

Verifiable Outsourced Attribute-Based Encryption Scheme for Cloud-Assisted Mobile E-Health System

Yinbin Miao , *Member, IEEE*, Feng Li , Xinghua Li , Jianting Ning , Hongwei Li , *Senior Member, IEEE*, Kim-Kwang Raymond Choo , *Senior Member, IEEE*, and Robert H. Deng , *Fellow, IEEE*

Abstract—The cloud-assisted mobile electronic health (e-health) system facilitates e-health data sharing between healthcare providers and patients, but also raises the security and privacy concerns of e-health data. Although Ciphertext-Policy Attribute-Based Encryption (CP-ABE) has been a promising technique to achieve fine-grained access control over encrypted e-health data, it still incurs high encryption and decryption burdens on mobile users such as smartphones and sensors. In addition, malicious cloud servers may conduct incorrect operations due to various interest incentives (e.g., leaking sensitive information to illegal users, saving computation and storage costs). To solve the above issues, in this paper we first propose an Outsourced CP-ABE (OABE) with verifiable encryption scheme by splitting secret keys corresponding to an attribute set and using the short signature, which not only reduces the encryption and decryption complexities of mobile users but also guarantees that cloud servers correctly perform encryption operations. Then, we extend OABE to construct outsourced CP-ABE with verifiable decryption (OABE+) by utilizing the verifiable tag mechanism, which guarantees that cloud servers correctly conduct the ciphertext transformation. Formal security analysis proves that our schemes are selectively secure against unauthorized accesses and malicious operations. Extensive experiments using various real-world datasets demonstrate that our schemes are efficient and feasible in real applications.

Index Terms—Mobile e-health system, outsourced CP-ABE, verifiable decryption, verifiable encryption, verifiable tag mechanism.

I. INTRODUCTION

THE mobile e-health system [1], [2] generally allows doctors to monitor patients' health status such as blood pressure and heart rate anywhere. The massive e-health data generated by mobile devices incur high storage and computation burdens [3]. Outsourcing e-health data to cloud servers is becoming a viable option, but the security and privacy of outsourced data become main concerns due to semi-honest cloud servers. For secure sharing of outsourced data in the multi-user setting, Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [4] has been a promising cryptographic primitive. Many outsourced CP-ABE schemes [5], [6], [7], [8], [9], [10] have been proposed. For example, a resource-constrained smart bracelet that can detect human heart rate is also very impractical to perform many encryption operations. Therefore, the smart bracelet can outsource data encryption operations to Huawei Cloud through data outsourcing. As Huawei Cloud may not be fully trusted, incorrect outsourcing encryption operations can be performed, thereby causing privacy issues. However, the above schemes are not extensively deployed in real applications due to two issues shown in Fig. 1.

The first issue of outsourced CP-ABE is that the encryption complexity grows with the number of attributes in access structures, which incurs high computation and storage burdens on resource-limited mobile users. Three solutions have been proposed to reduce the complexity of the encryption. The first solution uses Disjunctive Normal Form (DNF) [8], [11] to describe an access structure. Although it can improve the encryption efficiency to some extent as each clause in DNF contains a set of attributes, its encryption complexity still grows with the number of clauses in DNF. The second solution utilizes the aggregate algorithm [12], [13], [14] of broadcast encryption or AND-gate policy [15] to reduce the storage complexity of the encryption, but its computation complexity is still linear with the number of attributes. The third solution employs the offline/online encryption mechanism [16], [17] to perform heavy and lightweight operations in the offline phase and online phase respectively. This solution is suitable for resource-limited mobile users in the online phase, but does not reduce the total encryption complexity. Therefore, the existing solutions and

Manuscript received 8 October 2022; revised 17 April 2023; accepted 30 June 2023. Date of publication 4 July 2023; date of current version 11 July 2024. This work was supported by the National Natural Science Foundation of China under Grant 62072361, in part by the Key Research and Development Program of Shaanxi under Grant 2022GY-019, in part by the Shaanxi Fundamental Science Research Project for Mathematics and Physics under Grant 22JSY019, in part by the National Natural Science Foundation of China under Grants 62125205, 62032005, 61972094, in part by the Fundamental Research Funds for the Central Universities under Grant QTZX23091, in part by the Fellowship pf China Postdoctoral Science Foundation under Grant 2022T150507, in part by the Opening Project of Intelligent Policing Key Laboratory of Sichuan Province under Grant ZNJW2023KFMS002. The work of Kim-Kwang Raymond Choo was supported only by the Cloud Technology Endowed Professorship. (Corresponding author: Yinbin Miao.)

Yinbin Miao and Feng Li are with the School of Cyber Engineering, Xidian University, Xi'an, Shaanxi 710071, China (e-mail: ybmiao2014@163.com; feng.li@stu.xidian.edu.cn).

Xinghua Li is with the State Key Laboratory of Integrated Services Networks, School of Cyber Engineering, Xidian University, Xi'an, Shaanxi 710071, China (e-mail: xhli1@mail.xidian.edu.cn).

Jianting Ning is with the College of Computer and Cyber Security, Fujian Normal University, Fuzhou, Fujian 350007, China, and also with the City University of Macau, Macau, China (e-mail: jtning88@gmail.com).

Hongwei Li is with the Department of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan 610051, China (e-mail: hongweili@uestc.edu.cn).

Kim-Kwang Raymond Choo is with the Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249-0631 USA (e-mail: raymond.choo@fulbrightmail.org).

Robert H. Deng is with the School of Information Systems, Singapore Management University, Singapore 178902 (e-mail: robertdeng@smu.edu.sg).

Digital Object Identifier 10.1109/TDSC.2023.3292129

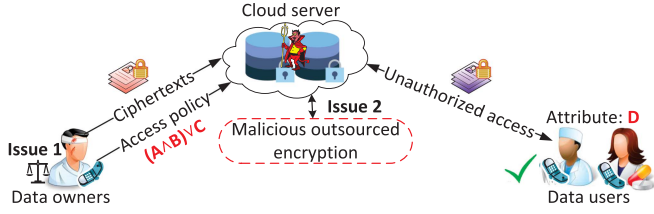


Fig. 1. Two issues of existing outsourced ABE schemes.

technologies cannot be adapted for mobile devices with limited resources.

The second issue is that existing outsourced CP-ABE schemes cannot prevent cloud servers from conducting incorrect encryption operations. In the traditional outsourced attribute-based encryption, the data owner will upload the partially encrypted data and the access structure to the cloud server simultaneously, so that the cloud server can judge whether the attributes of the data user match the access structure of encrypted data. Therefore, the cloud server can directly get access structure. In mobile cloud computing, the computing resources of data owner are very limited. To reduce the computing burden, the data owner will outsource partial encryption operations to the cloud server. In real applications, a malicious encryptor (e.g., data publisher, delegated cloud server) can allow unauthorized users to access the data by embedding illegal attributes into ciphertexts. For example, in an E-mail system, the data owner uses ABE to encrypt the mail and sends the intermediate ciphertexts to E-mail system. However, when the E-mail system is compromised by a hacker, the hacker may tamper with the access structure of the mail and generate the final ciphertexts with modified access structure, which makes the unauthenticated data users to access the email. As shown in Fig. 1, the original access structure for the data that is $(A \wedge B) \vee C$. However, in outsourced encryption, a malicious cloud server may modify the access structure of the data owner as $(A \wedge B) \vee D$, so that the user who has the unauthorized attribute D can successfully obtain the data. To solve this issue, three solutions have been proposed. The first solution uses the access control policy $\mathcal{P}(i, j) \rightarrow \{0, 1\}$ and non-interactive zero-knowledge proofs [18] to construct the access control encryption mechanism [19], [20], [21], [22], note that $\mathcal{P}(i, j) \rightarrow 1$ if the sender i is allowed to send information to receiver j . Unfortunately, this mechanism incurs high policy verification burdens when deployed in the multi-user setting. The second solution utilizes the sanitizable identity-based broadcast encryption and structure-preserving signature [23] to design the cross-domain access control encryption mechanism [24], which achieves the constant-size access control policy for a set of receivers. This mechanism needs to frequently update the encryption key when the set dynamically changes. The third solution uses the notion of sanitized ABE to construct a sanitizable access control mechanism [25], which achieves the access structure authenticity [26], but it still incurs high encryption burdens on data owners. As a result, the existing solutions cannot prevent cloud servers from performing incorrect encryption operations while accommodating mobile devices with limited resources.

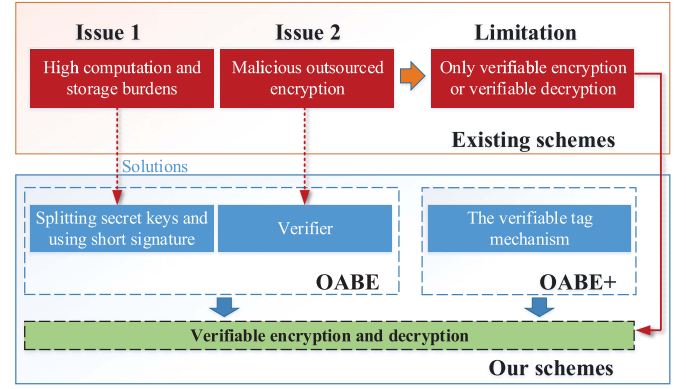


Fig. 2. Comparison between previous schemes and our schemes in encryption phase.

And the above solutions can only implement outsourced encryption verification or outsourced decryption verification, but cannot support them simultaneously.

From the above, the existing schemes are not suitable for some resource-limited terminals or applications that require highly real-time performance. As we can see in Fig. 2, the existing outsourced CP-ABE solutions cannot reduce the encryption complexity and verify the correctness of the encryption simultaneously. In addition, existing solutions cannot achieve both verifiable encryption and decryption. To solve these issues, we first propose an Outsourced CP-ABE (OABE) scheme with verifiable outsourced encryption in a cloud-assisted mobile e-health system by splitting secret keys and using short signature [27]. For achieving outsourced encryption, OABE distributes partial secret keys corresponding to each mobile user's attributes to cloud servers that will conduct most of the time-consuming encryption operations. To further achieve verifiable outsourced encryption, a verifier is introduced to verify the proof information generated by cloud servers and sanitize the final ciphertexts. Then, we improve OABE to support verifiable outsourced decryption (called OABE+) by utilizing the verifiable tag mechanism [28]. Compared with existing state-of-the-art schemes, we achieve both outsourced encryption verification and outsourced decryption verification, without incurring significant costs on resource-constrained data owners and data users. The contributions of our work are shown as follows:

- We first design an outsourced CP-ABE mechanism to support both outsourced encryption and decryption. Through splitting secret keys, our schemes allow resource-limited users to conduct constant operations and leave heavy operations to cloud servers.
- We then design a verifiable outsourced encryption mechanism to prevent malicious cloud servers from performing incorrect encryption operations. Our basic scheme OABE enables a public verifier to check the correctness of proof information generated by a cloud server, and also allows the public verifier to generate the sanitized ciphertexts for preventing unauthorized accesses.
- We utilize a verifiable tag generation mechanism to guarantee the correctness of ciphertext transformation conducted

by cloud servers, so that our extended scheme OABE+ achieves verifiable outsourced decryption.

The remaining of this paper is organized as follows: Section II reviews some related works. Section III presents the preliminaries involved in our schemes. Section IV defines the problem formulation consisting of the system model, problem definition, threat model and security model. Section V demonstrates the technical overview, and the constructions of our schemes. Section VI gives the formal security analysis of our schemes. Section VII demonstrates the theoretical complexities and experimental tests of our schemes by using various real-world datasets. Section VIII summarizes our work and discusses future research directions.

II. RELATED WORK

Due to its fine-grained access control over encrypted data, CP-ABE [29], [30] has been extensively deployed in real applications to prevent unauthorized access. There is a line of researches about CP-ABE in various aspects such as policy-hiding [31], expressive policy [32] and multi-authority [33]. In this section, we mainly focus on the outsourced CP-ABE and verifiable CP-ABE.

Outsourced CP-ABE: In a CP-ABE system, the data owners encrypt ciphertexts by embedding an access structures into ciphertexts, and data users can decrypt them if their attributes satisfy the specified access structures. However, the encryption and decryption complexities linearly grow with the number of policy attributes and user attributes, respectively. Thus, traditional CP-ABE schemes do not scale well in real applications due to heavy burdens on resource-limited mobile users. To eliminate burdens from mobile users, Green et al. achieved the outsourced decryption [5] by enabling the cloud server to conduct time-consuming operations (e.g., pairing operation) with a transformation key, which does not consider the limitation of user access permissions. Thus, Ning et al. proposed a σ -time outsourced CP-ABE [34] for limiting users' access privileges in a specified period. The above-outsourced decryption mechanisms just support serial computing. Then, Feng et al. used Spark and MapReduce to design a parallel outsourced decryption model that can be deployed in all CP-ABE schemes with tree-based access structures. Unfortunately, its access structure still leaks sensitive information of data. Thus, Zhang et al. used the Viète's formula [15] to construct the policy-hidden outsourced CP-ABE [35]. Wang et al. [36] proposed a CP-ABE-OD-DHP scheme that can hide access policy and alleviate the decryption burdens of data users, but cannot alleviate the computational burdens of data owners. To improve the practicability, outsourced CP-ABE has been extended to support keyword search [37] and policy update [38].

The CP-ABE with outsourced decryption still cannot reduce the encryption complexity. To solve this issue, Malluhi et al. utilized DNF rather than access tree or LSSS to specify access structures, thereby reducing the encryption complexity to some extent. But its encryption complexity still linearly increases with the number of clauses. Zhou et al. used the broadcast encryption [39] to design the attribute-based broadcast encryption

scheme [14], which achieves the constant storage complexity of the encryption. However, the above schemes still incur high computation complexity of the encryption, and thus are inapplicable for resource-limited mobile users. Then, Hohenberger et al. developed the online/offline CP-ABE scheme [40] by splitting the encryption algorithm into two phases. This solution allows data owners to conduct the majority of the encryption tasks in the offline phase and lightweight tasks in the online phase, but are vulnerable to undetectable vulnerabilities that can expose secrets. Thus, Ma et al. proposed a concessive online/offline CP-ABE scheme by using the cryptographic reverse firewalls [41], which resists the exfiltration of secrets. To support multi-authority setting, De et al. proposed a decentralized CP-ABE scheme [42] that achieves both offline/online encryption and outsourced decryption.

Verifiable CP-ABE: In practice, the cloud servers may maliciously conduct partial outsourced computations for saving computation and storage costs. Thus, it is vital to verify outsourced encryption and decryption processes. To guarantee the correctness of ciphertext transformation in the outsourced decryption process, Lai et al. proposed a verifiable outsourced decryption mechanism [9] by adding the random message ciphertext to the original message ciphertext, but incurs double ciphertext computation and storage overheads due to parallel encryption. Then, Lin et al. used the key encapsulation mechanism [28] and commitment scheme to construct a more efficient verifiable outsourced algorithm [43]. However, the above solutions do not allow offline users to timely check the correctness of outsourced decryption. Xu et al. designed a verifiable delegation mechanism [44] by using a monotone boolean circuit and authenticated encryption. To further verify the correctness of outsourced decryption for both authorized and unauthorized users, Li et al. proposed the full verifiability [10] for outsourced decryption. Although the verifiable outsourced decryption mechanisms guarantee the correctness of transformed ciphertexts, they cannot achieve verifiable outsourced encryption.

