A Ciphertext-Policy Attribute-based Encryption Scheme With Optimized Ciphertext Size And Fast Decryption

Qutaibah M. Malluhi
Qatar University
Doha, Qatar
qmalluhi@qu.edu.qa

Abdullatif Shikfa
Qatar University
Doha, Qatar
ashikfa@qu.edu.qa

Viet Cuong Trinh
Qatar University and Hong Duc University
Thanh Hoa, Viet Nam
trinhvietcuong@hdu.edu.vn

ABSTRACT

We address the problem of ciphertext-policy attribute-based encryption with fine access control, a cryptographic primitive which has many concrete application scenarios such as Pay-TV, e-Health, Cloud Storage and so on. In this context we improve on previous LSSS based techniques by building on previous work of Hohenberger and Waters at PKC'13 and proposing a construction that achi- eves ciphertext size linear in the minimum between the size of the boolean access formula and the number of its clauses. Our construction also supports fast decryption. We also propose two interesting extensions: the first one aims at reducing storage and computation at the user side and is useful in the context of lightweight devices or devices using a cloud operator. The second proposes the use of multiple authorities to mitigate key escrow by the authority.

Keywords

Ciphertext-policy attribute-based encryption, DNF access policy, LSSS access policy, Fast decryption, multi-authority.

1. INTRODUCTION

In the "era of modern cryptography", cryptographic schemes become more and more complex to satisfy the needs of modern applications. Regarding data encryption, many new scenarios require advanced capabilities and flexible ways to do it beyond simple semantically secure encryption with a key. For instance a desirable property is the ability to encrypt a message according to a specific policy. In such scenario, only receivers who possess enough attributes satisfying this specific policy can decrypt the encrypted message.

We consider the following practical scenario, in a faculty of computer science, there are faculty members and adminsitrative staff and there are three international research groups: Crypto, Wireless Communications, and Image processing. In the Crypto group we have two projects: Garbled computer and Security for IoT-based Applications. In the Wireless Communications group we also have two projects: Fog Computing and Internet of Things. In such system, the attributes are: Faculty members, Administrative staff, Crypto, Wireless Communications, Image processing, Garbled computer, Security for IoT-based Applications, Fog computing, and Internet of Things. When the dean of the faculty wants to put a document in the cloud server for the faculty members in the Crypto group who are working in the Garbled computer project, as well as faculty members in the Wireless Communications group who are working in the Fog Computing project, the access policy should be:

(Faculty member and Crypto and Garbled computer) or (Faculty member and Wireless Communications and Fog Computing).

In this access policy, the size of the boolean formula is six (counting the reused attribute Faculty member) while the number of clauses in DNF form is only two. Recently, this type of encryption is applied to more and more contexts such as pay-TV system, e-Health, or internet of things.

Addressing this problem, Sahai and Waters [25] introduced the concept of *attribute-based encryption* (ABE) where both the encryption and decryption steps are performed by using the user's attributes. Recent researches have investigated two variants of ABE: the first one is named *ciphertext-policy* attribute-based encryption (CP-ABE) and the second one is named *key-policy* attribute-based encryption (KP-ABE). In a CP-ABE scheme, the secret key is associated with a set of attributes and the ciphertext is associated with an access policy (structure) over a universe of attributes: a user can then decrypt a given ciphertext if the set of attributes related to his/her secret key satisfies the access policy underlying the ciphertext. In contrast, in a KP-ABE scheme, the access policy is for the secret key and the set of attributes is for the ciphertext.

In the context of ABE, the set of privileged users is determined by the access policy. To date, several types of access policy have been investigated. Two limited ones are AND-gates and threshold. In [11, 9], the access structure is constructed by AND-gates on multi-valued attributes. In [17, 13, 8], the access policy is *threshold*, meaning that there is no distinction among attributes in the access policy: anyone who possesses enough attributes (equal or bigger than a threshold chosen by the sender) will be able to decrypt. In some modern applications, finer-grained access control is needed such as boolean formula, and the common technique so far to construct such access control is based on LSSS matrix [26] (LSSS-based scheme - so called). In LSSS-based schemes, the ciphertext size is usually linear in the size of the access boolean formula. We note that, in several specific types of boolean formula especially the DNF form (i.e., with disjunctions (OR) of conjunctions (AND) of attributes), the

ASIA CCS '17, April 02-06, 2017, Abu Dhabi, United Arab Emirates

DOI: http://dx.doi.org/10.1145/3052973.3052987

size of the boolean formula could be much bigger than the number of clauses (as we showed in the above example). In contrast, in CNF form (i.e., with conjunctions (AND) of disjunctions (OR) of attributes) the size of the boolean formula could be much smaller than the number of clauses in the corresponding DNF form. This leads us to research a construction where the ciphertext size could be linear in either the size of the boolean formula or the number of clauses depending on which is smaller between them.

In some practical contexts such as mobile pay-TV system or internet of things, the power of user's devices are restricted which means that the decrypting complexity and the user's storage are essential. In addition, in those contexts a CP-ABE scheme is more suitable than a KP-ABE scheme as shown in [19], therefore designing a CP-ABE scheme with fast decryption, limited user's storage, and supporting fine-grained access control such as boolean formula is desirable. Another problem that we were keen on adressing is that CP-ABE, as some other cryptographic primitives, suffers from the key escrow problem since the authority knows all private keys of users. We thus researched the possibility to implement a multi-authority ABE scheme to mitigate this risk.

1.1 Related Work

Attribute-base encryption has been deeply researched in recent years with a lot of proposed papers [25, 16, 22, 11, 19, 17, 26, 3, 21, 18, 23, 8, 28, 27, 10, 12, 14, 6], to name a few. Regarding the ABE scheme supporting the fine-grained access control, the first construction [16] is based on the tree structure where the authors extended the Sahai and Waters's work [25] to propose the schemes supporting fine-grained access control, specified by a boolean formula. Subsequent works in this direction such as [26, 18, 23, 8, 28, 27, 10], are constructed based on the linear secret sharing scheme (LSSS). The shortcoming of this technique is that the ciphertext size is usually linear in the size of the access boolean formula which is not as small as expected especially when the access boolean formula is in DNF form. Very recently, in [1] and [2] the authors based on LSSS technique proposed two CP-ABE schemes with constant-size ciphertext and supporting the fine-grained access control. However, the public key size and secret key size of those schemes are very impractical. Multi-authority ABE scheme supporting fine-grained access control have been investigated in [20, 24] where each authority takes responsibility for a disjoint set of attributes. Regarding the fast decryption, only one of them [18] proposes an efficient CP-ABE scheme with fast decryption and supports LSSS access policy. Some other schemes can achieve the fast decryption property but either they are in *key-policy* attribute-based style [18, 3, 16] or they supports limited access policy such as threshold or AND-gates [11, 17].

1.2 Our Contribution and Organization of the Paper

In this paper, we propose a CP-ABE scheme supporting fine-grained access control and achieving highly desirable properties:

- small ciphertext size: the ciphertext-size of our scheme is linear in either the size of the access boolean formula or the the number of clauses depending on which is smaller between them;
- fast decryption: in all cases, our scheme just needs two Pairings for decryption;
- mitigation of key escrow: our scheme can be extended to support non colluding multiple authorities that cannot decrypt messages;
- minimization of user storage: our scheme can make a large part of the encryption and decryption material public, thus saving storage space and enabling delegation of part of the computation to an outsourced server.

More precisely, we extend the scheme [18] (section 3.5) to allow the encryptor to choose the most efficient of two encryption algorithms at the time of encryption. Starting with an access boolean formula, the encryptor first describes this formula in the DNF form (i.e., with disjunctions (OR) of conjunctions (AND) of attributes). She then compares between the number of clauses in the DNF form and the size of the original boolean formula (the number of attributes in the access boolean formula - counting also the reused attributes). Finally, she will produce the ciphertext depending on which is smaller between them.

We emphasize that our scheme still takes advantage of LSSS technique (the ciphertext size is linear in the size of the original boolean formula), while it overcomes the weakness of LSSS technique when the size of the access boolean formula is bigger than the number of clauses. It is therefore fair to say that our scheme is an improvement of the scheme [18] (section 3.5).

Note that, it seems not difficult to transform some AND-gates schemes such as [11, 9] to achieve a new scheme supporting DNF access policy where the ciphertext-size is linear in the number of clauses. However, since such schemes do not take advantage of LSSS technique, it is not efficient in case the size of the access boolean formula is smaller than the number of clauses, for example when the access boolean formula is in CNF form. The reason why our scheme can take advantage of LSSS technique while others cannot is that our scheme shares a similar key structure with the scheme [18] (section 3.5).

