Attribute-Based Encryption with Partially Hidden Encryptor-Specified Access Structures

Takashi Nishide[1,2], Kazuki Yoneyama[1], and Kazuo Ohta[1]

[1] Department of Information and Communication Engineering,
The University of Electro-Communications, 1-5-1 Chofugaoka Chofu-shi,
Tokyo 182-8585 Japan
{t-nishide,yoneyama,ota}@ice.uec.ac.jp
[2] Hitachi Software Engineering Co., Ltd.; 4-12-7 Higashi-Shinagawa Shinagawa-ku,
Tokyo, 140-0002 Japan

Abstract. We propose attribute-based encryption schemes where encryptor-specified access structures (also called ciphertext policies) are hidden. By using our schemes, an encryptor can encrypt data with a hidden access structure. A decryptor obtains her secret key associated with her attributes from a trusted authority in advance and if the attributes associated with the decryptor's secret key do not satisfy the access structure associated with the encrypted data, the decryptor cannot decrypt the data or guess even what access structure was specified by the encryptor. We prove security of our construction based on the Decisional Bilinear Diffie-Hellman assumption and the Decision Linear assumption. In our security notion, even the legitimate decryptor cannot obtain the information about the access structure associated with the encrypted data more than the fact that she can decrypt the data.

Keywords: Attribute-Based Encryption, Recipient Anonymity, Access Control on Encrypted Data, Ciphertext Policy.

1 Introduction

In the distributed setting, we need to enforce access control polices to protect various resources. In such settings, it may be suitable to specify access control policies based on attributes rather than individual identities, because an identity may not have enough information about its entity. Attribute-based encryption (ABE) is a mechanism by which we can realize such access control in a cryptographic way. There are two kind of ABE schemes, key-policy and ciphertext-policy ABE schemes.

In the key-policy ABE schemes [12, 18, 19, 15], ciphertexts are associated with sets of attributes and users' secret keys are associated with access structures. If the attributes associated with the ciphertext satisfy the access structure of the secret key, the secret key holder can decrypt the ciphertext successfully. Also the concept of searchable and predicate encryption [5, 21] is related to key-policy ABE in the sense that successful decryption is conditional on access structure associated with secret keys.

On the other hand, in the ciphertext-policy ABE (CP-ABE) schemes [2, 11, 15, 16], the situation is reversed. That is, attributes are associated with secret keys and access structures are associated with ciphertexts and called ciphertext policies. The access structures are described with the attributes and therefore the concept of CP-ABE is closely related to Role-Based Access Control.

In this work, we focus on CP-ABE and construct a CP-ABE scheme where we can hide encryptor-specified access structures associated with ciphertexts. Our scheme can be considered as a recipient-anonymous targeted broadcast and the relation of our scheme to a normal CP-ABE scheme is similar to that of anonymous identity-based encryption (IBE) to normal IBE. For example, suppose a company wants to hire certain qualified people who satisfy the policy the company specified and the policy may contain the useful information about the company's business strategy. The company can post a message encrypted by our CP-ABE scheme on a public bulletin board to seek applications. By doing so, the company can keep the important policy confidential. Since the policy is hidden, the rival companies cannot know what kind of policy the company used to hire its employees.

In the ABE schemes, *collusion-resistance* is an important property. We do not want the secret key holders to be able to combine their secret keys to decrypt ciphertexts neither of them can decrypt. By building on the previous schemes [2, 11], we can also realize collusion-resistant CP-ABE schemes.

Our Results. We construct two CP-ABE schemes with partially hidden ciphertext policies in the sense that possible values of each attribute in the system should be known to an encryptor in advance and the encryptor can hide what subset of possible values for each attribute in the ciphertext policy can be used for successful decryption. In our schemes, encryptors can use wildcards to mean that certain attributes are not relevant to the ciphertext policy in a hidden way. The security proof of our first construction is given under the Decisional Bilinear Diffie-Hellman assumption and the Decision Linear assumption. Since these assumptions are general, we can use a large variety of elliptic curves (including both asymmetric and symmetric bilinear pairings) to implement our first scheme though we use the symmetric notation for ease of exposition. The security proof of our second construction is given in the generic group model and the second construction needs DDH-hard groups, but with a property inherited from [2], the second construction is more flexible than the first construction in that new attributes can be added in the ciphertext policy securely with the existing public parameters being unchanged even after the system setup is done. We mention this aspect in Sect. 5 in more detail. We describe our constructions in the multi-valued attribute setting where an attribute can take multiple values and this setting is a generalization of the access structures used in [11]. In our security notion, even the legitimate decryptor cannot obtain the information about the ciphertext policy more than the fact that she can decrypt the data.

Related Work. Bethencourt, Sahai, and Waters [2] proposed the first CP-ABE scheme. Their scheme allows the ciphertext policies to be very expressive, but the security proof is in the generic group model and the policies need to be revealed in the ciphertexts because decryptors must know how they should combine their secret key components for decryption. Cheung and Newport [11] proposed a provably secure CP-ABE scheme and their scheme deals with negative attributes explicitly and supports wildcards in the ciphertext policies but the policies need to be revealed as in [2]. Kapadia et al. [14] also proposed a CP-ABE scheme and their scheme realizes hidden ciphertext policies that have the same expressiveness as [11], but their scheme is not collusion-resistant and needs an online semi-trusted server that must know the attributes' values every user in the system has and re-encrypt ciphertexts for each user when the user retrieves the ciphertexts. Such an online semi-trusted server can be a performance bottleneck in the system while, in our schemes, encryptors can just post or broadcast ciphertexts. Lubicz and Sirvent [16] proposed another CP-ABE scheme that has the same expressiveness as [11] and only 3 pairing computations are needed for decryption, but the ciphertext policies need to be revealed for decryption. Shi et al. [21] proposed a predicate encryption scheme that focuses on range queries over huge numbers, the dual of which can also realize a CP-ABE scheme where an encryptor can specify a number range in the ciphertext policy. The security proof of [21] is based on the security notion weaker than ours, which is called *match-revealing security* in [21] and the number of attributes must be small because the decryption cost is exponential in the number of attributes. Boneh and Waters [5] proposed a predicate encryption scheme based on the primitive called *Hidden Vector Encryption* or HVE for short. The scheme in [5] can realize the same functionality as ours by using the opposite semantics of subset predicates described in [5] (see Appendix A for the details). However, it needs bilinear groups the order of which is a product of two large primes, so it needs to deal with large group elements and the numbers of both attributes and possible values for each attribute specified in the ciphertext policy are fixed at the system setup while, in our constructions, the number of possible attribute values in the ciphertext policy can be increased and furthermore in our second construction, the number of attributes in the ciphertext policy can be increased securely even after the system setup with the existing public parameters being unchanged.