To solve this issue, Li et al. proposed an efficient ABE solution with verifiable outsourced encryption [46] by introducing an untrusted encryption service provider. For guaranteeing that the ciphertexts are just designed for authorized users, Fuchsbaauer et al. proposed a novel access control encryption primitive [20] by using the structure-preserving signature, which just supports the one-to-one model. Wang et al. constructed a cross-domain access control encryption mechanism [24] by utilizing the sanitizable identity-based broadcast encryption, and just supports the fixed user set. Then, Kim et al. designed a secure ABE scheme [26] that resists the fake policy attack in the multi-user setting. Unfortunately, this scheme cannot check the correctness of the encryption. Later, Susilo et al. presented a sanitizable access control mechanism [25] against data publishers by leveraging the sanitized ABE. This solution can guarantee that the encryption is correctly executed, which still incurs high encryption burdens on data owners. Unfortunately, the above schemes do not support outsourced encryption. Through outsourcing massive modular exponentiations to cloud servers, these solutions [45], [47] can eliminate data owners from computation burdens, but cannot check the correctness of outsourced

TABLE I
COMPARATIVE SUMMARY BETWEEN OUR SCHEMES
AND PREVIOUS SCHEMES

Schemes	Outsourced Encryption	Verifiable Encryption	Outsourced Decryption	Verifiable Decryption	Encryption Complexity	Decryption Complexity
[8]	✗	✗	✓	✓	$\mathcal{O}(m)$	$\mathcal{O}(1)$
[9]	✗	✗	✓	✓	$\mathcal{O}(2l)$	$\mathcal{O}(1)$
[11]	✗	✗	✗	✗	$\mathcal{O}(m)$	$\mathcal{O}(S_u)$
[14]	✗	✗	✗	✗	$\mathcal{O}(U)$	$\mathcal{O}(U)$
[25]	✗	✓	✗	✗	$\mathcal{O}(l)$	$\mathcal{O}(S_u)$
[26]	✗	✓	✗	✗	$\mathcal{O}(l)$	$\mathcal{O}(S_u)$
[34]	✗	✗	✓	✓	$\mathcal{O}(l)$	$\mathcal{O}(1)$
[35]	✗	✗	✓	✗	$\mathcal{O}(U)$	$\mathcal{O}(1)$
[28]	✗	✗	✓	✓	$\mathcal{O}(l)$	$\mathcal{O}(1)$
[45]	✓	✗	✓	✗	$\mathcal{O}(1)$	$\mathcal{O}(1)$
OABE	✓	✓	✓	✗	$\mathcal{O}(1)$	$\mathcal{O}(1)$
OABE+	✓	✓	✓	✓	$\mathcal{O}(1)$	$\mathcal{O}(1)$

Notes. “ l ”: Number of attributes in access structure using LSSS; “ m ”: Number of clauses in access structure using DNF; “ $|U|$ ”: Number of attributes in the system.

encryption by malicious CSP. To solve the above challenging issues, we propose two outsourced ABE schemes with verifiable encryption and decryption respectively. Compared with existing solutions, our schemes have versatile features in Table I. The outsourced CP-ABE algorithm mainly studies how to reduce the computing burden of the client. Therefore, the encryption complexity and decryption complexity respectively refer to the computation burdens of data owner and data user in Table I.

III. PRELIMINARIES

In this section, we review some cryptographic background associated with the our schemes.

A. Bilinear Map

Let $\mathbb{G}, \mathbb{G}_T$ are two cyclic groups of prime order p , e is the bilinear map $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ and g is the generator of $\mathbb{G}$. The bilinear map has three properties:

- *Bilinearity*: $e(g^a, g^b) = e(g, g)^{ab}$ holds for all $a, b \in \mathbb{Z}_p^*$, where $\mathbb{Z}_p$ is a field of prime order p .
- *Non-degeneracy*: $e(g, g) \neq 1$.
- *Computability*: There is an efficient algorithm to compute $e(g, g)$.

B. Ciphertext-Policy Attribute-Based Encryption

A ciphertext-policy attribute based encryption scheme [4] consists of four fundamental algorithms: Setup, KeyGen, Enc, and Dec.

- **Setup**. The setup algorithm takes no input other than the implicit security parameter. It outputs the public key pk and a master key msk .
- **KeyGen(msk, S_u)**. The KeyGen algorithm takes as input the master key msk and a set of attributes S_u . It outputs a secret key sk .
- **Enc(pk, M, Γ)**. The Enc algorithm takes as input the public key pk , a message M , and an access structure Γ over the authorized attributes. The algorithm will encrypt M and produce a ciphertext cph such that only a data user

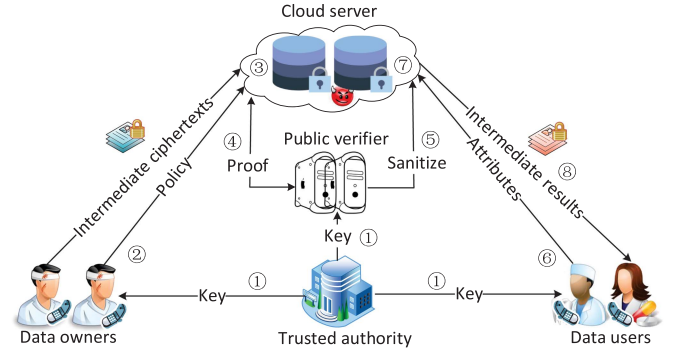


Fig. 3. System model of our schemes.

whose attributes satisfy the access structure will be able to decrypt the message. It is worth noticing that the ciphertext implicitly contains Γ .

- **Dec(pk, cph, sk)**. The Dec algorithm takes as input the public key pk , a ciphertext cph containing an access structure Γ , and a secret key sk corresponding to attributes S_u . If the attribute set S_u satisfies the access structure Γ , this algorithm will successfully decrypt the ciphertext and return a message M .

IV. PROBLEM FORMULATION

In this section, we define the system model, problem definition, threat model and security model.

A. System Model

In this paper, we consider a cloud-assisted mobile e-health system. As depicted in Fig. 3, the system model consists of five entities, namely Trusted Authority (TA), Data Owner (DO), Public Verifier (PV), Cloud Server (CS) and Data Users (DUs). The role of each entity is defined as follows:

- TA is responsible for initializing the system and generating keys for all entities including DO, DUs and PV.
- DO specifies an access structure and then uses it to encrypt e-health data, finally uploads the intermediate ciphertexts to CS.
- CS re-encrypts the intermediate ciphertexts according to DO's access structure and then sends the final ciphertexts to PV. CS also needs to generate the proof information in response to PV's challenge. Besides, CS partially decrypts the sanitized ciphertexts and returns the intermediate results to authorized DUs.
- PV first checks whether CS has correctly generated the final ciphertexts by sending the challenging information to it, and then returns the sanitized ciphertexts to CS.
- DUs make queries, and decrypt the intermediate results to obtain the plaintext data.

Fig. 3 shows the whole process of our scheme, which consists of eight steps. Step ①: TA generates the secret keys for CS, DO and DUs, and it also generates the public/secret key pair for PV. Step ②: For sharing e-health data with authorized DUs, each DO first specifies an access structure and generates intermediate

ciphertexts of e-health data, then sends them to CS. Step ③: According to the access structure, CS converts the intermediate ciphertexts into final ciphertexts. Step ④: To check whether CS has correctly generated the final ciphertexts based on the specified access structure, PV sends the challenging attribute set to CS that is required to return the proof information and final ciphertexts. Step ⑤: Upon receiving the final ciphertexts and proof information, PV first checks the validity of proof information. If the proof information is valid, PV then generates the sanitized ciphertexts to further prevent unauthorized access. Step ⑥: To access encrypted e-health data, a certain DU needs to submit his attributes to CS. Step ⑦: If DU's attributes satisfy the specified access structure, CS performs partial decryption operations. Step ⑧: CS returns the intermediate results to DU.

B. Problem Definition

In a CP-ABE system, the encryption and decryption complexities depend on the access structure specified by DO. Prior to defining the problems of existing CP-ABE schemes, we first introduce LSSS (Linear Secret Sharing Scheme) Π [48], [49] which is induced from a monotone span program [50] and can specify any access formula over attributes. Specifically, we embed the secret value s into a labeled matrix $(M_{l \times n}, \rho(\cdot))$ and treat this matrix as an access structure Γ , and thus any authorized DU can recover s if his attributes satisfies Γ . The secret-sharing scheme Π over an attribute set U is called linear over the field $\mathbb{Z}_p$ if it has the following features:

- The shares for each attribute form a vector over $\mathbb{Z}_p$.
- For a share-generating matrix M with l rows and n columns, its i -th row M_i is labeled by an attribute $\rho(i)$. Let $s \in \mathbb{Z}_p^*$ be a secret, and $\vec{x} = (s, y_2, \dots, y_n) \in \mathbb{Z}_p^n$ be a random column vector. For any authorized attribute set S that satisfies Γ , we can get the shares $\{\lambda_i = M_i \vec{x}\}_{i \in I}$, where $I = \{i | \rho(i) \in S\}$. We also can use the authorized attribute set to compute the reconstruction constants $\{w_i\}_{i \in I}$ satisfying (1), such that $\sum_{i \in I} w_i M_i = \vec{e}$. Finally, we can recover the secret value s by (2).

$$\left\{ \begin{aligned} \sum_{i \in I} w_i M_i &= \vec{e} = (1, 0, \dots, 0). \\ \sum_{i \in I} w_i \lambda_i &= \sum_{i \in I} w_i (M_i \vec{v}) = \vec{e} \cdot \vec{v} = s. \end{aligned} \right. \quad (1) \quad (2)$$

It is worth noticing that any authorized DU whose attribute set satisfies Γ can gain the target vector $\vec{e} = (1, 0, \dots, 0)$ in the span of I , but the unauthorized DUs cannot. To facilitate understanding, we show an example with setting $\mathbb{Z}_p = \mathbb{Z}_{47}$. Given the system attribute set $U = \{A, B, C, D\}$ and access structure $\Gamma = A \wedge ((B \wedge C) \vee D)$, we choose a random vector $\vec{v} = (5, 6, 3)$ and then utilize the labeled matrix $(M_{4 \times 3}, \rho(\cdot))$ to represent Γ in Fig. 4, where “5” is the secret value. An authorized DU with attributes (A, D) first finds two reconstruction constants $w_1 = 2, w_2 = 46$ such that $\sum_{i \in \{1, 2\}} w_i M_i = \vec{e} = (1, 0, 0)$, then recovers the secret value $s = 5 = \vec{e} \cdot \vec{v}$.

From the above discussion, we notice that the encryption and decryption complexities are linear to the variables $l, |I|$, respectively. With higher values of $l, |I|$, the CP-ABE system incurs higher computation burdens on mobile users. Outsourcing time-consuming computation operations to cloud servers

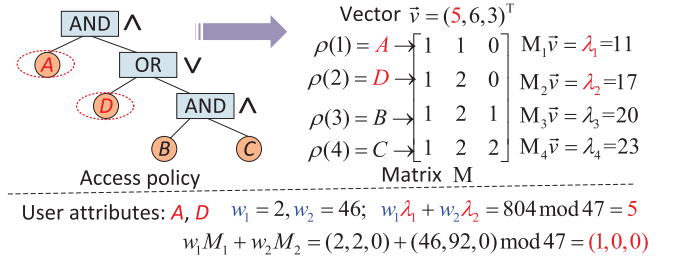


Fig. 4. Example of access structure.

reduces the burdens of mobile users, but does not guarantee that cloud servers correctly conduct outsourced encryption and decryption operations as cloud servers may be malicious. For example, due to various interest incentives such as leaking sensitive information to unauthorized mobile users and saving computation resources, CS can embed illegal attributes in ciphertexts in the outsourced encryption process, and even conduct incorrect ciphertext transformation in the outsourced decryption process. Thus, it is still necessary to provide a verification mechanism for checking the correctness of outsourced encryption and decryption. There are many traditional verifiable techniques such as signatures, Merkle hash trees and accumulators can be used to verify data integrity and data correctness. For example, the signature and Merkle hash tree are mainly used to verify whether the original data has been tampered. The accumulator not only supports data integrity verification, but also can be used to verify whether the cloud server has performed the operation correctly. However, the accumulator will involve the operation of expensive bilinear pairing operations, which brings heavy computational burden to the data user. If directly deploying traditional verifiable techniques on existing outsourced CP-ABE schemes, the problems of verifiable outsourcing encryption and outsourcing decryption cannot be solved simultaneously. Furthermore, these traditional mechanisms may result in large ciphertext size or computational overhead. Thus, our scheme aims to simultaneously check the correctness of outsourced encryption and decryption conducted by the malicious cloud server, which can mitigate computation burdens from data owners and data users.

C. Threat Model

In our schemes, TA and DOs are fully trusted. PV is assumed to be *honest-but-curious*, which honestly performs the established protocols but may use background information to deduce some sensitive information. Most of the CP-ABE schemes assume that CS is a *honest-but-curious* entity. However, we consider that CS is *malicious*, which may perform incorrect outsourced encryption or even return incorrect ciphertext transformation due to various interest incentives. Besides, considering the problem of key collusion attacks, consistent with most CP-ABE schemes [4], we consider that DUs are *malicious*. It is worth noticing that malicious DUs cannot collude with CS. The potential security threats caused by PV, CS and DUs are shown as follows:

- *Data confidentiality threats*: As DOs lose direct control over their e-health data, PV, CS and unauthorized DUs may try to deduce as much plaintext information as possible, thereby incurring privacy leakage.
- *Unauthorized outsourced encryption*: CS may perform outsourced encryption maliciously for leaking sensitive information to unauthorized DUs.
- *Incorrect outsourced decryption*: CS may maliciously output incorrect intermediate results for saving storage and computation overheads.
- *Key collusion attacks*: Malicious DUs with various attribute sets may try to collude with each other to gain the new valid secret key, which incurs unauthorized accesses.

D. Security Model

For guaranteeing the data confidentiality, the unauthorized DUs cannot decrypt the sanitized ciphertexts. Similar to the CP-ABE scheme [48], our schemes allow any adversary $\mathcal{A}$ to make secret key generation queries that cannot be utilized to decrypt the challenging ciphertext. We formalize this security using the following security game in which $\mathcal{A}$'s attribute set S_u does not satisfy the access structure Γ^* in the challenge phase. The security game simulated between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ is demonstrated as follows:

Setup: Given the security parameter λ , $\mathcal{C}$ runs Setup to generate the public key pk and master key msk , and then sends pk to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ adaptively makes multiple queries to the following oracles:

- $\mathcal{O}_{KeyGen}$: $\mathcal{A}$ submits an attribute set S_u of an mobile user u (including DO and DU). $\mathcal{C}$ calls $KeyGen_u$ to the secret key $sk = (sk_{CS,u}, sk_u)$ and then sends sk to $\mathcal{A}$. We denote the attribute sets queried by $\mathcal{A}$ as $\mathbb{S} = \{S_u\}$.
- $\mathcal{O}_{Enc}$: $\mathcal{A}$ is required to submit a message $\mathcal{M}$, DO's attribute set S_{DO} and an access structure $\Gamma = (M_{l \times n}, \rho(\cdot))$ including an attribute set $\{\rho(i)\} \subseteq S_{DO}$, where $i \in [1, l]$. $\mathcal{C}$ calls Enc to generate the final ciphertext cph , and then sends cph to $\mathcal{A}$.
- $\mathcal{O}_{Sanitize}$: $\mathcal{A}$ submits the tuple $(\{\rho(i)\}, cph)$. $\mathcal{C}$ needs to run Sanitize to check whether CS is authorized to generate cph . If CS is not authorized, $\mathcal{C}$ aborts this process. Otherwise, $\mathcal{C}$ returns the sanitized ciphertext $\overline{cph}$ to $\mathcal{A}$.

Challenge: $\mathcal{A}$ submits two challenging messages $\mathcal{M}_0, \mathcal{M}_1$ with the same length, an access structure $\Gamma^* = (M_{l^* \times n^*}, \rho^*(\cdot))$ and DO's attribute set S_{DO}^* . The restriction is that any $S_u \in \mathbb{S}$ does not satisfy Γ^* . $\mathcal{C}$ first chooses a random bit $b \in \{0, 1\}$, then successively runs $KeyGen_u$, Enc and Sanitize to generate the sanitized ciphertext $\overline{cph}^*$ for $\mathcal{A}$.

Phase 2: This phase is the same as *Phase 1*, but the requirement is that $S_u \in \mathbb{S}$ does not satisfy Γ^* in the oracle $\mathcal{O}_{KeyGen}$.

Guess: $\mathcal{A}$ outputs a guess bit $b' \in \{0, 1\}$. We say that $\mathcal{A}$ wins the above security game if $b' = b$.

Definition 1: The CP-ABE schemes are secure against unauthorized accesses issued by illegal DUs if the adversary $\mathcal{A}$'s advantage $Adv_{\mathcal{A}}$ in breaking the above security game is negligible, namely $Adv_{\mathcal{A}} = |Pr[b = b'] - \frac{1}{2}| < \epsilon$, where ϵ is a negligible function.

Our schemes are selectively secure if we add *Init* phase prior to *Setup* phase in the above security game. In *Init* phase, the adversary is allowed to submit the challenging access structure $\Gamma = (M_{l \times n}, \rho(\cdot))$ before obtaining the public key. Thus, our schemes can be proved secure in the selective security model.

