White-Box Traceable Ciphertext-Policy Attribute-Based Encryption Supporting Flexible Attributes

Jianting Ning, Xiaolei Dong, Zhenfu Cao, *Senior Member, IEEE*, Lifei Wei, and Xiaodong Lin, *Senior Member, IEEE*

***Abstract*— Ciphertext-policy attribute-based encryption (CP-ABE) enables fine-grained access control to the encrypted data for commercial applications. There has been significant progress in CP-ABE over the recent years because of two properties called traceability and large universe, greatly enriching the commercial applications of CP-ABE. Traceability is the ability of ABE to trace the malicious users or traitors who intentionally leak the partial or modified decryption keys for profits. Nevertheless, due to the nature of CP-ABE, it is difficult to identify the original key owner from an exposed key since the decryption privilege is shared by multiple users who have the same attributes. On the other hand, the property of large universe in ABE enlarges the practical applications by supporting flexible number of attributes. Several systems have been proposed to obtain either of the above properties. However, none of them achieve the two properties simultaneously in practice, which limits the commercial applications of CP-ABE to a certain extent. In this paper, we propose two practical large universe CP-ABE systems supporting white-box traceability. Compared with existing systems, both the two proposed systems have two advantages: 1) the number of attributes is not polynomially bounded and 2) malicious users who leak their decryption keys could be traced. Moreover, another remarkable advantage of the second proposed system is that the storage overhead for traitor tracing is constant, which are suitable for commercial applications.**

***Index Terms*— Attribute-based encryption, ciphertext-policy, large universe, white-box traceablity, commercial applications.**

Manuscript received July 24, 2014; revised October 8, 2014 and December 22, 2014; accepted February 10, 2015. Date of publication February 20, 2015; date of current version April 29, 2015. This work was supported in part by the Specialized Research Fund for the Doctoral Program of Higher Education of China through the Prioritized Development Projects under Grant 20130073130004, in part by the National Natural Science Foundation of China under Grant 61371083, Grant 61373154, Grant 61033014, Grant 61402282, and Grant 61411146001, and in part by the Natural Science Foundation of Shanghai of Yang-Fan Plan under Grant 14YF1410400. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Willy Susilo. *(Corresponding author: Xiaolei Dong and Zhenfu Cao.)*

J. Ning is with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: jtning@sjtu.edu.cn).

X. Dong and Z. Cao are with the Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai 200062, China (e-mail: dongxiaolei@sei.ecnu.edu.cn; zfcao@sei.ecnu.edu.cn).

L. Wei is with the College of Information Technology, Shanghai Ocean University, Shanghai 201306, China (e-mail: lfwei@shou.edu.cn).

X. Lin is with the Faculty of Business and Information Technology, University of Ontario Institute of Technology, Oshawa, ON L1H 7K4, Canada (e-mail: xiaodong.lin@uoit.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2015.2405905

I. Introduction

In traditional public key encryption, a user is privileged to share his/her data with others in a private manner. The access of a targeted user or device to the shared data is *all* or *nothing*. In other words, one can get the entire access capability to the shared data if given the secret key; otherwise, nothing will be revealed. In many cases, however, this may not be enough. For example, a user may expect to share his/her data through a more general and expressive way based on the targeted user or a device's credentials. To address this issue, Sahai and Waters [36] introduced the notion of Fuzzy Identity-Based Encryption (FIBE). Goyal *et al.* [14] proposed two complementary forms of Attribute-Based Encryption (ABE): Key-Policy Attribute-Based Encryption (KP-ABE) and Ciphertext-Policy Attribute-Based Encryption (CP-ABE). In the KP-ABE, users' decryption keys are issued according to an access policy and the ciphertexts are annotated by attributes. In the CP-ABE, users' decryption keys are issued according to the attributes they possess and the encrypting party specifies an access policy for the ciphertexts. A series of KP-ABE and CP-ABE schemes have been proposed [4], [12], [16], [18], [22], [26], [29], [30], [34], [35], [38], aiming at better expressiveness, efficiency or security. In particular, large universe and traceability are the two significant progresses in ABE, we will discuss following.

Recently, Rouselakis and Waters [34] proposed a new construction and proof method for Large Universe Attribute-Based Encryption (LU-ABE). In general, an ABE system can be classified to "small universe" and "large universe" constructions. In the "small universe" construction, the attributes are fixed at system setup and the size of the attributes is polynomially bounded, and furthermore the size of public parameters grows linearly with the number of attributes. While in the "large universe" construction, the attributes need not be specified at system setup and the size of the attribute universe is unbounded. The "large universe" construction for ABE system brings an obvious advantage that the designer of the ABE system need not bother to choose a particular bound of the attributes at system setup.

On the other hand, several CP-ABE systems supporting traceability have been proposed [23], [24], [26]. In CP-ABE, each user possesses a set of attributes and can decrypt the ciphertext if his/her attributes satisfy the ciphertext's access policy. This results in an obvious consequence that the

encryptor or system does not know who leaks the decryption key to others intentionally. Due to the fact that the attributes are shared by multiple users and different users may have the same subset of attributes, the encryptor or system has no feasible method to trace the suspicious receiver if the decryption key is leaked. We take Alice (with attributes {Alice, Assistant Professor, Computer Science}) and Bob (with attributes {Bob, Assistant Professor, Computer Science}) as an example. They both have the same decryption keys corresponding to attributes {Assistant Professor, Computer Science} and can decrypt such a ciphertext encrypted by the attributes {Assistant Professor, Computer Science}. Suppose no other receiver in the system has both attributes ({Assistant Professor} and {Computer Science}) at the same time. If it happens to exist a user who can decrypt the ciphertext except Alice and Bob, it is significant to find out who leaks such decryption key to him, Alice or Bob? This drawback should be fixed in practice in case of leaking decryption key. It is necessary to add the property of traceability to the original ABE scheme, to identify who exactly leaks the decryption key. The above traceability is called *white-box traceability* [26], which means that any user who leaks his/her decryption key to the third user or device intentionally or unintentionally will be identified. Also note that there exists a relatively stronger notion named *black-box traceability* [25]: the leakage of the user is the decryption equipment instead of its decryption key.

However, up to now, there exists no practical traceable CP-ABE system supporting the property of large universe as the (non-traceable) CP-ABE system in [34]. Large universe CP-ABE system with white-box traceability is not yet achieved in practice: (1) The CP-ABE systems supporting traceability proposed in [23], [24], and [26] do not support the property of large universe, the attributes need to be fixed at system setup and the size of the attributes is polynomially bounded. Besides, public parameters' size grows linearly with the number of attributes. (2) The large universe CP-ABE system proposed in [34] is the first large universe CP-ABE system secure in the standard model; however, it does not support the property of traceability.

A Motivating Story: Consider a commercial application such as a pay-TV system with huge number of users for example. Each user is labeled with lots of related attributes, which are defined as TV channels that the user have ordered. As a versatile one-to-many encryption mechanism, CP-ABE system is quite suitable in this scenario. The pay-TV system provides several TV channels for users, and those who have paid for the TV channels could satisfy the access policy to decrypt the ciphertext and enjoy the ordered TV channels. CP-ABE enables fine-grained access control to the encrypted data according to attributes in users' ordered lists. However, there are two problems with this approach. First, if someone (who does not have the privilege to access to those TV channels) illegally buys the decryption key from the Internet at a lower cost, she/he could also get access to the TV channels. It is necessary to find out who is selling the decryption key. Second, as the TV channels of the pay-TV system expand, an increasing number of new attributes need to be added to the system to describe the new channels. If the number of the attributes exceeds the bound set during the initial deployment of the pay-TV system, then the entire system has to be re-deployed and possibly all its data will have to be re-encrypted [34], which would be a disaster to the pay-TV in the commercial applications.

The problems, as described above, are the main obstacles when CP-ABE is implemented in commercial applications such as pay-TV systems and social networks. Due to the nature of CP-ABE, if a malicious user leaks its decryption key to others for profits (such as selling the decryption key on the Internet), it is difficult to find out the original key owner from an exposed key since the decryption key is shared by multiple users who have the same attributes. As such, the pay-TV company will suffer severe financial loss. Thus, it is necessary for the pay-TV system to trace the malicious users who intentionally leak the partial or modified decryption keys. Also, as the pay-TV system expands, an increasing new attributes (which describe new TV channels) have to be added into the system. In previous CP-ABE constructions, the attributes are fixed at system setup and the number of the attributes are bounded. If the bound is not specified large enough, the attributes may exhaust if the number of the users exceeds the threshold and the entire system needs to be completely re-built [34]. On the other hand, if the bound is specified too large, it will increase the storage and communication burden of the entire system due to the corresponding increase of the public parameters' size. Thus, it is necessary for the pay-TV system to support flexible number of attributes. Lastly, since the number of users in a pay-TV system could grow fast, the storage for traceability should not increase linearly with the number of users. Otherwise, the storage for traceability will become relatively huge and exhaust if the users increase dramatically. Thus, the storage cost for traceability needs to be at a constant level in an ideal case.

A. Our Contribution

In this paper, we propose two new large universe CP-ABE systems, the Traceable Large Universe CP-ABE system (the T-LU-CPABE system) and the enhanced Traceable Large Universe CP-ABE system (the eT-LU-CPABE system), which are white-box[1] traceable on prime order bilinear groups. To the best of our knowledge, both the T-LU-CPABE system and the eT-LU-CPABE system are the first practical CP-ABE systems that simultaneously support the following two properties: white-box traceability and large universe. Moreover, the eT-LU-CPABE system achieves another remarkable property: constant storage for traitor tracing. Compared with other constructions based on composite order groups, we build our constructions on the efficient prime order bilinear groups. We also prove our new systems selectively secure in the standard model.

[1]In this paper, we mainly aim to obtain a large universe CP-ABE system with white-box traceability. The realization of black-box traceability for large universe CP-ABE system will be our future work.

We solve the obstacles of CP-ABE implementation in the commercial applications such as pay-TV systems and social networks as follows:

1) *White-box traceability*. Both the T-LU-CPABE system and the eT-LU-CPABE system achieve the property of white-box traceability in CP-ABE. Our new systems can trace the malicious users or traitors who may leak the partial or modified decryption keys to others for profits.
2) *Large universe*. Both the T-LU-CPABE system and the eT-LU-CPABE system obtain the property of large universe in white-box traceable CP-ABE. In our new systems attributes need not be fixed at system setup, the attributes' size is not polynomially bounded and the public parameters' size does not grow linearly with the number of attributes.
3) *Constant storage overhead*. The eT-LU-CPABE system does not need to maintain an identity table for tracing as used in [26] and the T-LU-CPABE system. Instead, we adopt the Shamir's $(\bar{t}, \bar{n})$ threshold scheme in tracing the malicious users or traitors, the storage cost for traceability does not grow linearly with the number of the users, it is constant which only depends on the threshold $\bar{t}$.
4) *Dynamical scalability*. It yields another result that the stored data for traceability need not be updated when new users are added into the system or malicious users are removed from the system, which makes the system more practical for applications.

In addition, we also provide some detailed comparisons between our work and some other related work in terms of performances and features, showing that our systems are suitable for practical commercial applications. Additionally, we give some inspirational discussions of the proposed systems which lead to some possible directions of future work.

B. Our Technique

In this subsection, we briefly introduce the main idea we utilize to realize the properties of large universe and white-box traceability before giving the full details in Section V and Section VI.

To realize large universe construction, we adopt the "individual randomness" and "layer" technique from [21] and [34]. We use the "layer" technique to encrypt data securely and to be able to decrypt. We employ two "layers": the "attribute" layer and the "secret sharing" layer, and use a "binder term" to connect these two layers securely. In the "attribute" layer, we utilize u, h terms to provide a Boneh-Boyen-style [5] hash function ($u^A h$). As for the "secret sharing" layer, during **KeyGen** and **Encrypt** phases we use w term to hold the secret randomness r and the secret randomness s's shares respectively. Finally, we use the v term to "bind" this two layers together.

To realize traceability, we use the Boneh-Boyen-style signature [5] in both the T-LU-CPABE system and the eT-LU-CPABE system. Furthermore, we find that the identity table T with the tuple identity and its randomness used in [26] and the T-LU-CPABE system grows linearly with the number of the users.[2] With the number of the users in a system scaling large, the corresponding identity table T for traceability will expand as a result, which leads to heavy burden of the storage space for T. Besides, the corresponding cost of searching K' in T during the **Trace** phase is relatively huge. In our eT-LU-CPABE system, we utilize the Shamir's $(\bar{t}, \bar{n})$ threshold scheme to optimize the property of traceability. We only need store $\bar{t} - 1$ points and the value $f(0)$ on a polynomial $f(x)$ at system setup. Consequentially, our storage for traceability does not grow linearly with the number of the users and is a constant.

The main idea of our traceability in the eT-LU-CPABE system is as follows.