Recently, Katz, Sahai, and Waters [15] proposed a novel predicate (or *functional*) encryption scheme supporting *inner product* predicates and their scheme is very general and can realize both key-policy and ciphertext-policy ABE schemes. Their scheme can also realize hidden ciphertext policies that can be more expressive than ours. However, their scheme is based on a special type of bilinear groups the order of which is a product of three (or two) large primes while ours are not. Therefore, their scheme needs to deal with large group elements and requires new complexity assumptions for the security proof. By using the dual of the predicate corresponding to polynomial evaluation, the scheme in [15] can realize the same access structure of ciphertext policies that our schemes can support (see Appendix B for the details) and then the ciphertext size of our schemes $O(\sum_{i=1}^{n} n_i)$ is comparable to that of [15] where n is the number of attributes in ciphertext policies and n_i is the number of possible values for each attribute i. Fox example, if attribute i is boolean, $n_i = 2$. In the CP-ABE

Table 1. Comparison of different schemes

	Expressiveness of policy	Anonymity	Complexity assumption	Type of bilinear group	Add attrs after setup
[5]	**AND**-gates on multi-valued attributes with wildcards	yes	cDBDH, C3DH	group of composite order $N = pq$	no
[2]	all boolean formula	no	generic group model	any	yes
[11]	**AND**-gates on postive and negative attributes with wildcards	no	DBDH	any	no
[15]	all boolean formula	yes	new assumptions based on composite order group	group of composite order $N = pqr$	no
This work1	**AND**-gates on multi-valued attributes with wildcards	yes	DBDH, D-Linear	any	no
This work2	**AND**-gates on multi-valued attributes with wildcards	yes	generic group model	DDH-hard group	yes

scheme of [15], the maximum size of the subset of attribute values for each attribute specified in the ciphertext policy for successful decryption is fixed at the system setup while, in our constructions, the size can be increased. Also, the number of attributes specified in the ciphertext policy is fixed at the system setup while, in our second construction, the number of attributes in the ciphertext policy can be increased securely even after the system setup with the existing public parameters being unchanged. However, when the number of possible attribute values is huge, the scheme in [15] is more advantageous than ours because it can enjoy the smaller ciphertext size and still realize the wildcard functionality.

Chase [10] proposed a multi-authority ABE where multiple authorities generate secret keys for their monitored attributes. The technique of [10] can be applicable to our schemes too. Abdalla et al. [1] proposed an identity-based encryption scheme where an encryptor can use wildcards to specify recipients of the ciphertext, but the positions of the wildcards and other ID components need to be revealed in the ciphertexts.

We summarize the comparison of major different schemes in Table 1.

2 Preliminaries

2.1 Bilinear Maps

We assume that there is an efficient algorithm $\boldsymbol{Gen}$ for generating bilinear groups. The algorithm $\boldsymbol{Gen}$, on input a security parameter 1^{κ}, outputs a tuple, $\boldsymbol{G} = [p, \mathbb{G}, \mathbb{G}_T, g \in \mathbb{G}, \boldsymbol{e}]$ where $\log_2(p) = \Theta(\kappa)$. A function $\boldsymbol{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear map. Here, $\mathbb{G}$ and $\mathbb{G}_T$ are multiplicative groups of prime order p, generated by g and $\boldsymbol{e}(g, g)$ respectively. The bilinear map $\boldsymbol{e}$ has the following properties:

1. Bilinearity: for all $a, b \in \mathbb{Z}_p$, $\boldsymbol{e}(g^a, g^b) = \boldsymbol{e}(g, g)^{ab}$.
2. Non-degeneracy: $\boldsymbol{e}(g, g) \neq 1$.

2.2 Complexity Assumptions

We describe complexity assumptions used in our security proofs.

2.2.1 The Decisional Bilinear Diffie-Hellman (DBDH) Assumption

We use the decisional version of the bilinear DH assumption [4,13]. Let $z_1, z_2, z_3, z \in \mathbb{Z}_p^*$ be chosen at random and $g \in \mathbb{G}$ be a generator. The DBDH assumption is that no probabilistic polynomial-time algorithm can distinguish the tuple $[g, g^{z_1}, g^{z_2}, g^{z_3}, \boldsymbol{e}(g, g)^{z_1 z_2 z_3}]$ from the tuple $[g, g^{z_1}, g^{z_2}, g^{z_3}, \boldsymbol{e}(g, g)^{z}]$ with non-negligible advantage.

2.2.2 The Decision Linear (D-Linear) Assumption

The D-Linear assumption was first proposed in [3]. Let $z_1, z_2, z_3, z_4, z \in \mathbb{Z}_p^*$ be chosen at random and $g \in \mathbb{G}$ be a generator. The D-Linear assumption is that no probabilistic polynomial-time algorithm can distinguish the tuple $[g, g^{z_1}, g^{z_2}, g^{z_1 z_3}, g^{z_2 z_4}, g^{z_3 + z_4}]$ from the tuple $[g, g^{z_1}, g^{z_2}, g^{z_1 z_3}, g^{z_2 z_4}, g^{z}]$ with non-negligible advantage.

2.3 Access Structure for Ciphertext

In the CP-ABE scheme, an encryptor specifies an access structure for a ciphertext, which is called a ciphertext policy. If a decryptor has a secret key whose associated set of attributes satisfies the access structure, she can decrypt the ciphertext. The access structures used in [2] are the most flexible and expressive. For example, we can use an access structure such as ((A **AND** B) **OR** (C **AND** D)) in [2]. This means that a recipient must have attributes A and B simultaneously or attributes C and D simultaneously in order to decrypt the ciphertext. Therefore, if a recipient has a secret key associated with a set of attributes {A, B, C}, she can satisfy the access structure and decrypt the ciphertext. However, if the recipient has a secret key associated with a set of attributes {A, C}, she can not satisfy the access structure or decrypt the ciphertext. Actually **AND**, **OR**, and threshold gates can be used for expressing the access structures in [2].

However, the security proof of [2] is in the *generic group model*. In order to obtain a reduction-based security proof, Cheung and Newport proposed another CP-ABE scheme [11] which is proved to be secure under *standard* complexity assumptions. The price of obtaining such security proofs is that the expressiveness of ciphertext policies in [11] is somewhat restricted as compared with [2]. However, the expressiveness is not too restrictive and still remains useful.