The outsourced encryption can reduce DO's computation burdens, but causes privacy leakage if CS maliciously performs outsourced encryption. Thus, our schemes should guarantee that only authorized CS can perform outsourced encryption. We formalize the security by the following security game simulated between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

Setup: Given the security parameter λ , $\mathcal{C}$ runs Setup to generate the public key pk and master key msk , and then sends pk to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ adaptively makes queries to the following oracles:

- $\mathcal{O}_{KeyGen}$: $\mathcal{A}$ submits an attribute set S_u of an mobile user u (including DO and DU). $\mathcal{C}$ calls $KeyGen_u$ to generate the secret key $sk = (sk_{CS,u}, sk_u)$ and then sends sk to $\mathcal{A}$. We denote the attribute sets queried by $\mathcal{A}$ as $\mathbb{S} = \{S_u\}$.
- $\mathcal{O}_{Enc}$: $\mathcal{A}$ is required to submit a message $\mathcal{M}$, DO's attribute set S_{DO} and an access structure $\Gamma = (M_{l \times n}, \rho(\cdot))$ including an attribute set $\{\rho(i)\} \subseteq S_{DO}$, where S_{DO} is the DO's attribute set. $\mathcal{C}$ calls Enc to generate the final ciphertext cph , and then sends cph to $\mathcal{A}$, note that the set of $\{cph\}$ queried by $\mathcal{A}$ is set as $\mathbb{C}$.
- $\mathcal{O}_{Sanitize}$: $\mathcal{A}$ is asked to submit the two-tuple $(\{\rho(i)\}, cph)$. $\mathcal{C}$ runs Sanitize to check whether CS correctly generates cph . If CS is not authorized, $\mathcal{C}$ aborts this process. Otherwise, $\mathcal{C}$ returns the sanitized ciphertexts $\overline{cph}$ to $\mathcal{A}$.

Challenge: $\mathcal{A}$ is allowed to submit a message $\mathcal{M}^*$, an access structure $\Gamma^* = (M_{l^* \times n^*}, \rho^*(\cdot))$ associated with attributes $\{\rho^*(i)\}$ and DO's attribute set S_{DO}^* , but the restriction is that $S_{DO}^* \notin \mathbb{S}$. $\mathcal{C}$ successively calls $KeyGen_u$ and Enc to generate the final ciphertext $cph^* \notin \mathbb{C}$.

Phase 2: This phase is the same as *Phase 1*, and the requirement is that $S_{DO} \notin \mathbb{S}$ in the oracle $\mathcal{O}_{KeyGen}$ and $cph^* \notin \mathbb{C}$ in the oracle $\mathcal{O}_{Enc}$.

Guess: $\mathcal{A}$ outputs the proof information $proof^*$ and wins this security game if $proof^*$ passes the verification mechanism in Sanitize.

Definition 2: The CP-ABE schemes guarantee that the outsourced encryption is correctly performed if the adversary $\mathcal{A}$'s proof information $proof^*$ is valid with a negligible advantage.

Our schemes are selectively secure if we add *Init* phase prior to *Setup* phase in the above security game. In *Init* phase, the adversary is allowed to submit the attributes that it intends to query in the oracles including $\mathcal{O}_{KeyGen}$, $\mathcal{O}_{Enc}$ and $\mathcal{O}_{Sanitize}$. Thus, our schemes can be proved secure in the selective security model. In fact, our scheme can achieve a higher level of security by only modifying the underlying ABE framework. For example, we can

use an optimized matrix-based CP-ABE scheme to deploy our scheme, which further improves the security level. Waters et al. [48] proposed a new approach, which achieves expressive access control by using a Linear Secret Sharing Scheme (LSSS) matrix M over the attributes in the system. Meanwhile, Waters et al. provided two other constructions that could achieve provable security under weaker assumptions including Decisional-Bilinear Diffie-Hellman Exponent (DBDHE) and Decisional Bilinear Diffie-Hellman (DBDH) assumptions. However, our schemes do not focus on the original ABE framework, our interest focuses on how to solve the problem of outsourced cryptographic verification and outsourced decryption verification of the full process under the ABE framework. That is, our schemes can achieve higher security by just using available ABE framework, but in this paper we focus on providing verification for both outsourced encryption and decryption.

V. OUR PROPOSED SCHEMES

In this section, we first introduce the technical overview of our schemes, then demonstrate their concrete constructions.

A. Technical Overview

Outsourced encryption and decryption: CP-ABE [4] is a promising technology to implement fine-grained access control, in which the encryption and decryption overheads linearly increase with the complexity of access formula over attributes. To the best of our knowledge, two corresponding technologies can partially solve these problems. Regarding the outsourced encryption, the online/offline mechanism [40], [41], [51] preprocesses the time-consuming encryption operations in the offline phase, and enables DOs to execute lightweight operations in the online phase. In terms of outsourced decryption, the transformation key [7], [10], [34] allows CS to perform the outsourced decryption, which relieves DUs from high decryption burdens. The naive solution is to combine the above two technologies, but cannot reduce DOs' computation complexities in the entire encryption process. Thus, it is still a challenging issue to design an efficient outsourced CP-ABE mechanism for both resource-limited DOs and DUs.

To solve this issue, we show the main idea of our technology as follow. DO first employs the LSSS scheme to generate the secret value $s \in \mathbb{Z}_p^*$, then encapsulates it in the term $e(g, g)^{s\alpha}$ for security concerns, where $\alpha \in \mathbb{Z}_p^*$ is a master key initialized by TA. To achieve outsourced encryption and decryption, TA first splits α into $\alpha_1, \alpha_2 \in \mathbb{Z}_p^*$ for achieving secret sharing, then uses α_1, α_2 to respectively generate secret keys for CS and mobile users $\{u\}$ (including DOs and DUs) based on their attributes S , which are defined by (3), where $t_u \in \mathbb{Z}_p^*$, $h_x \in \mathbb{G}$.

$$\begin{cases} \text{sk}_{CS,u} = (g^{\alpha_1} g^{at_u}, g^{t_u}, \{h_x^{t_u}\}_{x \in S}); \\ \text{sk}_u = g^{\alpha_2} g^{at_u}, \end{cases} \quad (3)$$

In a nutshell, the mobile user u first executes the constant encryption operations by embedding s in the term $e(g, g)^{s\alpha} = e(g, g)^{s(\alpha_1 + \alpha_2)}$, where $e(g, g)^\alpha$ is a public parameter. Then, CS conducts the time-consuming encryption operations to generate the final ciphertexts. In the decryption phase, CS uses the key

$\text{sk}_{CS,u}$ to perform heavy decryption operations, and deduces the intermediate result $e(g, g)^{s\alpha_1} / e(g, g)^{ast_u}$. However, it cannot compute the record encryption key K_R blinded by $e(g, g)^{s\alpha}$. The authorized DU can use his secret key sk_u to compute $e(g, g)^{s\alpha}$ by (4), where g^s is the returned ciphertext component and $a \in \mathbb{Z}_p^*$.

$$e(g, g)^{s\alpha} = \frac{e(g, g)^{s\alpha_1}}{e(g, g)^{ast_u}} e(\text{sk}_u, g^s). \quad (4)$$

Verifiable outsourced encryption and decryption: When implementing the outsourced encryption, DO first specifies the access structure $\Gamma = (M_{l \times n}, \rho(\cdot))$ and the column vector $\vec{v} = \{s, y_2, \dots, y_n\}$, then generates the intermediate ciphertexts $C = \text{Me}(g, g)^{\alpha s}$, $C' = g^s$ and computes $\lambda_i = M_i \vec{v}$. To prevent CS from generating the final ciphertexts that are based on incorrect access structure $\Gamma^* = (M_{l^* \times n^*}, \rho^*(\cdot))$, we introduce a ciphertext sanitization mechanism conducted by PV. Specifically, PV randomly chooses an attribute set S^* satisfying Γ and sends it to CS. CS uses its key $\text{sk}_{CS,DO}$ delegated by DO to generate the proof information. If the proof information is valid, CS correctly outputs the final ciphertexts. Otherwise, CS outputs the invalid ciphertexts. Suppose CS uses the forged Γ^* rather than Γ specified by DO to generate the final ciphertext $C_i^* = g^{a\lambda_i^*} h_{\rho^*(i)}^{-r_i^*}$, $D_i^* = g^{r_i^*}$, then it cannot recover $e(g, g)^{s\alpha_1}$ by (5). Accordingly, CS cannot cancel $e(g, g)^{s\alpha}$ by (4). That is to say, CS cannot share the sensitive information with each unauthorized DU with attributes S^* . Our schemes can guarantee that CS correctly generates the final ciphertexts only if it has authorized by DO. Thus, our schemes just need to check whether CS has been authorized by using the short signature scheme [27].

$$\frac{e(C', g^{\alpha_1} g^{at_u})}{\prod_{i \in I} (e(C_i^*, g^{t_u}) e(D_i^*, h_{\rho^*(i)}^{t_u}))^{w_i}} = \frac{e(g, g)^{s\alpha_1} e(g, g)^{ast_u}}{e(g, g)^{as^* t_u}}. \quad (5)$$

To prevent the malicious CS from returning previous or modified intermediate results, we utilize the verifiable outsourced decryption mechanism [28]. Its main idea is to generate a session key $r_R \in \mathbb{G}_T$ for each e-health record R and encrypt it as C_R by using the traditional symmetric encryption algorithm such as AES. Instead of directly using the session key r_R as the symmetric key to encrypt R , we first map it to a short string $S_R = H_0(r_R)$ by using a hash function H_0 , note that $H_0(r_R)$ will be used to check the correctness of session key r_R . However, the single hash value $H_0(r_R)$ will make r_R non-uniform. Then, we use a key extractor H_1 to extract a nearly uniform symmetric encryption key $K_R^* = H_1(r_R)$ which is used to encrypt R as $C_R^* = \text{Enc}_{K_R^*}(R)$. Finally, DO outputs the tag $\text{tag} = H_2(S_R || C_R^*)$ by using the hash function H_2 , where " $||$ " is the concatenation operation. It is worth noticing that tag is utilized to check the correctness of outsourced decryption, and H_0, H_1, H_2 are defined by (6).

$$\begin{aligned} H_0 : \{0, 1\}^* &\rightarrow \{0, 1\}^{l_0}, H_1 : \{0, 1\}^* \rightarrow \{0, 1\}^{l_1}; \\ H_2 : \{0, 1\}^* &\rightarrow \{0, 1\}^{l_2}. \end{aligned} \quad (6)$$

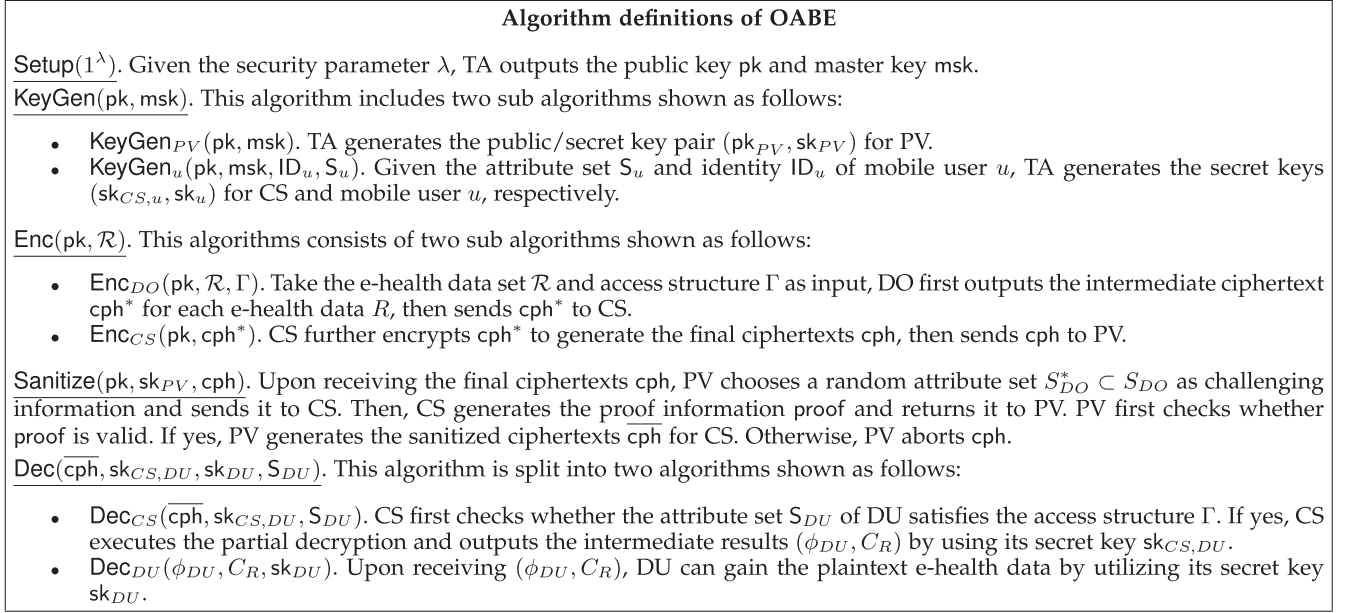


Fig. 5. Algorithm definitions of our OABE.

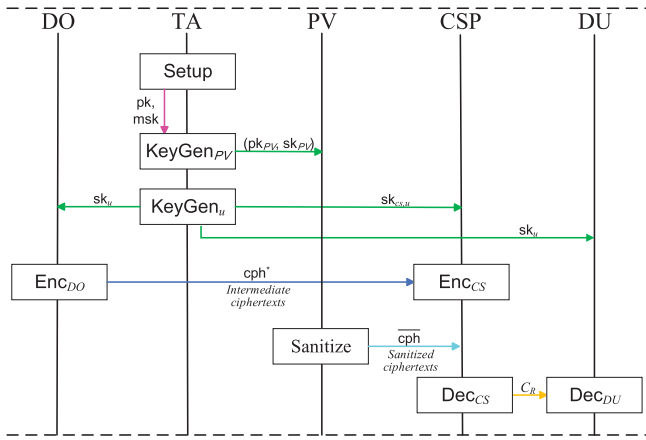


Fig. 6. Framework of OABE.

B. Construction of Outsourced CP-ABE With Verifiable Encryption (OABE)

In this section, we attempt to design an outsourced CP-ABE with verifiable encryption (OABE) scheme, in which CS is delegated to execute time-consuming operations on behalf of DO and DUs. As mentioned in the threat model, CS and DUs may be malicious, OABE should prevent CS from leaking sensitive information to unauthorized DUs by conducting incorrect outsourced encryption. The construction of OABE is composed of five algorithms including Setup, KeyGen, Enc, Sanitize and Dec, note that both Enc and Dec are divided into two sub algorithms conducted by mobile user and CS, respectively. The algorithm definition of OABE is shown in Fig. 5 and the framework of OABE is shown in Fig. 6.

The concrete construction of OABE is demonstrated as follows:

Setup (1^λ). Given the security parameter λ , TA runs this algorithm to output the public Bilinear parameters $\mathcal{BP} = (\mathbb{G}, \mathbb{G}_T, e, p, g)$, and chooses random elements $h_1, \dots, h_{|U|} \in \mathbb{G}$ for the system attribute set U , where $|U|$ is the size of system attributes. Then, TA chooses random elements $h, w \in \mathbb{G}$, $\alpha, \alpha_1, \alpha_2, a, b \in \mathbb{Z}_p^*$ and a collision-resistant hash function $H : \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$, where $\alpha_1 + \alpha_2 = \alpha$. Finally, TA computes $e(g, g)^\alpha, e(g, g)^{\alpha_1}, g^a, g^b$, and outputs the public key pk and master key msk by (7)

$$\begin{aligned} pk &= (\mathcal{BP}, e(g, g)^\alpha, e(g, g)^{\alpha_1}, h, w, g^a, g^b); \\ msk &= (\alpha, \alpha_1, \alpha_2, b). \end{aligned} \quad (7)$$

KeyGen (pk, msk). Given the public key pk and master key msk , TA needs to generate keys for both PV and mobile user u . The two sub algorithms are shown as follows:

- 1) **KeyGen_{PV}**(pk, msk). TA chooses a random element $c \in \mathbb{Z}_p^*$ and generates the public/secret key pair (pk_{PV}, sk_{PV}) for PV, which is shown by (8).

$$pk_{PV} = g^c, sk_{PV} = c. \quad (8)$$

- 2) **KeyGen_u**(pk, msk, ID_u, S_u). Let ID_u and S_u be respectively the identity and attribute sets of mobile user u such as DO and DU. When u registers with the system, TA first selects a random element $t_u \in \mathbb{Z}_p^*$ and generates the complete secret key $sk = (sk_{CS,u}, sk_u)$ by modifying (3). The secret key component sk_u is sent to the mobile user u , and the secret key component $sk_{CS,u} = (K_u, L_u, R_u, \{K_{u,x}\}_{x \in S_u})$ is sent to CS. The secret key sk is defined by (9).

$$\begin{aligned} K_u &= g^{\alpha_1} g^{at_u} w^{1/(b+H(ID_u))}, L_u = g^{t_u}, \\ R_u &= g^{1/(b+H(ID_u))}, K_{u,x} = h_x^{t_u}; \end{aligned}$$

Algorithm 1: The Specific Process of Enc.