Regarding decryption efficiency, the new decryption algorithm of our scheme just needs to compute two Pairings and $|I|$ multiplications, where $|I|$ is the number of attributes for a decryption key to satisfy a ciphertext access policy.

Furthermore, in our scheme the user just needs to keep one element secret, the other elements in the user's secret key can be made public, it is thus very suitable to the context of lightweight cryptography as it is sufficient for a user to store only one secret element in the smart card. It can also be useful in the context of outsourced data storage and computation, such as in cloud computing, since the user can store part of the key in the cloud and even have the cloud perform some steps of the encryption and decryption algorithms.

Our scheme can also be extended to support multiple non-colluding authorities by using the splitting technique, where each user constructs her secret key with the help of θ authorities in the system. The collusion of up to $\theta - 1$ curious authorities is not enough to compute the secret key of user. While this solution is not difficult to implement, to our knowledge we haven't seen any previous CP-ABE work mentioning it.

The paper is now organized as follows. The next section introduces preliminary security definitions and mathemati-

cal building blocks. In Section 3.1, we introduce our scheme and prove that it achieves selective security in the following section. In section 4 we analyze the performance of our scheme compared to the literature. Finally, in Section 5, we discuss how to minimize the user's storage as well as support of multiple authorities in our scheme.

2. PRELIMINARIES

We recall in this section several definitions and notions that are needed for our construction. We first define the security model of CP-ABE scheme, followed by access structures, bilinear maps and related security assumptions and finally LSSS matrices.

2.1 Ciphertext-Policy Attribute-Based Encryption

Formally, a CP-ABE scheme consists of four probabilistic algorithms.

Setup$(1^\lambda, \mathcal{B})$**:** The setup algorithm takes the security parameter λ and the description of the attributes' universe $\mathcal{B}$ as inputs. It generates the master key MSK, as well as the public parameters param of the system.

Extract$(u, \mathcal{B}(u), \mathsf{MSK}, \mathsf{param})$**:** Takes as input a user u and his set of attributes $\mathcal{B}(u)$, as well as the public parameters param and the master key MSK. It outputs the user's private key d_u.

Encrypt$(\mathcal{M}, \mathbb{A}, \mathsf{param})$**:** Takes as input a message $\mathcal{M}$, an access policy $\mathbb{A}$ over the universe of attributes and the public parameters param. It outputs the ciphertext ct along with a description of the access policy $\mathbb{A}$.

Decrypt$(ct, d_u, \mathsf{param})$**:** Takes as input the ciphertext ct, the private key d_u of user u, together with the parameters param. It outputs the message $\mathcal{M}$ if and only if $\mathcal{B}(u)$ satisfies $\mathbb{A}$. Otherwise, it outputs $\perp$.

Security Model:.

We now recall the security model for a CP-ABE scheme [26]. The security model consists of the following probabilistic game between an attacker $\mathcal{A}$ and a challenger $\mathcal{C}$.

Setup$(1^\lambda, \mathcal{B})$**.** The challenger runs the **Setup**$(1^\lambda, \mathcal{B})$ algorithm to generate the public parameters param of the system, as well as a master key MSK. The corruption list Λ_C is set to the empty list (the corruption list corresponds to the queries of the adversary as will be described in the next steps).

Query phase 1. The adversary $\mathcal{A}$ chooses a set of attributes $\mathcal{B}(u)$ and asks corruption query corresponding to this set of attributes: the challenger runs **Extract**$(u, \mathcal{B}(u),$ MSK, param) and forwards the resulting private key to the adversary. The user u is appended to the corruption list Λ_C.

Challenge. The adversary $\mathcal{A}$ outputs a target access policy $\mathbb{A}^*$ and two equal length message $\mathcal{M}_0^*, \mathcal{M}_1^*$. Next, the challenger picks a random $b \xleftarrow{\$} \{0,1\}$ and runs **Encrypt**$(\mathcal{M}_b^*, \mathbb{A}^*, \mathsf{param})$ to obtain ct^*. Finally the challenger outputs ct^*.

Query phase 2. The adversary $\mathcal{A}$ continues to adaptively ask queries as in the first phase.

Guess. The adversary $\mathcal{A}$ eventually outputs its guess $b' \in \{0,1\}$ for b.

We say the adversary wins the game if $b' = b$, and if $\mathcal{B}(u)$ does not satisfy $\mathbb{A}^*$ for all $u \in \Lambda_C$ (the corruption list). We then denote the advantage of adversary to win the game by

$$\mathbf{Adv}_{\mathcal{A}} = \Pr\left[b = b'\right] - \frac{1}{2}.$$

Definition 1 *A ciphertext-policy attribute-based encryption scheme is secure if all polynomial time adversaries have at most a negligible advantage in the above game.*

There is also a classical restricted scenario: a *selective* attacker provides the target access policy $\mathbb{A}^*$ at the beginning of the security game.

Definition 2 (Selective Security) *A* CP-ABE *scheme is said to be selectively secure if it is secure against a selective adversary in the above security game.*

2.2 Access Structures

Definition 3 (Access Structures) *Let* $\{Att_1, Att_2, \ldots, Att_n\}$ *be a set of attributes. A collection* $\mathbb{A} \subseteq 2^{\{Att_1, Att_2, \ldots, Att_n\}}$ *is monotone if* $\forall B, C$ *: if* $B \in \mathbb{A}$ *and* $B \subseteq C$ *then* $C \in \mathbb{A}$*. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection)* $\mathbb{A}$ *of non-empty subsets of* $\{Att_1, Att_2, \ldots, Att_n\}$*, i.e,* $\mathbb{A} \subseteq 2^{\{Att_1, Att_2, \ldots, Att_n\}} \setminus \{\emptyset\}$*. The sets in* $\mathbb{A}$ *are called the authorized sets, and the sets not in* $\mathbb{A}$ *are called the unauthorized sets.*

We note that the access structure $\mathbb{A}$ will contain the authorized sets of attributes. In this paper, we consider monotone access structures only. However, as explained in [26], it is also possible to achieve general access structures at the cost of doubling the number of attributes in the system.

2.3 Bilinear Maps and $(P, Q, f)-$GDDHE Assumptions

Let $\mathbb{G}$, $\widetilde{\mathbb{G}}$ and $\mathbb{G}_T$ denote three finite multiplicative abelian groups of large prime order $p > 2^\lambda$ where λ is the security parameter. Let g be a generator of $\mathbb{G}$ and $\tilde{g}$ be a generator of $\widetilde{\mathbb{G}}$

An admissible bilinear map is a function $e : \mathbb{G} \times \widetilde{\mathbb{G}} \to \mathbb{G}_T$, which verifies the following properties for all $a, b \in \mathbb{Z}_p$:

1. $e(g^a, \tilde{g}^b) = e(g, \tilde{g})^{ab}$,
2. $e(g^a, \tilde{g}^b) = 1$ iff $a = 0$ or $b = 0$,
3. $e(g^a, \tilde{g}^b)$ is efficiently computable.

If such a function exists, we say that $(p, \mathbb{G}, \widetilde{\mathbb{G}}, \mathbb{G}_T, e)$ is a bilinear map group system. We say that the bilinear map group system is in:

1. Type 1 Pairings if $\mathbb{G} = \widetilde{\mathbb{G}}$
2. Type 2 Pairings if $\mathbb{G} \neq \widetilde{\mathbb{G}}$ but there is an efficiently computable homomorphism $\phi : \widetilde{\mathbb{G}} \to \mathbb{G}$
3. Type 3 Pairings if $\mathbb{G} \neq \widetilde{\mathbb{G}}$ but there are no efficiently computable homomorphism between $\widetilde{\mathbb{G}}$ and $\mathbb{G}$

We now recall the generalization of the Diffie-Hellman exponent assumption in Type 1 Pairings bilinear map group system (which was first introduced in [5]).

