multi-party computation in the presence of malicious adversaries. However, the efficiency is also a bottleneck. Designing efficient multi-party computation and decentralizing trust among AUs (while maintaining the same level of security and efficiency) is also a part of our future work.

We use Paillier-like encryption to serve as an extractable commitment to achieve white-box traceability. From an abstract view point, any extractable commitment may be employed to achieve white-box traceability in theory. To improve the efficiency of tracing, we may make use of a more light-weight (pairing-suitable) extractable commitment.

Also, the trace algorithm in CryptCloud$^+$ needs to take the master secret key as input to achieve white-box traceability of malicious cloud users. Intuitively, the proposed CryptCloud$^+$ is private traceable[5]. Private traceability only allows the tracing algorithm to be run by the system administrator itself, while partial/full public traceability enables the administrator, authorized users and even anyone without the secret information of the system to fulfill the trace. Our future work will include extending CryptCloud$^+$ to provide "partial" and fully public traceability without compromising on performance.

Acknowledgments

We are grateful to the anonymous reviewers for their invaluable suggestions. We would like to thank Prof. Jun Shao for useful discussions, and Mr. Cong Zuo for many helps on the experiments. This work was supported by the National Natural Science Foundation of China (Grant No. 61632012, 61672239, 61411146001 and 61402282) and the Shanghai Science and Technology Commission Project (Grant No. 16511101400).

References

[1] Mazhar Ali, Revathi Dhamotharan, Eraj Khan, Samee U. Khan, Athanasios V. Vasilakos, Keqin Li, and Albert Y. Zomaya. Sedasc: Secure data sharing in clouds. *IEEE Systems Journal*, 11(2):395–404, 2017.

[2] Mazhar Ali, Samee U. Khan, and Athanasios V. Vasilakos. Security in cloud computing: Opportunities and challenges. *Inf. Sci.*, 305:357–383, 2015.

[3] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, et al. A view of cloud computing. *Communications of the ACM*, 53(4):50–58, 2010.

[4] Nuttapong Attrapadung and Hideki Imai. Attribute-based encryption supporting direct/indirect revocation modes. In *Cryptography and Coding*, pages 278–300. Springer, 2009.

[5] Amos Beimel. *Secure schemes for secret sharing and key distribution*. PhD thesis, PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.

[6] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In *Advances in Cryptology-CRYPTO'92*, pages 390–420. Springer, 1993.

[7] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In *EUROCRYPT - 2004*, pages 56–73, 2004.

[8] Hongming Cai, Boyi Xu, Lihong Jiang, and Athanasios V. Vasilakos. Iot-based big data storage systems in cloud computing: Perspectives and challenges. *IEEE Internet of Things Journal*, 4(1):75–87, 2017.

[9] Jie Chen, Romain Gay, and Hoeteck Wee. Improved dual system ABE in prime-order groups via predicate encodings. In *Advances in Cryptology - EUROCRYPT 2015*, pages 595–624, 2015.

[10] Angelo De Caro and Vincenzo Iovino. jpbc: Java pairing based cryptography. In *ISCC 2011*, pages 850–855. IEEE, 2011.

[11] Hua Deng, Qianhong Wu, Bo Qin, Jian Mao, Xiao Liu, Lei Zhang, and Wenchang Shi. Who is touching my cloud. In *Computer Security-ESORICS 2014*, pages 362–379. Springer, 2014.

[12] Zhangjie Fu, Fengxiao Huang, Xingming Sun, Athanasios Vasilakos, and Ching-Nung Yang. Enabling semantic search based on conceptual graphs over encrypted outsourced data. *IEEE Transactions on Services Computing*, 2016.

5. As noted in [36], there three types of traceability, namely: *private traceability*, *partial public traceability* and *fully public traceability*.

[13] Vipul Goyal. Reducing trust in the PKG in identity based cryptosystems. In *Advances in Cryptology-CRYPTO 2007*, pages 430–447. Springer, 2007.

[14] Vipul Goyal, Steve Lu, Amit Sahai, and Brent Waters. Black-box accountable authority identity-based encryption. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 427–436. ACM, 2008.

[15] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98. ACM, 2006.