Input: Public keys pk , e-health data set $\mathcal{R}$, access structure Γ
Output: Final ciphertexts cph

1 **DO side:**
2 **foreach** $R \in \mathcal{R}$ **do**
3 Choose a symmetric key $K_R \in \mathbb{G}_T$;
4 Encrypt R as $C_R = \text{Enc}_{K_R}(R)$;
5 Choose a vector $\vec{v} = (s, y_2, \dots, y_n)^\top \in \mathbb{Z}_p^n$;
6 Compute (C, C', C'') by Eq. 10 ;
7 **for** $i = 1; i \leq l; i++ = 1$ **do**
8 Compute $\lambda_i = M_i \vec{v}$;
9 Set the intermediate ciphertexts as
 $\text{cph}^* = (\Gamma, C_R, C, C', C'', \{\lambda_i\})$;
10 **CS side:**
11 **for** $i = 1; i \leq l; i++ = 1$ **do**
12 Choose a random element $r_i \in \mathbb{Z}_p^*$;
13 Compute $\{C_i, D_i\}$ by Eq. 11 ;
14 Set the final ciphertexts as $\text{cph} = (\Gamma, C_R, C, C', C'', \{C_i, D_i\})$;
15 **return** cph .

$$\text{sk}_u = g^{\alpha_2} g^{at_u}. \quad (9)$$

$\text{Enc}(\text{pk}, \mathcal{R})$. Given the e-health data set $\mathcal{R} = \{R\}$ and public key pk , this encryption process shown in Algorithm 1 is divided into two sub algorithms. DO conducts the lightweight encryption operations, and outsources the time-consuming operations to CS. The concrete sub algorithms are demonstrated as follows:

- 1) $\text{Enc}_{DO}(\text{pk}, \mathcal{R}, \Gamma)$. For each e-health data $R \in \mathcal{R}$, DO first chooses a key $K_R \in \mathbb{G}_T$ and encrypts R as $C_R = \text{Enc}_{K_R}(R)$ by using the traditional symmetric encryption algorithm Enc such as AES. Then, DO encrypts K_R by specifying an access structure $\Gamma = (M_{l \times n}, \rho(\cdot))$ for preventing unauthorized access permissions. Specifically, DO first chooses a random vector $\vec{v} = (s, y_2, \dots, y_n)^\top \in \mathbb{Z}_p^n$ and computes $\lambda_i = M_i \vec{v}$, then computes (C, C', C'') by (10). Finally, DO sends the intermediate ciphertexts $\text{cph}^* = (\Gamma, C_R, C, C', C'', \{\lambda_i\})$ to CS.

$$\begin{aligned} C &= K_R \cdot e(g, g)^{\alpha_s} \cdot e(g, \text{pk}_{PV})^s; \\ C' &= g^s, C'' = w^s \text{pk}_{PV}^s. \end{aligned} \quad (10)$$

- 2) $\text{Enc}_{CS}(\text{pk}, \text{cph}^*)$. Upon receiving each record ciphertext uploaded by DO, CS first chooses a random element $r_i \in \mathbb{Z}_p^*$ and computes $\{C_i, D_i\}$ by (11), where $i \in [1, l]$, $\{\rho(i)\} \subset S_{DO}$. Finally, CS stores the final ciphertexts $\text{cph} = (\Gamma, C_R, C, C', C'', \{C_i, D_i\})$.

$$C_i = g^{a\lambda_i} h_{\rho(i)}^{-r_i} \text{pk}_{PV}^{-r_i}, D_i = g^{r_i}, \quad (11)$$

$\text{Sanitize}(\text{pk}, \text{sk}_{PV}, \text{cph})$. For checking whether CS has correctly generated the final ciphertexts in Algorithm 2, PV first chooses a random attribute set $S_{DO}^* \subset S_{DO}$ that satisfies Γ , and sends it to CS. Then, CS generates the proof information and sends it to PV. If the proof information passes the verification mechanism conducted by PV, PV outputs the sanitized ciphertexts. Otherwise, PV aborts the final ciphertexts returned by CS. The interaction between PV and CS is shown as follows:

- 1) For proving that it has correctly generated the final ciphertexts based on access structure Γ , CS first chooses three random elements $t, y, z \in \mathbb{Z}_p^*$

and then computes the proof information $\text{proof} = (K'_{DO}, L'_{DO}, R'_{DO}, \prod_{x \in S_{DO}^*} K'_{DO,x}, \{A_j\}_{j \in [1,6]})$ by (12). Finally, CS sends the proof information proof to PV.

$$\begin{aligned} K'_{DO} &= K_{DO} \cdot g^{at}, L'_{DO} = L_{DO} \cdot g^t; \\ \prod_{x \in S_{DO}^*} K'_{DO,x} &= \prod_{x \in S_{DO}^*} K_{DO,x} \cdot h_x^t; \\ A_1 &= (L'_{DO})^{H(\text{cph})} w^y, R'_{DO} = R_{DO}^z; \\ A_2 &= g^y, A_3 = w^{1/z}, A_4 = g^{b/z} g^{H(ID_{DO})/z}; \\ A_5 &= e(g, g)^{H(ID_{DO})/z}, A_6 = g^{1/z}. \end{aligned} \quad (12)$$

- 2) After gaining the proof information proof , PV needs to check its validity by checking whether (13) holds. If (13) holds, PV accepts the final ciphertexts cph and continues. Otherwise, it rejects and aborts.

$$\begin{aligned} e(K'_{DO}, g) &\stackrel{?}{=} e(g, g)^{\alpha_1} e(L'_{DO}, g^a) e(R'_{DO}, A_3); \\ e(A_3, g) &\stackrel{?}{=} e(w, A_6), e(A_4, g) \stackrel{?}{=} e(g^b, A_6) A_5; \\ e(R'_{DO}, A_4) &\stackrel{?}{=} e(g, g); \\ e\left(\prod_{x \in S_{DO}^*} K'_{DO,x}, g\right) &\stackrel{?}{=} e\left(\prod_{x \in S_{DO}^*} h_x, L'_{DO}\right); \\ e(A_1, g) &\stackrel{?}{=} e(L'_{DO}, g)^{H(\text{cph})} e(A_2, w). \end{aligned} \quad (13)$$

- 3) If CS has generated the valid proof information, PV will sanitize it so that only authorized DUs can access these ciphertexts. First, PV chooses a random element $r \in \mathbb{Z}_p^*$ and then computes $\overline{\text{cph}} = (C_0, \overline{C}, \overline{C'}, \overline{C''}, \{\overline{C_i}, \overline{D_i}\})$ by (14). Finally, PV returns the sanitized ciphertexts cph to CS.

$$\begin{aligned} C_0 &= g^{ar}, \overline{C} = C \cdot e(g, C')^{-c} e(g, g)^{ar}; \\ \overline{C'} &= C' g^r; \overline{C''} = C'' (C')^{-c} w^r; \\ \overline{C_i} &= C_i D_i^c h_{\rho(i)}^{-r_i}, \overline{D_i} = D_i g^r. \end{aligned} \quad (14)$$

$\text{Dec}(\overline{\text{cph}}, \text{sk}_{CS, DU}, \text{sk}_{DU}, S_{DU})$. Suppose a certain DU attempts to access the record ciphertexts, he will submit his attribute S_{DU} to CS. If S_{DU} satisfies the access structure Γ , CS partially decrypts the sanitized ciphertexts $\overline{\text{cph}}$. Otherwise, CS returns $\perp$. This algorithm is also divided into two sub algorithms. First, CS converts the sanitized ciphertexts $\overline{\text{cph}}$ into the intermediate result ϕ_{DU} by using its key $\text{sk}_{CS, DU}$. Then, DU decrypts the intermediate result ϕ_{DU} to gain the secret key K_R that is used to decrypt C_R . The concrete algorithms are shown as follows:

- 1) $\text{Dec}_{CS}(\overline{\text{cph}}, \text{sk}_{CS, DU}, S_{DU})$. Let $I = \{i | \rho(i) \in S_{DU}\} \subset \{1, \dots, l\}$ be the set that has corresponding attributes satisfying Γ , and $\{w_i \in \mathbb{Z}_p^*\}_{i \in I}$ be constants such that $\sum_{i \in I} w_i \lambda_i = s$. After receiving the sanitized ciphertexts $\overline{\text{cph}}$, CS first computes ϕ_{DU} by (5) and (15), then sends

Algorithm 2: The Specific Progress of Sanitize.

Input: Public keys pk , PV's secret key sk_{PV} , final ciphertexts cph
Output: Sanitized ciphertexts cph or abort final ciphertexts cph

- 1 **PV side:**
- 2 Given the final ciphertexts cph ;
- 3 Choose a random attribute set $S_{DO}^* \subset S_{DO}$ and send it to CS;
- 4 **CS side:**
- 5 Choose random elements $t, y, z \in \mathbb{Z}_p^*$;
- 6 Compute $K'_{DO}, L'_{DO}, R'_{DO}, \prod_{x \in S_{DO}^*} K'_{DO,x}, \{A_j\}_{j \in [1,6]}$ by Eq. 12;
- 7 Send the proof information proof to PV;
- 8 **PV side:**
- 9 Check the correctness of proof by Eq. 13;
- 10 **if** Eq. 13 **does not hold** **then**
- 11 Abort cph and return the empty set $\perp$;
- 12 **else**
- 13 Choose a random element $r \in \mathbb{Z}_p^*$;
- 14 Compute $C_0, \bar{C}, \bar{C}', \bar{C}'', \{\bar{C}_i, \bar{D}_i\}$ by Eq. 14;
- 15 Set the sanitized ciphertexts as $cph = (C_0, \bar{C}, \bar{C}', \bar{C}'', \{\bar{C}_i, \bar{D}_i\})$;
- 16 **return** $\perp$ or cph .

ϕ_{DU} and encrypted e-health data C_R to DU.

$$\frac{e(\bar{C}', K_{DU})e(R_{DU}, \bar{C}'')^{-1}}{(\prod_{i \in I} [e(\bar{C}_i, L_{DU})e(\bar{D}_i, K_{DU, \rho(i)})]^{w_i} e(L_{DU}, C_0))^2} = e(g, g)^{\alpha_1(s+r)} / e(g, g)^{at_{DU}(s+r)} = \phi_{DU}, \quad (15)$$

- 2) $Dec_{DU}(\phi_{DU}, C_R, sk_{DU})$. When receiving the intermediate result ϕ_{DU} , DU decrypts it to obtain the record encryption key K_R by (16) and (32).

$$K_R = \frac{\bar{C}}{\phi_{DU} \cdot e(sk_{DU}, \bar{C}')} \cdot \quad (16)$$

Remark: In real applications, CS may maliciously perform outsourced encryption operations due to various interest incentives. To solve this issue, our scheme OABE introduces PV that can check whether CS has generated the correct intermediate ciphertexts, thereby achieving the verifiable outsourced encryption and incurring fewer computation burdens on DOs. However, it is still necessary to check whether CS has returned the intermediate results honestly. This is because the malicious CS may return previous or modified intermediate results for saving computation and storage resources. Thus, it is vital to support verifiable outsourced decryption in our scheme.

C. Extended Outsourced CP-ABE With Verifiable Decryption (OABE+)

Our basic scheme OABE achieves the outsourced CP-ABE with verifiable encryption, but still cannot guarantee the correctness of ciphertext transformation when CS is malicious. Thus, several mechanisms with providing verifiable outsourced decryption have been proposed. For example, the verifiable computation technique [52], [53] can be used to construct verifiable outsourced CP-ABE, but incurs heavy computations in the decryption phase. The key encapsulation mechanism [5] achieves above goal by checking whether the random element derived from transformed ciphertext is equal to one in the tag, which

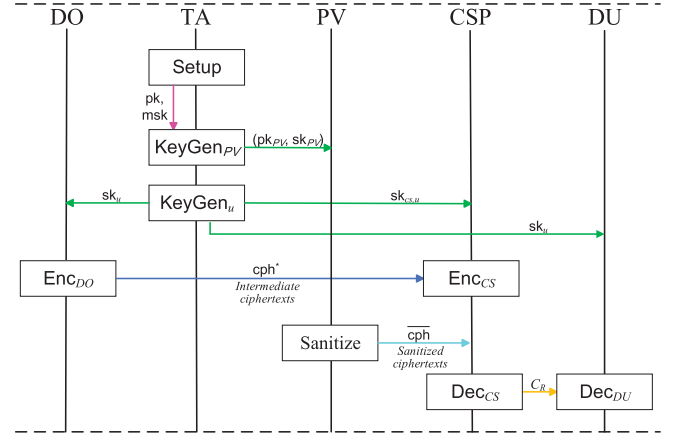


Fig. 7. Framework of OABE+.

cannot resist reply attacks. The redundant random ciphertext [9], [10] appended to original message ciphertext can guarantee the correctness of ciphertext transformation, but incurs double encryption overheads compared with the above ones. For extending OABE to support verifiable outsourced decryption, we utilize the verifiable tag mechanism [28] to prevent CS from returning incorrect intermediate results. To simplify the presentation, we just show the modified content in Setup, Enc, Sanitizer and Dec. KeyGen is still the same as that of our basic scheme OABE. The framework of OABE+ is shown in Fig. 7 and the modified algorithms are demonstrated as follows:

Setup (1^λ). Given the security parameter λ , TA not only generates the public key pk and master key msk but also selects three hash functions H_0, H_1, H_2 by (6). Finally, TA outputs the public key pk^* and master key msk^* by (17)

$$\begin{aligned} pk^* &= (pk, H_0, H_1, H_2); \\ msk^* &= msk. \end{aligned} \quad (17)$$

Enc ($pk, \mathcal{R}$). In this algorithm, DO modifies the intermediate ciphertexts cph^* and final ciphertexts as follows:

- 1) $Enc_{DO}(pk, \mathcal{R}, \Gamma)$. For each e-health data $R \in \mathcal{R}$, DO first chooses a session key $r_R \in \mathbb{G}_T$ and computes $S_R, K_R^*, C_R^*, \text{tag}, C^*$ by (18), where K_R^* is the key of symmetric encryption algorithm Enc. Then, DO calls (10) to generate the intermediate ciphertext $cph^* = (\Gamma, C_R^*, \text{tag}, C^*, C', C'', \{\lambda_i\})$.

$$\begin{aligned} S_R &= H_0(r_R), K_R^* = H_1(r_R); \\ C_R^* &= \text{Enc}_{K_R^*}(R), \text{tag} = H_2(S_R || C_R^*); \\ C^* &= r_R \cdot e(g, g)^{\alpha s} \cdot e(g, pk_{PV})^s, \end{aligned} \quad (18)$$

- 2) $Enc_{CS}(pk, cph^*)$. CS calls (11) to output final ciphertexts $cph = (\Gamma, C_R^*, \text{tag}, C^*, C', C'', \{C_i, D_i\})$.

Sanitize (pk, sk_{PV}, cph). Upon checking that CS has generated the correct final ciphertexts cph based on the proof information proof, PV computes $\bar{C}^*$ by (19) and returns sanitized ciphertexts $\bar{cph} = (C_0, \bar{C}^*, \bar{C}', \bar{C}'', \{\bar{C}_i, \bar{D}_i\})$.

$$\bar{C}^* = C^* \cdot e(g, C')^{-c} e(g, g)^{\alpha r}, \quad (19)$$

Dec $(\overline{\text{cph}}, \text{sk}_{CS,DU}, \text{sk}_{DU}, \text{pk}^*)$. Upon receiving the intermediate result ϕ_{DU} , record ciphertext C_R^* as well as the tag tag, DU first computes the session key r'_R by (20). Then, DU computes $S'_R = H_0(r'_R)$ and checks whether (21) holds. If (21) holds, DU computes $K_R^* = H_1(r'_R)$ and uses it to decrypt C_R^* , which indicates that CS has correctly executed the outsourced decryption.

$$r'_R = \frac{\overline{C^*}}{\phi_{DU} \cdot e(\text{sk}_{DU}, \overline{C^*})}. \quad (20)$$

$$\text{tag} \stackrel{?}{=} H_2(S'_R || C_R^*). \quad (21)$$

Remark: In addition to having the advantages of OABE, OABE+ supports the verifiable outsourced decryption with incurring small computation and storage overheads. Thus, OABE+ achieves the verifiable outsourced CP-ABE. It not only reduces DO's encryption burdens and DUs' decryption burdens by outsourcing heavy burdens to CS, but also guarantees that CS correctly performs outsourced encryption and decryption operations, which scales well in a resource-limited mobile e-health system.