The access structure and the attribute set associated with the secret key used in [11] are as follows. Let's assume that the total number of attributes in the system is n and the attributes are indexed as $\{\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_i, \ldots, \mathbb{A}_n\}$ or we may use just i to refer to $\mathbb{A}_i$. We use the notation such as $L = [L_1, \ldots, L_n] = [1, 0, 1, \ldots, 0]$ in order to describe attribute-value pairs for a user, which we call the attribute list. For example, the user has the value 1 for $\mathbb{A}_1$, 0 for $\mathbb{A}_2$, 1 for $\mathbb{A}_3$, ..., and 0 for $\mathbb{A}_n$ in this case. A trusted authority generates a secret key for the user based on the user's attribute list.

In order to specify the access structure for a ciphertext, we use the notation such as $W = [W_1, \ldots, W_n] = [1, 1, *, *, 0]$ where $n = 5$, which we call the ciphertext policy. The wildcard $*$ can be used in the ciphtertext policies and it plays the role of "don't care" value. This can be considered as an **AND**-gate on all the attributes. For example, the above ciphertext policy means that the recipient who wants to decrypt must have the value 1 for $\mathbb{A}_1$ and $\mathbb{A}_2$ and 0 for $\mathbb{A}_5$, and the values for $\mathbb{A}_3$ and $\mathbb{A}_4$ do not matter in the **AND**-gate. If the recipient has the secret key associated with, let us say, $[1, 1, 1, 0, 0]$, she can decrypt the ciphertext, but not if the secret key is associated with $[1, 1, 1, 0, 1]$.

Formally, given an attribute list $L = [L_1, L_2, \ldots, L_n]$ and a ciphertext policy $W = [W_1, W_2, \ldots, W_n]$, L satisfies W if, for all $i = 1, \ldots, n$, $L_i = W_i$ or $W_i = *$, and otherwise L does not satisfies W. We use the notation $L \models W$ to mean that L satisfies W.

In our constructions, we generalize the access structures in [11]. In [11], each attribute can take two values 1 and 0, but in our generalized access structures each attribute can take two or more values and each W_i in W can be any subset of possible values for $\mathbb{A}_i$. More formally, let $S_i = \{v_{i,1}, v_{i,2}, \ldots, v_{i,t}, \ldots, v_{i,n_i}\}$ be a set of possible values for $\mathbb{A}_i$ where n_i is the number of the possible values for $\mathbb{A}_i$. Then the attribute list L for a user is $L = [L_1, L_2, \ldots, L_i, \ldots, L_n]$ where $L_i \in S_i$ and the generalized ciphertext policy W is $W = [W_1, W_2, \ldots, W_i, \ldots, W_n]$ where $W_i \subseteq S_i$. The generalized ciphertext policy W means, let us say,

$$\begin{aligned}
&(\mathbb{A}_1 = v_{1,1} \vee \mathbb{A}_1 = v_{1,3}) \\
&\quad \wedge (\mathbb{A}_2 = v_{2,2}) \wedge \ldots \\
&\quad \wedge (\mathbb{A}_i = v_{i,5} \vee \ldots \vee \mathbb{A}_i = v_{i,n_i}) \wedge \ldots \\
&\quad \wedge (\mathbb{A}_n = v_{n,1} \vee \mathbb{A}_n = v_{n,2} \vee \mathbb{A}_n = v_{n,3}).
\end{aligned}$$

When the encryptor specifies a wildcard for $\mathbb{A}_i$, it corresponds to specifying $W_i = S_i$ for $\mathbb{A}_i$. The attribute list L satisfies the ciphertext policy W iff $L_i \in W_i$ for $1 \leq i \leq n$. We achieve recipient anonymity by hiding what subset W_i for each $\mathbb{A}_i$ is specified in the access structure of the **AND**-gate of all the attributes.

2.4 Syntax of CP-ABE

Our CP-ABE schemes consist of the following four algorithms.

***Setup*(1^κ).** This algorithm takes the security parameter κ as input and generates a public key PK and a master secret key MK.

***KeyGen*(MK, L).** This algorithm takes MK and an attribute list L as input and generates a secret key SK_L associated with L.

***Encrypt*(PK, M, W).** This algorithm takes PK, a message M, and an ciphertext policy W as input, and generates a ciphertext CT.

***Decrypt*(CT, SK_L).** This algorithm takes CT and SK_L associated with L as input and returns the message M if the attribute list L satisfies the ciphertext policy W specified for CT, that is, $L \models W$. If $L \not\models W$, it returns $\perp$ with overwhelming probability.

2.5 Security Model

We describe the security models for our CP-ABE. Based on [21, 5, 15], we use the following security game. A CP-ABE scheme is selectively secure if no probabilistic polynomial-time adversary has non-negligible advantage in the following game.

Selective Game for CP-ABE

Init: The adversary commits to the challenge ciphertext policies W_0, W_1.

Setup: The challenger runs the ***Setup*** algorithm and gives PK to the adversary.

Phase 1: The adversary submits the attribute list L for a ***KeyGen*** query. If $(L \models W_0 \wedge L \models W_1)$ or $(L \not\models W_0 \wedge L \not\models W_1)$, the challenger gives the adversary the secret key SK_L. The adversary can repeat this polynomially many times.

Challenge: The adversary submits messages M_0, M_1 to the challenger. If the adversary obtained the SK_L whose associated attribute list L satisfies both W_0 and W_1 in Phase 1, then it is required that $M_0 = M_1$. The challenger flips a random coin b and passes the ciphertext ***Encrypt***$(PK,\ M_b,\ W_b)$ to the adversary.

Phase 2: Phase 1 is repeated. If $M_0 \neq M_1$, the adversary cannot submit L such that $L \models W_0 \wedge L \models W_1$.

Guess: The adversary outputs a guess b' of b.

The advantage of an adversary in this game is defined as $\left|\Pr[b' = b] - \frac{1}{2}\right|$ where the probability is taken over the random bits used by the challenger and the adversary. Since the adversary must commit to the challenge ciphertext policies before the setup phase, this model can be considered to be analogous to the selective-ID model [8, 9] used in identity-based encryption schemes. The non-selective-ID model can be found in [2] where the proof is in the generic group model. In the game, the adversary can submit L such that $L \models W_0$ and $L \models W_1$ if possible and then the adversary can decrypt the ciphertext. This definition captures that the adversary cannot obtain the useful information about the ciphertext policy more than the fact that she can decrypt the ciphertext. The above notion of security is called *match-concealing security* in [21].