[16] Qi Jing, Athanasios V. Vasilakos, Jiafu Wan, Jingwei Lu, and Dechao Qiu. Security of the internet of things: perspectives and challenges. *Wireless Networks*, 20(8):2481–2501, 2014.

[17] Allison Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In *Advances in Cryptology–EUROCRYPT 2012*, pages 318–335. Springer, 2012.

[18] Allison Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *Advances in Cryptology–EUROCRYPT 2010*, pages 62–91. Springer, 2010.

[19] Allison Lewko and Brent Waters. New proof methods for attribute-based encryption: Achieving full security through selective techniques. In *Advances in Cryptology–CRYPTO 2012*, pages 180–198. Springer, 2012.

[20] Jiguo Li, Xiaonan Lin, Yichen Zhang, and Jinguang Han. KSF-OABE: outsourced attribute-based encryption with keyword search function for cloud storage. *IEEE Trans. Services Computing*, 10(5):715–725, 2017.

[21] Jiguo Li, Wei Yao, Yichen Zhang, Huiling Qian, and Jinguang Han. Flexible and fine-grained attribute-based data storage in cloud computing. *IEEE Trans. Services Computing*, 10(5):785–796, 2017.

[22] Jin Li, Qiong Huang, Xiaofeng Chen, Sherman SM Chow, Duncan S Wong, and Dongqing Xie. Multi-authority ciphertext-policy attribute-based encryption with accountability. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, ASIACCS 2011*, pages 386–390. ACM, 2011.

[23] Jin Li, Kui Ren, and Kwangjo Kim. A2be: Accountable attribute-based encryption for abuse free access control. *IACR Cryptology ePrint Archive*, 2009:118, 2009.

[24] Jiaqiang Liu, Yong Li, Huandong Wang, Depeng Jin, Li Su, Lieguang Zeng, and Thanos Vasilakos. Leveraging software-defined networking for security policy enforcement. *Inf. Sci.*, 327:288–299, 2016.

[25] Qiang Liu, Hao Zhang, Jiafu Wan, and Xin Chen. An access control model for resource sharing based on the role-based access control intended for multi-domain manufacturing internet of things. *IEEE Access*, 5:7001–7011, 2017.

[26] Zhen Liu, Zhenfu Cao, and Duncan S Wong. 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*, pages 475–486. ACM, 2013.

[27] Zhen Liu, Zhenfu Cao, and Duncan S Wong. White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures. *IEEE Transactions on Information Forensics and Security*, 8(1):76–88, 2013.

[28] Ben Lynn et al. The pairing-based cryptography library. *Internet: crypto. stanford. edu/pbc/[Mar. 27, 2013]*, 2006.

[29] Dalit Naor, Moni Naor, and Jeff Lotspiech. Revocation and tracing schemes for stateless receivers. In *Advances in Cryptology - CRYPTO 2001*, pages 41–62. Springer, 2001.

[30] Jianting Ning, Zhenfu Cao, Xiaolei Dong, Junqing Gong, and Jie Chen. Traceable cp-abe with short ciphertexts: How to catch people selling decryption devices on ebay efficiently. In *Computer Security-ESORICS 2016*, pages 551–569. Springer, 2016.

[31] Jianting Ning, Zhenfu Cao, Xiaolei Dong, Kaitai Liang, Hui Ma, and Lifei Wei. Auditable -time outsourced attribute-based encryption for access control in cloud computing. *IEEE Transactions on Information Forensics and Security*, 13(1):94–105, 2018.

[32] Jianting Ning, Zhenfu Cao, Xiaolei Dong, and Lifei Wei. Traceable and revocable CP-ABE with shorter ciphertexts. *SCIENCE CHINA Information Sciences*, 59(11):119102:1–119102:3, 2016.

[33] Jianting Ning, Zhenfu Cao, Xiaolei Dong, and Lifei Wei. White-box traceable cp-abe for cloud storage service: How to catch people leaking their access credentials effectively. *IEEE Transactions on Dependable and Secure Computing*, 2016.

[34] Jianting Ning, Zhenfu Cao, Xiaolei Dong, Lifei Wei, and Xiaodong Lin. Large universe ciphertext-policy attribute-based encryption with white-box traceability. In *Computer Security-ESORICS 2014*, pages 55–72. Springer, 2014.