Let $(p, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot))$ be a bilinear map group system and $g \in \mathbb{G}$ be a generator of $\mathbb{G}$. Set $g_T = e(g,g) \in \mathbb{G}_T$. Let s, n be positive integers and $P, Q \in \mathbb{F}_p[X_1, \ldots, X_n]^s$ be two s-tuples of n-variate polynomials over $\mathbb{F}_p$. In other words, P and Q are two lists containing s multivariate polynomials each. We write $P = (p_1, p_2, \ldots, p_s)$ and $Q = (q_1, q_2, \ldots, q_s)$ and impose that $p_1 = q_1 = 1$. For any function $h : \mathbb{F}_p \to \Omega$ and vector $(x_1, \ldots, x_n) \in \mathbb{F}_p^n$, $h(P(x_1, \ldots, x_n))$ stands for $(h(p_1(x_1, \ldots, x_n)), \ldots, h(p_s(x_1, \ldots, x_n))) \in \Omega^s$. We use a similar notation for the s-tuple Q. Let $f \in \mathbb{F}_p[X_1, \ldots, X_n]$. We say that f depends on (P, Q), denoted $f \in \langle P, Q \rangle$, when there exists a linear decomposition

$$f = \sum_{1 \le i,j \le s} a_{i,j} \cdot p_i \cdot p_j + \sum_{1 \le i \le s} b_i \cdot q_i, \qquad a_{i,j}, b_i \in \mathbb{Z}_p$$

Let P, Q be as above and $f \in \mathbb{F}_p[X_1, \ldots, X_n]$. The $(P, Q, f)-$ GDHE problem is defined as follows.

Definition 4 $((P, Q, f) - \mathsf{GDHE})$ *[5].*
Given the tuple $H(x_1, \ldots, x_n) = (g^{P(x_1,\ldots,x_n)}, g_T^{Q(x_1,\ldots,x_n)}) \in \mathbb{G}^s \times \mathbb{G}_T^s$ *compute* $g_T^{f(x_1,\ldots,x_n)}$.

Definition 5 $((P, Q, f) - \mathsf{GDDHE})$ *[5].*
Given the tuple $H(x_1, \ldots, x_n) = (g^{P(x_1,\ldots,x_n)}, g_T^{Q(x_1,\ldots,x_n)}) \in \mathbb{G}^s \times \mathbb{G}_T^s$ *and* $T \in \mathbb{G}_T$ *decide whether* $T = g_T^{f(x_1,\ldots,x_n)}$.

In this paper, we will prove that our scheme is semantically secure under $(P, Q, f) -$ GDDHE assumption. Note that our scheme can be naturally extended to the Type 3 Pairings.

2.4 LSSS Matrices

Let p be a prime and $\mathcal{B}$ be the attributes' universe. If $\mathbb{A}$ is an access structure on $\mathcal{B}$, then one can find an LSSS matrix $M \in \mathbb{Z}_p^{\ell \times n}$, and a function ρ, that labels the rows of M with the attributes from $\mathcal{B}$ that appear in $\mathbb{A}$ (making use of the standard techniques in [4] if needed), i.e. $\rho \in \mathcal{F}([\ell] \to \mathcal{B})$. The pair (M, ρ) is called an LSSS access policy. Define the vector $\overrightarrow{y} = (s, y_2, \ldots, y_n)^\perp \xleftarrow{\$} \mathbb{Z}_p^n$ with sharing secret value s, and denote the vector shares $\overrightarrow{\lambda} = M.\overrightarrow{y}$. Let S denote an authorized set for $\mathbb{A}$ encoded by the policy (M, ρ), I be the set of rows of M whose labels are in S, i.e. $I = \{i | i \in [\ell] \wedge \rho(i) \in S\}$. There exists constants $\{\omega_i\}_{i \in I}$ in $\mathbb{Z}_p$ such that for any valid shares $\{\lambda_i = (M.\overrightarrow{y})_i\}_{i \in I}$ of a sharing secret s, $\sum_{i \in I} \omega_i \lambda_i = s$, and the constants $\{\omega_i\}_{i \in I}$ can be found in time polynomial in the size of matrix M. For completeness, we recall from [20] the algorithm to convert from a boolean formula to a corresponding LSSS matrix in Appendix B.

We now consider the following example, assume that the set of attributes are:

- Faculty member(FM), Administrative staff(AS), Crypto, Wireless Communications(WC), Image processing(IP), Garbled computer(GC), Security for IoT-based Applications(SIoTA), Fog computing(FC), and Internet of Things(IoT).

We define the following access policy on the set of attributes:

- (Faculty member and Crypto and Garbled computer) or (Faculty member and Wireless Communications and Fog Computing).

This access policy is already in DNF form and contains only two clauses.

By following the algorithm defined in Appendix B we obtain the access tree and corresponding LSSS matrix as in the figure 1. Where the LSSS matrix is:

$$\begin{matrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{matrix}$$

Here, the number of rows of the LSSS matrix is 6 (counting the reused attribute FM) which is bigger than the number of clauses (which is 2). Here the ratio is only three because we took a simple example with few attributes and clauses. In a more general DNF formula where each clause contains many attributes, the number of rows of the LSSS matrix will even be orders of magnitude bigger than the number of clauses. This is what motivated our new construction that we describe in the next section.

3. CONSTRUCTION

3.1 Overview of Our Approach

In this construction, we reuse the Setup algorithm and the Key generation algorithm of [18]. Regarding the encryption phase, starting with an access boolean formula, the encryptor first describes this formula in the DNF form (the disjunctions (OR) of conjunctions (AND) of attributes) and then compares between the number of clauses in the resulting DNF form and the size of the original access boolean formula. Next, the encryptor produces the ciphertext depending on which is smaller between them, by using either our new encryption algorithm or the existing encryption algorithm from [18] (section 3.5). The decryption phase automatically follows using either our new decryption algorithm or the existing decryption algorithm from [18] (section 3.5).

Regarding our new encryption and decryption algorithms, to achieve a constant number of Pairing computations for decryption we do not use the LSSS matrix in the construction, instead of that we only make use of LSSS technique in the proof. Note that, when using LSSS matrix in the construction almost existing schemes [26, 23, 8, 28, 27, 10] to name a few require at least $|I|$ Pairing computations for decryption, $|I|$ is the number of attributes for a decryption key to satisfy a ciphertext access policy.

3.2 Detailed Construction

The construction of our scheme is detailed as follows.

Setup$(\lambda, \mathcal{B})$**:** The algorithm takes as input the security parameter and attribute universe description, it generates the parameters of the system as follows: Let $N = |\mathcal{B}|$ be the maximal number of attributes in the system, let $(p, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot))$ be a bilinear group system. The algorithm first picks a random generator $g \in \mathbb{G}$, random scalars $a, \alpha \in \mathbb{Z}_p$, and then computes g^a, g^α. Next, the algorithm generates N group elements in $\mathbb{G}$ associated with N attributes in the system $h_1, \ldots, h_N$.

The master secret key is set as:

$$\mathsf{MSK} = g^\alpha$$

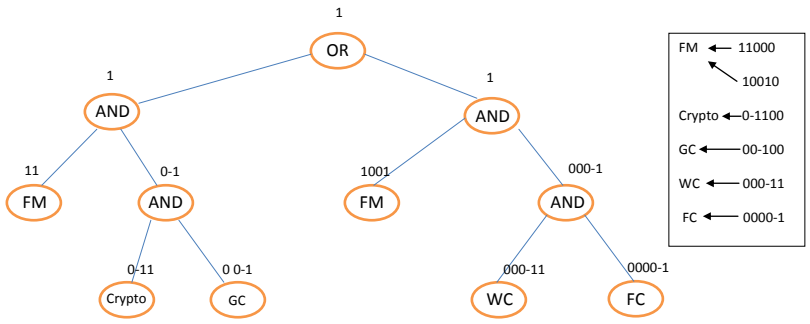

Figure 1: Access tree on the left and corresponding LSSS matrix on the right.

The global parameters are set as:

$$\mathsf{param} = (g, g^a, e(g,g)^\alpha, h_1, \ldots, h_N)$$

Extract$(u, \mathcal{B}(u), \mathsf{MSK}, \mathsf{param})$**:** The set of attributes of user u is $\mathcal{B}(u)$. The algorithm picks randomly a scalar $s_u \in \mathbb{Z}_p$, and computes the secret key for user u as

$$d_u = (d_{u_0}, d'_{u_0}, \{d_{u_i}\}_{i \in \mathcal{B}(u)})$$

where:

$$d_{u_0} = g^\alpha \cdot g^{a \cdot s_u}, d'_{u_0} = g^{s_u}, \{d_{u_i} = h_i^{s_u}\}_{i \in \mathcal{B}(u)}$$

In section 5.1 we show that user just needs to keep d_{u_0} secret, she can publish the rest of her secret key to the public domain.

Encrypt$(\mathcal{M}, \beta, \mathsf{param})$**:** Starting with an access boolean formula β, assume that the size of β is $|\beta|$. The encryptor first describes β in the form of DNF access policy as $\beta = (\beta_1 \vee \cdots \vee \beta_m)$, where each β_i is a set of attributes, $i = 1, \ldots, m$.