3 Proposed Schemes

We construct two CP-ABE schemes that achieve recipient anonymity. In [11], the ciphertext policy needs to be revealed in the ciphertext so that a decryptor can know which secret key components should be used. Furthermore, in order to support wildcards for ciphertext policies, the public key components for the wildcards are prepared in [11] and the decryptor uses the secret key components corresponding to the wildcards if the wildcards are specified in the ciphertext policies. In our constructions, we can hide how the ciphertext policy is specified successfully.

First we show the construction of [11] and later explain the intuition behind our approach we take to make it recipient-anonymous. We assume, for notational simplicity, that the total number of attributes in the system is n and the attributes are indexed as $\{1, 2, \ldots, i, \ldots, n\}$.

3.1 Construction of [11]

The four algorithms are as follows:

Setup(1^κ)**.** A trusted authority generates a tuple $\boldsymbol{G} = [p, \mathbb{G}, \mathbb{G}_T, g \in \mathbb{G}, \boldsymbol{e}] \leftarrow$ ***Gen***(1^κ), and random $w \in \mathbb{Z}_p^*$. For each attribute i where $1 \leq i \leq n$, the authority generates random values $a_i, \widehat{a}_i, a_i^*, \in \mathbb{Z}_p^*$. The authority computes $Y = \boldsymbol{e}(g,g)^w$ and $A_i = g^{a_i}, \widehat{A}_i = g^{\widehat{a}_i}, A_i^* = g^{a_i^*}$. The public key PK consists of $\langle Y, p, \mathbb{G}, \mathbb{G}_T, g, \boldsymbol{e}, \{A_i, \widehat{A}_i, A_i^*\}_{1 \leq i \leq n} \rangle$. The master secret key MK is $\langle w, \{a_i, \widehat{a}_i, a_i^*\}_{1 \leq i \leq n} \rangle$.

KeyGen(MK, L)**.** Let $L = [L_1, L_2, \ldots, L_n]$ be the attribute list for the user who will obtain the corresponding secret key. The trusted authority picks up random values $s_i \in \mathbb{Z}_p^*$ for $1 \leq i \leq n$, sets $s = \sum_{i=1}^n s_i$, and computes $D_0 = g^{w-s}$. For $1 \leq i \leq n$, the authority also computes $[D_i, D_i^*] = [g^{s_i/a_i}, g^{s_i/a_i^*}]$ if $L_i = 1$, and $[D_i, D_i^*] = [g^{s_i/\widehat{a}_i}, g^{s_i/a_i^*}]$ if $L_i = 0$. The secret key SK_L is $\langle D_0, \{D_i, D_i^*\}_{1 \leq i \leq n} \rangle$.

Encrypt(PK, M, W)**.** An encryptor encrypts a message $M \in \mathbb{G}_T$ under a ciphertext policy $W = [W_1, W_2, \ldots, W_n]$. The encryptor picks up a random value $r \in \mathbb{Z}_p^*$ and sets $\widetilde{C} = MY^r$ and $C_0 = g^r$. Also for $1 \leq i \leq n$, the encryptor computes C_i as follows: if $W_i = 1$, $C_i = A_i^r$; if $W_i = 0$, $C_i = \widehat{A}_i^r$; if $W_i = *$, $C_i = A_i^{*r}$. The ciphertext CT is $\langle \widetilde{C}, C_0, \{C_i\}_{1 \leq i \leq n} \rangle$. The encryptor needs to reveal W in CT so that recipients can know which secret key components should be used for each C_i.

Note that if W is hidden in CT, the recipients need to try all the possible combinations of the secret key components for decryption and it takes exponential time in n, which seems impractical.

***Decrypt*(CT, SK_L).** The recipient can check W to know whether $L \models W$. If $L \models W$, she can proceed. The recipient decrypts the CT, $\langle \widetilde{C}, C_0, \{C_i\}_{1 \le i \le n} \rangle$ by using her SK_L, $\langle D_0, \{D_i, D_i^*\}_{1 \le i \le n} \rangle$ associated with the attribute list L, as follows:

1. For $1 \le i \le n$,
$$D_i' = \begin{cases} D_i & \text{if } W_i \neq * \\ D_i^* & \text{if } W_i = *. \end{cases}$$
2. $$M = \frac{\widetilde{C}}{\boldsymbol{e}(C_0, D_0) \prod_{i=1}^{n} \boldsymbol{e}(C_i, D_i')}.$$

3.2 Main Idea

We describe how to make the construction of [11] recipient-anonymous. To achieve our goal, the recipients need to be able to decrypt CT without knowing W and we also want to support wildcards in a hidden way. For that, we remove the public key components $\{A_i^*\}_{1 \le i \le n}$ for the wildcards and the secret key components $\{D_i^*\}_{1 \le i \le n}$ are not included in SK_L. Furthermore, instead of the ciphertext components $\{C_i\}_{1 \le i \le n}$, $\{C_i, \widehat{C}_i\}_{1 \le i \le n}$ are generated with $C_0 = g^r$ as follows: let $\{C_i, \widehat{C}_i\} = \{A_i^{r_1}, \widehat{A}_i^{r_2}\}$; if $W_i = 1$, we set $r_1 = r$ and r_2 is random; if $W_i = 0$, r_1 is random and $r_2 = r$; if $W_i = *$, $r_1 = r_2 = r$. That is, if $C_i = A_i^r$ or $\widehat{C}_i = \widehat{A}_i^r$, these ciphertext components are "*well-formed*" and can be used for successful decryption and otherwise "*malformed*" (or random). Each decryptor uses C_i for decryption if $L_i = 1$ and uses $\widehat{C}_i$ if $L_i = 0$ without knowing what is specified for W_i. By generating the ciphertext like this, we can realize the functionality of wildcards. We can generalize this idea to adapt to the multi-valued attribute setting.

Finally to make it hard to distinguish the well-formed components from the malformed components, we use the linear splitting technique in [6,21] and make our first construction provably secure as shown in Sect. 4.

3.3 Our First Construction

The four algorithms are as follows:

***Setup*(1^κ).** A trusted authority generates a tuple $\boldsymbol{G} = [p, \mathbb{G}, \mathbb{G}_T, g \in \mathbb{G}, \boldsymbol{e}] \leftarrow$ ***Gen*(1^κ)** and random $w \in \mathbb{Z}_p^*$. For each attribute i where $1 \le i \le n$, the authority generates random values $\{a_{i,t}, b_{i,t} \in \mathbb{Z}_p^*\}_{1 \le t \le n_i}$ and random points $\{A_{i,t} \in \mathbb{G}\}_{1 \le t \le n_i}$ [1]. The authority computes $Y = \boldsymbol{e}(g, g)^w$. The public key

[1] In the asymmetric bilinear groups, $A_{i,t}$ must be generated such that $A_{i,t} = g^{c_{i,t}}$ where $c_{i,t} \in_R \mathbb{Z}_p^*$ and $c_{i,t}$ is known to the authority so that the authority can use $c_{i,t}$ in ***KeyGen***.