[35] Jianting Ning, Xiaolei Dong, Zhenfu Cao, and Lifei Wei. Accountable authority ciphertext-policy attribute-based encryption with white-box traceability and public auditing in the cloud. In *Computer Security–ESORICS 2015*, pages 270–289. Springer, 2015.

[36] Jianting Ning, Xiaolei Dong, Zhenfu Cao, Lifei Wei, and Xiaodong Lin. White-box traceable ciphertext-policy attribute-based encryption supporting flexible attributes. *IEEE Transactions on Information Forensics and Security*, 10(6):1274–1288, 2015.

[37] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 195–203. ACM, 2007.

[38] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology - EUROCRYPT '99*, pages 223–238, 1999.

[39] Yannis Rouselakis and Brent Waters. Practical constructions and new proof methods for large universe attribute-based encryption. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 463–474. ACM, 2013.

[40] Amit Sahai, Hakan Seyalioglu, and Brent Waters. Dynamic credentials and ciphertext delegation for attribute-based encryption. In *Advances in Cryptology–CRYPTO 2012*, pages 199–217. Springer, 2012.

[41] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *Advances in Cryptology–EUROCRYPT 2005*, pages 457–473. Springer, 2005.

[42] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *Public Key Cryptography–PKC 2011*, pages 53–70. Springer, 2011.

[43] Lifei Wei, Haojin Zhu, Zhenfu Cao, Xiaolei Dong, Weiwei Jia, Yunlu Chen, and Athanasios V. Vasilakos. Security and privacy for storage and computation in cloud computing. *Inf. Sci.*, 258:371–386, 2014.

[44] Hu Xiong, Kim-Kwang Raymond Choo, and Athanasios V Vasilakos. Revocable identity-based access control for big data with verifiable outsourced computing. *IEEE Transactions on Big Data*, 2017.

[45] Boyi Xu, Lida Xu, Hongming Cai, Lihong Jiang, Yang Luo, and Yizhi Gu. The design of an m-health monitoring system based on a cloud computing platform. *Enterprise IS*, 11(1), 2017.

[46] Fei Xu, Fangming Liu, Hai Jin, and Athanasios V. Vasilakos. Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions. *Proceedings of the IEEE*, 102(1):11–31, 2014.

[47] Zheng Yan, Xueyun Li, Mingjun Wang, and Athanasios V. Vasilakos. Flexible data access control based on trust and reputation in cloud computing. *IEEE Trans. Cloud Computing*, 5(3):485–498, 2017.

[48] Zheng Yan, Mingjun Wang, Yuxiang Li, and Athanasios V. Vasilakos. Encrypted data management with deduplication in cloud computing. *IEEE Cloud Computing*, 3(2):28–35, 2016.

[49] Kan Yang and Xiaohua Jia. Expressive, efficient, and revocable data access control for multi-authority cloud storage. *IEEE transactions on parallel and distributed systems*, 25(7):1735–1744, 2014.

[50] Kan Yang, Zhen Liu, Xiaohua Jia, and Xuemin Sherman Shen. Time-domain attribute-based access control for cloud-based video content sharing: A cryptographic approach. *IEEE Transactions on Multimedia*, 18(5):940–950, 2016.

[51] Yanjiang Yang, Joseph K Liu, Kaitai Liang, Kim-Kwang Raymond Choo, and Jianying Zhou. Extended proxy-assisted approach: achieving revocable fine-grained encryption of cloud data. In *Computer Security-ESORICS 2015*, pages 146–166. Springer, Cham, 2015.

[52] Shucheng Yu, Cong Wang, Kui Ren, and Wenjing Lou. Attribute based data sharing with attribute revocation. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, pages 261–270. ACM, 2010.

[53] Yong Yu, Liang Xue, Man Ho Au, Willy Susilo, Jianbing Ni, Yafang Zhang, Athanasios V. Vasilakos, and Jian Shen. Cloud data integrity checking with an identity-based auditing mechanism from RSA. *Future Generation Comp. Syst.*, 62:85–91, 2016.