The encryptor picks a scalar $s \xleftarrow{\$} \mathbb{Z}_p$, then computes the first two elements of the ciphertext:

$$C = \mathcal{M} \cdot e(g,g)^{\alpha \cdot s}, C_0 = g^s$$

To compute other elements, the encryptor compares between m and $|\beta|$. If $m \leq |\beta|$ the encryptor uses our new algorithm and computes:

$$C_1 = (g^a \prod_{i \in \beta_1} h_i)^s, \ldots, C_m = (g^a \prod_{i \in \beta_m} h_i)^s$$

Else, she reverts to the encryption algorithm of [18]. The encryptor constructs an LSSS matrix M representing the original boolean formula β, and a map function ρ such that $(M, \rho) \in (\mathbb{Z}_p^{\ell \times n}, \mathcal{F}([\ell] \to [N]))$. She then chooses a random vector $\overrightarrow{v} = (s, y_2, \ldots, y_n) \in \mathbb{Z}_p^n$. For $i = 1, \ldots, \ell$ she computes $\lambda_i = \overrightarrow{v} \cdot M_i$, where M_i is the vector corresponding to the i'th row of M. She computes:

$$C_i = g^{a \cdot \lambda_i} h_{\rho(i)}^{-s}, i = 1, \ldots, \ell$$

Finally, the output is $ct = (C, C_0, \ldots, C_m)$ along with a description of β, or $ct = (C, C_0, \ldots, C_\ell)$ along with a description of (M, ρ).

Decrypt$(ct, d_u, \mathsf{param})$**:** The user u first parses the ct and checks the number of elements in ct. If it is equal to $m+1$, it means that she needs to use our new decryption algorithm. She parses the ct as $(C_0, C_1, \ldots C_m)$, then she finds j such that $\beta_j \subset \mathcal{B}(u)$, and computes:

$$\frac{e(C_0, d_{u_0} \prod_{i \in \beta_j} d_{u_i})}{e(d'_{u_0}, C_j)} = \frac{e(g^s, g^\alpha (g^a \prod_{i \in \beta_j} h_i)^{s_u})}{e(g^{s_u}, (g^a \prod_{i \in \beta_j} h_i)^s)}$$

$$= e(g,g)^{\alpha \cdot s} = K$$

Then she recovers the plaintext as $\mathcal{M} = C \cdot K^{-1}$.

Else, she reverts to the decryption algorithm of [18]. She defines the set $I \subset \{1, 2, \ldots, \ell\}$ such that $I = \{i : \rho(i) \in \mathcal{B}(u)\}$. Let $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ be a set of constants such that if $\{\lambda_i\}$ are valid shares of any secret s according to M then $\sum_{i \in I} \omega_i \lambda_i = s$. Note that from the relation $\sum_{i \in I} \omega_i M_i = (1, 0, \ldots, 0)$ where M_i is the i-th row of the matrix M, she can determine these constants. She parses the ct as $(C_0, C_1, \ldots C_\ell)$ and computes:

$$e(\prod_{i \in I} C_i^{-\omega_i}, d'_{u_0}) \cdot e(C_0, d_{u_0} \prod_{i \in I} d_{u_{\rho(i)}}^{-\omega_i}) = K$$

Then she recovers the plaintext as $\mathcal{M} = C \cdot K^{-1}$.

Remark 1 *In our scheme, to achieve the selective security, all sets β_i must be disjoint subsets, $i = 1, \ldots, m$. That means the attributes cannot be reused in the access formula. To overcome this drawback, we allow each attribute to have k_{max} copies of itself as in [18] (section 3.5). This means that, as in [18], the key-size in our scheme will increase by a factor of k_{max}, where k_{max} is the maximum number of times an attribute can appear in the access formula. In section 5.1, we show that in our scheme the user just needs to keep one element d_{u_0} secret, the rest of the user's secret key can be made public.*

3.3 Security

We will now prove the security of our construction in the model defined in section 2.1. We first define a modified BDHE assumption [7], and then prove the selective security of our scheme under this assumption.

Definition 6 ***Modified-*BDHE *problem:*** *Let* $(p, \mathbb{G}, \mathbb{G}_T, e)$ *be a bilinear group system, choose* $a, t, s, q \xleftarrow{\$} \mathbb{Z}_p$*, a generator* $g \in \mathbb{G}$*. Given* $\vec{Y} =$

$$g, g^s, g^a, \ldots, g^{a^q}, g^{a^{q+2}}, \ldots, g^{a^{2q}}, g^{s(at+a)}, g^{at},$$

$$g^{a^2 t}, \ldots, g^{a^q t}, g^{a^{q+2} t}, \ldots, g^{a^{2q} t}$$

it must remain hard to distinguish between $T = e(g,g)^{a^{q+1} s} \in \mathbb{G}_T$ *and a random element* $T = R \in \mathbb{G}_T$*.*

An adversary $\mathcal{A}$ that outputs $b \in \{0,1\}$ has advantage ϵ in solving Modified-BDHE problem in $\mathbb{G}$ if

$$\left| \Pr\left[\mathcal{A}(\vec{Y}, T = e(g,g)^{a^{q+1} s}) = 0\right] - \Pr\left[\mathcal{A}(\vec{Y}, T = R) = 0\right] \right| \geq \epsilon$$

Definition 7 *We say that the Modified-*BDHE *assumption holds if no polytime adversary has a non-negligible advantage in solving the Modified-*BDHE *problem.*

Intuitively, to compute $e(g,g)^{a^{q+1} s}$ one needs to know the values $g^{a^{q+1}}$ or $g^{a^{q+1} t}$, but these elements are not provided in $\vec{Y}$. For completeness, we prove that this assumption holds in the generic group model.

Proof. We first rewrite the Modified-BDHE assumption in the form of GDDHE assumption as follows:

$$\begin{aligned} P &= \{1, s, sa(t+1), a, a^2, \ldots, a^q, a^{q+2}, \ldots, a^{2q}, at, \\ &\quad a^2 t, \ldots, a^q t, a^{q+2} t, \ldots, a^{2q} t\} \\ Q &= \{1\} \\ f &= \{a^{q+1} s\} \end{aligned}$$

Suppose that f is not independent to (P, Q), *i.e.,* one can find $b_{i,j}, c_i$ such that the following equation holds

$$f = \sum_{\{p_i, p_j\} \subset P} b_{i,j} \cdot p_i \cdot p_j + c_i$$

We will use s to analyze f, so it is easy to deduce that one needs to find the constant $b_1, b_2, b_3, d_i, e_j, 1 \leq i \leq 2q, 1 \leq j \leq 2q$ such that the following equation holds

$$a^{q+1} s = (b_1.s + b_2.s.a(s+1))(b_3 + d_1 a + d_2 a^2 + \cdots + d_q a^q +$$

$$d_{q+2} a^{q+2} + \ldots d_{2q} a^{2q} + e_1 at + e_2 a^2 t + \cdots + e_q a^q t +$$

$$e_{q+2} a^{q+2} t + \cdots + e_{2q} a^{2q} t)$$

$$\iff a^{q+1} = (b_1 + b_2.a(t+1))(b_3 + d_1 a + d_2 a^2 + \cdots + d_q a^q +$$

$$d_{q+2} a^{q+2} + \ldots d_{2q} a^{2q} + e_1 at + e_2 a^2 t + \cdots + e_q a^q t +$$

$$e_{q+2} a^{q+2} t + \cdots + e_{2q} a^{2q} t)$$

We manage to put the term related to a^{q+1} in the left equation, we have:

$$a^{q+1}(1 - b_2(d_q + e_q.t)(t+1)) = L \qquad (1)$$

where there is no term related to a^{q+1} appearing in L. Since a, t, s are chosen at random, so to let the equation (1) hold, one needs to choose b_2, e_q, d_q such that $(1 - b_2(d_q + e_q.t)(t+1)) = t^2 b_2 e_q + (b_2 d_q + b_2 e_q)t + b_2 d_q - 1 = 0, \forall t$, this cannot hold because one cannot find b_2, e_q, d_q such that simultaneously $b_2 e_q = 0$ and $b_2 d_q + b_2 e_q = 0$ and $b_2 d_q - 1 = 0$. That means the equation (1) cannot hold or f is independent to (P, Q). □

Theorem 1 *Assume that* β^* *is the challenge access policy and from* β^* *construct the corresponding challenge LSSS matrix L' of size* $\ell' \times n'$ *and map function* ρ'*. Describe* $\beta^* = \beta_1^* \vee \cdots \vee \beta_m^*$ *where* $\beta_i^*, i = 1, \ldots, m$ *are disjoint sets and then construct the corresponding challenge LSSS matrix* L^* *of size* $\ell^* \times n^*$ *and map function* ρ^**. If those LSSS matrices satisfy* $\ell', n', \ell^*, n^* \leq q$*, our scheme is selectively secure under the Modified-*BDHE *assumption.*

Proof. The simulator $\mathcal{S}$ is first given an instance of Modified - BDHE assumption, it will simulate an adversary $\mathcal{A}$ who attacks our scheme in the selective secure game with non-negligible advantage and then use the output of $\mathcal{A}$ to break the security of Modified-BDHE assumption.