PK consists of $\langle Y, p, \mathbb{G}, \mathbb{G}_T, g, \boldsymbol{e}, \{\{A_{i,t}^{a_{i,t}}, A_{i,t}^{b_{i,t}}\}_{1 \le t \le n_i}\}_{1 \le i \le n}\rangle$. The master secret key MK is $\langle w, \{\{a_{i,t}, b_{i,t}\}_{1 \le t \le n_i}\}_{1 \le i \le n}\rangle$.

***KeyGen*(MK, L).** Let $L = [L_1, L_2, \ldots, L_n] = [v_{1,t_1}, v_{2,t_2}, \ldots, v_{n,t_n}]$ be the attribute list for the user who obtains the corresponding secret key. The trusted authority picks up random values $s_i, \lambda_i \in \mathbb{Z}_p^*$ for $1 \le i \le n$, sets $s = \sum_{i=1}^{n} s_i$, and computes $D_0 = g^{w-s}$. For $1 \le i \le n$, the authority also computes $[D_{i,0}, D_{i,1}, D_{i,2}] = [g^{s_i}(A_{i,t_i})^{a_{i,t_i} b_{i,t_i} \lambda_i}, g^{a_{i,t_i}\lambda_i}, g^{b_{i,t_i}\lambda_i}]$ where $L_i = v_{i,t_i}$. The secret key SK_L is $\langle D_0, \{\{D_{i,j}\}_{0 \le j \le 2}\}_{1 \le i \le n}\rangle$.

***Encrypt*(PK, M, W).** An encryptor encrypts a message $M \in \mathbb{G}_T$ under a ciphertext policy $W = [W_1, W_2, \ldots, W_n]$. The encryptor picks up a random value $r \in \mathbb{Z}_p^*$ and sets $\widetilde{C} = MY^r$ and $C_0 = g^r$. Also for $1 \le i \le n$, the encryptor picks up random values $\{r_{i,t} \in \mathbb{Z}_p^*\}_{1 \le t \le n_i}$ and computes $\{C_{i,t,1}, C_{i,t,2}\}_{1 \le t \le n_i}$ as follows: if $v_{i,t} \in W_i$, $[C_{i,t,1}, C_{i,t,2}] = [(A_{i,t}^{b_{i,t}})^{r_{i,t}}, (A_{i,t}^{a_{i,t}})^{r - r_{i,t}}]$ (*well-formed*); if $v_{i,t} \notin W_i$, $[C_{i,t,1}, C_{i,t,2}]$ are random (*mal-formed*). The ciphertext CT is $\langle \widetilde{C}, C_0, \{\{C_{i,t,1}, C_{i,t,2}\}_{1 \le t \le n_i}\}_{1 \le i \le n}\rangle$.

***Decrypt*(CT, SK_L).** The recipient tries decrypting the CT, $\langle \widetilde{C}, C_0, \{\{C_{i,t,1}, C_{i,t,2}\}_{1 \le t \le n_i}\}_{1 \le i \le n}\rangle$ without knowing W by using her SK_L, $\langle D_0, \{\{D_{i,j}\}_{0 \le j \le 2}\}_{1 \le i \le n}\rangle$ associated with the attribute list L, as follows:

1. For $1 \le i \le n$,

$$[C'_{i,1}, C'_{i,2}] = [C_{i,t_i,1}, C_{i,t_i,2}] \text{ where } L_i = v_{i,t_i}.$$

2.

$$M = \frac{\widetilde{C} \prod_{i=1}^{n} \boldsymbol{e}(C'_{i,1}, D_{i,1}) \boldsymbol{e}(C'_{i,2}, D_{i,2})}{\boldsymbol{e}(C_0, D_0) \prod_{i=1}^{n} \boldsymbol{e}(C_0, D_{i,0})}.$$

If the attribute list L satisfies the hidden ciphertext policy W of the CT, the recipient can decrypt the CT correctly. For the recipient to know whether the decryption was successful without knowing the ciphertext policy W, we can use the technique used in [5] in practice. As in [5], the encryptor picks up a random $k \in \mathbb{G}_T$ and derives two uniform and independent μ-bit symmetric keys (k_0, k_1) from k. The final ciphertext consists of $\langle k_1, \textbf{\textit{Encrypt}}(PK, k, W), E_{k_0}(M)\rangle$ where $\textbf{\textit{Encrypt}}(PK, k, W)$ is the ciphertext of k encrypted under PK and W, and $E_{k_0}(M)$ is the ciphertext of M encrypted under k_0 by using a symmetric encryption scheme. The recipient can use k_1 to check whether the decryption was successful after decrypting k where the false positive probability is approximately $1/2^{\mu}$. If successful, the recipient can decrypt M by using k_0 derived from k. The security proof is given in Sect. 4.

3.4 Second Construction with More Flexibility

We can also apply the technique in Sect. 3.2 to [2] and make it recipient-anonymous. With a property inherited from [2], this scheme is more flexible though the security proof is in the generic group model. The scheme in [2] uses a symmetric bilinear group while we use an asymmetric bilinear group. That is, we assume $\boldsymbol{Gen}(1^\kappa)$ outputs $\boldsymbol{G} = [p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1 {\in} \mathbb{G}_1, g_2 {\in} \mathbb{G}_2, \boldsymbol{e}]$ where $\boldsymbol{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is a bilinear map. We also use the External Diffie-Hellman (XDH) assumption used in, for example, [3, 20, 7] to achieve recipient anonymity, which holds on MNT curves [17]. In the XDH assumption, it holds that the Decisional Diffie-Hellman (DDH) problem is hard in $\mathbb{G}_1$ and this implies that there does not exist an efficiently-computable isomorphism $\psi : \mathbb{G}_1 \rightarrow \mathbb{G}_2$. The four algorithms are as follows:

Setup(1^κ)**.** A trusted authority generates a tuple $\boldsymbol{G} = [p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2, \boldsymbol{e}]$. and random $w, \beta \in \mathbb{Z}_p^*$. For each attribute i where $1 \leq i \leq n$, the authority generates random values $\{a_{i,t} \in \mathbb{Z}_p^*\}_{1 \leq t \leq n_i}$ and computes points $\{A_{i,t} = g_1^{a_{i,t}}\}_{1 \leq t \leq n_i}$. The authority computes $Y = \boldsymbol{e}(g_1, g_2)^w$ and $B = g_1^\beta$. The public key PK consists of $\langle Y, B, p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, \boldsymbol{e}, \{\{A_{i,t}\}_{1 \leq t \leq n_i}\}_{1 \leq i \leq n}\rangle$. The master secret key MK is $\langle w, \beta, \{\{a_{i,t}\}_{1 \leq t \leq n_i}\}_{1 \leq i \leq n}\rangle$.