VI. SECURITY ANALYSIS

For proving that our schemes are secure against unauthorized DUs, we may encounter an attribute associated with multiple rows in the access structure $\Gamma^* = (M_{l^* \times n^*}^*, \rho^*(\cdot))$, which is analogous to an attribute denoted by multiple leaf-nodes in an access tree [4]. Thus, the significant obstacle is how to program the challenging ciphertexts into the public parameters. To solve this problem, we utilize the decisional q -parallel BDHE (Bilinear Diffie-Hellman Exponent) assumption to program multiple rows of $M_{l^* \times n^*}^*$ into one group element corresponding to an attribute. Before proofing the security, we first review the decisional q -parallel BDHE assumption as follows:

$$\Phi = \begin{cases} g, g^s, g^a, \dots, g^{a^q}, g^{a^{q+2}}, \dots, g^{a^{2q}}; \\ g^{sb_j}, g^{a/b_j}, \dots, g^{a^q/b_j}, g^{a^{q+2}/b_j}, \dots, g^{a^{2q}/b_j}; \\ g^{asbk/b_j}, \dots, g^{a^q sbk/b_j}, \end{cases} \quad (22)$$

Definition 3 (q -parallel BDHE assumption): Let $\mathbb{G}, \mathbb{G}_T$ be two groups of prime order p , g be a generator of $\mathbb{G}$, and $a, s, b_1, \dots, b_q$ be random elements in the field $\mathbb{Z}_p$. The adversary $\mathcal{A}$'s goal of q -parallel BDHE problem is to distinguish $e(g, g)^{a^{q+1}s}$ from a random element $\mathcal{T}$ in $\mathbb{G}_T$ when $\mathcal{A}$ is given a set Φ defined by (22), where $j, k \in [1, q]$, $k \neq j$. We can say that q -parallel BDHE assumption holds if $\mathcal{A}$ can solve the q -parallel BDHE problem with negligible advantage ϵ , namely

$$\text{Adv}_{\mathcal{A}} = \left| \Pr[\mathcal{A}(\Phi, \Lambda = e(g, g)^{a^{q+1}s}) = 0] - \Pr[\mathcal{A}(\Phi, \Lambda = \mathcal{T}) = 0] \right| \leq \epsilon.$$

Theorem 1: Our schemes are selectively secure against unauthorized access permissions when the q -parallel BDHE assumption holds.

Proof: Assume that there exists an adversary $\mathcal{A}$ that can break the securities of our schemes, then we construct a simulator $\mathcal{S}$ that can use $\mathcal{A}$ to break the q -parallel BDHE assumption as follows. Given the q -parallel BDHE instance Φ shown by (22),

the challenger $\mathcal{C}$ chooses a guess bit $b \in \{0, 1\}$. If $b = 0$, $\mathcal{C}$ sends a two-tuple $(\Phi, \Lambda = e(g, g)^{a^{q+1}s})$ to $\mathcal{S}$. Otherwise, $\mathcal{C}$ sends a two-tuple $(\Phi, \Lambda = \mathcal{T})$ to $\mathcal{S}$. Then, $\mathcal{S}$ will output his guess b' of b .

Init: $\mathcal{A}$ first submits a challenging access structure $\Gamma^* = (M_{l^* \times n^*}^*, \rho^*(\cdot))$, where $l^*, n^* \leq q$.

Setup: Suppose X is the set of indices i such that $\rho^*(i) = x$, $\mathcal{S}$ first selects $\alpha'_1, \alpha'_2, b, \nu \in \mathbb{Z}_p^*$ such that $\alpha_1 = \alpha'_1 + a^{q+1}/2$, $\alpha_2 = \alpha'_2 + a^{q+1}/2$, $\alpha_1 + \alpha_2 = \alpha$, $\alpha'_1 + \alpha'_2 = \alpha'$, then computes $e(g, g)^\alpha = e(g, g)^{\alpha'} e(g, g)^{a^{q+1}}$, g^a, g^b and $w = g^\nu$. For each attribute $x \in \mathbb{U}$, $\mathcal{S}$ chooses $t_x \in \mathbb{Z}_p^*$ and computes h_x by (23). It is worth noticing that $h_x = g^{t_x}$ if X is empty. Finally, $\mathcal{S}$ chooses $c \in \mathbb{Z}_p^*$ and computes g^c . $\mathcal{S}$ sends the public key $\text{pk} = (e(g, h), e(g, g)^\alpha, g^a, g^b, g^c, h_1, \dots, h_{|\mathbb{U}|})$ to $\mathcal{A}$.

$$h_x = g^{t_x} \prod_{i \in X} g^{aM_{i,1}^*/b_i} g^{aM_{i,2}^*/b_i} \dots g^{aM_{i,n^*}^*/b_i}. \quad (23)$$

Phase 1: $\mathcal{A}$ adaptively makes multiple queries to the following oracles:

- $\mathcal{O}_{\text{KeyGen}}$. $\mathcal{A}$ submits an identity ID_u and an attribute S_u that does not satisfy Γ^* . $\mathcal{S}$ chooses a random element $\eta_u \in \mathbb{Z}_p^*$ and a vector $\vec{v}^* = (v_1, \dots, v_{n^*})$ such that $v_1 = -1$ and $\vec{v}^* M_i^* \notin S_u$, then computes t_u, L_u, K_u, R_u by (24). If the attribute $x \in S_u$, $\mathcal{S}$ sets $K_{u,x} = L_u^{t_x}$. Otherwise, $\mathcal{S}$ sets $K_{u,x}$ by (25). Finally, $\mathcal{S}$ sends $(L_u, R_u, K_u, \{K_{u,x}\})$ to $\mathcal{A}$.

$$\begin{aligned} t_u &= \eta_u + v_1 a^q + v_2 a^{q-1} + \dots + w_{n^*} a^{q-n^*+1}; \\ L_u &= g^{t_u} = g^{\eta_u} \prod_{j=1}^{n^*} (g^{a^{q+1}-j})^{v_j}, R_u = g^{\frac{1}{b+H(\text{ID}_u)}}; \\ K_u &= g^{\alpha'_1} g^{a\eta_u} \prod_{j=2}^{n^*} (g^{a^{q-j+2}})^{v_j} w^{\frac{1}{H(b+\text{ID}_u)}}. \end{aligned} \quad (24)$$

$$K_{u,x} = L_u^{t_x} \prod_{i \in X} \prod_{j=1}^{n^*} g^{\left(\frac{a_j}{b_i}\right)^{\eta_u}} \prod_{d=1}^{n^*} g^{\left(\frac{(a^{q+1}+j-d)}{b_i}\right)^{v_k}} \quad (25)$$

- $\mathcal{O}_{\text{Enc}}$. $\mathcal{A}$ is also allowed to submit a message $\mathcal{M}$, identity ID_{DO} , attribute set S_{DO} and access structure $\Gamma = (M_{l \times n}, \rho(\cdot))$. $\mathcal{S}$ first generates the secret keys $(L_{DO}, R_{DO}, K_{DO}, \{K_{DO,x}\})$ corresponding to S_{DO} , then selects a random vector $\vec{v} = (s', y_2, \dots, y_n)$ and random elements $r_1, \dots, r_l \in \mathbb{Z}_p^*$, finally computes C, C', C'', C_i, D_i by (26), where $i \in [1, l]$. The final ciphertexts $\text{cph} = (C, C', C'', \{C_i, D_i\})$ are returned to $\mathcal{A}$.

$$\begin{aligned} C &= \mathcal{M} e(g, g)^{s'\alpha} e(g^c, g)^{s'}, C' = g^{s'}, \\ C'' &= w^{s'} g^{cs}, C_i = g^{a\lambda_i} h_{\rho(i)}^{-r_i} g^{-cr_i}, D_i = g^{r_i}, \end{aligned} \quad (26)$$

- $\mathcal{O}_{\text{Sanitize}}$. $\mathcal{A}$ submits the final ciphertexts cph and its chosen access structure Γ . $\mathcal{S}$ chooses a random element $r \in \mathbb{Z}_p^*$ and then computes $\overline{C} = C \cdot e(g, C')^{-c} e(g, g)^{r\alpha}$, $\overline{C'} = C' g^r$, $\overline{C''} = C'' (C')^{-c} w^r$, $C_0 = g^{ar}$, $\overline{C_i} = g^{a\lambda_i} h_{\rho(i)}^{-r_i-r}$,

$\overline{D_i} = D_i g^r = g^{r_i+r}$. The sanitized ciphertexts $\overline{\text{cph}} = (C_0, \overline{C}, \overline{C'}, \overline{C''}, \{\overline{C_i}, \overline{D_i}\})$ are sent to $\mathcal{A}$.

Challenge: $\mathcal{A}$ submits two challenging messages $\mathcal{M}_0, \mathcal{M}_1$ and access structure $\Gamma^* = (M_{l \times n}^*, \rho^*(\cdot))$. $\mathcal{S}$ chooses a guess bit $b' \in \{0, 1\}$ and random elements $y_2', \dots, y_{n'}' \in \mathbb{Z}_p^*$, then sets the vector $\vec{v}^* = (s, sa + y_2, sa^{n^*-1} + y_{n'})$. Finally, $\mathcal{S}$ chooses random elements $r_1', \dots, r_{l'}' \in \mathbb{Z}_p^*$ and defines the set $\psi_i = \{j | \rho^*(j) = \rho^*(i) \wedge j \neq i\}$ and computes $C_{b'}, C_{b'}, C_{b'}, D_{b',i}, C_{b',i}$ by (27). The challenging ciphertexts $\text{cph}^* = (C_{b'}, C_{b'}, C_{b'}, \{C_{b',i}, D_{b',i}\})$ are sent to $\mathcal{A}$.

$$C_{b'} = \mathcal{M}_{b'} \mathcal{T}e(g^s, g^{\alpha'})^s, C_{b'}' = g^s;$$

$$C_{b'}'' = g^{s\nu}, D_{b',i} = g^{-r_i' - sb_i};$$

$$C_{b',i} = h_{\rho^*(i)}^{r_i'} (g^{b_i s})^{-r_i} \prod_{j=2}^n g^{a M_{i,j}^* y_j'} \prod_{j \in \psi_i} \prod_{k=1}^{n^*} \left(g^{\frac{a^j s b_i}{b_k}} \right)^{M_{j,k}^*}. \quad (27)$$

Phase 2: This phase is similar to *Phase 1*, but the requirement is that S_u should not satisfy Γ^* .

Guess: $\mathcal{A}$ outputs a guess bit $b'' \in \{0, 1\}$ of b' . If $b'' = b'$, $\mathcal{S}$ outputs $b' = 0$. Otherwise, $\mathcal{S}$ outputs $b' = 1$.

- If $b = 0$, cph^* is a correct ciphertext form, thus $\mathcal{A}$'s probability of outputting $b'' = b'$ is $\Pr[b'' = b' | b = 0] \geq \frac{1}{2} + \epsilon$. When $\mathcal{A}$ outputs $b'' = b'$, then $\mathcal{S}$'s probability outputting $b' = 0$ is $\Pr[b' = b | b = 0] \geq \frac{1}{2} + \epsilon$.
- If $b = 1$, cph^* is a correct ciphertext form, thus $\mathcal{A}$'s probability of outputting $b'' \neq b'$ is $\Pr[b'' \neq b' | b = 1] = \frac{1}{2}$. When $\mathcal{A}$ outputs $b'' \neq b'$, $\mathcal{S}$'s probability outputting $b' = 1$ is $\Pr[b' = b | b = 1] = \frac{1}{2}$.

$\mathcal{S}$'s advantage in breaking the q -parallel BDHE assumption is defined as by (28). That is, $\mathcal{S}$ can break the q -parallel BDHE assumption with a non-negligible advantage, which conflicts with the q -parallel BDHE assumption. Thus, our scheme is selectively secure against unauthorized accesses if the q -parallel BDHE assumption holds.

$$\begin{aligned} \text{Adv}_{\mathcal{S}} &= \left| \frac{1}{2} \Pr[b' = b | b = 0] - \frac{1}{2} \Pr[b' = b | b = 1] \right| \\ &\geq \frac{1}{2} \times \left(\frac{1}{2} + \epsilon \right) - \frac{1}{2} \times \frac{1}{2} = \frac{\epsilon}{2}. \end{aligned} \quad (28)$$

□

For guaranteeing that outsourced encryption has been correctly performed, our basic scheme OABE should prove that CS has been authorized by DO, which is achieved by the following theorem. Before demonstrating the proof of Theorem 2, we first review the q -SDH (Strong Diffie-Hellman) assumption.

Definition 4 (q -SDH assumption): Let $(\mathbb{G}, \mathbb{G}_T, e, p, \tilde{g})$ be the bilinear pairing parameters and $d \in \mathbb{Z}_p^*$ be a random element $x \in \mathbb{Z}_p^*$. Given a set of elements $(\tilde{g}, \tilde{g}^x, \dots, \tilde{g}^{x^q})$ the goal of q -SDH problem is to output $(d, \tilde{g}^{1/(x+d)})$, where $d \in \mathbb{Z}_p^*$. We can say that the q -SDH assumption holds if there exists no adversary $\mathcal{A}$

that can solve the q -SDH problem with non-negligible advantage ϵ , namely $\text{Adv}_{\mathcal{A}} = \Pr[\mathcal{A}(\tilde{g}, \tilde{g}^x, \dots, \tilde{g}^{x^d}) = (d, \tilde{g}^{1/(x+d)})] \geq \epsilon$.

Theorem 2: Our scheme OABE is selectively secure against unauthorized outsourcing encryption under the q -SDH assumption, where $q \geq q_1 + q_2 + 1$, q_1, q_2 respectively denote the number of key generation queries and encryption queries issued by an adversary $\mathcal{A}$.

Proof: Suppose $\mathcal{A}$ can break our scheme OABE with a non-negligible advantage, then we construct a simulator $\mathcal{S}$ that will use $\mathcal{A}$ to break the q -SDH assumption. Given the q -SDH instance $(e, p, \mathbb{G}, \mathbb{G}_T, \tilde{g}, \tilde{g}^x, \dots, \tilde{g}^{x^q})$, $\mathcal{S}$ tries to output $(d, \tilde{g}^{1/(x+d)})$ by simulating the following security game:

Init: $\mathcal{A}$ first submits q' mobile users' identities $\Phi = (\text{ID}_1, \dots, \text{ID}_{q'})$.

Setup: Given $q \geq q' + 1$, $\mathcal{S}$ first defines three polynomials $f(x), f'(x), f_i(x)$ by (29). Then, $\mathcal{S}$ chooses multiple random elements

$\alpha, \alpha_1, \alpha_2, a, b, \nu, x_1, \dots, x_{|U|}, y_1, \dots, y_{q'} \in \mathbb{Z}_p^*$ and computes $g = \tilde{g}^{f(x)} = \prod_{i=0}^{q'} (\tilde{g}^{x^i})^{\alpha_i}$, $w = g^\nu = \tilde{g}^{\nu f(x)}$, $e(g, g)^\alpha = e(\tilde{g}^{f(x)}, \tilde{g}^{f(x)})^\alpha$, $e(g, g)^{\alpha_1} = e(\tilde{g}^{f(x)}, \tilde{g}^{f(x)})^{\alpha_1}$, $e(g, g)^{\alpha_2} = e(\tilde{g}^{f(x)}, \tilde{g}^{f(x)})^{\alpha_2}$, $g^a, g^b = g^x = \tilde{g}^{x f(x)}$, $h_i = g^{x_i}$, where $i \in [1, |U|]$, $\alpha = \alpha_1 + \alpha_2$. In addition, $\mathcal{S}$ chooses a random element $c \in \mathbb{Z}_p^*$ and computes g^c . Finally, $\mathcal{S}$ returns the public key $\text{pk} = (g, e(g, g)^\alpha, w, g^a, g^b, g^c, h_1, \dots, h_{|U|})$ to $\mathcal{A}$.

$$\begin{aligned} f(x) &= \prod_{i=1}^{q'} (x + H(\text{ID}_i)) = \sum_{i=0}^{q'} a_i x^i; \\ f'(x) &= f(x)^2 = \sum_{i=0}^{2q'} c_i x^i, f_i(x) \\ &= f(x)/(x + H(\text{ID}_i)). \end{aligned} \quad (29)$$

Phase 1: $\mathcal{A}$ makes multiple queries to the following oracles:

- $\mathcal{O}_{\text{KeyGen}}$. $\mathcal{A}$ submits the identity $\text{ID}_u \in \Phi$ and attributes S_u of DU. $\mathcal{S}$ chooses a random element $t_u \in \mathbb{Z}_p^*$ and computes $K_u = g^{\alpha_1} g^{at_u} \tilde{g}^{\nu f(x)/(x+H(\text{ID}_u))}$, $\text{sk}_u = g^{\alpha_2} g^{at_u} \tilde{g}^{\nu f(x)/(x+H(\text{ID}_u))}$, $L_u = g^{t_u}$, $R_u = \tilde{g}^{f_i(x)} = g^{1/(x+H(\text{ID}_u))}$, $K_{u,i} = h_i^{t_u}$, where $i \in S_u$. Finally, $\mathcal{S}$ sends the secret keys $(K_u, \text{sk}_u, L_u, R_u, \{K_{u,i}\})$ to $\mathcal{A}$.
- $\mathcal{O}_{\text{Enc}}$. $\mathcal{A}$ is allowed to submit a message $\mathcal{M}$, DO's attribute set S_{DO} and a specified access structure $\Gamma = (M_{l \times n}, \rho(\cdot))$. The simulator $\mathcal{S}$ first generates the secret keys $(K_{DO}, \text{sk}_{DO}, L_{DO}, R_{DO}, \{K_{DO,i}\})$ that are associated with S_{DO} . Then, the simulator $\mathcal{S}$ selects a random vector $\vec{v} = (s, y_2, \dots, y_n)$ and random elements $r_1, \dots, r_n \in \mathbb{Z}_p^*$, and computes $C = \mathcal{M}e(g, g^c)^s e(g, g)^{s\alpha}$, $C' = g^s$, $C'' = w^s g^{cs}$, $C_i = g^{a\lambda_i} h_{\rho(i)}^{-r_i} g^{-cr_i}$, $D_i = g^{r_i}$, where $i \in [1, l]$, $\lambda_i = M_i \vec{v}$. Besides, $\mathcal{S}$ chooses random elements $t, y, z \in \mathbb{Z}_p^*$ and calls Sanitize to output $K'_{DO}, L'_{DO}, A_1, A_2, R'_{DO}, A_3, A_4, A_5, A_6, \prod_{i \in S_{DO}} K'_{DO,i}$. Finally, the simulator $\mathcal{S}$ sends the final ciphertexts $\text{cph} = (\Gamma, C, C', C'', \{C_i, D_i\})$ and proof $= (K'_{DO}, L'_{DO}, R'_{DO}, \prod_{i \in S_{DO}} K'_{DO,i}, A_1, A_2, A_3, A_4, A_5, A_6)$ to $\mathcal{A}$.