Firstly, the **Setup** algorithm initializes an instance of Shamir's $(\bar{t}, \bar{n})$ threshold scheme $\mathcal{INS}_{(\bar{t},\bar{n})}$ and keeps a polynomial $f(x)$ and $\bar{t} - 1$ points $\{(x_1, y_1), (x_2, y_2), \ldots, (x_{\bar{t}-1}, y_{\bar{t}-1})\}$ on $f(x)$ secret. Then we insert c into the decryption key sk during the **KeyGen** phase where $c = Enc_{\bar{k}_2}(x||y), x = Enc_{\bar{k}_1}(id), y = f(x)$. Note that the tuple (x, y) is a point on $f(x)$. During the **Trace** phase, the algorithm extracts $(x^* = x', y^* = y')$ from $x'||y' = Dec_{\bar{k}_2}(K')$ in the decryption key sk, and then it checks whether sk is issued by system. If $(x^* = x', y^* = y') \in \{(x_1, y_1), (x_2, y_2), \ldots, (x_{\bar{t}-1}, y_{\bar{t}-1})\}$, the algorithm computes $Dec_{\bar{k}_1}(x^*)$ to get id to identify the malicious user directly. Otherwise, the algorithm computes the secret of $\mathcal{INS}_{(\bar{t},\bar{n})}$ by interpolating with $\bar{t} - 1$ points $\{(x_1, y_1), (x_2, y_2), \ldots, (x_{\bar{t}-1}, y_{\bar{t}-1})\}$ and (x^*, y^*). If the recovered secret is equal to $f(0)$, the algorithm computes $Dec_{\bar{k}_1}(x^*)$ to get id to identify the malicious user. If the equation fails, sk is not issued by the system. In this way, we could trace the owner of the decryption key. Meanwhile, it brings the benefit that the system only stores $f(0)$ and $\bar{t} - 1$ points on $f(x)$, and thus the storage for traceability is a constant.

C. Related Work

Sahai and Waters introduced the notion of Fuzzy Identity-Based Encryption in [36]. Goyal *et al.* [14] later formalized two notions of ABE: CP-ABE (where user keys are labeled with sets of attributes and ciphertexts are associated with policies) and KP-ABE (where ciphertexts are labeled with sets of attributes and private keys are associated with access structures). Subsequently, many constructions of selectively secure KP-ABE and CP-ABE systems were proposed [1], [4], [8], [12], [18], [29], [30], [38] Many advances have been made for ABE as the following directions: new proof techniques to obtain fully secure [18], [19], [22], [29], decentralizing trust by setting multiple authorities [6], [7], [20] and outsourcing computation [15], [32]. The first large universe KP-ABE construction was proposed in [21]. It was built on composite order groups and proved selectively secure in the standard model. Then the first large universe KP-ABE construction on prime order groups was proposed in [17] inspired by the

[2]Note that in the extension of [26], it gives another signature scheme for the purpose of removing the identify table T, but unfortunately the new signature scheme is not as efficient as the original one. Besides, it brings some other parameters, which will cause additional computation overhead.

dual pairing vector space framework [27]–[29]. Recently, the first large universe CP-ABE construction [34] built on prime order bilinear groups was proposed by Rouselakis and Waters. It was proved selectively secure in the standard model under "q-type" assumption. Another branch of ABE research considers the problem of traceability. The notion of accountable CP-ABE was first proposed in [24] to prevent illegal key sharing among colluding users. Then a multi-authority ciphertext-policy (AND gates with wildcard) ABE scheme with accountability was proposed in [23], which allowed tracing the identity of a misbehaving user who leaked the decryption key to others. Liu, Cao and Wong lately proposed a white-box [26] and black-box [25] traceability CP-ABE system which supported policies expressed in any monotone access structures.

D. Organization

Section II gives the formal definition of our traceable large universe CP-ABE system (the T-LU-CPABE system) and its security model. Section III gives the formal definition of our enhanced traceable large universe CP-ABE system (the eT-LU-CPABE system) and its security model. Section IV introduces the relevant background. Section V presents the construction of our new T-LU-CPABE system as well as the security proof. Section VI presents the construction of our new eT-LU-CPABE system as well as the security proof. Section VII gives the comparison between our work and some other related work. Some extensions of our work are discussed in Section VIII. Some discussions of our new T-LU-CPABE system and eT-LU-CPABE system are presented in Section IX. Section X presents a briefly conclusion and foresees our future work.

II. Traceable Large Universe CP-ABE System

A. Definition

A Traceable Large Universe CP-ABE system (T-LU-CPABE system) is a CP-ABE system where attributes need not be fixed at system setup and can trace the user by his/her decryption key. We enhance the original large universe CP-ABE system by adding users' identities and a **Trace** algorithm to it according to [26]. In particular, following the notation of the large universe CP-ABE system introduced in [34], a T-LU-CPABE system consists of six algorithms as follows:

- **Setup**$(1^\lambda) \rightarrow (pp, msk)$: The algorithm takes as inputs a security parameter $\lambda \in \mathbb{N}$ encoded in unary. It outputs the public parameters pp and the master secret key msk. We assume that the description of the attribute universe U is contained in the public parameters.[3] In addition, it initializes an identity table $T = \phi$.
- **KeyGen**$(1^\lambda, pp, msk, id, S) \rightarrow sk_{id,S}$: The key generation algorithm takes as inputs the public parameters pp, the master secret key msk and a set of attributes $S \subseteq U$ for a user with identity id. The security parameter in the inputs ensures that it is polynomial time in λ. The algorithm outputs a secret key $sk_{id,S}$ corresponding to S. Then, it puts id into the identity table T.
- **Encrypt**$(1^\lambda, pp, m, \mathbb{A}) \rightarrow ct$: The encryption algorithm takes as inputs the public parameters pp, a plaintext message m, and an access structure $\mathbb{A}$ over U. It outputs the ciphertext ct.
- **Decrypt**$(1^\lambda, pp, sk_{id,S}, ct) \rightarrow m$ or $\perp$: The decryption algorithm takes as inputs the public parameters pp, a secret key $sk_{id,S}$, and a ciphertext ct. It outputs the plaintext m or $\perp$.
- **KeySanityCheck**$(pp, sk) \rightarrow 1$ or 0: The decryption algorithm takes as inputs the public parameters pp and a secret key sk. If sk passes the key sanity check, it outputs 1. Otherwise, it outputs 0. The key sanity check is a deterministic algorithm [11], [13], which is used to guarantee the secret key to be well-formed in the decryption process.
- **Trace**$(pp, sk, T) \rightarrow id$ or $\intercal$: The tracing algorithm takes as inputs the public parameters pp, a secret key sk and an identity table T. The algorithm first verifies whether sk is well-formed to determine whether sk needs to be traced. If sk is well-formed, the algorithm outputs an identity id from the identity table T implying that sk is linked to id. Otherwise, it outputs a special symbol $\intercal$ implying that sk does not need to be traced. We define a secret key sk is *well-formed* which means that **KeySanityCheck**$(pp, sk) \rightarrow 1$.

B. T-LU-CPABE System Selective Security

The security model of our T-LU-CPABE system is similar to that of the LU-CPABE system [34], excepting every key query is companied with an explicit identity. In this subsection we present the definition of selective security for our T-LU-CPABE system. It is parameterized by the security parameter $\lambda \in \mathbb{N}$ and is described by a game between an attacker and a challenger. We define Q be the total number of key queries that the attacker makes and $Q_1 \in \{0, 1, 2, \ldots, Q\}$. The phases of the game are as follows:

- **Initialization**: The attacker claims the challenge access structure $\mathbb{A}^*$ he will attack, and then sends it to the challenger.
- **Setup**: The challenger runs the **Setup**(1^λ) algorithm and sends the public parameters pp to the attacker.
- **Query Phase 1**: In this phase the attacker can adaptively ask for secret keys for the sets of attributes $(id_1, S_1), (id_2, S_2), \ldots, (id_{Q_1}, S_{Q_1})$. For each (id_i, S_i) the challenger calls **KeyGen**$(1^\lambda, pp, msk, id, S) \rightarrow sk_{id,S}$ and sends $sk_{id,S}$ to the attacker. The only restriction is that the attacker cannot query the sets that satisfies the challenge access structure $\mathbb{A}^*$, i.e. $\forall_i \in [Q_1] : S_i \notin \mathbb{A}^*$.
- **Challenge**: The attacker declares two equal length messages m_0 and m_1 and sends them to the challenger. The challenger flips a random coin $\beta \in \{0, 1\}$ and calls **Encrypt**$(1^\lambda, pp, m_\beta, \mathbb{A}^*) \rightarrow ct$. It gives ct to the attacker.

[3] In the previous CP-ABE systems, the attribute universe U was one of the arguments in the **Setup** algorithm. In the large universe case, the attribute universe only depends on the size of the security parameter and the group generation algorithm [34].

- **Query Phase 2**: This is the same as query phase 1. The attacker asks for the secret key for the sets $(id_{Q_1+1}, S_{Q_1+1}), \ldots, (id_Q, S_Q)$ with the same restriction: $\forall_i \in [Q] : S_i \notin \mathbb{A}^*$.
- **Guess**: The attacker outputs a guess $\beta' \in \{0, 1\}$ for β.

The advantage of an attacker is defined to be $Adv = |\Pr[\beta' = \beta] - 1/2|$ in this game.

Definition 1: A traceable large universe ciphertext-policy attribute-based encryption system is selectively secure if all probabilistic polynomial-time (PPT) attackers have at most negligible advantage in λ in the above security game.

C. Traceability

In this subsection, we give the traceability definition for our T-LU-CPABE system. It is described by a game between an attacker and a challenger. We define q be the total number of key queries that the attacker makes. The phases of the game are as follows:

- **Setup**: Here the challenger calls the $\mathbf{Setup}(1^\lambda)$ algorithm and sends the public parameters pp to the attacker.
- **Key Query**: The attacker submits the sets of attributes $(id_1, S_1), \ldots, (id_q, S_q)$ to request the corresponding decryption keys.
- **Key Forgery**: The attacker will output a decryption key sk_*. If $\mathbf{Trace}(pp, sk_*, T) \neq \top$ and $\mathbf{Trace}(pp, sk_*, T) \notin \{id_1, \ldots, id_q\}$, then the attacker wins the game. The advantage of an attacker in this game is defined to be $\Pr[Trace(pp, sk_*, T) \notin \{\top, id_1, \ldots, id_q\}]$.

Definition 2: A traceable large universe ciphertext-policy attribute-based encryption system is fully traceable if there has no polynomial time attacker has non-negligible advantage in the above game.

D. Key Sanity Check

In this subsection, we give the definition of Key Sanity Check for our T-LU-CPABE system according to [2]. It is described by the following game between an adversary and a simulator. On input a security parameter 1^λ ($\lambda \in \mathbb{N}$), a simulator invokes an adversary $\mathcal{A}$ on 1^λ. The adversary $\mathcal{A}$ returns the public parameters pp, a ciphertext ct and two different secret keys $sk_{id,S}$ and $sk'_{id,S}$ corresponding to the same set of attributes S for a user with identity id. $\mathcal{A}$ wins the game if

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

The advantage of the adversary $\mathcal{A}$ in the above game is defined as $\Pr[\mathcal{A}\ wins]$. And it is easy to see that the intuition of "Key Sanity Check" is captured combining the notion captured in the above game and the related algorithms (**KeySanityCheck** and **Decrypt**) defined in this section.

III. Enhanced Traceable Large Universe CP-ABE System

A. Definition

An enhanced Traceable Large Universe CP-ABE system (eT-LU-CPABE system) is a CP-ABE system where attributes need not be fixed at system setup and can trace the user by his/her decryption key. Moreover, we enhance the T-LU-CPABE system by eliminating the identity table T. In particular, following the notation of the T-LU-CPABE system, an eT-LU-CP-ABE system consists of six algorithms as follows:

- $\mathbf{Setup}(1^\lambda) \rightarrow (pp, msk)$: The algorithm takes as inputs a security parameter $\lambda \in \mathbb{N}$ encoded in unary. It outputs the public parameters pp and the master secret key msk. We assume that the description of the attribute universe U is contained in the public parameters. In addition, it initializes an instance of Shamir's $(\bar{t}, \bar{n})$ threshold scheme denoted by $\mathcal{INS}_{(\bar{t},\bar{n})}$.
- $\mathbf{KeyGen}(1^\lambda, pp, msk, id, S) \rightarrow sk_{id,S}$: The key generation algorithm takes as inputs the public parameters pp, the master secret key msk and a set of attributes $S \subseteq U$ for a user with identity id. The security parameter in the inputs ensures that it is polynomial time in λ. The algorithm outputs a secret key $sk_{id,S}$ corresponding to S.
- $\mathbf{Encrypt}(1^\lambda, pp, m, \mathbb{A}) \rightarrow ct$: The encryption algorithm takes as inputs the public parameters pp, a plaintext message m, and an access structure $\mathbb{A}$ over U. It outputs the ciphertext ct.
- $\mathbf{Decrypt}(1^\lambda, pp, sk_{id,S}, ct) \rightarrow m$ or $\perp$: The decryption algorithm takes as inputs the public parameters pp, a secret key $sk_{id,S}$, and a ciphertext ct. It outputs the plaintext m or $\perp$.
- $\mathbf{KeySanityCheck}(pp, sk) \rightarrow 1$ or 0: The decryption algorithm takes as inputs the public parameters pp and a secret key sk. If sk passes the key sanity check, it outputs 1. Otherwise, it outputs 0. The key sanity check is a deterministic algorithm [11], [13], which is used to guarantee the secret key to be well-formed in the decryption process.
- $\mathbf{Trace}(pp, \mathcal{INS}_{(\bar{t},\bar{n})}, msk, sk) \rightarrow id$ or $\top$: The tracing algorithm takes as inputs the public parameters pp, an instance of of Shamir's $(\bar{t}, \bar{n})$ threshold scheme $\mathcal{INS}_{(\bar{t},\bar{n})}$, the master secret key msk, and a secret key sk. The algorithm first verifies whether sk is well-formed to determine whether sk needs to be traced. If sk is well-formed and could recover the secret of $\mathcal{INS}_{(\bar{t},\bar{n})}$, the algorithm outputs an identity id implying that sk is linked to id. Otherwise, it outputs a special symbol $\top$ implying that sk does not need to be traced. We define a secret key sk is *well-formed* which means that $\mathbf{KeySanityCheck}(pp, sk) \rightarrow 1$.