KeyGen$(MK,\ L)$**.** Let $L = [L_1, L_2, \ldots, L_n] = [v_{1,t_1}, v_{2,t_2}, \ldots, v_{n,t_n}]$ be the attribute list for the user who obtains the corresponding secret key. The trusted authority picks up random values $s, \lambda_i \in \mathbb{Z}_p^*$ for $1 \leq i \leq n$ and computes $D_0 = g_2^{\frac{w+s}{\beta}}$. For $1 \leq i \leq n$, the authority also computes $[D_{i,1}, D_{i,2}] = [g_2^{s + a_{i,t_i}\lambda_i}, g_2^{\lambda_i}]$ where $L_i = v_{i,t_i}$. The secret key SK_L is $\langle D_0, \{D_{i,1}, D_{i,2}\}_{1 \leq i \leq n}\rangle$.

Encrypt$(PK,\ M,\ W)$**.** An encryptor encrypts a message $M \in \mathbb{G}_T$ under a ciphertext policy $W = [W_1, W_2, \ldots, W_n]$. The encryptor picks up a random value $r \in \mathbb{Z}_p^*$ and sets $\widetilde{C} = MY^r$ and $C_0 = B^r$. Also for $1 \leq i \leq n$, the encryptor picks up random values $r_i \in \mathbb{Z}_p^*$ such that $r = \sum_{i=1}^n r_i$, sets $C_{i,1} = g_1^{r_i}$ and computes $\{C_{i,t,2}\}_{1 \leq t \leq n_i}$ as follows: if $v_{i,t} \in W_i$, $C_{i,t,2} = A_{i,t}^{r_i}$ (*well-formed*); if $v_{i,t} \notin W_i$, $C_{i,t,2}$ is random (*malformed*). The ciphertext CT is $\langle \widetilde{C}, C_0, \{C_{i,1}, \{C_{i,t,2}\}_{1 \leq t \leq n_i}\}_{1 \leq i \leq n}\rangle$.

Decrypt$(CT,\ SK_L)$**.** The recipient decrypts the CT, $\langle \widetilde{C}, C_0, \{C_{i,1}, \{C_{i,t,2}\}_{1 \leq t \leq n_i}\}_{1 \leq i \leq n}\rangle$ by using her SK_L, $\langle D_0, \{D_{i,1}, D_{i,2}\}_{1 \leq i \leq n}\rangle$ associated with the attribute list L as follows:

1. For $1 \leq i \leq n$,
$$C'_{i,2} = C_{i,t_i,2} \text{ where } L_i = v_{i,t_i}.$$
2.
$$M = \frac{\widetilde{C} \prod_{i=1}^n \boldsymbol{e}(C_{i,1}, D_{i,1})}{\boldsymbol{e}(C_0, D_0) \prod_{i=1}^n \boldsymbol{e}(C'_{i,2}, D_{i,2})}.$$

Under the XDH assumption, it is hard to guess from CT what subset W_i the encryptor specified for each attribute $\mathbb{A}_i$ in the ciphertext policy. The security proof will be similar to that of [2] and is omitted due to space limitation. We discuss the flexibility of this scheme in Sect. 5 in more detail.

4 Overview of Security Proofs

We prove that our first scheme is selectively secure under the DBDH assumption and the D-Linear assumption. We will give the high-level arguments of the proofs here and the detailed proofs of the lemmas are given in Appendix C.

Suppose the adversary commits to the challenge ciphertext policies W_0, W_1 at the beginning of the game. We use the notation $W_b = [W_{b,1}, W_{b,2}, \ldots, W_{b,i}, \ldots, W_{b,n}]$.

The proof uses a sequence of hybrid games to argue that the adversary cannot win the original security game denoted by G with non-negligible probability. We begin by slightly modifying the game G into a game G_0. Games G and G_0 are the same except for how the challenge ciphertext is generated. In G_0, if the adversary did not obtain the SK_L whose associated attribute list L is such that $L \models W_0 \wedge L \models W_1$, then the challenge ciphtertext component $\widetilde{C}$ is a random element of $\mathbb{G}_T$ regardless of the random coin b. The rest of the ciphertext is generated as usual. If the adversary obtained the SK_L whose associated attribute list L is such that $L \models W_0 \wedge L \models W_1$ then the challenge ciphertext in G_0 is generated correctly. That is, $G = G_0$ in this case.

Lemma 1 *Under the DBDH assumption, for any polynomial time adversary $\mathcal{A}$, the difference of advantage of $\mathcal{A}$ in game G and game G_0 is negligible in the security parameter κ.*

Next, we modify G_0 by changing how to generate the ciphertext components $\{\{C_{i,t,1}, C_{i,t,2}\}_{1 \le t \le n_i}\}_{1 \le i \le n}$ and define a sequence of games as follows.

For $v_{i,t}$ such that $(v_{i,t} \in W_{0,i} \wedge v_{i,t} \in W_{1,i})$ or $(v_{i,t} \notin W_{0,i} \wedge v_{i,t} \notin W_{1,i})$, the components $\{C_{i,t,1}, C_{i,t,2}\}$ are generated as in the real scheme through the sequence of all the games.

If there is $v_{i,t}$ such that $(v_{i,t} \in W_{0,i} \wedge v_{i,t} \notin W_{1,i})$ or $(v_{i,t} \notin W_{0,i} \wedge v_{i,t} \in W_{1,i})$, the components $\{C_{i,t,1}, C_{i,t,2}\}$ generated properly in game $G_{\ell-1}$ are replaced with the random values in the new modified game G_ℓ regardless of the random coin b. Every time we replace such components $\{C_{i,t,1}, C_{i,t,2}\}$ with the random values, we define a new modified game. We repeat this replacement one by one until we have no component that satisfies $(v_{i,t} \in W_{0,i} \wedge v_{i,t} \notin W_{1,i})$ or $(v_{i,t} \notin W_{0,i} \wedge v_{i,t} \in W_{1,i})$. In the last game of the sequence, the advantage of the adversary is zero because the adversary is given a ciphertext chosen from the same distribution regardless of the random coin b.