- $\mathcal{O}_{\text{Sanitize}}$. $\mathcal{A}$ submits the final ciphertexts cph . $\mathcal{S}$ first chooses a random element $r \in \mathbb{Z}_p^*$, then computes $C_0 = g^{ar}$, $\overline{C} = C \cdot e(g, C')^{-c} e(g, g)^{r\alpha}$, $\overline{C'} = C' g^r$, $\overline{C''} = C'' (C')^{-c} w^r$, $\overline{C_i} = C_i D_i^c h_{\rho(i)}^{-r}$, $\overline{D_i} = D_i g^r$. Finally, $\mathcal{S}$ sends the sanitized ciphertexts $\text{cph} = (C_0, \overline{C}, \overline{C'}, \overline{C''}, \{\overline{C_i}, \overline{D_i}\})$ to $\mathcal{A}$.

Challenge: $\mathcal{A}$ submits a certain DO's attribute set S_{DO}^* , a message $\mathcal{M}^*$ and an access structure $\Gamma^* = (\mathcal{M}^*, \rho^*(\cdot))$, but its identity $\text{ID}_{DO} \notin \Phi$. $\mathcal{S}$ first chooses a random vector $\vec{v}^* = (s^*, y_2, \dots, y_n)$ and random elements $r_1, \dots, r_n \in \mathbb{Z}_p^*$, then computes $C^*, \overline{C'}, \overline{C''}, C_i^*, D_i^*$ by (30), where $\lambda_i^* = M_i^* \vec{v}^*$, $i \in [1, l]$. Finally, $\mathcal{S}$ sends the challenging ciphertexts $\text{cph}^* = (C^*, \overline{C'}, \overline{C''}, \{C_i^*, D_i^*\})$ to $\mathcal{A}$.

$$\begin{aligned} C^* &= \mathcal{M}^* e(g, g)^{s^* \alpha} e(g, g^c)^{s^*}, \overline{C'} = g^{s^*}; \\ \overline{C''} &= w^{s^*} g^{cs^*}, C_i^* = g^{\alpha \lambda_i^*} h_{\rho^*(i)}^{-r_i} g^{-cr_i}, D_i^* = g^{r_i}, \end{aligned} \quad (30)$$

Guess: $\mathcal{A}$ first outputs the proof information $\text{proof}^* = (K'_{DO^*}, L'_{DO^*}, R'_{DO^*}, \prod_{i \in S_{DO^*}} K'_{DO^*, i}, A_1^*, A_2^*, A_3^*, A_4^*, A_5^*, A_6^*)$. If the proof information proof^* can pass the verification mechanism, then we have $K'_{DO^*} = g^\alpha g^{at} w^{1/(x+H(\text{ID}_{DO^*}))}$, $L'_{DO^*} = g^{t_{DO^*}}$. Given $\Omega, f(x)$ by (31), we can derive (32), where $\mu(x) = \mu_0 + \mu(x)_1 x + \dots + \mu_{q'-2} x^{q'-2}$. We can further deduce that $\tilde{g}^{1/(x+H(\text{ID}_{DO^*}))} = (\Omega^{1/\nu} \tilde{g}^{\mu(x)})^{1/\rho} = (\Omega^{1/\nu} \prod_{i=0}^{q'-2} \tilde{g}^{x^i})^{1/\rho}$. We can say that the simulator $\mathcal{S}$ can use $(H(\text{ID}_{DO^*}), \tilde{g}^{1/(x+H(\text{ID}_{DO^*}))})$ to solve the q -SDH problem, which conflicts the q -SDH assumption. Thus, $\mathcal{A}$ cannot break OABE with non-negligible advantage.

$$\begin{aligned} \Omega &= K'_{DO^*} / (g^\alpha (L'_{DO^*})^a) = w^{1/(x+H(\text{ID}_{DO^*}))}; \\ f(x) &= \mu(x)(x + H(\text{ID}_{DO^*})) + \rho, \quad (31) \\ \frac{f(x)}{(x + H(\text{ID}_{DO^*}))} &= \mu(x) + \frac{\rho}{(x + H(\text{ID}_{DO^*}))}; \\ \Omega^{1/\nu} &= g^{\frac{1}{(x+H(\text{ID}_{DO^*}))}} = \tilde{g}^{\frac{f(x)}{(x+H(\text{ID}_{DO^*}))}} \\ &= \tilde{g}^{\mu(x)} \tilde{g}^{\frac{\rho}{(x+H(\text{ID}_{DO^*}))}}. \end{aligned} \quad (32)$$

□

Similar to the reference [28], our OABE+ guarantees the verifiable outsourced decryption is privately verifiable, which is proved by the following theorem.

Theorem 3: Assume that H_0, H_2 are collision-resistant hash functions, then our scheme OABE+ is privately verifiable.

Proof: Assume that there exists an adversary $\mathcal{A}$ that can break the verifiability of outsourced decryption, then we construct a simulator $\mathcal{S}$ then can break the collision-resistance of hash functions H_0, H_2 . Given two challenging hash functions H_0^*, H_2^* , $\mathcal{S}$ presents the proof sketch of $\mathcal{S}$ that simulates the security game [28] as follows:

$\mathcal{S}$ first calls Setup to output the public key pk and master key msk and sends pk to $\mathcal{A}$. Then, $\mathcal{S}$ simulates $\mathcal{A}$'s queries in both Phase 1 and Phase 2: When $\mathcal{A}$ submits the challenging message r_R^* corresponding to record $R^* \in \mathcal{R}$ and access structure Γ^* , $\mathcal{S}$ calls Enc to generate $S_R^* = H_0^*(r_R^*)$, $K_R^* = H_1^*(r_R^*)$, $C_R^* = \text{Enc}_{K_R^*}(R^*)$, $\text{tag} = H_2^*(S_R^* || C_R^*)$, and sends C_R^*, tag to $\mathcal{A}$. Then, $\mathcal{A}$ submits an attribute set S_u^* that satisfies Γ^* and

outputs the intermediate results (ϕ_u^*, C_R^*) . If $\mathcal{A}$ breaks the verifiability of outsourced decryption, $\mathcal{S}$ can recover a message $R \notin \{R^*, \perp\}$, where $\perp$ is null. The $\mathcal{A}$'s probability in breaking the verifiability of outsourced decryption is shown as follows:

- In the case $(S_R, C_R) \neq (S_R^*, C_R^*)$, $\mathcal{S}$ can obtain a collision of hash function H_2^* as it has S_R^*, C_R^* .
- In the case $(S_R, C_R) = (S_R^*, C_R^*)$ and $r_R \neq r_R^*$, $\mathcal{S}$ can break the collision-resistance of H_0^* by observing $H_0^*(r_R) = S_R = S_R^* = H_0^*(r_R^*)$.

From the above analysis, our proposed OABE+ can guarantee the verifiability of outsourced decryption. For more detailed proof, please refer to the reference [28]. We also can use an optimized matrix-based CP-ABE architecture to easily improve the security level of our scheme without affecting desired functions. □

VII. PERFORMANCE ANALYSIS

In this section, we evaluate the performance of our schemes in terms of theoretical complexities and experimental tests using various real-world datasets.

A. Theoretical Complexities

To evaluate the efficiency and feasibility of our schemes including OABE and OABE+, we compare them with the latest Sanitizable Access Control System (SACS) [25].¹ Regarding the theoretical complexities, we mainly focus on several time-consuming operations such as pairing operation P and modular exponentiation operation E (or E_T) in group $\mathbb{G}$ or $\mathbb{G}_T$. Let $\mathcal{L}_{\mathbb{G}_T}, \mathcal{L}_{\mathbb{G}}, \mathcal{L}_{\mathbb{Z}_p}$ be the length of elements in $\mathbb{G}_T, \mathbb{G}, \mathbb{Z}_p$ respectively. As the extended scheme OABE+ has an approximately equal computational complexity with the basic scheme OABE, we just show the theoretical complexities (including computation and storage costs) of OABE and SACS in Table II.

In KeyGen, OABE generates keys for both mobile user (e.g., DO or DUs) and PV that separately take $\mathbb{E}$ and $6\mathbb{E} + |S_u|\mathbb{E}$ operations, while SACS takes $6\mathbb{E} + 2|S_u|\mathbb{E}$ operations to generate the secret key for each DU. Thus, OABE has fewer computation cost than SACS as the number of mobile user attributes (namely the variable $|U|$) increases. Besides, the total storage cost of OABE is $\mathcal{L}_{\mathbb{Z}_p} + 5\mathcal{L}_{\mathbb{G}} + |S_u|\mathcal{L}_{\mathbb{G}}$, which is still fewer than $4\mathcal{L}_{\mathbb{G}} + 2|S_u|\mathcal{L}_{\mathbb{G}}$ of SACS. In Enc, OABE consists of two phases executed sequentially by DO and CS. DO in OABE just performs $3\mathbb{E} + \mathbb{P} + 2\mathbb{E}_T$, and CS needs to conduct time-consuming operations $4l\mathbb{E} + 2l\mathcal{L}_{\mathbb{G}}$. While SACS only allows DO to perform encryption operation, which takes about $2\mathbb{E}_T + \mathbb{E} + 3l\mathbb{E}$. Thus, OABE has a slightly higher computation cost than SACS if the value of variable l is larger, but its DO's computation burden is not affected by l . For the same reason, the storage cost of OABE is also slightly higher than that of SACS. However, in practice, the encryption process of OABE is just one operation, which does not affect the user experience.

In Sanitize, CS in OABE first generates the *proof* information with executing $10\mathbb{E} + \mathbb{P} + \mathbb{E}_T + |S_{DO}^*|\mathbb{E}$ operations, then PV

¹This scheme introduces a sanitizer to prevent malicious data publishers generating unauthorized ciphertexts that leak sensitive information to unauthorized DUs.

TABLE II
THEORETICAL COMPUTATION AND STORAGE COSTS: A COMPARATIVE SUMMARY

Algorithms			Computation costs		Storage costs	
KeyGen	OABE	PV	$\mathbb{E}$		$\mathcal{L}_{\mathbb{Z}_p} + \mathcal{L}_{\mathbb{G}}$	
		User	$6\mathbb{E} + \mathbb{S}_u \mathbb{E}$		$4\mathcal{L}_{\mathbb{G}} + \mathbb{S}_u \mathcal{L}_{\mathbb{G}}$	
		Total	$7\mathbb{E} + \mathbb{S}_u \mathbb{E}$		$\mathcal{L}_{\mathbb{Z}_p} + 5\mathcal{L}_{\mathbb{G}} + \mathbb{S}_u \mathcal{L}_{\mathbb{G}}$	
	SACS	DU	$6\mathbb{E} + 2 \mathbb{S}_u \mathbb{E}$		$4\mathcal{L}_{\mathbb{G}} + 2 \mathbb{S}_u \mathcal{L}_{\mathbb{G}}$	
Enc	OABE	DO	$3\mathbb{E} + \mathbb{P} + 2\mathbb{E}_T$		$2\mathcal{L}_{\mathbb{G}} + \mathcal{L}_{\mathbb{G}_T}$	
		CS	$4\mathbb{E}$		$2\mathcal{L}_{\mathbb{G}}$	
		Total	$3\mathbb{E} + \mathbb{P} + 2\mathbb{E}_T + 4\mathbb{E}$		$2\mathcal{L}_{\mathbb{G}} + \mathcal{L}_{\mathbb{G}_T} + 2\mathcal{L}_{\mathbb{G}}$	
	SACS	DO	$2\mathbb{E}_T + \mathbb{E} + 3\mathbb{E}$		$2\mathcal{L}_{\mathbb{G}_T} + \mathcal{L}_{\mathbb{G}} + 2\mathcal{L}_{\mathbb{G}}$	
Sanitize	OABE	CS	$10\mathbb{E} + \mathbb{P} + \mathbb{E}_T + \mathbb{S}_{DO}^* \mathbb{E}$		$9\mathcal{L}_{\mathbb{G}} + \mathcal{L}_{\mathbb{G}_T}$	
		PV	$15\mathbb{P} + 3\mathbb{E}_T + 4\mathbb{E} + 2\mathbb{E}$		$15\mathcal{L}_{\mathbb{G}_T} + 3\mathcal{L}_{\mathbb{G}} + 2\mathcal{L}_{\mathbb{G}}$	
		Total	$16\mathbb{P} + 4\mathbb{E}_T + 14\mathbb{E} + 3\mathbb{E}$		$16\mathcal{L}_{\mathbb{G}_T} + 12\mathcal{L}_{\mathbb{G}} + 2\mathcal{L}_{\mathbb{G}}$	
	SACS	CS	$4\mathbb{E} + 2\mathbb{E}_T + 2\mathbb{P} + \mathbb{S}_{DU} \mathbb{E} + l(2\mathbb{P} + \mathbb{E}_T + 3\mathbb{E})$		$3\mathcal{L}_{\mathbb{G}} + 4\mathcal{L}_{\mathbb{G}_T} + \mathbb{S}_{DU} \mathcal{L}_{\mathbb{G}} + 4l\mathcal{L}_{\mathbb{G}_T}$	
Dec	OABE	CS	$3\mathbb{P} + \mathbb{E}_T + I (2\mathbb{P} + \mathbb{E}_T)$		$3\mathcal{L}_{\mathbb{G}_T} + 2 I \mathcal{L}_{\mathbb{G}_T}$	
		DU	$\mathbb{P}$		$\mathcal{L}_{\mathbb{G}_T}$	
	SACS	DU	$4\mathbb{P} + \mathbb{E}_T + I (2\mathbb{P} + \mathbb{E}_T)$		$4\mathcal{L}_{\mathbb{G}_T} + 2 I \mathcal{L}_{\mathbb{G}_T}$	
			$2\mathbb{P} + 2 I (2\mathbb{P} + \mathbb{E}_T)$		$2\mathcal{L}_{\mathbb{G}_T} + 4 I \mathcal{L}_{\mathbb{G}_T}$	

Notes. “ $|\mathbb{S}_u|$ ”: Number of each mobile user’s attributes; “ $|\mathbb{S}_{DO}|$ ”: Number of DO’s attributes; “ $|\mathbb{S}_{DU}|$ ”: Number of DU’s attributes; “ $|\mathbb{S}_{DO}^*|$ ”: Number of attributes chosen from DO’s attributes; “ l ”: Number of attributes in access structure; “ $|I|$ ”: Number of attributes that satisfy the access structure;

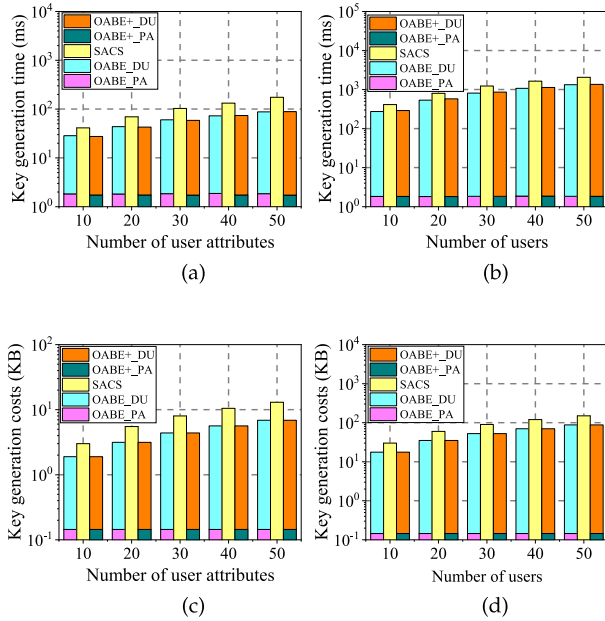


Fig. 8. Performance of KeyGen.

checks *proof* and sanitizes the final ciphertexts by performing $15\mathbb{P} + 3\mathbb{E}_T + 4\mathbb{E} + 2\mathbb{E}$ operations. While CS in SACS has to verify the correctness of ciphertexts and output the sanitized ciphertexts with performing $4\mathbb{E} + 2\mathbb{E}_T + 2\mathbb{P} + |\mathbb{S}_{DU}|\mathbb{E}$. As the computation cost of SACS is affected by both variables $|\mathbb{S}_{DU}|, l$, it has higher computation and storage burdens than OABE. In Dec, CS in OABE conducts most operations ($3\mathbb{P} + \mathbb{E}_T + |I|(2\mathbb{P} + \mathbb{E}_T)$), and DU does the constant decryption operations ($\mathbb{P}$). While DU in SACS performs the whole operations ($2\mathbb{P} + 2|I|(2\mathbb{P} + \mathbb{E}_T)$). As SACS requires each DU to execute double decryption operations, it has higher total

decryption costs (including computation and decryption costs) than OABE. Thus, OABE is more suitable for resource-limited end uses in real applications.