B. eT-LU-CPABE System Selective Security

In this subsection, we present the definition of selective security for our eT-LU-CPABE system. The security model of our eT-LU-CPABE system is the same with that of the T-LU-CPABE system. We refer the interested readers to the

game in Subsection II-B for the game of the security model of our eT-LU-CPABE system.

Definition 3: An enhanced traceable large universe ciphertext-policy attribute-based encryption system is selectively secure if all probabilistic polynomial-time attackers have at most negligible advantage in λ in the above security game.

C. Traceability

In this subsection, we give the traceability definition for our eT-LU-CPABE system. Some of the text below is taken verbatim from the traceability definition of the T-LU-CPABE system. It is described by a game between an attacker and a challenger. We define q be the total number of key queries that the attacker makes. The phases of the game are as follows:

- **Setup**: Here the challenger calls the $\mathbf{Setup}(1^\lambda)$ algorithm and sends the public parameters pp to the attacker.
- **Key Query**: The attacker submits the sets of attributes $(id_1, S_1), \ldots, (id_q, S_q)$ to request the corresponding decryption keys.
- **Key Forgery**: The attacker will output a decryption key sk_*. If $\mathbf{Trace}(pp, \mathcal{INS}_{(\bar{t},\bar{n})}, msk, sk_*) \neq \top$ and $\mathbf{Trace}(pp, \mathcal{INS}_{(\bar{t},\bar{n})}, msk, sk_*) \notin \{id_1, \ldots, id_q\}$, then the attacker wins the game. The advantage of an attacker in this game is defined to be $\Pr[Trace(pp, \mathcal{INS}_{(\bar{t},\bar{n})}, msk, sk_*) \notin \{\top, id_1, \ldots, id_q\}]$.

Definition 4: An enhanced traceable large universe ciphertext-policy attribute-based encryption system is fully traceable if there has no polynomial time attacker has non-negligible advantage in the above game.

D. Key Sanity Check

In this subsection, we give the key sanity check definition for our eT-LU-CPABE system according to [2]. The definition is the same with that of the T-LU-CPABE system, and we refer the interested readers to the definition in Subsection II-D.

IV. Background

A. Notation

We define $[l] = \{1, 2, \ldots, l\}$ for $l \in \mathbb{N}$. By PPT we denote probabilistic polynomial-time. We denote $\mathbb{Z}_p^{l\times n}$ be the set of matrices of size $l \times n$ with elements in $\mathbb{Z}_p$. The set of row vectors of length n (i.e. $\mathbb{Z}_p^{1\times n}$) and the set of column vectors of length n (i.e. $\mathbb{Z}_p^{n\times 1}$) are the two special subsets. We denote $(s_1, s_2, \ldots, s_n)$ be a row vector and $(s_1, s_2, \ldots, s_n)^\perp$ be a column vector. By v_i we denote the i-th element in a vector $\vec{v}$. And by $M\vec{v}$ we denote the inner product of matrix M with vector $\vec{v}$. We define $\mathcal{F}(U_1 \to U_2)$ be the set of functions from set U_1 to U_2.

We denote $GD = (p, \mathbb{G}, \mathbb{G}_T, e)$ be the groups and the bilinear mapping description where $\mathbb{G}$ and $\mathbb{G}_T$ are two multiplicative cyclic groups of prime order p and $e : \mathbb{G} \times \mathbb{G}_T$ is a bilinear map.

B. Access Policy

This subsection presents the definition of access structure referred to [3] and [34].

Definition 5 (Access Structure [3]): Let U denote the attribute universe. A collection $\mathbb{A} \in 2^U$ of non-empty sets of attributes is an access structure on U. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets. A collection $\mathbb{A} \in 2^U$ is called monotone if $\forall B, C \in \mathbb{A} : if\ B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$.

The main idea in ABE is that the role of the users is taken by the attributes. Thus, the access structure $\mathbb{A}$ will contain the authorized sets of attributes. For CP-ABE, if a user of the system posses an authorized set of attributes then he can decrypt the ciphertext, otherwise, he can't get any information from ciphertext if the set he possed is unauthorized. In our construction, we restrict our attention to monotone access structure.

C. Linear Secret-Sharing Schemes

It is shown in [3] that a linear secret sharing scheme can realize any monotone access structure. In this subsection, we will present the definition of linear secret-sharing scheme (LSSS) referred to [3] and [35].

Definition 6 (Linear Secret-Sharing Schemes (LSSS) [3], [34]): Let U denote the attribute universe and p denote a prime. A secret-sharing scheme $\prod$ with domain of secrets $\mathbb{Z}_p$ realizing access structure on U in called linear (over $\mathbb{Z}_p$) if

1. *The shares of a secret $s \in \mathbb{Z}_p$ for each attribute form a vector over $\mathbb{Z}_p$.*
2. *For each access structure $\mathbb{A}$ on U, there exists a matrix M with l rows and n columns called the share-generating matrix. For $i = 1, \ldots, l$, we define a function ρ labels row i of M with attribute $\rho(i)$ from the attribute universe U, i.e. $\rho \in \mathcal{F}([l] \to U)$. When we consider the column vector $\vec{v} = (s, r_2, \ldots, r_n)^\perp$, where $r_2, \ldots, r_n \in \mathbb{Z}_p$ are randomly chosen. Then $M\vec{v} \in \mathbb{Z}_p^{l\times 1}$ is the vector of l shares of the secret s according to $\prod$. The share $(M\vec{v})_j$ "belongs" to attribute $\rho(j)$, where $j \in [l]$.*

As shown in [3], every linear secret-sharing scheme according to the above definition also enjoys the linear reconstruction property, defined as follows: Let $\prod$ be an LSSS for the access structure $\mathbb{A}$, $S \in \mathbb{A}$ be any authorized set and let $I \subset \{1, 2, \ldots, l\}$ be defined as $I = \{i \in [l] \wedge \rho(i) \in S\}$. Then, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i\in I}$ such that for any valid shares $\{\lambda_i = (M\vec{v})_i\}_{i\in I}$ of a secret s according to $\prod$, then $\sum_{i\in I} \omega_i \lambda_i = s$. Additionally, it is shown in [3] that these constants $\{\omega_i\}_{i\in I}$ can be found in time polynomial in the size of the share-generating matrix M. On the other hand, for any unauthorized set S', no such constants $\{\omega_i\}$ exist.

In our construction, an LSSS matrix (M, ρ) will be used to express an access policy associated to a ciphertext.

D. Prime Order Bilinear Groups

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}$ and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map. The bilinear map e has the

following properties:

1. Bilinearity: $\forall u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degeneracy: $e(g, g) \neq 1$.

We say that $\mathbb{G}$ is a bilinear group if the group operations in $\mathbb{G}$ and the bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ can both be computed efficiently. Notice that the map $e(\cdot, \cdot)$ is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

E. Assumptions

We adopt the "q-type" assumption of [34] as this construction's assumption.

Assumption 1 ("q-Type" Assumption [34]): We define the q-type problem as follows. Initially choose a group generation algorithm with input the security parameter, pick a random group element $g \in \mathbb{G}$, and $q + 2$ random exponents $d, s, b_1, b_2, \ldots, b_q \in \mathbb{Z}_p$. If the attacker is given the group description $(p, \mathbb{G}, \mathbb{G}_T, e)$ and $\vec{y}$ including the following terms:

$$
\begin{aligned}
&g, g^s \\
&g^{d^i}, g^{b_j}, g^{sb_j}, g^{d^i b_j}, g^{d^i/b_j^2} \quad \forall (i, j) \in [q, q] \\
&g^{d^i/b_j} \quad \forall (i, j) \in [2q, q] \text{ with } i \neq q + 1 \\
&g^{d^i b_j/b_{j'}^2} \quad \forall (i, j, j') \in [2q, q, q] \text{ with } j \neq j' \\
&g^{sd^i b_j/b_{j'}}, g^{sd^i b_j/b_{j'}^2} \quad \forall (i, j, j') \in [q, q, q] \text{ with } j \neq j',
\end{aligned}
$$

it is hard for the attacker to distinguish $e(g, g)^{sd^{q+1}} \in \mathbb{G}_T$ from an element which is randomly chosen from $\mathbb{G}_T$.

An algorithm $\mathcal{A}$ that outputs $\beta \in \{0, 1\}$ has advantage ϵ in solving the above assumption if $|\Pr[\mathcal{A}(\vec{y}, e(g, g)^{sd^{q+1}}) = 0] - \Pr[\mathcal{A}(\vec{y}, R) = 0]| \geq \epsilon$.

Definition 7: We say that the q-type assumption holds if no PPT algorithm has a non-negligible advantage in solving the q-type problem.

Assumption 2 (l-SDH Assumption [5], [11]): Let $\mathbb{G}$ be a bilinear group of prime order p and g be a generator of $\mathbb{G}$, the l-Strong Diffie-Hellman (l-SDH) problem in $\mathbb{G}$ is defined as follows: given a $(l + 1)$-tuple $(g, g^x, g^{x^2}, \ldots, g^{x^l})$ as inputs, output a pair $(c, g^{1/(c+x)}) \in \mathbb{Z}_p \times \mathbb{G}$. An algorithm $\mathcal{A}$ has advantage ϵ in solving l-SDH in $\mathbb{G}$ if $\Pr[\mathcal{A}(g, g^x, g^{x^2}, \ldots, g^{x^l}) = (c, g^{1/(c+x)})] \geq \epsilon$, where the probability is over the random choice of x in $\mathbb{Z}_p^*$ and the random bits consumed by $\mathcal{A}$.

Definition 8: We say that the (l, t, ϵ)-SDH assumption holds in $\mathbb{G}$ if no t-time algorithm has advantage at least in solving the l-SDH problem in $\mathbb{G}$.

F. Shamir's $(\bar{t}, \bar{n})$ Threshold Scheme

It is well known for Shamir's $(\bar{t}, \bar{n})$ threshold scheme [37] (or Shamir's secret sharing scheme) in cryptography. The essential idea of that scheme is that $\bar{t}$ points on a $\bar{t} - 1$ degree curve are sufficient to confirm such a curve, that is, $\bar{t}$ points are enough to determine a $\bar{t} - 1$ degree polynomial. For a $(\bar{t}, \bar{n})$ threshold scheme, a secret can be divided into $\bar{n}$ parts (or even more), which are sent to each participant a unique part. All of them can be used to reconstruct the secret. Suppose that the secret is assumed to be an element in a finite field $\mathbb{F}_p^*$. Choose $\bar{t} - 1$ random coefficients $a_1, a_2, \cdots, a_{\bar{t}-2} \in \mathbb{F}_p$ and $a_{\bar{t}-1} \in \mathbb{F}_p^*$ and set the secret in the constant term a_0. Note that, we have such a polynomial: $f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{\bar{t}-1} x^{\bar{t}-1}$. Every participant is given a point (x, y) on the above curve, that is, the input to the polynomial x and its output $y = f(x)$. Given a subset with any $\bar{t}$ points, we can recover the constant term a_0 using the Lagrange interpolation.

G. Probabilistic Encryption

Probabilistic encryption [10] is an encryption algorithm with some randomness during the encryption, which leads that encrypting same messages yields different ciphertexts in the various times. The first provably-secure probabilistic public-key encryption scheme was proposed by Goldwasser an Micali [10], based on the hardness of the quadratic residuosity assumption. Later, some efficient probabilistic encryption schemes appeared including ElGamal [9], Paillier [31], and various constructions under the random oracle model. In our scheme, we only use the property of probabilism to output a ciphertext that cannot be distinguished from a random number from the view of the adversary. Without loss of generality, we define such a probabilistic encryption $(Enc_{\bar{k}}, Dec_{\bar{k}})$ in our scheme where $\bar{k}$ is the secret key for encryption and decryption. From the point of efficiency, symmetric encryption scheme is quite suitable since encryption and decryption are easy to preform.