By replacing the well-formed ciphertext components in $G_{\ell-1}$ with the random values in G_ℓ in this way, we can embed a D-Linear challenge in the ciphertext such that the distinguisher of $G_{\ell-1}$ and G_ℓ leads to the distinguisher of the D-Linear challenge.

Lemma 2 *Under the D-Linear assumption, for any polynomial time adversary $\mathcal{A}$, the difference of advantage of $\mathcal{A}$ in game $G_{\ell-1}$ and game G_ℓ is negligible in the security parameter κ.*

By considering the sequence $G, G_0, G_1, \ldots$ of games starting with the original game G, no polynomial adversary can win the game G with non-negligible advantage by the lemmas above.

5 Adding Attributes After *Setup*

In our schemes, it is easy to add new possible values $v_{i,t}$'s of each attribute $\mathbb{A}_i$ in the ciphertext policy even after ***Setup*** is executed, because we have only to add the public key components for the new values of $\mathbb{A}_i$ and the existing public parameters can remain unchanged. That is, the access structure for the ciphertext policy can be extended accordingly though the ciphertext size is also increased. However, in our first scheme, it cannot be done securely to simply add new attributes $\mathbb{A}_{i'}$'s in the ciphertext policy with the existing public parameters being unchanged after ***Setup*** is executed and some users already have their secret keys. The reason is as follows. Suppose there are three attributes $\mathbb{A}_1, \mathbb{A}_2, \mathbb{A}_3$ in the system when ***Setup*** is executed and a user obtains her secret key SK_L where the attribute list $L = [L_1, L_2, L_3] = [1, 1, 0]$. After that, a new attribute $\mathbb{A}_4$ is added in the system and the corresponding public key components for $\mathbb{A}_4$ are generated and published. Then an encryptor may specify a ciphertext policy $W = [W_1, \ldots, W_4] = [*, *, 0, 1]$, requiring the legitimate recipients to have the value 1 for $\mathbb{A}_4$. In this case, the user who has the above SK_L can decrypt the ciphertext encrypted under the ciphertext policy W even if she does not have the secret key component for $\mathbb{A}_4$, because L satisfies $[W_1, W_2, W_3]$ partially and it enables the user to combine all the secret key components to reconstruct $s = \sum_{i=1}^{n} s_i$ in the exponent for decryption. The similar situations can also happen in [11, 15, 5, 21] if we consider the setting where new attributes may be added in the ciphertext policy dynamically after ***Setup*** is executed. As mentioned in [18], we may be able to prepare redundant filler attributes reserved for future use, but it increases the ciphertext size unnecessarily.

The second scheme can avoid this situation with the property inherited from [2] and we can add new attributes in the ciphertext policy securely after ***Setup*** is executed where the existing public parameters can remain unchanged. Note that in this scheme, the encryptor splits random r in the ciphertext CT such that $r = \sum_{i=1}^{n} r_i$ and it forces decryptors to have the secret key components for all the attributes specified in the ciphertext policy even if the attributes in the ciphertext policy were added after the decryptors obtained their secret keys. If a user wants to decrypt the ciphertext with the ciphertext policy including newly added attributes, she must obtain a new secret key including the newly added attributes from the trusted authority again.

Additionally, in the second scheme, an encryptor can specify a variable-length ciphertext policy. For example, the encryptor can specify the ciphertext policy $W = [W_{i_1}, W_{i_2}, \ldots, W_{i_m}]$ where $m < n$ and n is the number of all the attributes in the system. Since there are several attributes that do not appear in the ciphertext policy, the partial information on the ciphertext policy is leaked. That is, it means that the wildcards are specified for the attributes not appearing in the ciphertext policy. However, it may be acceptable to the encryptor in some cases and it can reduce the size of the ciphertext.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments.

References

1. Abdalla, M., Catalano, D., Dent, A., Malone-Lee, J., Neven, G., Smart, N.: Identity-based encryption gone wild. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 300–311. Springer, Heidelberg (2006)
2. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: Proc. IEEE Symposium on Security and Privacy, pp. 321–334 (2007)
3. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
4. Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
5. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007)
6. Boyen, X., Waters, B.: Anonymous hierarchical identity-based encryption (without random oracles). In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 290–307. Springer, Heidelberg (2006)
7. Camenisch, J., Hohenberger, S., Lysyanskaya, A.: Compact e-cash. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 302–321. Springer, Heidelberg (2005)
8. Canetti, R., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 255–271. Springer, Heidelberg (2003)
9. Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)
10. Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007)
11. Cheung, L., Newport, C.: Provably secure ciphertext policy ABE. In: Proc. ACM Conference on Computer and Communications Security (CCS), pp. 456–465 (2007)
12. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proc. ACM Conference on Computer and Communications Security (CCS), pp. 89–98 (2006)
13. Joux, A.: A one round protocol for tripartite Diffie-Hellman. In: Bosma, W. (ed.) ANTS 2000. LNCS, vol. 1838, pp. 385–394. Springer, Heidelberg (2000)
14. Kapadia, A., Tsang, P.P., Smith, S.W.: Attribute-based publishing with hidden credentials and hidden policies. In: Proc.Network & Distributed System Security Symposium (NDSS), pp. 179–192 (2007)

15. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Eurocrypt 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
Cryptology ePrint Archive 2007/404 (2007)
16. Lubicz, D., Sirvent, T.: Attribute-based broadcast encryption scheme made efficient. In: Africacrypt 2008. LNCS, vol. 5023. Springer, Heidelberg (to appear, 2008)
17. Miyaji, A., Nakabayashi, M., Takano, S.: New explicit conditions of elliptic curve traces for FR-reductions. IEICE Trans., Fundamentals E84-A(5), 1234–1243 (2001)
18. Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: Proc. ACM Conference on Computer and Communications Security (CCS), pp. 195–203 (2007)
19. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
20. Scott, M.: Authenticated ID-based key exchange and remote log-in with simple token and PIN number. Cryptology ePrint Archive 2002/164 (2002)
21. Shi, E., Bethencourt, J., Chan, H., Song, D., Perrig, A.: Multi-dimensional range query over encrypted data. In: Proc. IEEE Symposium on Security and Privacy, pp. 350–364 (2007)

A Realization of CP-ABE with [5]