[54] Jun Zhou, Zhenfu Cao, Xiaolei Dong, and Athanasios V. Vasilakos. Security and privacy for cloud-based iot: Challenges. *IEEE Communications Magazine*, 55(1):26–33, 2017.

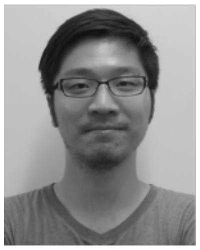
[55] Jun Zhou, Zhenfu Cao, Xiaolei Dong, Naixue Xiong, and Athanasios V. Vasilakos. 4s: A secure and privacy-preserving key management scheme for cloud-assisted wireless body area network in m-healthcare social networks. *Inf. Sci.*, 314:255–276, 2015.

[56] Jun Zhou, Xiaolei Dong, Zhenfu Cao, and Athanasios V. Vasilakos. Secure and privacy preserving protocol for cloud-based vehicular dtns. *IEEE Trans. Information Forensics and Security*, 10(6):1299–1314, 2015.

Jianting Ning received the Ph.D. degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University in 2016. He is currently a research fellow at Department of Computer Science, National University of Singapore. His research interests include applied cryptography and cloud security, in particular, Public Key Encryption, Attribute-Based Encryption, and Secure Multiparty Computation.

Zhenfu Cao (SM'10) received the Ph.D. degree in mathematics from Harbin Institute of Technology, Harbin, China, in 1999. His research interests mainly include Number Theory, Cryptography and Information Security. Up to now (since 1981), more than 400 academic papers have been published in Journals or conferences. He was exceptionally promoted to Associate Professor in 1987, became a Professor in 1991 and is currently a Distinguished Professor in East China Normal University, China. He also serves as a member of the expert panel of the National Nature Science Fund of China. He has received a number of awards, including the Youth Research Fund Award of the Chinese Academy of Science in 1986, the Ying-Tung Fok Young Teacher Award in 1989, the National Outstanding Youth Fund of China in 2002, the Special Allowance by the State Council in 2005. Prof. Cao is also the leaders of Asia 3 Foresight Program (61161140320) and the key project (61033014) of National Natural Science Foundation of China. He is a senior member of the IEEE.

Xiaolei Dong is a Distinguished Professor in East China Normal University. After her graduation with a doctorate degree from Harbin Institute of Technology, she pursued her postdoctoral study in SJTU from September 2001 to July 2003. Her primary research interests include Number Theory, Cryptography, Trusted Computing, etc. Her "Number Theory and Modern Cryptographic Algorithms" project won the first prize of China University Science and Technology Award in 2002. Her "New Theory of Cryptography and Some Basic Problems" project won the second prize of Shanghai Nature Science Award in 2007. Her "Formal Security Theory of Complex Cryptographic System and Applications" won the second prize of Ministry of Education Natural Science Progress Award in 2008.



Kaitai Liang received the PhD degree from the Department of Computer Science, City University of Hong Kong in 2014. He is currently a lecturer (assistant professor) with the Department of Computer Science, University of Surrey, U.K. His research interests are applied cryptography and information security in particular, encryption, network security, big data security, privacy-enhancing technology and security in cloud computing.

Lifei Wei received the B.Sc. and M.Sc. degrees in applied mathematics from University of Science and Technology Beijing, Beijing, China, in 2005 and 2007, respectively and the Ph.D. degree in computer science from Shanghai Jiao Tong University, Shanghai, China in 2013. He is currently an Assistant Professor with the Department of Information and Computing Sciences, Shanghai Ocean University. His research interests include applied cryptography, cloud computing, and wireless network security.

Kim-Kwang Raymond Choo received the Ph.D. in Information Security in 2006 from Queensland University of Technology, Australia. He currently holds the Cloud Technology Endowed Professorship at The University of Texas at San Antonio. He is the recipient of ESORICS 2015 Best Paper Award, Winning Team of the Germany's University of ErlangenNuremberg (FAU) Digital Forensics Research Challenge 2015, 2014 Highly Commended Award by the Australia New Zealand Policing Advisory Agency, Fulbright Scholarship in 2009, 2008 Australia Day Achievement Medallion, British Computer Society's Wilkes Award in 2008, etc. He is a Fellow of the Australian Computer Society, and a senior member of the IEEE.