V. Our T-LU-CPABE System

In this section we propose the construction of our new large universe CP-ABE system with white-box traceability.

A. Construction

- **`Setup`**$(1^\lambda) \rightarrow (pp, msk)$: The algorithm runs the group generator algorithm $\mathfrak{g}(1^\lambda)$ and gets the groups and the bilinear mapping description $GD = (p, \mathbb{G}, \mathbb{G}_T, e)$, where $(\mathbb{G}, \mathbb{G}_T)$ are groups of order p and e is the bilinear mapping. Let $U \subseteq \mathbb{Z}_p$ be the attribute universe. The algorithm randomly chooses $g, u, h, w, v \in \mathbb{G}$ and $\alpha, a \in \mathbb{Z}_p$. Besides, it initializes an empty table T. It sets $(GD, g, u, h, w, v, g^a, e(g, g)^\alpha)$ as pp and (α, a) as msk.
- **`KeyGen`**$(1^\lambda, pp, msk, id, S = \{A_1, A_2, \ldots, A_k\} \subseteq \mathbb{Z}_p) \rightarrow sk_{id,S}$: The algorithm chooses $c \in \mathbb{Z}_p^*$[4] and $r, r_1, r_2, \ldots, r_k \in \mathbb{Z}_p$. The decryption key $sk_{id,S}$ is set as follows:

$$
\begin{aligned}
&\langle K = g^{\alpha/(a+c)} w^r, K' = c, L = g^r, L' = g^{ar}, \\
&\{K_{\tau,1} = g^{r_\tau}, K_{\tau,2} = (u^{A_\tau} h)^{r_\tau} v^{-(a+c)r}\}_{\tau \in [k]} \rangle
\end{aligned}
$$

  Then, the algorithm puts the tuple (id, c) into the identity table T.
- **`Encrypt`**$(1^\lambda, pp, m \in \mathbb{G}_T, (M, \rho) \in (\mathbb{Z}_p^{l \times n}, \mathcal{F}([l] \rightarrow \mathbb{Z}_p))) \rightarrow ct$: The algorithm takes the public parameters pp, a plaintext message m and randomly chooses $\vec{y} = (s, y_2, \ldots, y_n)^\perp \in \mathbb{Z}_p^{n \times 1}$, where s is the random secret to be shared according to Subsection IV-C. It gets the vector of the shares $\vec{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_l)^\perp$ by

[4]Note that c will be used to compute $1/(a + c)$ mod p. The algorithm will choose another value $c \in \mathbb{Z}_p^*$ randomly if the unlikely events (c has been in T or $gcd(a + c, p) \neq 1$) happen.

computing the inner product $\lambda_i = M_i \overrightarrow{y}$, where M_i is the i-th row of M. Then it randomly picks l exponents $t_1, t_2, \ldots, t_l \in \mathbb{Z}_p$. The ciphertext ct is set as follows:

$$\langle (M,\rho), C = m \cdot e(g,g)^{\alpha s}, C_0 = g^s, C_0' = g^{as}, \{C_{i,1} = w^{\lambda_i} v^{t_i}, C_{i,2} = (u^{\rho(i)} h)^{-t_i}, C_{i,3} = g^{t_i}\}_{i\in[l]}\rangle$$

It outputs the ciphertext ct.

- **Decrypt**$(1^\lambda, pp, sk_{id,S}, ct) \to m$ or $\perp$: The algorithm first computes the set of rows in M that produces a share to attributes in S, that is, $I = \{i : \rho(i) \in S\}$. If the attribute set S is not an authorized set of the access policy, then it cannot satisfy the access structure of (M,ρ), the algorithm outputs $\perp$. Otherwise, the algorithm lets $\{\omega_i \in \mathbb{Z}_p\}_{i\in I}$ be a set of constants such that $\sum_{i\in I} \omega_i M_i = (1,0,\ldots,0)$, where M_i is the matrix M's i-th row. Note that $\sum_{i\in I} \omega_i \lambda_i = s$ if the attribute set S is authorized, and there may exists other different ways to choose the values of ω_i to satisfy this. Then it computes:

$$\begin{aligned} E &= e(K, C_0^{K'} C_0') = e(g,g)^{\alpha s} e(w,g)^{(a+c)sr} \\ D &= \prod\nolimits_{i\in I} (e(L^{K'} L', C_{i,1}) e(K_{\tau,1}, C_{i,2}) e(K_{\tau,2}, C_{i,3}))^{\omega_i} \\ &= e(g,w)^{(a+c)rs} \\ F &= E/D = e(g,g)^{\alpha s} \end{aligned}$$

where τ is the attribute $\rho(i)$'s index in S (it depends on i). It outputs the plaintext $m = C/F$.

Correctness:

$$F = \frac{E}{D} = \frac{e(g,g)^{\alpha s} e(w,g)^{(a+c)sr}}{\prod_{i\in I} D_1 \cdot D_2 \cdot D_3 \cdot D_4 \cdot D_5} = \frac{e(g,g)^{\alpha s} e(w,g)^{(a+c)sr}}{e(g,w)^{(a+c)r\sum_{i\in I}\omega_i\lambda_i}} = e(g,g)^{\alpha s}$$

where

$$\begin{aligned} &D_1 = e(g,w)^{(a+c)r\lambda_i\omega_i}, \quad D_2 = e(g,v)^{(a+c)rt_i\omega_i}, \\ &D_3 = e(g, u^{\rho(i)}h)^{-r_\tau t_i \omega_i}, \quad D_4 = e(u^{\rho(i)}h, g)^{r_\tau t_i \omega_i}, \\ &D_5 = e(v,g)^{-(a+c)rt_i\omega_i}. \end{aligned}$$

- **KeySanityCheck**$(pp, sk) \to 1$ or 0: The algorithm takes as inputs the public parameters pp and a secret key sk. The secret key sk passes the key sanity check if
  (1) sk is in the form of $(K, K', L, L', \{K_{\tau,1}, K_{\tau,2}\}_{\tau\in k})$ and $K' \in \mathbb{Z}_p^*$, $K, L, L', K_{\tau,1}, K_{\tau,2} \in \mathbb{G}$.
  (2) $e(L', g) = e(L, g^a)$.
  (3) $e(K, g^a g^{K'}) = e(g,g)^\alpha e(L' L^{K'}, w)$.
  (4) $\exists \tau \in [k],\ s.t.\ e(K_{\tau,2}, g) e(L^{K'} L', v) = e(K_{\tau,1}, h) \cdot e(K_{\tau,1}, u)^{A_\tau}$.

  If sk passes the key sanity check, the algorithm outputs 1. Otherwise, it outputs 0.
- **Trace**$(pp, sk, T) \to id$ or $\top$: If **KeySanityCheck**$(pp, sk) \to 0$, the algorithm will output $\top$. Otherwise, sk is a well-formed decryption key, and the algorithm will do as follows:
  (1) The algorithm searches K' in the identity table T. If K' can be found in the table T, it outputs the corresponding id to identity the malicious user. Otherwise, go to (2).
  (2) The algorithm outputs $\top$, which never appears in the identity table T.

B. Selective Security Proof

In the selective security proof, although we can proof that directly based on the Assumption 1 as [34] does, for simplicity, we will reduce the selective security of our T-LU-CPABE system to that of Rouselakis and Waters's system in [34] which is proved selectively secure under Assumption 1.

For simplicity, we denote by $\Sigma_{lucpabe}$, $\Sigma_{tlucpabe}$ the LU-CPABE system in [34] and our system respectively. Note that the security model of our system $\Sigma_{tlucpabe}$ is almost same with that of the system $\Sigma_{lucpabe}$ in [34], excepting every key query is companied with an explicit identity.

Lemma 1 [34]: If the assumption 1 holds, then the LU-CPABE system $\Sigma_{lucpabe}$ is selectively secure.

Selective security of our new T-LU-CPABE system:

Lemma 2: If the LU-CPABE system $\Sigma_{lucpabe}$ is selectively secure in the game of [34], then our new T-LU-CPABE system $\Sigma_{tlucpabe}$ is selectively secure in the game of Subsection II-B.

Proof: Suppose there exists a PPT adversary $\mathcal{A}$ with a challenge matrix M^* that has advantage $Adv_{\mathcal{A}}\Sigma_{tlucpabe}$ in selectively breaking our T-LU-CPABE system $\Sigma_{tlucpabe}$, where M^* is an $l \times n$ matrix satisfies the restriction that $l, n \le q$. We construct a PPT algorithm $\mathcal{B}$ that has advantage $Adv_{\mathcal{B}}\Sigma_{lucpabe}$ in selectively breaking the underlying LU-CPABE system $\Sigma_{lucpabe}$, which equals to $Adv_{\mathcal{A}}\Sigma_{tlucpabe}$.

- **Initialization**: $\mathcal{B}$ gets a challenge policy (M^*, ρ^*) from $\mathcal{A}$ and then sends this received challenge policy to $\Sigma_{lucpabe}$. Note that M^* is an $l \times n$ matrix, where $l, n \le q$, and $\rho^* \in \mathcal{F}([l] \to \mathbb{Z}_p)$.
- **Setup:** $\Sigma_{lucpabe}$ gives $\mathcal{B}$ the public parameter $pp_{\Sigma_{lucpabe}}$ as follows:

$$\begin{aligned} g &= g, \quad w = g^d \\ v &= g^{\tilde{v}} \cdot \prod\nolimits_{(j,k)\in[l,n]} (g^{d^k/b_j})^{M^*_{j,k}} \\ u &= g^{\tilde{u}} \cdot \prod\nolimits_{(j,k)\in[l,n]} (g^{d^k/b_j^2})^{M^*_{j,k}} \\ h &= g^{\tilde{h}} \cdot \prod\nolimits_{(j,k)\in[l,n]} (g^{d^k/b_j^2})^{-\rho^*(j)M^*_{j,k}} \\ e(g,g)^\alpha &= e(g^d, g^{d^q}) \cdot e(g,g)^{\tilde{\alpha}} \end{aligned}$$

  Then $\mathcal{B}$ randomly chooses $a \in \mathbb{Z}_p$, adds g^a to $pp_{\Sigma_{lucpabe}}$ as a new public parameter pp and gives the new $pp = (D, g, u, h, w, v, g^a, e(g,g)^\alpha)$ to $\mathcal{A}$. Finally, it initializes an identity table $T = \phi$.
- **Query Phase 1**: The adversary $\mathcal{A}$ will submit (id, S) to $\mathcal{B}$ to query a decryption key, then $\mathcal{B}$ submits S to $\Sigma_{lucpabe}$ and gets the corresponding decryption key as follows:

$$\begin{aligned} \hat{K}_0 &= g^{\tilde{\alpha}} (g^d)^{\tilde{r}} \prod_{i=2}^{n} (g^{d^{q+2-i}})^{w_i}, \\ \hat{K}_1 &= g^{\tilde{r}} \prod\nolimits_{i\in[n]} (g^{d^{q+1-i}})^{w_i} \\ \hat{K_{\tau,2}} &= g^{\tilde{r_\tau}} \cdot \prod_{i'\in[l], \rho^*(i')\notin S} (g^{b_{i'}})^{\frac{\tilde{r}}{A_\tau - \rho^*(i')}} \\ &\quad \cdot \prod_{(i,i')\in[n,l], \rho^*(i')\notin S} (g^{b_{i'} d^{q+1-i}})^{\frac{w_i}{A_\tau - \rho^*(i')}} \end{aligned}$$

$$\hat{K_{\tau,3}} = \hat{\Psi} \cdot \hat{\Phi}\ ,\quad where$$
$$\hat{\Psi} = (u^{A_\tau} h)^{\tilde{r}_\tau} \cdot (\hat{K_{\tau,2}}/g^{\tilde{r}_\tau})^{\tilde{u} A_\tau + \tilde{h}} \cdot \prod_{(i',j,k)\in[l,l,n],\, \rho^*(i')\notin S} (g^{\frac{b_{i'} d^k}{b_j^2}})^{\frac{\tilde{r}(A_\tau - \rho^*(j)) M^*_{j,k}}{A_\tau - \rho^*(i')}} \cdot \prod_{\substack{(i,i',j,k)\in[n,l,l,n] \\ \rho^*(i')\notin S,\,(j\neq i'\vee i\neq k)}} (g^{\frac{b_{i'} d^{q+1+k-i}}{b_j^2}})^{\frac{(A_\tau - \rho^*(j)) w_i M^*_{j,k}}{A_\tau - \rho^*(i')}}$$
$$\hat{\Phi} = v^{-\tilde{r}} \prod_{i\in[n]} (g^{d^{q+1-i}})^{-\tilde{v} w_i} \cdot \prod_{(i,j,k)\in[n,l,n],\, i\neq k} (g^{\frac{d^{q+1+k-i}}{b_j}})^{-w_i M^*_{j,k}}$$

$\mathcal{B}$ chooses chooses $c \in \mathbb{Z}_p^*$.[5] Then $\mathcal{B}$ sets $r = \tilde{r}/(a+c)$ and $K' = c$ implicitly and randomly chooses $g' \in \mathbb{G}$, then computes

$$K = (\hat{K_0})^{\frac{1}{a+c}} = g^{\frac{\tilde{a}}{a+c}} ((g^d)^{\tilde{r}} \prod_{i=2}^{n} (g^{d^{q+2-i}})^{w_i})^{\frac{1}{a+c}}$$
$$L = (\hat{K_1})^{\frac{1}{a+c}} = g^{\frac{\tilde{r}}{a+c}} (\prod_{i\in[n]} (g^{d^{q+1-i}})^{w_i})^{\frac{1}{a+c}} = g^r (\prod_{i\in[n]} (g^{d^{q+1-i}})^{w_i})^{\frac{1}{a+c}}$$
$$L' = (\hat{K_1})^{\frac{a}{a+c}} g' = g^{\frac{a\tilde{r}}{a+c}} (\prod_{i\in[n]} (g^{d^{q+1-i}})^{w_i})^{\frac{a}{a+c}} g' = g^{ar} (\prod_{i\in[n]} (g^{d^{q+1-i}})^{w_i})^{\frac{a}{a+c}} g'$$
$$K_{\tau,1} = \hat{K_{\tau,2}},\quad K_{\tau,2} = \hat{K_{\tau,3}} = \hat{\Psi} \cdot \hat{\Phi}$$

$\mathcal{B}$ sends the decryption key $sk_{id,S} = \langle K, K', L, L', \{K_{\tau,1}, K_{\tau,2}\}_{\tau\in[|S|]}\rangle$ to $\mathcal{A}$. Note that g' makes L' uncorrelated to L. Then, $\mathcal{B}$ puts the tuple (id, c) into the identity table T.