Setup The simulator is first given an instance of Modified-BDHE assumption, and then receives challenge access policy β^* from $\mathcal{A}$. Assume the size of β^* is $|\beta^*|$.

She first describes β^* in the DNF form as $\beta^* = \beta_1^* \vee \cdots \vee \beta_m^*$ where $\beta_i^*, i = 1, \ldots, m$ are disjoint sets (using the copies of attributes as needed as explained in Remark 1). Then she compares between m and $|\beta^*|$ (the size of the original β^*). There are two cases:

First Case: $m > |\beta^*|$**.** Our scheme now is exactly the same as the scheme in [18] (section 3.5). We thus refer the reader to the proof of scheme in [18] (section 3.5). Note that the instance of Modified-BDHE assumption includes the instance of BDHE assumption.

Second Case: $m \leq |\beta^*|$**.** The simulator first constructs LSSS matrix $(M^*_{\ell^* \times n^*}, \rho^*)$ from $\beta^* = \beta_1^* \vee \cdots \vee \beta_m^*$ where both $\ell^*, n^* \leq q$. To program the value $e(g,g)^\alpha$, the simulator picks $\alpha' \xleftarrow{\$} \mathbb{Z}_p$ and implicitly sets $\alpha = \alpha' + a^{q+1}$. She computes $e(g,g)^\alpha = e(g^a, g^{a^q}) e(g,g)^{\alpha'}$.

The simulator finds disjoint sets of rows of matrix M^*: $I_1, \ldots, I_m$ where $\{\rho(i), i \in I_j\} = \beta_j^*$.

β^* is now described as: $(\wedge \rho(i))_{i \in I_1} \vee (\wedge \rho(i))_{i \in I_2} \vee \cdots \vee (\wedge \rho(i))_{i \in I_m}$.

To program the group elements $h_1, \ldots, h_N$, the simulator implicitly defines the vector

$$\overrightarrow{y} = (t, ta, ta^2, \ldots, ta^{n^*-1})^\perp \in \mathbb{Z}_p^{n^*}$$

Let $\overrightarrow{\lambda} = M^* \cdot \overrightarrow{y}$ be the vector shares, thus for $j = 1, \ldots, \ell^*$

$$\lambda_j = \sum_{i \in [n^*]} M^*_{j,i} t a^{i-1}$$

She then finds the set $\{\omega_i\}_{1 \leq i \leq \ell^*}$ such that for all $j = 1, \ldots, m$:

$$\sum_{i \in I_j} \omega_i \cdot \lambda_i = t$$

For each $h_j, 1 \leq j \leq N$, where there exists an indice $i \in [\ell^*]$ such that $j = \rho(i)$ (note that the function ρ is injective), the simulator chooses $z_j \xleftarrow{\$} \mathbb{Z}_p$ and computes: Note that the simulator knows matrix M^* and g^{ta^k} where $k \in [n^*]$ from the instance of Modified-BDHE assumption.

$$h_j = g^{z_j} \cdot g^{\omega_i \sum_{k \in [n^*]} M^*_{i,k} t a^k} = g^{z_j} \cdot g^{a \omega_i \lambda_i}$$

Otherwise, the simulator chooses $z_j \xleftarrow{\$} \mathbb{Z}_p$ and computes $h_j = g^{z_j}$. We note that $\{h_j\}_{j=1,\ldots,N}$ are distributed randomly due to choosing randomly z_j. Finally, the simulator gives

$$\mathsf{param} = (g, g^a, e(g,g)^\alpha, h_1, \ldots, h_N)$$

to $\mathcal{A}$ and ends the Setup phase.

Query phase 1 the simulator needs to answer the corrupted queries. To this aim, $\mathcal{A}$ first sends the set of indices of attributes $S \subset [N]$ to simulator with the requirement that the set of attributes associated with S doesn't satisfy M^*. The simulator first finds a vector $\overrightarrow{x} = (x_1, \ldots, x_{n^*}) \in \mathbb{Z}_p^{n^*}$ such that $x_1 = -1$ and for all i where $\rho^*(i) \in S$ the product $\langle \overrightarrow{x} \cdot M_i^* \rangle = 0$. Based on the property of LSSS matrix, such vector $\overrightarrow{x}$ exists. The simulator continues to pick $r \xleftarrow{\$} \mathbb{Z}_p$ and implictly define the value s_u as:

$$s_u = r + x_1 a^q + x_2 a^{q-1} + \cdots + x_{n^*} a^{q-n^*+1}$$

The simulator computes

$$d_{u_0} = g^{\alpha'} g^{ar} \prod_{i=2,\ldots,n^*} (g^{a^{q+1-i}})^{x_i} = g^\alpha \cdot g^{a \cdot s_u}$$

Note that $x_1 = -1$ thus $g^{a \cdot s_u}$ contains the term $g^{-a^{q+1}}$ which cancels out the unknown term $g^{a^{q+1}}$ in g^α. With the known vector $\overrightarrow{x}$, she continues to compute:

$$d'_{u_0} = g^{s_u} = g^r \prod_{i=1,\ldots,n^*} (g^{a^{q+1-i}})^{x_i}$$

For $j \in S$ such that there is no $i \in [\ell^*]$ satisfying $\rho^*(i) = j$. The simulator knows values z_j and computes

$$h_j^{s_u} = (g^{s_u})^{z_j}$$

For $j \in S$ such that there is an indice $i \in [\ell^*]$ satisfying $\rho^*(i) = j$. The simulator computes

$$h_j^{s_u} = (g^{s_u})^{z_j} \cdot$$
$$\cdot g^{(r + x_1 a^q + x_2 a^{q-1} + \cdots + x_{n^*} a^{q-n^*+1}) \omega_i \sum_{k \in [n^*]} M_{i,k}^* t a^k}$$

Note that the product $\langle \overrightarrow{x} \cdot M_i^* \rangle = 0$ thus the simulator doesn't need to know the unknown term of form $g^{a^{q+1}t}$ to compute $h_j^{s_u}$, all other terms he knows from the assumption. If j is outside the set S and there exists $i \in [\ell^*]$ such that $\rho^*(i) = j$, the simulator cannot compute $h_j^{s_u}$ since $\langle \overrightarrow{x} \cdot M_i^* \rangle \neq 0$ (this is exactly the classical partition technique proof).

Challenge The adversary gives two equal length message $\mathcal{M}_0^*, \mathcal{M}_1^*$ to the simulator. The simulator picks a random bit b, computes:

$$C^* = \mathcal{M}_b^* \cdot T \cdot e(g^s, g^{\alpha'}), C_0^* = g^s$$

and the other elements $(C_1^*, \ldots, C_m^*) =$

$$\left(g^{s(a+at)} g^{\sum_{i \in I_1} s z_{\rho(i)}}, \ldots, g^{s(a+at)} g^{\sum_{i \in I_m} s z_{\rho(i)}} \right) =$$

$$\left((g^a \cdot \prod_{i \in I_1} g^{z_{\rho(i)}} \cdot g^{a \omega_i \lambda_i})^s, \ldots, (g^a \cdot \prod_{i \in I_m} g^{z_{\rho(i)}} \cdot g^{a \omega_i \lambda_i})^s \right)$$

$$= \left((g^a \prod_{i \in I_1} h_{\rho(i)})^s, \ldots, (g^a \prod_{i \in I_m} h_{\rho(i)})^s \right)$$

$$= \left((g^a \prod_{i \in \beta_1^*} h_i)^s, \ldots, (g^a \prod_{i \in \beta_m^*} h_i)^s \right)$$

The disjoint sets $\{\rho(i), i \in I_j\}$ is β_j^*. Note that if $T = e(g,g)^{a^{q+1}s}$ then ct^* is in valid form.