B. Experimental Tests

To show the actual performance of our schemes and SACS, we conduct extensive experiments on macOS Catalina 10.15.4 with Intel Core i7 CPU 2.9 GHz by using Python3 language and Paring Based Cryptography (PBC) Library.² The real-world dataset used in these empirical experiments comes from the public dataset: NSF Research Awards Abstracts³ which consists of 12900 abstracts describing NSF awards for basic research. The common Type A curve $E(F_q) : y^2 = x^3 + x$ selected has 80-bit security level, and $\mathbb{G}, \mathbb{G}_T$ of order p are treated as subgroups of $E(F_q)$, where the lengths of p and q are 160 bits and 512 bits, respectively. Thus, the size of each element in groups $\mathbb{G}_T, \mathbb{G}$ is 1024 bits, and that of an element in the file $\mathbb{Z}_p$ is 160 bits. For simplicity, we choose 1000 files from the above dataset and set $|\mathbb{S}_u| = |\mathbb{S}_{DO}| = |\mathbb{S}_{DU}| = |\mathbb{S}_{DO}^*| = l = |I| \in [1, 50]$.

From Fig. 8 we notice that the computation and storage overheads of KeyGen linearly increase with both the number of user attributes and users. Our schemes need to generate keys for both PV and each user, while SACS only generates the secret key for each user. In Fig. 8(a), (c), we vary the number of each user attributes from 1 to 50, namely the variable $|\mathbb{S}_u| \in [1, 50]$. The computation and storage overheads of DU in OABE, OABE+ and SACS linearly increase with the variable $|\mathbb{S}_u|$, and that of PV in our schemes keeps almost unchanged. However, the total computation and storage burdens of key generation in our schemes are still fewer than those of SACS. For example, when $|\mathbb{S}_u|$ is set as 50, the computation and storage costs of PV and each user in OABE, OABE+ and SACS are respectively ((1.84 ms, 0.14 KB), (85.76 ms, 6.75 KB)), ((1.73 ms, 0.14 KB), (86.64 ms, 6.75 KB)), (174.48 ms, 13.00 KB). In Fig. 8(b), (d), we vary the number of users from 1 to 50 and fix the value of $|\mathbb{S}_u|$ as 10. We then notice that the computation and storage costs of key generation in the above three schemes also grow with the number of users, and the key generation cost of PV in our schemes is also independent of the number of users. However, our schemes still incur fewer key generation costs than SACS. Thus, our schemes are efficient than SACS in terms of key generation no matter which variable varies. It is worth noticing that OABE+ and OABE have equal key generation overheads.

The ciphertext encryption costs (including computation and storage costs) of Enc are demonstrated in Fig. 9, which linearly grow with the number of policy attributes and data. Our schemes including OABE and OABE+ support outsourced encryption, in which DO conducts constant operations and CS performs time-consuming operations, while SACS requires DO to execute the whole encryption. In Fig. 9(a), (c), take record encryption as an example, the encryption costs of DO in our schemes keep unchanged, and those of CS in our schemes linearly increase with the number of policy attributes, namely

²<https://crypto.stanford.edu/pbc/download.html>

³<http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html>

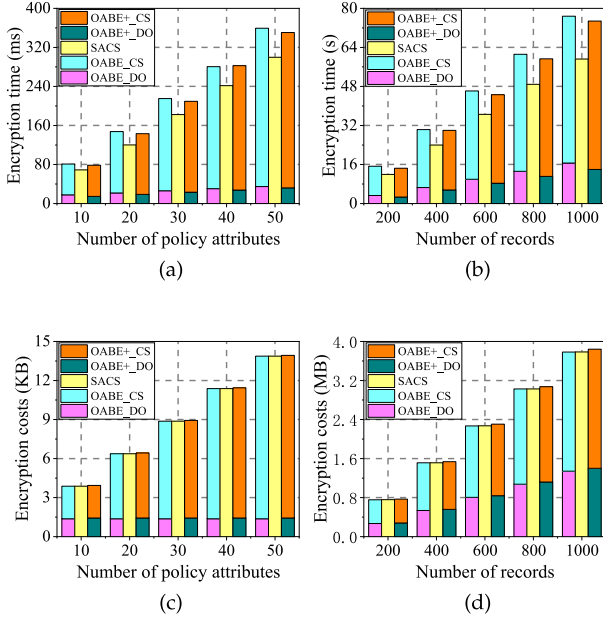


Fig. 9. Performance of Enc.

the variable $l \in [1, 50]$. The encryption cost of DO in SACS also linearly grows as the value of l increases. OABE+ supports the verifiable outsourced decryption, but it just incurs three extra hash operations. Thus, our schemes have approximately equal encryption costs. When compared with SACS, our schemes have slightly higher computation overheads and equal storage overheads. For example, when setting $l = 50$, the computation and storage costs of DO and CS in OABE and OABE+ are ((34.85 ms, 1.38 KB), (324.48 ms, 12.50 KB)), ((32.03 ms, 1.43 KB), (318.52 ms, 12.50 KB)), while those of DO in SACS are (299.84 ms, 13.88 KB). In Fig. 9(b), (d), we set $l = 10$ and vary the number of e-health data from 1 to 1000. The encryption costs of our schemes and SACS linearly increase with the number of e-health data, note that those of both DO and CS in our schemes linearly increase as the number of e-health data increases. With the same reason shown in Fig. 9(a), (c), our schemes have slightly higher computation burdens than SACS, but eliminate high computation burdens from DO.

Fig. 10 shows the computation and storage costs of Sanitize, which linearly grow with the number of user attributes (namely the variable $|S_{DO}^*|$) and the number of e-health data. In our schemes, CS generates the proof information for verifying its legitimacy, and PV checks the correctness of proof information and generates the sanitized ciphertexts, whose computation and storage costs both grow as the values of the above two variables increase. In SACS, CS not only checks the correctness of ciphertexts but also outputs the sanitized ciphertexts, whose computation and storage costs also linearly increase with the above two variables. In Fig. 10(a), (c), the computation and storage costs of each entity in our schemes and SACS linearly grow with the variable $|S_{DO}^*| = |S_{DU}| \in [1, 50]$. While the whole computation and storage costs of our schemes are fewer than those of SACS. In addition, as OABE+ does not modify

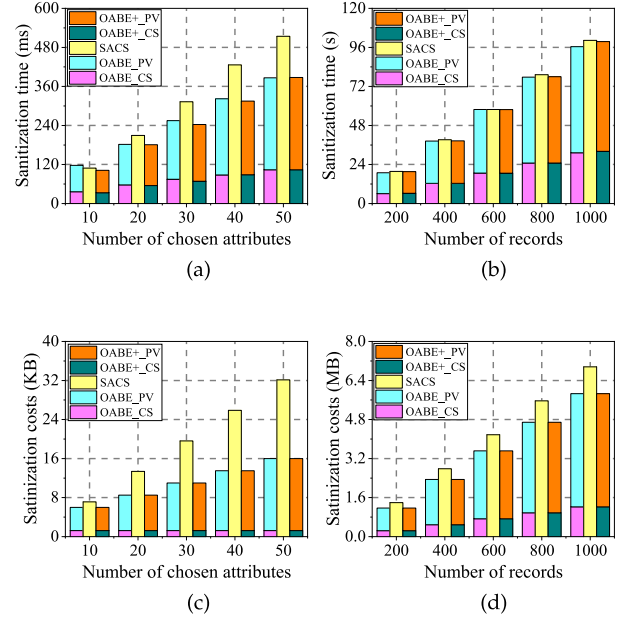


Fig. 10. Performance of sanitize.

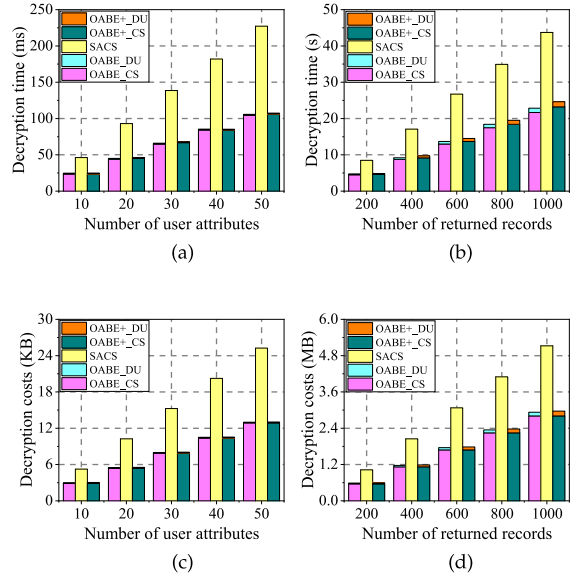


Fig. 11. Performance of dec.

Sanitize, CS and PV in our schemes have equal computation and storage costs. For example, with setting $|S_{DO}^*| = |S_{DU}| = 50$, the computation and storage costs of CS and PV in OABE and OABE+ are ((103.74 ms, 1.25 KB), (282.80 ms, 14.75 KB)), ((103.98 ms, 1.25 KB), (283.62 ms, 14.75 KB)), while those of CS in SACS are (514.21 ms, 32.13 KB). In Fig. 10(b), (d), we set $|S_{DO}^*| = |S_{DU}| = 10$ and vary the number of e-health data from 1 to 1000. As each record encryption uses different access structures, CS in our schemes still needs to generate different proof information for different record encryptions, thereby resulting in linear overheads. We also draw the same conclusion as Fig. 10(a), (c).

TABLE III
EXAMPLE: PERFORMANCE TESTS IN VARIOUS DATASETS

Datasets	Schemes	Aspects	KeyGen		Enc		Sanitize		Dec	
Enron	OABE	Computation costs	PV	DU	DO	CS	CS	PV	CS	DU
			1.83 ms	26.69 ms	20.29 s	61.19 s	31.25 s	66.26 s	21.58 s	1.24 s
	Storage costs	0.14 KB	1.75 KB	1.35 MB	2.45 MB	1.23 MB	4.65 MB	2.82 MB	0.13 MB	
	OABE+	Computation costs	1.73 ms	25.68 ms	13.77 s	61.51 s	31.07 s	65.53 s	21.72 s	1.24 s
		Storage costs	0.14 KB	1.75 KB	1.40 MB	2.45 MB	1.23 MB	4.65 MB	2.81 MB	0.17 MB
	SACS	Computation costs	DU		DO		CS		DU	
		41.28 ms		60.62 s		97.81 s		41.91 s		
	Storage costs	3.00 KB		3.79 MB		6.96 MB		5.13 MB		
Newsgroups	OABE	Computation costs	PV	DU	DO	CS	CS	PV	CS	DU
			1.83 ms	26.69 ms	20.63 s	61.46 s	31.36 s	66.96 s	22.13 s	1.27 s
	Storage costs	0.13 KB	1.74 KB	1.34 MB	2.44 MB	1.22 MB	4.64 MB	2.81 MB	0.12 MB	
	OABE+	Computation costs	1.73 ms	25.68 ms	13.67 s	61.92 s	31.34 s	66.17 s	21.87 s	1.25 s
		Storage costs	0.13 KB	1.74 KB	1.40 MB	2.44 MB	1.22 MB	4.64 MB	2.81 MB	0.16 MB
	SACS	Computation costs	DU		DO		CS		DU	
		41.28 ms		60.49 s		99.70 s		42.56 s		
	Storage costs	2.95 KB		3.74 MB		6.92 MB		5.11 MB		
NSF	OABE	Computation costs	PV	DU	DO	CS	CS	PV	CS	DU
			1.83 ms	26.69 ms	20.30 s	61.20 s	31.26 s	66.43 s	21.76 s	1.26 s
	Storage costs	0.13 KB	1.74 KB	1.36 MB	2.46 MB	1.23 MB	4.65 MB	2.84 MB	0.14 MB	
	OABE+	Computation costs	1.73 ms	25.68 ms	13.37 s	61.76 s	31.30 s	65.91 s	21.83 s	1.26 s
		Storage costs	0.13 KB	1.74 KB	1.42 MB	2.45 MB	1.24 MB	4.66 MB	2.82 MB	0.18 MB
	SACS	Computation costs	DU		DO		CS		DU	
		41.28 ms		60.18 s		98.54 s		44.00 s		
	Storage costs	2.95 KB		3.76 MB		6.94 MB		5.12 MB		

Fig. 11 presents the computation and storage costs of Dec, which are affected by the number of user attributes (namely $|S_{DU}| \in [1, 50]$) and the number of returned e-health data. In our schemes, CS executes most of the decryption operations and DU conducts the constant decryption operations. CS in SACS performs the entire decryption operations. From Fig. 11 we notice that the computation and storage costs of CS in our schemes and DU in SACS both linearly increase with the above two variables. While the decryption costs of DU in our schemes just grow as the number of returned e-health data increases, which are not affected by the variable $|S_{DU}|$. In Fig. 11(a), (c), the decryption costs of DU in our schemes keep almost unchanged, but those of CS in our schemes and DU in SACS linearly grow as the value of $|S_{DU}|$ increases. It is worth noticing that the decryption costs of our schemes are much fewer than those of SACS. A For example, when setting $|S_{DU}| = 50$, the computation and storage costs of CS and DU in OABE and OABE+ are ((104.47 ms, 12.88 KB), (1.30 ms, 0.13 KB)), ((106.08 ms, 12.88 KB), (1.41 ms, 0.16 KB)), while those of SACS are (227.20 ms, 25.25 KB). In Fig. 11(b), (d), the computation and storage costs of all entities in the above three schemes linearly increase with the number of returned e-health data, but our schemes are still much more efficient than SACS in terms of record decryption.

To show the practicability and feasibility of our schemes in real applications, in addition to NSF Research Awards Abstracts dataset, we also conduct extensive experiments using other two public datasets including Enron Email Dataset and 20 Newsgroups. The NSF Research Awards Abstracts consists of 12900 abstracts describing NSF awards for basic research. And each data of NSF contains the title, type, time, abstract content of research, etc. Enron Email Dataset contains data from about 150 users, mostly senior management of Enron, and is organized into folders. Enron Email Dataset is publicly available and frequently used database. Each data in Enron Email Dataset contains email

sending time, email content, etc. And the Newsgroups is a collection of approximately 20,000 newsgroup documents, and is partitioned (nearly) evenly across 20 different newsgroups. Each data of Newsgroups contains the subject, specific content of news, etc. As shown in Table III, we test the computation and storage costs of three schemes at different phases by using three public datasets respectively. From Table III, our schemes are superior to SACS as these schemes are more applicable to resource-limited mobile users, which scale well in practice.