- **Challenge**: The attacker $\mathcal{A}$ outputs two equal length messages (m_0, m_1) and sends them to $\mathcal{B}$. Then $\mathcal{B}$ submits (m_0, m_1) to $\Sigma_{lucpabe}$, and gets the challenge ciphertext as follows:

$$\langle (M^*, \rho^*),\ \hat{C} = m_\beta \cdot T' \cdot e(g, g^s)^{\tilde{\alpha}},\ \hat{C_0} = g^s,$$
$$\hat{C_{i,1}} = w^{\tilde{\lambda}_i} \cdot (g^{s b_i})^{-\tilde{v}} \cdot \prod_{(j,k)\in[l,n],\, j\neq i} (g^{s d^k b_i / b_j})^{-M^*_{j,k}},$$
$$\hat{C_{i,2}} = (g^{s b_i})^{-(\tilde{u}\rho^*(i) + \tilde{h})} \cdot \prod_{(j,k)\in[l,n],\, j\neq i} (g^{s d^k b_i / b_j^2})^{-(\rho^*(i) - \rho^*(j)) M^*_{j,k}},$$
$$\hat{C_{i,3}} = g^{t_i} = (g^{s b_i})^{-1}\rangle$$

where T' is the challenge term and g^s the corresponding term of the assumption in $\Sigma_{lucpabe}$.

Then $\mathcal{B}$ sets $C = \hat{C}, C_0 = \hat{C_0}, C_0' = (\hat{C_0})^a = g^{as}$, $C_{i,1} = \hat{C_{i,1}}, C_{i,2} = \hat{C_{i,2}}, C_{i,3} = \hat{C_{i,3}}$. Finally, $\mathcal{B}$ gives the challenge ciphertext $ct = \langle (M^*, \rho^*), C, C_0, C_0', \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i\in[l]}\rangle$ to $\mathcal{A}$.

- **Query Phase 2**: This phase is the same with Phase 1.
- **Guess**: The attacker outputs his guess β', and gives it to $\mathcal{B}$. Then $\mathcal{B}$ gives β' to $\Sigma_{lucpabe}$.

[5]As described in Subsection V-A, c will be used to compute $1/(a+c)$ mod p. The algorithm will choose another value $c \in \mathbb{Z}_p^*$ randomly if the unlikely events (c has been in T or $gcd(a+c, p) \neq 1$) happen.

Since the distributions of the public parameters, decryption keys and challenge ciphertext in the above game are the same as that in the real system, we have $Adv_{\mathcal{B}}\Sigma_{lucpabe} = Adv_{\mathcal{A}}\Sigma_{tlucpabe}$. □

Theorem 1: If Assumption 1 holds, then our T-LU-CPABE system is selectively secure.

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

C. Traceability Proof

In this subsection, we will give the traceability proof of our T-LU-CPABE system based on l-SDH assumption. We use a proof method from [5] and [26].

Theorem 2: If the l-SDH assumption holds, then our T-LU-CPABE system is fully traceable provided that $q < l$.

Proof: Suppose there exists a PPT adversary $\mathcal{A}$ that has non-negligible advantage in winning the traceability game after making q key queries, w.l.o.g., assuming $l = q + 1$, we construct a PPT algorithm $\mathcal{B}$ that has non-negligible advantage in breaking the l-SDH assumption. $\mathcal{B}$ is given an instance of l-SDH assumption problem as follows.

Let $\mathbb{G}$ be a bilinear group of order p and g be a generator of $\mathbb{G}$, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_\mathbb{T}$ be a bilinear map, $a \in \mathbb{Z}_p^*$. $\mathcal{B}$ is given an instance $\mathcal{INS}_{SDH} = (\mathbb{G}, \mathbb{G}_\mathbb{T}, p, e, \bar{g}, \bar{g}^a, \bar{g}^{a^2}, \ldots, \bar{g}^{a^l})$. $\mathcal{B}$'s goal is to output a tuple $(c_i, w_i) \in \mathbb{Z}_p \times \mathbb{G}$ satisfying $w_i = \bar{g}^{1/(a+c_i)}$ for solving the l-SDH problem. Before starting the traceability game with $\mathcal{A}$, $\mathcal{B}$ takes $(\mathbb{G}, \mathbb{G}_\mathbb{T}, p, e, \bar{g}, \bar{g}^a, \bar{g}^{a^2}, \ldots, \bar{g}^{a^l})$ as inputs and sets $B_i = \bar{g}^{a^i}$ for $i = 0, 1, \ldots, l$, then interacts with $\mathcal{A}$ in the traceability game as follows:

- **Setup**: $\mathcal{B}$ randomly chooses q distinct values $c_1, \ldots, c_q \in \mathbb{Z}_p^*$. Let $f(y)$ be the polynomial $f(y) = \prod_{i=1}^{q} (y + c_i)$. Expand $f(y)$ and write $f(y) = \sum_{i=0}^{q} \alpha_i y^i$, where $\alpha_0, \ldots, \alpha_q \in \mathbb{Z}_p$ are the coefficients of the polynomial $f(y)$. Then $\mathcal{B}$ computes $g \leftarrow \prod_{i=0}^{q} (B_i)^{\alpha_i} = \bar{g}^{f(a)}$ and $g^a \leftarrow \prod_{i=1}^{q+1} (B_i)^{\alpha_{i-1}} = \bar{g}^{f(a)\cdot a}$. $\mathcal{B}$ randomly chooses $\alpha, \theta \in \mathbb{Z}_p$ and $u, h, v \in \mathbb{G}$, and then gives $\mathcal{A}$ the public parameter $pp = (GD, g, u, h, w = g^\theta, v, g^a, e(g,g)^\alpha)$. Also, it initializes an identity table $T = \phi$.
- **Key Query**: $\mathcal{A}$ submits (id_i, S_i) to $\mathcal{B}$ to query a decryption key. When it comes to the i-th query, we assume $i \leq q$. Let $f_i(y)$ be the polynomial $f_i(y) = f(y)/(y + c_i) = \prod_{j=1, j\neq i}^{q} (y + c_j)$. Expand $f_i(y)$ and write $f_i(y) = \sum_{j=0}^{q-1} \beta_j y^j$. $\mathcal{B}$ computes $\sigma_i \leftarrow \prod_{j=0}^{q-1} (B_j)^{\beta_j} = \bar{g}^{f_i(a)} = \bar{g}^{f(a)/(a+c_i)} = g^{1/(a+c_i)}$. $\mathcal{B}$ chooses $r, r_1, \ldots, r_k \in \mathbb{Z}_p$ randomly, then gives $\mathcal{A}$ the sk_{id_i, S_i} as follows:

$$\langle K = (\sigma_i)^\alpha w^r = g^{\alpha/(a+c_i)} w^r,\ K' = c_i,$$
$$L = g^r,\ L' = g^{ar},\ \{K_{\tau,1} = g^{r_\tau},$$
$$K_{\tau,2} = (u^{A_\tau} h)^{r_\tau} (v^a \cdot v^{c_i})^{-r} = (u^{A_\tau} h)^{r_\tau} v^{-(a+c_i)r}\}_{\tau\in[k]}\rangle$$

Then, $\mathcal{B}$ puts the tuple (id_i, c_i) into the identity table T.

- **Key Forgery**: $\mathcal{A}$ submits a decryption key sk_* to $\mathcal{B}$. Note that the distributions of public parameters and decryption keys in the above game are the same as that in the real system. Let $\epsilon_\mathcal{A}$ denote the event that $\mathcal{A}$ wins

the game, i.e. sk_* is in the form of $sk_* = (K, K', L, L', \{K_{\tau,1}, K_{\tau,2}\}_{\tau\in[|S_*|]})$, satisfies the key sanity check and $K' \notin \{c_1, \ldots, c_q\}$. If $\epsilon_{\mathcal{A}}$ does not happen, $\mathcal{B}$ chooses a random tuple $(c_s, w_s) \in \mathbb{Z}_p \times \mathbb{G}$ as the solution to l-SDH problem. If $\epsilon_{\mathcal{A}}$ happens, using long division $\mathcal{B}$ writes the polynomial f as $f(y) = \gamma(y)(y + K') + \gamma_{-1}$ for some polynomial $\gamma(y) = \sum_{i=0}^{q-1}(\gamma_i y^i)$ and some $\gamma_{-1} \in \mathbb{Z}_p$. Note that $\gamma_{-1} \neq 0$, since $f(y) = \prod_{i=1}^{q}(y + c_i), c_i \in \mathbb{Z}_p^*$ and $K' \notin \{c_1, .., c_q\}$, as thus $y + K'$ does not divide $f(y)$. And $\mathcal{B}$ computes a tuple $(c_s, w_s) \in \mathbb{Z}_p \times \mathbb{G}$ as follows:

Assuming $L = g^r$ where $r \in \mathbb{Z}_p$ is unknown, we have $L' = g^{ar}, K = g^{\alpha/(a+K')}w^r$. Then $\mathcal{B}$ computes $1/\gamma_{-1} \pmod p$ (since $gcd(\gamma_{-1}, p) = 1$) and then does following computation:

$$\sigma \leftarrow (K/L^{\theta})^{\alpha^{-1}} = g^{\frac{1}{a+K'}} = \bar{g}^{\frac{f(a)}{a+K'}} = \bar{g}^{\gamma(a)}\bar{g}^{\frac{\gamma_{-1}}{(a+K')}}$$
$$w_s \leftarrow (\sigma \cdot \prod_{i=0}^{q-1} B_i^{-\gamma_i})^{\frac{1}{\gamma_{-1}}} = \bar{g}^{\frac{1}{a+K'}},$$
$$c_s \leftarrow K' mod\ p \in \mathbb{Z}_p$$

Note that $e(\bar{g}^a \cdot \bar{g}^{c_s}, w_s) = e(\bar{g}^a \cdot \bar{g}^{K'}, \bar{g}^{\frac{1}{a+K'}}) = e(\bar{g}, \bar{g})$. Therefore (c_s, w_s) is a solution for the l-SDH problem.

Now we evaluate the advantage of $\mathcal{B}$ in breaking l-SDH assumption.

Let $\epsilon_{SDH}(c_s, w_s)$ denote the event that (c_s, w_s) is a solution for the l-SDH problem, which can be verified by checking whether $e(\bar{g}^a \cdot \bar{g}^{c_s}, w_s) = e(\bar{g}, \bar{g})$ holds. Note that when $\mathcal{B}$ randomly chooses (c_s, w_s), $\epsilon_{SDH}(c_s, w_s)$ happens with negligible probability, say zero for simplicity. And the probability of (c_s, w_s) satisfies $e(\bar{g}^a \cdot \bar{g}^{c_s}, w_s) = e(\bar{g}, \bar{g})$ is 1 in the case of $(\mathcal{A}\ win \wedge gcd(\gamma_{-1}, p) = 1)$ when $\mathcal{B}$ outputs (c_s, w_s).