Query phase 2 The same as Phase 1

Guess $\mathcal{A}$ sends his guess b' to simulator, the simulator then outputs 0 to guess that $T = e(g,g)^{a^{q+1}s}$ if $b' = b$; otherwise, it outputs 1 to guess that T is a random group element in $\mathbb{G}_T$.

When $T = e(g,g)^{a^{q+1}s}$ the simulator $\mathcal{S}$ gives a perfect simulation so we have that

$$\Pr\left[\mathcal{S}(\vec{Y}, T = e(g,g)^{a^{q+1}s}) = 0\right] = \frac{1}{2} + \mathbf{Adv}_{\mathcal{A}}$$

When T is a random group element the message $\mathcal{M}_b^*$ is completely hidden from the adversary and we have

$$Pr[\mathcal{S}(\vec{Y}, T = R) = 0] = \frac{1}{2}$$

Therefore, the simulator can play the Modified - BDHE game with non-negligible advantage (equal to $\mathbf{Adv}_{\mathcal{A}}$) or she can break the security of Modified-BDHE assumption. □

4. PERFORMANCE ANALYSIS

From a performance perspective, it is worth mentioning that our construction being an improvement of the scheme proposed in [18], it will perform as good as in the worst case, which is when the number of clauses is bigger than the size of the boolean formula. And [18] was already one of the best performing CP-ABE schemes. In the more favorable case where the number of clauses is smaller than the size of the boolean formula our scheme performs much better since the ciphertext becomes linear in the number of clauses instead of size of the LSSS matrix, and decryption time remains the smallest of all CP-ABE schemes proposed in the literature. If we dig deeper, we can observe that our new decryption algorithm just needs two pairings and $|I|$ multiplications in $\mathbb{G}$, while in [18] it needs two pairings, $2|I|$ exponentiations, and $2|I|$ multiplications in $\mathbb{G}$. Furthermore, since the user just needs to keep d_{u_0} secret, she can delegate some computing works to a third party so that she only needs to compute one pairing when decrypting, as we will explain in section 5.1. To be more precise we give a comparison table in table 2 where we see various schemes proposed in the literature compared to ours.

Remark 2 *Our scheme targets CP-ABE with fine grained access policies that can be expressed as boolean formulas and is the most efficient to do so. There is a category of efficient algorithms targeting a more specific policy, the (t,n)-threshold policy. Such policy can be expressed with boolean formulas as a DNF and be supported by our construction. In that case, our construction will be less efficient than the dedicated algorithms, because the resulting DNF will have a larger number of clauses. However our construction is*

	Policy	Ciphertext	Private Key	Decryption Time	Assumption
[15]	Tree	$O(N \cdot \ell^{3.42})$	$O(\lvert\mathcal{B}(u)\rvert \cdot \ell^{3.42})$	$O(N \cdot \ell^{3.42})P$	d-BDH
[19]	CNF	$(2m+1)$	$(2N + \lvert\mathcal{B}(u)\rvert)$	$(2m+1)P$	GDDHE
[26].1	LSSS	$(2\ell+1)$	$(\lvert\mathcal{B}(u)\rvert+2)$	$(2\lvert I\rvert+1)P$	*q-parallel* BDHE
[18](3.5)	LSSS	$(\ell+1)$	$(k_{max}\lvert\mathcal{B}(u)\rvert+2)$	$2P$	BDHE
[23]	LSSS	$(3\ell+2)$	$(2\lvert\mathcal{B}(u)\rvert+2)$	$(3\lvert I\rvert+1)P$	*q-type*
[2]	LSSS	$O(1)$	$O((\mathcal{B}(u)^*.\ell^*)^4)$	$3\lvert I\rvert P$	Parametrized
Ours	LSSS/DNF	$(m'+1)$	$(k_{max}\lvert\mathcal{B}(u)\rvert+2)$	$2P$	Modified-BDHE

Figure 2: N denotes the maximal number of attributes in the system, ℓ denotes the number of rows of the LSSS matrix, ℓ^* denotes the maximum of ℓ (that should be N), $|\mathcal{B}(u)|$ denotes the size of attribute set of a decryption key, $\mathcal{B}(u)^*$ denotes the maximum of $|\mathcal{B}(u)|$, m is the number of clauses in a CNF or DNF. Note that $m' = m$ if $m \leq \ell$, else $m' = \ell$. $|I|$ is the number of attributes for a decryption key to satisfy a ciphertext policy, P denotes Pairing computation, k_{max} denotes the maximal number of times where one attribute can be reused in an access formula.

more generic as it supports more expressive policies than simply threshold ones. Moreover threshold policy is not a fine-grained access policy since there is no distinction among attributes. It is therefore fair to say that our approach is the most efficient for generic and fine-grained policies.

5. EXTENSIONS

In this section we discuss two extensions that can further improve the performance or security of our construction.

5.1 Minimizing The User's Storage

We first show that our scheme is still secure under the same assumption if the user keeps only one element d_{u_0} secret. This is a very useful property in cloud-based scenario as it implies that the user can store a minimal amount of information and delegate some computations in the encryption and decryption phases to the cloud. This means that the client can thus save both storage and computation power.

To establish this property, we need to prove that the simulator still can simulate the adversary when the user keeps only the element d_{u_0} secret. More precisely, regarding a challenge key d_u^* (which can be used to decrypt the challenge ciphertext), the simulator now needs to provide d_u^* for the adversary except the secret part $d_{u_0}^*$. We sketch the proof as follows.

First, the simulator does the same as in the proof above, implicitly sets $\alpha = \alpha' + a^{q+1}$ where a^{q+1} is the challenge term from the assumption, therefore she can embed the challenge term from the assumption to g^α. Although simulator doesn't know $g^{a^{q+1}}$, she can manage to choose s_u such that $g^{a \cdot s_u}$ contains the term $g^{-a^{q+1}}$ which cancels out the unknown term $g^{a^{q+1}}$ in g^α. Thus, for the not challenge key the simulator compute

$$d_{u_0} = g^{\alpha'} g^{ar} \prod_{i=2,\ldots,n^*} (g^{a^{q+1-i}})^{x_i} = g^\alpha \cdot g^{a \cdot s_u}$$

Note that with such chosen s_u, the simulator only can compute the not challenge key $d_u \notin \{d_u^*\}$ as explained in the proof. Fortunately, for the challenge key d_u^* the simulator doesn't need to compute $d_{u_0}^*$, moreover the term g^α doesn't appear anywhere except in the part $d_{u_0}^*$, therefore she doesn't need to manage to choose s_u^* to cancel out $g^{a^{q+1}}$, she in fact can choose randomly s_u^* and then uses the public parameter to compute the secret key d_u^* (of course except the $d_{u_0}^*$ part). This leads to the fact that the simulator still successfully simulates the adversary which means that the security of the scheme is still unchanged. Finally, we remark that the above trick also can be applied to some other schemes such as [26, 18, 23] which share a similar key structure to ours. That shows that our scheme is still secure in the case $m > \ell$.

5.2 Multi-Authorities

In this section, we show that our scheme can be extended to support multi-authorities. The idea is to use the splitting technique to *split* the master key g^α into θ parts corresponding to θ authorities. This efficiently solves the problem of key escrow by the authority which is one of the weaknesses of current CP-ABE schemes.

To be more precise, in the extract phase, the user needs to get θ partial secrete keys from θ authorities to compute the full secret key. On the other hand, the colluding of up to $\theta - 1$ curious authorities cannot compute the full secret key of user. For completeness we provide the definition of multi-authority ABE scheme and the security model in Appendix A.

Regarding the construction, we will make some changes on the setup algorithm and extract algorithm while we retain the encryption and decryption algorithms. For the security, in our system we have an additional new type of adversary called the curious authority (Type II adversary), we need to prove that our scheme is secure against this adversary. We detail the construction and the security as follows.

5.2.1 Description of the multi-authorities construction

Setup$(\lambda, \mathcal{B}, \theta)$**:** The algorithm takes as input the security parameter, attribute universe description, and the maximum number of authorities θ in the system. It generates the parameters of the system as follows: Let $N = |\mathcal{B}|$ be the maximal number of attributes in the system and let $(p, \mathbb{G}, \mathbb{G}_T, e(\cdot,\cdot))$ be a bilinear group system. The algorithm first picks a random generator $g \in \mathbb{G}$, a random scalar $a \in \mathbb{Z}_p$, and then computes g^a. Next, the algorithm generates N group elements in $\mathbb{G}$ associated with N attributes in the system $h_1, \ldots, h_N$.