VIII. CONCLUSION

In this paper, we studied the verifiable outsourced CP-ABE scheme in a cloud-assisted mobile e-health system. We first designed the outsourced CP-ABE with verifiable encryption scheme, which could reduce the resource-limited mobile users' encryption and decryption complexities and check whether CS had correctly executed outsourced encryption. Then, we extended our basic scheme to construct outsourced CPA-ABE with verifiable decryption scheme, which further guaranteed that CS had generated the correct ciphertext transformation. We formally proved that our schemes were selectively secure against unauthorized accesses. Through theoretical complexity and experimental test analysis, we demonstrated the efficiency and feasibility of our schemes.

As part of future work, we focus on designing expressive or dynamic access structure in the verifiable outsourced CP-ABE mechanism.

REFERENCES

- [1] Y. Yang, X. Liu, R. H. Deng, and Y. Li, "Lightweight sharable and traceable secure mobile health system," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 1, pp. 78–91, Jan./Feb. 2020.
- [2] S. Sengupta and S. S. Bhunia, "Secure data management in cloudlet assisted IoT enabled e-health framework in smart city," *IEEE Sensors J.*, vol. 20, no. 16, pp. 9581–9588, Aug. 2020.

- [3] Y. Lu and J. Li, "Lightweight public key authenticated encryption with keyword search against adaptively-chosen-targets adversaries for mobile devices," *IEEE Trans. Mobile Comput.*, vol. 21, no. 12, pp. 4397–4409, Dec. 2022, doi: [10.1109/TMC.2021.3077508](https://doi.org/10.1109/TMC.2021.3077508).
- [4] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy*, 2007, pp. 321–334.
- [5] M. Green et al., "Outsourcing the decryption of ABE ciphertexts," in *Proc. USENIX Secur. Symp.*, 2011, Art. no. 34.
- [6] J. Li, X. Chen, J. Li, C. Jia, J. Ma, and W. Lou, "Fine-grained access control system based on outsourced attribute-based encryption," in *Proc. Eur. Symp. Res. Comput. Secur.*, 2013, pp. 592–609.
- [7] C. Feng, K. Yu, M. Aloqaily, M. Alazab, Z. Lv, and S. Mumtaz, "Attribute-based encryption with parallel outsourced decryption for edge intelligent IoT," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 13 784–13 795, Nov. 2020.
- [8] H. Wang, D. He, and J. Han, "VOD-ADAC: Anonymous distributed fine-grained access control protocol with verifiable outsourced decryption in public cloud," *IEEE Trans. Serv. Comput.*, vol. 13, no. 3, pp. 572–583, May/Jun. 2020.
- [9] J. Lai, R. H. Deng, C. Guan, and J. Weng, "Attribute-based encryption with verifiable outsourced decryption," *IEEE Trans. Inf. Forensics Secur.*, vol. 8, no. 8, pp. 1343–1354, Aug. 2013.
- [10] J. Li, Y. Wang, Y. Zhang, and J. Han, "Full verifiability for outsourced decryption in attribute based encryption," *IEEE Trans. Serv. Comput.*, vol. 13, no. 3, pp. 478–487, May/Jun. 2020.
- [11] Q. M. Malluhi, A. Shikfa, and V. C. Trinh, "A ciphertext-policy attribute-based encryption scheme with optimized ciphertext size and fast decryption," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, 2017, pp. 230–240.
- [12] N. Attrapadung, J. Herranz, F. Laguillaumie, B. Libert, E. De Panafieu, and C. R  fols, "Attribute-based encryption schemes with constant-size ciphertexts," *Theor. Comput. Sci.*, vol. 422, pp. 15–38, 2012.
- [13] J. Herranz, F. Laguillaumie, and C. R  fols, "Constant size ciphertexts in threshold attribute-based encryption," in *Proc. Int. Workshop Public Key Cryptogr.*, 2010, pp. 19–34.
- [14] Z. Zhou, D. Huang, and Z. Wang, "Efficient privacy-preserving ciphertext-policy attribute based-encryption and broadcast encryption," *IEEE Trans. Comput.*, vol. 64, no. 1, pp. 126–138, Jan. 2015.
- [15] T. V. X. Phuong, G. Yang, and W. Susilo, "Hidden ciphertext policy attribute-based encryption under standard assumptions," *IEEE Trans. Inf. Forensics Secur.*, vol. 11, no. 1, pp. 35–45, Jan. 2016.
- [16] Y. Miao, Q. Tong, K.-K. R. Choo, X. Liu, R. H. Deng, and H. Li, "Secure online/offline data sharing framework for cloud-assisted industrial Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8681–8691, Oct. 2019.
- [17] J. Li et al., "An efficient attribute-based encryption scheme with policy update and file update in cloud computing," *IEEE Trans. Ind. Informat.*, vol. 15, no. 12, pp. 6500–6509, Dec. 2019.
- [18] M. Bellare and S. Goldwasser, "New paradigms for digital signatures and message authentication based on non-interactive zero knowledge proofs," in *Proc. Conf. Theory Appl. Cryptol.*, 1989, pp. 194–211.
- [19] I. Damg  rd, H. Haagh, and C. Orlandi, "Access control encryption: Enforcing information flow with cryptography," in *Proc. Theory Cryptogr. Conf.*, 2016, pp. 547–576.
- [20] G. Fuchsbaauer, R. Gay, L. Kowalczyk, and C. Orlandi, "Access control encryption for equality, comparison, and more," in *Proc. IACR Int. Workshop Public Key Cryptogr.*, 2017, pp. 88–118.
- [21] W. C. Garrison, A. Shull, S. Myers, and A. J. Lee, "On the practicality of cryptographically enforcing dynamic access control policies in the cloud," in *Proc. IEEE Symp. Secur. Privacy*, 2016, pp. 819–838.
- [22] S. Kim and D. J. Wu, "Access control encryption for general policies from standard assumptions," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2017, pp. 471–501.
- [23] M. Abe, M. Chase, B. David, M. Kohlweiss, R. Nishimaki, and M. Ohkubo, "Constant-size structure-preserving signatures: Generic constructions and simple assumptions," *J. Cryptol.*, vol. 29, no. 4, pp. 833–878, 2016.
- [24] X. Wang and S. S. Chow, "Cross-domain access control encryption: Arbitrary-policy, constant-size, efficient," in *Proc. IEEE Symp. Secur. Privacy*, 2021, pp. 1–14.
- [25] W. Susilo, P. Jiang, J. Lai, F. Guo, G. Yang, and R. Deng, "Sanitizable access control system for secure cloud storage against malicious data publishers," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 3, pp. 2138–2148, May/Jun. 2022, doi: [10.1109/TDSC.2021.3058132](https://doi.org/10.1109/TDSC.2021.3058132).
- [26] I. Kim, W. Susilo, J. Baek, and J. Kim, "Harnessing policy authenticity for hidden ciphertext policy attribute based encryption," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 3, pp. 1856–1870, May/Jun. 2022, doi: [10.1109/TDSC.2020.3040712](https://doi.org/10.1109/TDSC.2020.3040712).
- [27] D. Boneh and X. Boyen, "Short signatures without random Oracles," in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, 2004, pp. 56–73.
- [28] B. Qin, R. H. Deng, S. Liu, and S. Ma, "Attribute-based encryption with efficient verifiable outsourced decryption," *IEEE Trans. Inf. Forensics Secur.*, vol. 10, no. 7, pp. 1384–1393, Jul. 2015.
- [29] C. Ge, W. Susilo, J. Baek, Z. Liu, J. Xia, and L. Fang, "A verifiable and fair attribute-based proxy re-encryption scheme for data sharing in clouds," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 5, pp. 2907–2919, Sep./Oct. 2022, doi: [10.1109/TDSC.2021.3076580](https://doi.org/10.1109/TDSC.2021.3076580).
- [30] C. Ge, W. Susilo, J. Baek, Z. Liu, J. Xia, and L. Fang, "Revocable attribute-based encryption with data integrity in clouds," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 5, pp. 2864–2872, Sep./Oct. 2022, doi: [10.1109/TDSC.2021.3065999](https://doi.org/10.1109/TDSC.2021.3065999).
- [31] Y. Zhang, D. Zheng, and R. H. Deng, "Security and privacy in smart health: Efficient policy-hiding attribute-based access control," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2130–2145, Jun. 2018.
- [32] N. Attrapadung, "Unbounded dynamic predicate compositions in attribute-based encryption," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, 2019, pp. 34–67.
- [33] M. Xie, Y. Ruan, H. Hong, and J. Shao, "A CP-ABE scheme based on multi-authority in hybrid clouds for mobile devices," *Future Gener. Comput. Syst.*, vol. 121, pp. 114–122, 2021, doi: [10.1016/j.future.2021.03.021](https://doi.org/10.1016/j.future.2021.03.021).
- [34] J. Ning, Z. Cao, X. Dong, K. Liang, H. Ma, and L. Wei, "Auditable time outsourced attribute-based encryption for access control in cloud computing," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 1, pp. 94–105, Jan. 2018.
- [35] L. Zhang, W. You, and Y. Mu, "Secure outsourced attribute-based sharing framework for lightweight devices in smart health systems," *IEEE Trans. Serv. Comput.*, vol. 15, no. 5, pp. 3019–3030, Sep./Oct. 2022, doi: [10.1109/TSC.2021.3073740](https://doi.org/10.1109/TSC.2021.3073740).
- [36] Z. Wang and W. Liu, "CP-ABE with outsourced decryption and directionally hidden policy," *Secur. Commun. Netw.*, vol. 9, no. 14, pp. 2387–2396, 2016.
- [37] S. Liu, J. Yu, Y. Xiao, Z. Wan, S. Wang, and B. Yan, "BC-SABE: Blockchain-aided searchable attribute-based encryption for cloud-IoT," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 7851–7867, Sep. 2020.
- [38] S. Belguith, N. Kaaniche, M. Hammoudeh, and T. Dargahi, "PROUD: Verifiable privacy-preserving outsourced attribute based signcryption supporting access policy update for cloud assisted IoT applications," *Future Gener. Comput. Syst.*, vol. 111, pp. 899–918, 2020.
- [39] C. Delerabl  e, "Identity-based broadcast encryption with constant size ciphertexts and private keys," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2007, pp. 200–215.
- [40] S. Hohenberger and B. Waters, "Online/offline attribute-based encryption," in *Proc. Int. Workshop Public Key Cryptogr.*, 2014, pp. 293–310.
- [41] H. Ma, R. Zhang, G. Yang, Z. Song, S. Sun, and Y. Xiao, "Concessive online/offline attribute based encryption with cryptographic reverse firewalls—secure and efficient fine-grained access control on corrupted machines," in *Proc. Eur. Symp. Res. Comput. Secur.*, 2018, pp. 507–526.
- [42] S. J. De and S. Ruj, "Efficient decentralized attribute based access control for mobile clouds," *IEEE Trans. Cloud Comput.*, vol. 8, no. 1, pp. 124–137, First Quarter 2020.
- [43] S. Lin, R. Zhang, H. Ma, and M. Wang, "Revisiting attribute-based encryption with verifiable outsourced decryption," *IEEE Trans. Inf. Forensics Secur.*, vol. 10, no. 10, pp. 2119–2130, Oct. 2015.
- [44] J. Xu, Q. Wen, W. Li, and Z. Jin, "Circuit ciphertext-policy attribute-based hybrid encryption with verifiable delegation in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 1, pp. 119–129, Jan. 2016.
- [45] Y. Xia, W. Chen, X. Liu, L. Zhang, X. Li, and Y. Xiang, "Adaptive multimedia data forwarding for privacy preservation in vehicular ad-hoc networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 10, pp. 2629–2641, Oct. 2017.
- [46] Z. Li, W. Li, Z. Jin, H. Zhang, and Q. Wen, "An efficient ABE scheme with verifiable outsourced encryption and decryption," *IEEE Access*, vol. 7, pp. 29 023–29 037, 2019.
- [47] X. Liu, Y. Xia, W. Chen, Y. Xiang, M. M. Hassan, and A. Alelaiwi, "SEMD: Secure and efficient message dissemination with policy enforcement in vanet," *J. Comput. System Sci.*, vol. 82, no. 8, pp. 1316–1328, 2016.
- [48] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Proc. Int. Workshop Public Key Cryptogr.*, 2011, pp. 53–70.

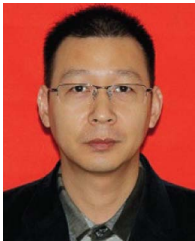
- [49] Z. Liu and Z. Cao, "On efficiently transferring the linear secret-sharing scheme matrix in ciphertext-policy attribute-based encryption," *IACR Cryptol. ePrint Arch.*, vol. 2010, 2010, Art. no. 374.
- [50] V. Nikov and S. Nikova, "New monotone span programs from old," *IACR Cryptol. ePrint Arch.*, vol. 2004, 2004, Art. no. 282.
- [51] R. Guo, G. Yang, H. Shi, Y. Zhang, and D. Zheng, "O³-R-CP-ABE: An efficient and revocable attribute-based encryption scheme in the cloud-assisted Iomt system," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 8949–8963, Jun. 2021, doi: [10.1109/JIOT.2021.3055541](https://doi.org/10.1109/JIOT.2021.3055541).
- [52] B. Parno, M. Raykova, and V. Vaikuntanathan, "How to delegate and verify in public: Verifiable computation from attribute-based encryption," in *Proc. Theory Cryptogr. Conf.*, 2012, pp. 422–439.
- [53] K.-M. Chung, Y. Kalai, and S. Vadhan, "Improved delegation of computation using fully homomorphic encryption," in *Proc. Annu. Cryptol. Conf. Adv. Cryptol.*, 2010, pp. 483–501.



Yinbin Miao (Member, IEEE) received the BE degree from the Department of Telecommunication Engineering, Jilin University, Changchun, China, in 2011, and the PhD degree from the Department of Telecommunication Engineering, Xidian University, Xi'an, China, in 2016. He is currently an associate professor with the Department of Cyber Engineering, Xidian University, Xi'an, China. His research interests include information security and applied cryptography.



Feng Li received the BE degree from the School of Cyberspace Security, Hangzhou Dianzi University, Hangzhou, China, in 2018. He is currently working toward the PhD degree with the School of Cyber Engineering, Xidian University. His current research interest includes cloud computing security.



Xinghua Li received the ME and PhD degrees in computer science from Xidian University, Xi'an, China, in 2004 and 2007, respectively. He is currently a professor with the School of Cyber Engineering, Xidian University, China. His research interests include wireless networks security, privacy protection, cloud computing, and security protocol formal methodology.



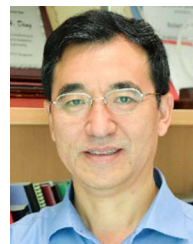
Jianting Ning received the PhD degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, in 2016. He was a research fellow with the Department of Computer Science, National University of Singapore. His research interests include applied cryptography and information security, in particular, public key encryption, attribute-based encryption, and secure multi-party computation.



Hongwei Li (Senior Member, IEEE) received the PhD degree in computer software and theory from the University of Electronic Science and Technology of China, Chengdu, China, in 2008. He is currently a professor with the School of Computer Science and Engineering, University of Electronic Science and Technology of China. He has worked as a postdoctoral fellow with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2012. His research interests include network security, applied cryptography, and trusted computing.



Kim-Kwang Raymond Choo (Senior Member, IEEE) received the PhD degree in information security from Queensland University of Technology, Australia, in 2006. He currently holds the Cloud Technology Endowed Professorship with The University of Texas at San Antonio. He is the founding co-editor-in-chief of ACM Distributed Ledger Technologies: Research & Practice, and the founding Chair of *IEEE Technology and Engineering Management Society Technical Committee (TC) on Blockchain and Distributed Ledger Technologies*. He is the recipient of the 2022 IEEE Hyper-Intelligence TC Award for Excellence in Hyper-Intelligence Systems (Technical Achievement award), the 2022 IEEE TC on Homeland Security Research and Innovation Award, the 2022 IEEE TC on Secure and Dependable Measurement Mid-Career Award, and the 2019 IEEE TC on Scalable Computing Award for Excellence in Scalable Computing (Middle Career Researcher).



Robert H. Deng (Fellow, IEEE) is AXA chair professor of cybersecurity and professor of information systems with the School of Information Systems, Singapore Management University since 2004. His research interests include data security and privacy, multimedia security, network and system security. He served/is serving on the editorial boards of many international journals, including *IEEE Transactions on Information Forensics and Security*, *IEEE Transactions on Dependable and Secure Computing*. He has received the Distinguished Paper Award (NDSS 2012), Best Paper Award (CMS 2012), Best Journal Paper Award (IEEE Communications Society 2017).