So $\mathcal{B}$ solves the l-SDH problem with probability

$$\begin{aligned}
&\Pr[\epsilon_{SDH}(c_s, w_s)] \\
&= \Pr[\epsilon_{SDH}(c_s, w_s)|\overline{\mathcal{A}\ win}] \cdot \Pr[\overline{\mathcal{A}\ win}] \\
&\quad + \Pr[\epsilon_{SDH}(c_s, w_s)|\mathcal{A}\ win \wedge gcd(\gamma_{-1}, p) \neq 1] \\
&\quad \cdot \Pr[\mathcal{A}\ win \wedge gcd(\gamma_{-1}, p) \neq 1] \\
&\quad + \Pr[\epsilon_{SDH}(c_s, w_s)|\mathcal{A}\ win \wedge gcd(\gamma_{-1}, p) = 1] \\
&\quad \cdot \Pr[\mathcal{A}\ win \wedge gcd(\gamma_{-1}, p) = 1] \\
&= 0 + 0 + 1 \cdot \Pr[\mathcal{A}\ win \wedge gcd(\gamma_{-1}, p) = 1] \\
&= \Pr[\mathcal{A}\ win \wedge gcd(\gamma_{-1}, p) = 1] \\
&= \Pr[\mathcal{A}\ win] \cdot \Pr[gcd(\gamma_{-1}, p) = 1] = \epsilon_{\mathcal{A}}
\end{aligned}$$

Therefore $\mathcal{B}$ can solve the l-SDH problem with non-negligible advantage, which conflicts with the l-SDH assumption. □

D. Key Sanity Check Proof

In this subsection, we will give the key sanity check proof of our T-LU-CPABE system. We use the proof method from [2].

Theorem 3: The advantage of an adversary in the key sanity check game (in Subsection II-D) is negligible for our T-LU-CPABE system.

Proof: Let the output of an adversary $\mathcal{A}$ be the public parameters pp, a ciphertext $ct = \langle (M, \rho), C, C_0, C_0', \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i\in[l]}\rangle$, two different secret keys $sk_{id,S} = \langle K, K', L, L', \{K_{\tau,1}, K_{\tau,2}\}_{\tau\in[k]}\rangle$ and $\tilde{sk}_{id,S} = \langle \tilde{K}, \tilde{K}', \tilde{L}, \tilde{L}', \{\tilde{K}_{\tau,1}, \tilde{K}_{\tau,2}\}_{\tau\in[k]}\rangle$. $\mathcal{A}$ wins implies that the following conditions (as defined in the key sanity check game in Subsection II-D) are all fulfilled.

Conditions (1)−(5):

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

Condition (1) implies

(1) $K' \in \mathbb{Z}_p^*, K, L, L', K_{\tau,1}, K_{\tau,2} \in \mathbb{G}$.
(2) $e(L', g) = e(L, g^a)$.
(3) $e(K, g^a g^{K'}) = e(g, g)^{\alpha} e(L'L^{K'}, w)$.
(4) $\exists\tau \in [k],\ s.t.\ e(K_{\tau,2}, g)e(L^{K'}L', v) = e(K_{\tau,1}, h) \cdot e(K_{\tau,1}, u)^{A_\tau}$.

Similarly, condition (2) implies

(1) $\tilde{K}' \in \mathbb{Z}_p^*, \tilde{K}, \tilde{L}, \tilde{L}', \tilde{K}_{\tau,1}, \tilde{K}_{\tau,2} \in \mathbb{G}$.
(2) $e(\tilde{L}', g) = e(\tilde{L}, g^a)$.
(3) $e(\tilde{K}, g^a g^{\tilde{K}'}) = e(g, g)^{\alpha} e(\tilde{L}'\tilde{L}^{\tilde{K}'}, w)$.
(4) $\exists\tau \in [k],\ s.t.\ e(\tilde{K}_{\tau,2}, g)e(\tilde{L}^{\tilde{K}'}\tilde{L}', v) = e(\tilde{K}_{\tau,1}, h) \cdot e(\tilde{K}_{\tau,1}, u)^{A_\tau}$.

From conditions (1) and (3), we have

$$\begin{aligned}
E &= e(K, C_0^{K'}C_0') = e(g, g)^{\alpha s} e(w, g)^{(a+c)sr} \\
D &= \prod_{i\in I}(e(L^{K'}L', C_{i,1})e(K_{\tau,1}, C_{i,2})e(K_{\tau,2}, C_{i,3}))^{\omega_i} \\
&= e(g, w)^{(a+c)rs} \\
F &= E/D = e(g, g)^{\alpha s},\ m = C/F.
\end{aligned}$$

Similarly, from conditions (2) and (4), we have

$$\begin{aligned}
\tilde{E} &= e(\tilde{K}, C_0^{\tilde{K}'}C_0') = e(g, g)^{\alpha s} e(w, g)^{(a+c)sr} \\
\tilde{D} &= \prod_{i\in I}(e(\tilde{L}^{\tilde{K}'}\tilde{L}', C_{i,1})e(\tilde{K}_{\tau,1}, C_{i,2})e(\tilde{K}_{\tau,2}, C_{i,3}))^{\omega_i} \\
&= e(g, w)^{(a+c)rs} \\
\tilde{F} &= \tilde{E}/\tilde{D} = e(g, g)^{\alpha s},\ m = C/\tilde{F}.
\end{aligned}$$

From conditions (1) − (4), we have

$$F = E/D = e(g, g)^{\alpha s} = \tilde{F} = \tilde{E}/\tilde{D}, m = C/F = C/\tilde{F} \quad (*)$$

However, condition (5) implies that $C/F \neq C/\tilde{F}$, where $F = E/D, \tilde{F} = \tilde{E}/\tilde{D}$, which contradicts to $(*)$. Thus $\mathcal{A}$ wins the game only with negligible probability. □

VI. Our eT-LU-CPABE System

In this section we present the new construction of our enhanced large universe CP-ABE system with white-box traceability. Note that the eT-LU-CPABE System is an enhanced system based on the T-LU-CPABE System. Some of the text below is taken verbatim from that of the T-LU-CPABE system. Compared with the T-LU-CPABE System, the significant and remarkable advantage of our new eT-LU-CPABE system is that the system does not need to maintain the identity table T and the storage overhead for traitor tracing is constant.

A. Construction

- **Setup**$(1^\lambda) \rightarrow (pp, msk)$: The algorithm runs the group generator algorithm $\mathfrak{g}(1^\lambda)$ and gets the groups and the bilinear mapping description $GD = (p, \mathbb{G}, \mathbb{G}_T, e)$, where $(\mathbb{G}, \mathbb{G}_T)$ are groups of order p and e is the bilinear mapping. Let $U \subseteq \mathbb{Z}_p$ be the attribute universe. The algorithm randomly chooses $g, u, h, w, v \in \mathbb{G}$ and α, $a \in \mathbb{Z}_p$. Besides, the algorithm chooses a probabilistic encryption scheme (Enc, Dec) [10] from a binary string to $\mathbb{Z}_p^*$ with different secret key $\bar{k_1}, \bar{k_2}$. Furthermore, it initializes an instance of Shamir's $(\bar{t}, \bar{n})$ threshold scheme $\mathcal{INS}_{(\bar{t},\bar{n})}$[6] [37] and keeps $f(x)$[7] and $\bar{t}-1$ points $\{(x_1, y_1), (x_2, y_2), \ldots, (x_{\bar{t}-1}, y_{\bar{t}-1})\}$ secret. It sets $(GD, g, u, h, w, v, g^a, e(g,g)^\alpha)$ as pp and $(\alpha, a, \bar{k_1}, \bar{k_2})$ as msk.
- **KeyGen**$(1^\lambda, pp, msk, id, S = \{A_1, A_2, \ldots, A_k\} \subseteq \mathbb{Z}_p) \rightarrow sk_{id,S}$: The algorithm computes: $x = Enc_{\bar{k_1}}(id)$, $y = f(x), c = Enc_{\bar{k_2}}(x||y)$. Note that the computing result c is not distinguished from a random number.[8] And it randomly chooses $r, r_1, r_2, \ldots, r_k \in \mathbb{Z}_p$ The decryption key $sk_{id,S}$ is set as follows:

$$\langle K = g^{\alpha/(a+c)} w^r, K' = c, L = g^r, L' = g^{ar},$$
$$\{K_{\tau,1} = g^{r_\tau}, K_{\tau,2} = (u^{A_\tau} h)^{r_\tau} v^{-(a+c)r}\}_{\tau \in [k]}\rangle$$

- **Encrypt**$(1^\lambda, pp, m \in \mathbb{G}_T, (M, \rho) \in (\mathbb{Z}_p^{l \times n}, \mathcal{F}([l] \rightarrow \mathbb{Z}_p))) \rightarrow ct$: The algorithm takes the public parameters pp, a plaintext message m and randomly chooses $\overrightarrow{y} = (s, y_2, \ldots, y_n)^\perp \in \mathbb{Z}_p^{n \times 1}$, where s is the random secret to be shared according to Subsection IV-C. It gets the vector of the shares $\overrightarrow{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_l)$ by computing the inner product $\lambda_i = M_i \overrightarrow{y}$, where M_i is the i-th row of M. Then it randomly picks l exponents $t_1, t_2, \ldots, t_l \in \mathbb{Z}_p$. The ciphertext ct is set as follows:

$$\langle (M, \rho), C = m \cdot e(g,g)^{\alpha s}, C_0 = g^s, C_0' = g^{as},$$
$$\{C_{i,1} = w^{\lambda_i} v^{t_i}, C_{i,2} = (u^{\rho(i)} h)^{-t_i}, C_{i,3} = g^{t_i}\}_{i \in [l]}\rangle$$

  It outputs the ciphertext ct.
- **Decrypt**$(1^\lambda, pp, sk_{id,S}, ct) \rightarrow m$ or $\perp$: The algorithm first computes the set of rows in M that produces a share to attributes in S, that is, $I = \{i : \rho(i) \in S\}$. If the attribute set S is not an authorized set of the access policy, then it cannot satisfy the access structure of (M, ρ), the algorithm outputs $\perp$. Otherwise, the algorithm lets $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ be a set of constants such that $\sum_{i \in I} \omega_i M_i = (1, 0, \ldots, 0)$, where M_i is the matrix M's i-th row. Note that $\sum_{i \in I} \omega_i \lambda_i = s$ if the attribute set S is authorized, and there may exists other different ways to choose the values of ω_i to satisfy this. Then it computes:

$$E = e(K, C_0^{K'} C_0') = e(g,g)^{\alpha s} e(w,g)^{(a+c)sr}$$
$$D = \prod_{i \in I} (e(L^{K'} L', C_{i,1}) e(K_{\tau,1}, C_{i,2}) e(K_{\tau,2}, C_{i,3}))^{\omega_i}$$
$$= e(g,w)^{(a+c)rs}$$
$$F = E/D = e(g,g)^{\alpha s}$$

  where τ is the attribute $\rho(i)$'s index in S (it depends on i). It outputs the plaintext $m = C/F$.

  Correctness: Correctness is the same with that of the T-LU-CPABE system.
- **KeySanityCheck**$(pp, sk) \rightarrow 1$ or 0: The algorithm takes as inputs the public parameters pp and a secret key sk. The key sanity check is the same with that of the T-LU-CPABE system. If sk passes the key sanity check, the algorithm outputs 1. Otherwise, it outputs 0.
- **Trace**$(pp, \mathcal{INS}_{(\bar{t},\bar{n})}, msk, sk) \rightarrow id$ or $\top$: If **KeySanityCheck** $\rightarrow 0$, the algorithm will output $\top$. Otherwise, sk is a well-formed decryption key, and the algorithm will do as follows:
  (1) The algorithm extracts $(x^* = x, y^* = y)$ from $x||y = Dec_{\bar{k_2}}(K')$ in sk.
  (2) If $(x^* = x, y^* = y) \in \{(x_1, y_1), (x_2, y_2), \ldots, (x_{\bar{t}-1}, y_{\bar{t}-1})\}$, the algorithm computes $Dec_{\bar{k_1}}(x^*)$ to get id to identify the malicious user (with id). Otherwise, go to (3).
  (3) The algorithm recovers the secret a_0^* of $\mathcal{INS}_{(\bar{t},\bar{n})}$ by interpolating with $\bar{t}-1$ points $\{(x_1, y_1), (x_2, y_2), \ldots, (x_{\bar{t}-1}, y_{\bar{t}-1})\}$ and $(x^* = x, y^* = y)$. If $a_0^* = f(0)$, it computes $Dec_{\bar{k_1}}(x^*)$ to get id to find out the malicious user. Otherwise, the algorithm outputs $\top$.

B. Selective Security Proof

In the selective security proof, the same with the selective security proof of the T-LU-CPABE system, we will reduce the selective security of our eT-LU-CPABE system to that of Rouselakis and Waters's system in [34] which is proved selectively secure under Assumption 1. Some of the text below is taken verbatim from the selective security proof of the T-LU-CPABE system.

For simplicity, we denote by $\Sigma_{lucpabe}$, $\Sigma_{etlucpabe}$ the LU-CPABE system in [34] and our system respectively. Note that the security model of our system $\Sigma_{etlucpabe}$ is almost same with that of the system $\Sigma_{lucpabe}$ in [34], excepting every key query is companied with an explicit identity.