Next, the authority $i \in [1, \ldots, \theta]$ picks a random scalar $\alpha_i \in \mathbb{Z}_p$. Computes $g^{\alpha_i}, e(g,g)^{\alpha_i}$, sets $\mathsf{MSK}_i = g^{\alpha_i}$ as his secret key and $e(g,g)^{\alpha_i}$ as his public key.

The algorithm implicitly sets $\alpha = \alpha_1 + \cdots + \alpha_\theta$, then computes $e(g,g)^{\alpha} = e(g,g)^{\alpha_1} \cdots e(g,g)^{\alpha_\theta}$.

The global parameters are set

$$\mathsf{param} = (g, g^a, h_1, \ldots, h_N, e(g,g)^{\alpha})$$

Extract$(u, \mathcal{B}(u), \mathsf{MSK}_i, \mathsf{param})$**:** This algorithm is run by the authority i-th. For notational simplicity, assume that the set of attributes of user u is $\mathcal{B}(u) = (Att_1, \ldots, Att_k)$, $k \leq N$. Picks randomly scalar $s_u^i \in \mathbb{Z}_p$, the i-th partial secret key for user u is $d_u^i = (d_{u_0}^i, d_{u_0}^{'i}, d_{u_1}^i, \ldots, d_{u_k}^i)$ where:

$$d_{u_0}^i = g^{\alpha_i} \cdot g^{a \cdot s_u^i}, d_{u_0}^{'i} = g^{s_u^i}, d_{u_1}^i = h_1^{s_u^i}, \ldots, d_{u_k}^i = h_k^{s_u^i}$$

The above algorithm is run θ times by θ different authorities. Finally, the full secret key of user is set

$$d_{u_0} = g^{\alpha} \cdot g^{a \cdot s_u}, d'_{u_0} = g^{s_u}, d_{u_1} = h_1^{s_u}, \ldots, d_{u_k} = h_k^{s_u}$$

where: $s_u = s_u^1 + \cdots + s_u^\theta$.

Encrypt$(\mathcal{M}, \beta, \mathsf{param})$**:** unchanged.
We notice that $(h_{\rho(i)})_{i \in I_j}, j = 1, \ldots, m$, are disjoint subsets.

Decrypt$(ct, d_u, \mathcal{B}(u), \mathsf{param})$**:** unchanged

Remark 3 *Compared to other existing multi-authority* ABE *schemes supporting fine-grained access policies [20, 24] the advantage of our scheme is that it achieves* ful *user privacy and not* partial *user privacy as in their schemes. In fact their scheme aim at distributing the set of attributes among different authorities and they do not efficiently solve the problem of key escrow, they simply distribute the load and responsibility among different authoritites by making each authority responsible for a disjoint set of attributes (but the authorities can still decrypt messages where the access policy is expressed with attributes from only one authority). Moreover, the efficiency of our scheme is better than the efficiency of their schemes in terms of ciphertext size and decryption time.*

5.2.2 Security Proof

Regarding the Type I adversary, it is the same as in the section 3.1, here we will focus on the Type II adversary.

Theorem 2 *Assume that β^* is the challenge access policy and from β^* construct the corresponding challenge LSSS matrix L' of size $\ell' \times n'$ and map function ρ'. Describe $\beta^* = \beta_1^* \vee \cdots \vee \beta_m^*$ where $\beta_i^*, i = 1, \ldots, m$ are disjoint sets and then construct the corresponding challenge LSSS matrix L^* of size $\ell^* \times n^*$ and map function ρ^*. If those LSSS matrices satisfy $\ell', n', \ell^*, n^* \leq q$, our scheme is selectively secure against Type II adversary under the Modified-*BDHE *assumption.*

Proof. Compare to the Type I adversary, for the Type II adversary the simulator needs to provide additional $\theta - 1$ partial master keys, without loss of generality we suppose that these are $\mathsf{MSK}_1, \ldots, \mathsf{MSK}_{\theta-1}$.

The simulator simply picks randomly $\theta - 1$ scalars $\alpha_1, \ldots, \alpha_{\theta-1}$ and implicitly sets $\alpha_\theta = \alpha - \alpha_1 - \cdots - \alpha_{\theta-1}$. Note that the simulator doesn't know α_θ, however she knows $e(g,g)^{\alpha}$ thus she still can compute $e(g,g)^{\alpha_\theta}$. Therefore, she can provide the master keys $\mathsf{MSK}_1, \ldots, \mathsf{MSK}_{\theta-1}$ to the adversary. The rest of the proof still remains. □

6. CONCLUSION

In this paper, we propose a CP-ABE scheme that extends the solution of [18]. Our scheme is more efficient than [18] when the size of the boolean formula is bigger than the number of clauses, otherwise it has the same efficiency. Our construction also achieves fast decryption time, hence it is already an improvement to [18] which was one of the best constructions for CP-ABE supporting fine grained policies. On top of that, our scheme can be extended to allow the user to keep only part of the encryption and decryption material secret, which lends itself well to the case where the user has limited storage and computation capabilities or when the client is taking advantage of a cloud. Finally we proposed a support for multiple non-colluding authorities to effectively mitigate the problem of key-escrow by the authority that extracts the key.

Acknowledgments

This publication was made possible by the NPRP award X-063-1–014 from the Qatar National Research Fund (a member of The Qatar Foundation). The authors thank Willy Susilo and Tran Viet Xuan Phuong for helpful discussions about this work. The statements made herein are solely the responsibility of the authors.

7. REFERENCES

[1] S. Agrawal and M. Chase. A study of pair encodings: predicate encryption in prime order groups. IACR Cryptology ePrint Archive, 2015:413, 2015.

[2] N. Attrapadung, G. Hanaokay, and S. Yamada. Conversions among several classes of predicate encryption and applications to abe with various compactness tradeoffs. IACR Cryptology ePrint Archive, 2015:431, 2015.

[3] N. Attrapadung, B. Libert, and E. de Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, editors, *PKC 2011: 14th International Workshop on Theory and Practice in Public Key Cryptography*, volume 6571 of *Lecture Notes in Computer Science*, pages 90–108, Taormina, Italy, Mar. 6–9, 2011. Springer, Berlin, Germany.

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

[5] D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical identity based encryption with constant size ciphertext. In R. Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany.

[6] D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan, and D. Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In P. Q. Nguyen and E. Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 533–556, Copenhagen, Denmark, May 11–15, 2014. Springer, Berlin, Germany.

[7] D. Boneh, C. Gentry, and B. Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In V. Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275, Santa Barbara, CA, USA, Aug. 14–18, 2005. Springer, Berlin, Germany.

[8] C. Chen, J. Chen, H. W. Lim, Z. Zhang, D. Feng, S. Ling, and H. Wang. Fully secure attribute-based systems with short ciphertexts/signatures and threshold access structures. In E. Dawson, editor, *Topics in Cryptology – CT-RSA 2013*, volume 7779 of *Lecture Notes in Computer Science*, pages 50–67, San Francisco, CA, USA, Feb. 25 – Mar. 1, 2013. Springer, Berlin, Germany.

[9] C. Chen, Z. Zhang, and D. Feng. Efficient ciphertext policy attribute-based encryption with constant-size ciphertext and constant computation-cost. In X. Boyen and X. Chen, editors, *ProvSec 2011: 5th International Conference on Provable Security*, volume 6980 of *Lecture Notes in Computer Science*, pages 84–101, Xi'an, China, Oct. 16–18, 2011. Springer, Berlin, Germany.

[10] J. Chen, R. Gay, and H. Wee. Improved dual system ABE in prime-order groups via predicate encodings. Lecture Notes in Computer Science, pages 595–624. Springer, Berlin, Germany, 2015.

[11] K. Emura, A. Miyaji, A. Nomura, K. Omote, and M. Soshi. A ciphertext-policy attribute-based encryption scheme with constant ciphertext length. In B. F, L. H, and W. G, editors, *Proceedings of ISPEC*, LNCS 5451, pages 13–23. Springer, 2009.

[12] S. Garg, C. Gentry, S. Halevi, A. Sahai, and B. Waters. Attribute-based encryption for circuits from multilinear maps. In R. Canetti and J. A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 479–499, Santa Barbara, CA, USA, Aug. 18–22, 2013. Springer, Berlin, Germany.

[13] A. Ge, R. Zhang, C. Chen, C. Ma, and Z. Zhang. Threshold ciphertext policy attribute-based encryption with constant size ciphertexts. In W. Susilo, Y. Mu, and J. Seberry, editors, *ACISP 12: 17th Australasian Conference on Information Security and Privacy*, volume 7372 of *Lecture Notes in Computer Science*, pages 336–349, Wollongong, NSW, Australia, July 9–11, 2012. Springer, Berlin, Germany.