Selective security of our new eT-LU-CPABE system:

Lemma 3: If the LU-CPABE system $\Sigma_{lucpabe}$ is selectively secure in the game of [34], then our new eT-LU-CPABE system $\Sigma_{etlucpabe}$ is selectively secure in the game of Subsection III-B.

Proof: Suppose there exists a PPT adversary $\mathcal{A}$ with a challenge matrix M^* that has advantage $Adv_{\mathcal{A}}\Sigma_{etlucpabe}$ in selectively breaking our eT-LU-CPABE system $\Sigma_{etlucpabe}$, where M^* is an $l \times n$ matrix satisfies the restriction that $l, n \leq q$. We construct a PPT algorithm $\mathcal{B}$ that has advantage $Adv_{\mathcal{B}}\Sigma_{lucpabe}$ in selectively breaking the underlying

[6] In our system, it requires $\bar{n}$ is greater than the number of the total users.

[7] If all of the users register and get the secret keys at the beginning of system initialization, the system could secretly store $f(0)$ instead of the polynomial $f(x)$ since the storage for $f(x)$ is much larger than that of $f(0)$.

[8] Due to the definition of probabilistic encryption, x is not distinguished from a random number. In addition, f is linear function and thus y is also a random number. Therefore, c, combined with x and y and through a probabilistic encryption, can also be a random number.

LU-CPABE system $\Sigma_{lucpabe}$, which equals to $Adv_{\mathcal{A}}\Sigma_{etlucpabe}$.

- **Initialization**: $\mathcal{B}$ gets a challenge policy (M^*, ρ^*) from $\mathcal{A}$ and then sends this received challenge policy to $\Sigma_{lucpabe}$. Note that M^* is an $l \times n$ matrix, where $l, n \leq q$, and $\rho^* \in \mathcal{F}([l] \to \mathbb{Z}_p)$.
- **Setup**: $\Sigma_{lucpabe}$ gives $\mathcal{B}$ the public parameter $pp_{\Sigma_{lucpabe}} = (D, g, w, v, u, h, e(g,g)^{\alpha})$, which is the same with that of the selective security proof of our T-LU-CPABE system. Then $\mathcal{B}$ randomly chooses $a \in \mathbb{Z}_p$, adds g^a to $pp_{\Sigma_{lucpabe}}$ as a new public parameter pp and gives the new pp to $\mathcal{A}$. Finally, it initializes an instance of Shamir's $(\bar{t}, \bar{n})$ threshold scheme and keeps $f(x)$ and $\bar{t}-1$ points $\{(x_1, y_1), (x_2, y_2), \ldots, (x_{\bar{t}-1}, y_{\bar{t}-1})\}$ secret.
- **Query Phase 1**: The adversary $\mathcal{A}$ will submit (id, S) to $\mathcal{B}$ to query a decryption key, then $\mathcal{B}$ submits S to $\Sigma_{lucpabe}$ and gets the corresponding decryption key $\langle \hat{K}_0, \hat{K}_1, \{\hat{K_{\tau,2}}, \hat{K_{\tau,3}}\}_{\tau \in [|S|]} \rangle$, which is the same with that of the selective security proof of our T-LU-CPABE system. $\mathcal{B}$ computes $x = Enc_{\bar{k_1}}(id)$, $y = f(x), c = Enc_{\bar{k_2}}(x||y)$ $(c \in \mathbb{Z}_p^*)$ and $1/(a+c)$ modulo p. Then $\mathcal{B}$ sets $r = \tilde{r}/(a+c)$ and $K' = c$ implicitly and randomly chooses $g' \in \mathbb{G}$, then computes $K, L, L', \{K_{\tau,1}, K_{\tau,2}\}_{\tau \in [|S|]}$, which adopts the same alogrithm with that of the selective security proof of our T-LU-CPABE system. $\mathcal{B}$ sends the decryption key $sk_{id,S} = \langle K, K', L, L', \{K_{\tau,1}, K_{\tau,2}\}_{\tau \in [|S|]} \rangle$ to $\mathcal{A}$. Note that g' makes L' uncorrelated to L.
- **Challenge**: The attacker $\mathcal{A}$ outputs two equal length messages (m_0, m_1) and sends them to $\mathcal{B}$. Then $\mathcal{B}$ submits (m_0, m_1) to $\Sigma_{lucpabe}$, and gets the challenge ciphertext $\langle (M^*, \rho^*), \hat{C}, \hat{C}_0, \{\hat{C_{i,1}}, \hat{C_{i,2}}, \hat{C_{i,3}}\}_{i \in [l]} \rangle$, which is the same with that of the selective security proof of our T-LU-CPABE system. Then $\mathcal{B}$ sets $C = \hat{C}, C_0 = \hat{C}_0$, $C_0' = (\hat{C}_0)^a = g^{as}, C_{i,1} = \hat{C}_{i,1}, C_{i,2} = \hat{C}_{i,2}, C_{i,3} = \hat{C}_{i,3}$. Finally, $\mathcal{B}$ gives the challenge ciphertext $ct = \langle (M^*, \rho^*), C, C_0, C_0', \{C_{i,1}, C_{i,2}, C_{i,3}\}_{i \in [l]} \rangle$ to $\mathcal{A}$.
- **Query Phase 2**: This phase is the same with Phase 1.
- **Guess**: The attacker outputs his guess β', and gives it to $\mathcal{B}$. Then $\mathcal{B}$ gives β' to $\Sigma_{lucpabe}$.

Since the distributions of the public parameters, decryption keys and challenge ciphertext in the above game are the same as that in the real system, we have $Adv_{\mathcal{B}}\Sigma_{lucpabe} = Adv_{\mathcal{A}}\Sigma_{etlucpabe}$. □

Theorem 4: If Assumption 1 holds, then our eT-LU-CPABE system is selectively secure.

Proof: It follows directly from Lemma 1 and Lemma 3. □

C. Traceability Proof

In this subsection, we will give the traceability proof of our eT-LU-CPABE system based on l-SDH assumption. Almost the same with the traceability proof of the T-LU-CP-ABE system, we will use a proof method from [5] and [26].

Theorem 5: If the l-SDH assumption holds, then our eT-LU-CP-ABE system is fully traceable provided that $q < l$.

TABLE I
FEATURES COMPARISON WITH OTHER RELATED WORK

	[24]	[23]	[26]	[35]	1 [5]	2 [6]
Large Universe [1]	×	×	×	√	√	√
Traceability	√	√	√	×	√	√
Constant Storage for Tracing [2]	√	√	×	−	×	√
Supporting Any Monotone Access Structures [3]	×	×	√	√	√	√
Constructed on Prime Order Groups [4]	√	×	×	√	√	√
Standard Model	×	√	√	√	√	√

[1] In [24],[23] and [26], their systems only support small universe.
[2] In [26], the storage for tracing is not constant. In [35], the proposed system does not support traceability.
[3] In [24] and [23], their systems do not support any monotone access structures.
[4] In [23] and [26], their systems are constructed on the composite order groups.
[5] 1 stands for our new T-LU-CPABE system.
[6] 2 stands for our new eT-LU-CPABE system.

TABLE II
PERFORMANCE COMPARISON WITH OTHER RELATED WORK[1,2]

	PubKS	PriKS	CS	PCD
[24]	$\|u\|+4$	$\|S\|+3$	$l+3$	2
[23]	$3\|u\|+3\rho+5$	$4\|S\|+3\rho$	$2l+4\rho+2$	$3\|I\|+3\rho$
[26]	$\|u\|+4$	$\|S\|+4$	$2l+3$	$2\|I\|+1$
[35]	7	$2\|S\|+2$	$3l+2$	$3\|I\|+1$
1 [3]	7	$2\|S\|+4$	$3l+3$	$3\|I\|+1$
2 [4]	7	$2\|S\|+4$	$3l+3$	$3\|I\|+1$

[1] PubKS stands for Public Key Size, PriKS stands for Private Key Size, CS stands for Ciphertext Size, PCD stands for Pairing Computation in Decryption, 1 stands for our new T-LU-CPABE system, and 2 stands for our new eT-LU-CPABE system.
[2] Let u be the size of the attribute universe, $|S|$ the size of the attribute set of a private key, l the size of an access policy, $|I|$ the number of attributes in a decryption key that satisfies a ciphertext's access policy, and ρ the bit length of the global identity in [23].
[3] 1 stands for our new T-LU-CPABE system.
[4] 2 stands for our new eT-LU-CPABE system.

Proof: The proof of this theorem is almost the same with that of the *Theorem 2* in the T-LU-CPABE system, we refer the interested readers to the proof of *Theorem 2* for the proof of this theorem. □

D. Key Sanity Check Proof

In this subsection, we will give the key sanity check proof of our eT-LU-CPABE system.

Theorem 6: The advantage of an adversary in the key sanity check game (in Subsection II-D) is negligible for our eT-LU-CPABE system.

Proof: The proof of this theorem is almost the same with that of the *Theorem 3* in the T-LU-CPABE system, we refer the interested readers to the proof of *Theorem 3*. □

VII. COMPARISON

Table I and Table II give the comparison between our work and several related work in terms of features (i.e. Large Universe, Traceability, etc.) and performance. In particular, our new T-LU-CPABE system and eT-LU-CPABE system feature a significantly shorter constant

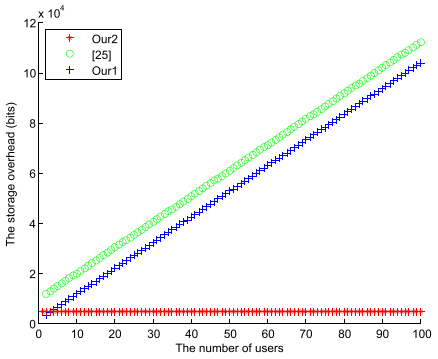

Fig. 1. System storage overhead for traitor tracing.[9]

set of public parameters than that of [23], [24], and [26], though the private key size, the ciphertext size and the pairing computation in decryption of our systems is a slightly longer or the same with that of [23], [24], and [26]. This makes our new systems more practical for applications. Compared with [34], our new systems only sacrifice tiny size of the private key and the ciphertext to achieve white-box traceability.

Fig. 1 illustrates that the system storage overhead for traitor tracing (including the public parameters, the master secret key and the storage for tracing) in [26] and our T-LU-CPABE system grows linearly with the number of the users. While our new eT-LU-CPABE system only needs to store $\overline{t}-1$ points and the value $f(0)$ on a polynomial $f(x)$, which is constant and only depends on the threshold value $\overline{t}$. Particularly, our new eT-LU-CPABE system need not to maintain an identity table for listing all the users' identities and pseudonyms when new users add into the system or malicious users are removed from the system. This makes our new eT-LU-CPABE system more suitable for practical commercial applications.

VIII. Extensions

A. Transform From One-Use (e)T-LU-CPABE System to Multi-Use (e)T-LU-CPABE System

The constructions[9] of our T-LU-CPABE system and eT-LU-CPABE system are both *one-use* CP-ABE constructions. Since the ρ in our systems is an injective function for each access policy associated to a ciphertext. During the row label of the share-generating matrix, the attributes are only used once. This kind of construction is called one-use CP-ABE construction.

We can extend our new T-LU-CPABE system and eT-LU-CPABE system to a *multi-use* system using the encoding technique in [18]: we take k copies of each attribute A instead of a single attribute. Then we have new "attributes": $\{A:1,\ldots,A:k\}$. Now we can label a row of the access matrix $\mathbb{A}$ with $\{A:i\}$. Thus the attribute can be used multiple times. Note that the size of the public parameters do not grow linearly with the number of the involved attributes, so that the size of the public parameters will remain the same size under this transformation. Besides the access matrix's size does not change under this transformation either, thus the size of the ciphertext also remains the same size. This makes our T-LU-CPABE system and eT-LU-CPABE system more suitable for commercial applications.

[9]We assume that the size of the attribute universe in [26] is 50, the length of a group element g is 160 bits, the length of the random number c in [26] and our T-LU-CPABE system is 1024 bits, and the threshold value $\overline{t}$ is 10. Our1 stands for our new T-LU-CPABE system and Our2 stands for our new eT-LU-CPABE system.

B. Revocable T-LU-CPABE System and eT-LU-CPABE System

Through our new T-LU-CPABE system and eT-LU-CPABE system proposed in this paper, it is easy to trace the malicious user who leak his/her decryption key for benefits. This evokes another significant issue to be considered: how to revoke the malicious users. Several work has focused on designing revocable ABE [33], [35]. With the technology of ciphertext delegation and piecewise key generation introduced in [35], we can achieve revocable T-LU-CPABE and eT-LU-CPABE constructions. Furthermore, since we make use of the Shamir's $(\bar{t},\bar{n})$ threshold scheme in the **`Trace`** algorithm, the eT-LU-CPABE system only need store $\bar{t}-1$ tuples for tracing, rather than an identify table T which contains all users' identifies. This brings an obvious advantage that the eT-LU-CPABE system need not update the identify table T when some users are revoked.

IX. Discussion

A. Removing the Identity Table Without Adding Additional Parameters

In the construction of our T-LU-CPABE system, the system adopts an identity table T to record the identities of the users who have obtained the decryption keys from the system. This gives rise to a corresponding consequence that the system has to maintain an identity table T. And the table T would grow linearly with the number of the users. Moreover, the table T will expand sharply in a large scale system, which causes a heavy burden for the storage space of T. Also, with the identity table T expanding, the corresponding cost for traceability (i.e. searching K' in T) would become higher. So it is a trend to eliminate the identity table T in order to obtain an efficient and practical construction.

As a result, in our enhanced construction (i.e. the eT-LU-CPABE system), we employ Shamir's $(\bar{t},\bar{n})$ threshold scheme [37] and probabilistic encryption scheme (Enc, Dec) [10] to replace the role of the identity table T. But actually, we need to introduce some additional parameters (such as the $f(x)$ and $\overline{t}-1$ points for Shamir's $(\bar{t},\bar{n})$ threshold scheme and $\bar{k_1},\bar{k_2}$ for probabilistic encryption scheme (Enc, Dec)) to achieve the elimination of the identity table T. It means that the system has to store these additional parameters, that is, we reach our goal at a low cost of some additional storage space. Obviously, it would be better to take advantage of the parameters from the original system itself (such as the group elements, the random secret parameter which has already been chosen and used) to achieve the goal of eliminating the table T. Hence, how to remove the identity

table T without introduce additional parameters still remains an interesting open problem for future work.

B. Public Traceability

In this paper, we propose two traceable large universe CP-ABE systems: the T-LU-CPABE system and the eT-LU-CPABE system. The T-LU-CPABE system adopts the identity table T to achieve the property of white-box traceability. And the eT-LU-CPABE system, which does not depend on the identity table T, employs the Shamir's $(\bar{t}, \bar{n})$ threshold scheme [37] and probabilistic encryption scheme (Enc, Dec) [10] to achieve the property of white-box traceability. Note that, both the T-LU-CPABE system and the eT-LU-CPABE system are *private traceable*, that is, only the system can run the tracing algorithm to identify the malicious users or traitors who intentionally leak their partial or modified decryption keys to others for profits. Here, combined existing work [25], we put forward a new stronger notion named *partial public traceability*: besides the system, someone else can run the tracing algorithm to trace the malicious users or traitors with no additional secret, as long as he/she is authorized. Furthermore, we put forward the strongest notion named *fully public traceability*[10] : anyone can run the tracing algorithm to identity the malicious users or traitors with no additional secret. How to obtain (e)T-LU-CPABE system with partial public traceability and (e)T-LU-CPABE system with fully public traceability remains an interesting problem.

X. Conclusion and Future Work

In this work, we have presented two practical large universe CP-ABE systems supporting white-box traceability. Specifically, we have achieved the property of white-box traceability in CP-ABE, which could trace the malicious users leaking the partial or modified decryption keys to others for profits. We have also obtained the property of large universe in white-box traceable CP-ABE where the attributes' size is unbounded and the public parameters' size does not grow linearly with the number of attributes. In addition, we optimize the system in tracing the malicious users to cut down the storage cost for traceability and to make the system efficient in the revocation of the users. Based on the above advantages, our new systems could be applied to many scenarios such as pay-TV systems and social networks. As far as we known, these are the first practical CP-ABE systems that simultaneously supports white-box traceability and large universe. We have also proved our new systems selectively secure in the standard model.

In our future work, we will focus on the stronger notion for traceability named black-box traceability. In that scenario, the malicious users leak their decryption devices instead of decryption keys. Specifically, the malicious users could hide the decryption algorithm by tweaking it, as well as the decryption keys. In this case, due to the fact that the decryption keys and decryption algorithm are both not well-formed, the new system supporting white-box traceability in this paper will fail. It will be our future work to obtain a large universe CP-ABE system, which supports black-box traceability.

There is another important issue about public auditing we need to pay attention to. Suppose a user Bob is identified as a malicious user by the system, but claims to be innocent and framed by the system. It is a big problem to judge whether Bob is in fact innocent or not. In this case, the suspected user does not trust the system and the system needs to provide some evidence persuasive enough to prove that the suspected user is guilty. To address this issue, a public auditor which is played by a trusted third party needs to be introduced. However, the suspected user does not want the public auditor to know the private information since in the **`Trace`** phase the auditor will obtain Bob's decryption keys and be able to decrypt all the data that Bob has. Achieving a traceable large universe CP-ABE system with public auditors is still an open problem, and we will keep working on it.

Also, according to discussions in Section IX, there are more work in the future: (1) how to remove the identity table T of our T-LU-CPABE system without adding additional parameters; and (2) how to realize partial public traceability and fully public traceability for our T-LU-CPABE system and eT-LU-CPABE system.

[10]The meaning of this notion is identical to that of which named *public traceability* appeared in related work (such as [25]).

Acknowledgements

The authors are grateful to the anonymous reviewers for their invaluable suggestions.

References

[1] N. Attrapadung, B. Libert, and E. de Panafieu, "Expressive key-policy attribute-based encryption with constant-size ciphertexts," in *Public Key Cryptography*. Berlin, Germany: Springer-Verlag, 2011, pp. 90–108.

[2] M. H. Au, Q. Huang, J. K. Liu, W. Susilo, D. S. Wong, and G. Yang, "Traceable and retrievable identity-based encryption," in *Proc. 6th Int. Conf. Appl. Cryptogr. Netw. Secur. (ACNS)*, New York, NY, USA, Jun. 2008, pp. 94–110.

[3] A. Beimel, "Secure schemes for secret sharing and key distribution," M.S. thesis, Technion-Israel Inst. Technol., Haifa, Israel, 1996.

[4] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2007, pp. 321–334.

[5] D. Boneh and X. Boyen, "Short signatures without random oracles," in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 3027, C. Cachin and J. L. Camenisch, Eds. Berlin, Germany: Springer-Verlag, 2004, pp. 56–73.

[6] M. Chase, "Multi-authority attribute based encryption," in *Theory of Cryptography*. Berlin, Germany: Springer-Verlag, 2007, pp. 515–534.

[7] M. Chase and S. S. M. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *Proc. 16th ACM Conf. Comput. Commun. Secur.*, 2009, pp. 121–130.

[8] L. Cheung and C. Newport, "Provably secure ciphertext policy ABE," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, 2007, pp. 456–465.

[9] T. El Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 1985, pp. 10–18.

[10] S. Goldwasser and S. Micali, "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984.

[11] V. Goyal, "Reducing trust in the PKG in identity based cryptosystems," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2007, pp. 430–447.

[12] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute based encryption," in *Automata, Languages and Programming*. Berlin, Germany: Springer-Verlag, 2008, pp. 579–591.

[13] V. Goyal, S. Lu, A. Sahai, and B. Waters, "Black-box accountable authority identity-based encryption," in *Proc. 15th ACM Conf. Comput. Commun. Secur.*, 2008, pp. 427–436.

[14] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. 13th ACM Conf. Comput. Commun. Secur.*, 2006, pp. 89–98.
[15] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of ABE ciphertexts," in *Proc. 20th USENIX Secur. Symp.*, 2011, p. 34.
[16] S. Hohenberger and B. Waters, "Attribute-based encryption with fast decryption," in *Public-Key Cryptography*. Berlin, Germany: Springer-Verlag, 2013, pp. 162–179.
[17] A. Lewko, "Tools for simulating features of composite order bilinear groups in the prime order setting," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2012, pp. 318–335.
[18] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2010, pp. 62–91.
[19] A. Lewko and B. Waters, "New techniques for dual system encryption and fully secure HIBE with short ciphertexts," in *Theory of Cryptography*. Berlin, Germany: Springer-Verlag, 2010, pp. 455–479.
[20] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2011, pp. 568–588.
[21] A. Lewko and B. Waters, "Unbounded HIBE and attribute-based encryption," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2011, pp. 547–567.
[22] A. Lewko and B. Waters, "New proof methods for attribute-based encryption: Achieving full security through selective techniques," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2012, pp. 180–198.
[23] J. Li, Q. Huang, X. Chen, S. S. M. Chow, D. S. Wong, and D. Xie, "Multi-authority ciphertext-policy attribute-based encryption with accountability," in *Proc. 6th ACM Symp. Inf., Comput., Commun. Secur.*, 2011, pp. 386–390.
[24] J. Li, K. Ren, and K. Kim, "A^2BE: Accountable attribute-based encryption for abuse free access control," IACR Cryptology ePrint Archive, 2009:118, 2009.
[25] Z. Liu, Z. Cao, and D. S. Wong, "Blackbox traceable CP-ABE: How to catch people leaking their keys by selling decryption devices on ebay," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2013, pp. 475–486.
[26] Z. Liu, Z. Cao, and D. S. Wong, "White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 1, pp. 76–88, Jan. 2013.
[27] T. Okamoto and K. Takashima, "Homomorphic encryption and signatures from vector decomposition," in *Pairing-Based Cryptography*. Berlin, Germany: Springer-Verlag, 2008, pp. 57–74.
[28] T. Okamoto and K. Takashima, "Hierarchical predicate encryption for inner-products," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2009, pp. 214–231.
[29] T. Okamoto and K. Takashima, "Fully secure functional encryption with general relations from the decisional linear assumption," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2010, pp. 191–208.
[30] R. Ostrovsky, A. Sahai, and B. Waters, "Attribute-based encryption with non-monotonic access structures," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, 2007, pp. 195–203.
[31] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 1999, pp. 223–238.
[32] B. Parno, M. Raykova, and V. Vaikuntanathan, "How to delegate and verify in public: Verifiable computation from attribute-based encryption," in *Theory of Cryptography*. Berlin, Germany: Springer-Verlag, 2012, pp. 422–439.
[33] J.-L. Qian and X.-L. Dong, "Fully secure revocable attribute-based encryption," *J. Shanghai Jiaotong Univ. (Sci.)*, vol. 16, no. 4, pp. 490–496, 2011.
[34] Y. Rouselakis and B. Waters, "Practical constructions and new proof methods for large universe attribute-based encryption," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2013, pp. 463–474.
[35] A. Sahai, H. Seyalioglu, and B. Waters, "Dynamic credentials and ciphertext delegation for attribute-based encryption," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2012, pp. 199–217.
[36] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2005, pp. 457–473.
[37] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.
[38] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Public Key Cryptography*. Berlin, Germany: Springer-Verlag, 2011, pp. 53–70.

Jianting Ning received the B.S. and M.S. degrees from Yangzhou University, in 2010 and 2013, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Shanghai Jiao Tong University.

His research interests include cryptography and information security, in particular, public key encryption, attribute-based encryption, noninteractive proof systems for bilinear groups, and zero-knowledge proof.

Xiaolei Dong received the Doctorate degree from the Harbin Institute of Technology in 2001. She pursued her post-doctoral study at Shanghai Jiao Tong University (SJTU) from 2001 to 2003. She joined the Department of Computer Science and Engineering, SJTU, in 2003. In 2014, she joined East China Normal University. She is currently a Distinguished Professor with East China Normal University. Her primary research interests include number theory, cryptography, and trusted computing. Her Number Theory and Modern Cryptographic Algorithms project received the first prize of the China University Science and Technology Award in 2002. Her New Theory of Cryptography and Some Basic Problems project received the second prize of the Shanghai Nature Science Award in 2007. Her Formal Security Theory of Complex Cryptographic System and Applications received the second prize of the Ministry of Education Natural Science Progress Award in 2008. She hosts a number of research projects supported by the National Basic Research Program of China (973 Program), the special funds on information security of the National Development and Reform Commission, and the National Natural Science Foundation of China.

Zhenfu Cao (SM'10) received the B.Sc. degree in computer science and technology and the Ph.D. degree in mathematics from the Harbin Institute of Technology, Harbin, China, in 1983 and 1999, respectively. He was exceptionally promoted to Associate Professor in 1987, became a Professor in 1991, and is currently a Distinguished Professor with East China Normal University, China. He has authored over 400 academic papers in journals or conferences since 1981. His research interests mainly include number theory, cryptography, and information security. He serves as a member of the Expert Panel of the National Nature Science Fund of China. He has received a number of awards, including the Youth Research Fund Award of the Chinese Academy of Science in 1986, the Ying-Tung Fok Young Teacher Award in 1989, the National Outstanding Youth Fund of China in 2002, and the Special Allowance by the State Council in 2005, and was a corecipient of the IEEE International Conference on Communications—Computer and Communications Security Symposium Best Paper Award in 2007. He is the Leader of the Asia 3 Foresight Program (61161140320) and the key project (61033014) of the National Natural Science Foundation of China.

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

Xiaodong Lin received the Ph.D. degree (awarded Outstanding Achievement in Graduate Studies) in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2008. He is currently an Associate Professor of Information Security with the University of Ontario Institute of Technology, Canada. His research interests are in the areas of information security, privacy enhancing technologies, digital forensics, and applied cryptography.