[14] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Attribute-based encryption for circuits. In D. Boneh, T. Roughgarden, and J. Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 545–554, Palo Alto, CA, USA, June 1–4, 2013. ACM Press.

[15] V. Goyal, A. Jain, O. Pandey, and A. Sahai. Bounded ciphertext policy attribute based encryption. In L. Aceto, I. Damgård, L. A. Goldberg, M. M. Halldórsson, A. Ingólfsdóttir, and I. Walukiewicz, editors, *ICALP 2008: 35th International Colloquium on Automata, Languages and Programming, Part II*, volume 5126 of *Lecture Notes in Computer Science*, pages 579–591, Reykjavik, Iceland, July 7–11, 2008. Springer, Berlin, Germany.

[16] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In A. Juels, R. N. Wright, and S. Vimercati, editors, *ACM CCS 06: 13th Conference on Computer and Communications Security*, pages 89–98, Alexandria, Virginia, USA, Oct. 30 – Nov. 3, 2006. ACM Press. Available as Cryptology ePrint Archive Report 2006/309.

[17] J. Herranz, F. Laguillaumie, and C. Ràfols. Constant size ciphertexts in threshold attribute-based encryption. In P. Q. Nguyen and D. Pointcheval, editors, *PKC 2010: 13th International Conference on Theory and Practice of Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 19–34, Paris, France, May 26–28, 2010. Springer, Berlin, Germany.

[18] S. Hohenberger and B. Waters. Attribute-based encryption with fast decryption. In K. Kurosawa and G. Hanaoka, editors, *PKC 2013: 16th International Workshop on Theory and Practice in Public Key Cryptography*, volume 7778 of *Lecture Notes in Computer Science*, pages 162–179, Nara, Japan, Feb. 26 – Mar. 1, 2013. Springer, Berlin, Germany.

[19] P. Junod and A. Karlov. An efficient public-key attribute-based broadcast encryption scheme allowing arbitrary access policies. In *ACM Workshop on Digital Rights Management*, pages 13–24. ACM Press, 2010.

[20] A. B. Lewko and B. Waters. Decentralizing attribute-based encryption. In K. G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 568–588, Tallinn, Estonia, May 15–19, 2011. Springer, Berlin, Germany.

[21] T. Okamoto and K. Takashima. Fully secure unbounded inner-product and attribute-based encryption. In X. Wang and K. Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 349–366, Beijing, China, Dec. 2–6, 2012. Springer, Berlin, Germany.

[22] R. Ostrovsky, A. Sahai, and B. Waters. Attribute-based encryption with non-monotonic access structures. In P. Ning, S. D. C. di Vimercati, and P. F. Syverson, editors, *ACM CCS 07: 14th Conference on Computer and Communications Security*, pages 195–203, Alexandria, Virginia, USA, Oct. 28–31, 2007. ACM Press.

[23] Y. Rouselakis and B. Waters. Practical constructions and new proof methods for large universe attribute-based encryption. In A.-R. Sadeghi, V. D. Gligor, and M. Yung, editors, *ACM CCS 13: 20th Conference on Computer and Communications Security*, pages 463–474, Berlin, Germany, Nov. 4–8, 2013. ACM Press.

[24] Y. Rouselakis and B. Waters. Efficient statically-secure large-universe multi-authority attribute-based encryption. In *FC 2015: 19th International Conference on Financial Cryptography and Data Security*, Lecture Notes in Computer Science, pages 315–332. Springer, Berlin, Germany, 2015.

[25] A. Sahai and B. R. Waters. Fuzzy identity-based encryption. In R. Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473,

Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany.

[26] B. Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, editors, *PKC 2011: 14th International Workshop on Theory and Practice in Public Key Cryptography*, volume 6571 of *Lecture Notes in Computer Science*, pages 53–70, Taormina, Italy, Mar. 6–9, 2011. Springer, Berlin, Germany.

[27] H. Wee. Dual system encryption via predicate encodings. In Y. Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 616–637, San Diego, CA, USA, Feb. 24–26, 2014. Springer, Berlin, Germany.

[28] S. Yamada, N. Attrapadung, G. Hanaoka, and N. Kunihiro. A framework and compact constructions for non-monotonic attribute-based encryption. In H. Krawczyk, editor, *PKC 2014: 17th International Workshop on Theory and Practice in Public Key Cryptography*, volume 8383 of *Lecture Notes in Computer Science*, pages 275–292, Buenos Aires, Argentina, Mar. 26–28, 2014. Springer, Berlin, Germany.

APPENDIX

A. MULTI-AUTHORITY CP - ABE

Formally, a multi-authority CP-ABE scheme consists of four probabilistic algorithms.

Setup$(1^\lambda, \mathcal{B}, \theta)$**:** The setup algorithm takes the security parameter λ, the description of the attributes' universe $\mathcal{B}$, and the maximum number of authorities in the system θ as inputs. It generates θ partial master key $\mathsf{MSK}_1, \ldots, \mathsf{MSK}_\theta$, as well as the global public parameters param of the system.

Extract$(u, \mathcal{B}(u), \mathsf{MSK}_i, \mathsf{param})$**:** Takes as input a user u and his set of attributes $\mathcal{B}(u)$, as well as the public parameters param and a partial master key MSK_i $(1 \leq i \leq \theta)$. It outputs the partial user's private key d_u^i. This algorithm is run θ times by θ different authorities. Finally, the full secret key of user is computed from θ partial user's private key.

Encrypt$(\mathcal{M}, \mathbb{A}, \mathsf{param})$**:** Takes as input a message $\mathcal{M}$, an access policy $\mathbb{A}$ over the universe of attributes and the global public parameters param. It outputs the ciphertext ct along with a description of the access policy $\mathbb{A}$.

Decrypt$(ct, d_u, \mathsf{param})$**:** Takes as input the ciphertext ct, the private key d_u of user u, together with the global public parameters param. It outputs the message $\mathcal{M}$ if and only if $\mathcal{B}(u)$ satisfies $\mathbb{A}$. Otherwise, it outputs $\perp$.

Security Model.

We consider two types of adversary for a multi-authority CP-ABE scheme, named Type I adversary $\mathcal{A}_1$ (related to Game I below) and Type II adversary $\mathcal{A}_2$ (related to Game II below).

In fact, $\mathcal{A}_1$ represents a third party adversary against the multi-authority CP-ABE scheme. Then, $\mathcal{A}_1$ does not know any partial master secret key. On contrary, the adversary $\mathcal{A}_2$ represents a curious authority who generates partial secret key of users. Then, besides knowing other information as $\mathcal{A}_1$ does $\mathcal{A}_2$ also knows additionally at most $\theta - 1$ partial master secret keys.

It is straightforward to realize that a multi-authority CP-ABE scheme is secure if it resists Type II adversary.

Game I: it is almost the same as the definition of security model in the section 2.1.
Game II: it is also almost the same as the definition of security model in the section 2.1, except that the simulator has to provide at most $\theta - 1$ partial master keys for the adversary.

B. CONVERSION FROM A BOOLEAN FORMULA TO A CORRESPONDING LSSS MATRIX

In this section, we recall from [20] the algorithm to convert from a boolean formula to a corresponding LSSS matrix. The algorithm works as follows.

We first consider the boolean formula as an access tree, where interior nodes are AND and OR gates and the leaf nodes correspond to attributes. We will use $(1, 0, \ldots, 0)$ as the sharing vector for the LSSS matrix. We begin by labeling the root node of the tree with the vector (1) (a vector of length 1). We then go down the levels of the tree, labeling each node with a vector determined by the vector assigned to its parent node. We maintain a global counter variable c which is initialized to 1.

If the parent node is an OR gate labeled by the vector v, then we also label its children by v (and the value of c stays the same). If the parent node is an AND gate labeled by the vector v, we pad v with 0's at the end (if necessary) to make it of length c. Then we label one of its children with the vector $v|1$ (where $|$ denotes concatenation) and the other with the vector $(0, \ldots, 0)| - 1$, where $(0, \ldots, 0)$ denotes the zero vector of length c. Note that these two vectors sum to $v|0$. We now increment the value of c by 1. Once we have finished labeling the entire tree, the vectors labeling the leaf nodes form the rows of the LSSS matrix. If these vectors have different lengths, we pad the shorter ones with 0's at the end to arrive at vectors of the same length.