We show how the scheme in [5] can realize the access structure of the ciphertext policy considered in this work by using HVE. For ease of exposition, suppose there are two attributes $\mathbb{A}_1, \mathbb{A}_2$ in the system and $\mathbb{A}_1$ can take values $v_{1,1}, v_{1,2}$ and $\mathbb{A}_2$ can take values $v_{2,1}, v_{2,2}, v_{2,3}$. When an encryptor encrypts a message, the encryptor specifies a vector corresponding to $(v_{1,1}, v_{1,2}, v_{2,1}, v_{2,2}, v_{2,3})$ as a ciphertext policy. For example, if $(v_{1,1}, v_{1,2}, v_{2,1}, v_{2,2}, v_{2,3}) = (1, 0, 1, 0, 1)$, this means $(\mathbb{A}_1 = v_{1,1}) \wedge (\mathbb{A}_2 = v_{2,1} \vee \mathbb{A}_2 = v_{2,3})$. A decryptor with $\mathbb{A}_1 = v_{1,1} \wedge \mathbb{A}_2 = v_{2,3}$ obtains her secret key the vector of which corresponds to $(1, *, *, *, 1)$. The decryptor can decrypt the ciphertext if the vectors of both the ciphertext and the secret key match up except the wildcards. In this scheme, the length of the vectors (5 in the example above) is fixed at the system setup. Therefore, the numbers of both attributes and possible values for each attribute specified in the ciphertext policy are fixed at the system setup.

B Realization of CP-ABE with [15]

We show how the scheme in [15] can realize the access structure of the ciphertext policy considered in this work by using the dual of the predicate corresponding polynomial evaluation. Similarly, for ease of exposition, suppose there are two attributes $\mathbb{A}_1, \mathbb{A}_2$ in the system and $\mathbb{A}_1$ can take values $v_{1,1}, v_{1,2}$ and $\mathbb{A}_2$ can take values $v_{2,1}, v_{2,2}, v_{2,3}$. In this scheme, decryption succeeds if the vector for the ciphertext $(a_1, a_2, \ldots, a_n)$ and the vector for the secret key $(x_1, x_2, \ldots, x_n)$ satisfy the condition that $\sum_{i=1}^{n} a_i x_i = 0$.

When an encryptor encrypts a message with the ciphertext policy $(\mathbb{A}_1 = v_{1,1}) \wedge (\mathbb{A}_2 = v_{2,1} \vee \mathbb{A}_2 = v_{2,3})$, she prepares two polynomials $f_1(x) = c_1 x + c_0$

and $f_2(x) = d_2x^2 + d_1x + d_0$ such that $f_1(v_{1,1}) = 0, f_2(v_{2,1}) = 0$ and $f_2(v_{2,3}) = 0$ and specifies the vector $(c_1, c_0, d_2, d_1, d_0)$ as the ciphertext policy. A decryptor with $\mathbb{A}_1 = v_{1,1} \wedge \mathbb{A}_2 = v_{2,3}$ obtains her secret key the vector of which corresponds to $(v_{1,1}, 1, v_{2,3}^2, v_{2,3}, 1)$. For example, when the encryptor specifies a wildcard for attribute $\mathbb{A}_2$ in the ciphertext policy, she simply uses $f_2(x) = 0$ where $d_2 = d_1 = d_0 = 0$. In this scheme, the length of the vectors (5 in the example above) is fixed at the system setup. Therefore, the maximum size of the subset of attribute values for each attribute specified in the ciphertext policy for successful decryption is fixed at the system setup. Also, the number of attributes specified in the ciphertext policy is fixed at the system setup. However, when the number of possible attribute values is huge and the maximum size of the subset of attribute values specified in the ciphertext policy is small, the scheme in [15] is more advantageous than ours because it can enjoy the smaller ciphertext size and still realize the wildcard functionality.

C Proofs of Lemmas

C.1 Proof of Lemma 1

Proof. We prove our lemma by assuming that a polynomial adversary $\mathcal{A}$ has non-negligible difference ϵ between its advantage in game G and its advantage in game G_0. We build a simulator $\mathcal{B}$ that can play the DBDH game with advantage ϵ.

Given a DBDH challenge $[g, g^{z_1}, g^{z_2}, g^{z_3}, Z]$ by the challenger where Z is either $\boldsymbol{e}(g, g)^{z_1z_2z_3}$ or random with equal probability, the simulator $\mathcal{B}$ creates the following simulation.

Init: The simulator $\mathcal{B}$ runs $\mathcal{A}$. $\mathcal{A}$ gives $\mathcal{B}$ two challenge chiphertext policies $W_0 = [W_{0,1}, \ldots, W_{0,n}], W_1 = [W_{1,1}, \ldots, W_{1,n}]$. Then $\mathcal{B}$ flips a random coin $b \in \{0, 1\}$.

Setup: To provide a public key PK to $\mathcal{A}$, $\mathcal{B}$ sets Y to $\boldsymbol{e}(g, g)^{z_1z_2}$. This implies $w = z_1z_2$. For each attribute i where $1 \leq i \leq n$, $\mathcal{B}$ generates $\{A_{i,t}\}_{1 \leq t \leq n_i}$ such that $A_{i,t} = g^{\alpha_{i,t}}$ if $v_{i,t} \in W_{b,i}$ and $A_{i,t} = g^{z_1\alpha_{i,t}}$ if $v_{i,t} \notin W_{b,i}$ where $\{\alpha_{i,t} \in \mathbb{Z}_p^*\}_{1 \leq t \leq n_i}$ are random. Then $\mathcal{B}$ publishes public parameters as in the real scheme by picking up $\{a_{i,t}, b_{i,t}\}_{1 \leq t \leq n_i}$ at random for $1 \leq i \leq n$.

Phase 1: $\mathcal{A}$ submits an attribute list $L = [L_1, \ldots, L_n]$ in a secret key query. We consider only the case where $L \not\models W_0 \wedge L \not\models W_1$. The reason for this is by our definition if $L \models W_0 \wedge L \models W_1$, then the challenge messages M_0, M_1 will be equal. In this case, the games G and G_0 are the same, so there is no difference of advantage of $\mathcal{A}$ in G and G_0. Therefore, $\mathcal{B}$ simply aborts and takes a random guess.
When $L \not\models W_0 \wedge L \not\models W_1$, there must be $k \in \{1, \ldots, n\}$ such that $L_k(= v_{k,t_k}) \notin W_{b,k}$.
For $1 \leq i \leq n$, $\mathcal{B}$ picks up $s'_i \in \mathbb{Z}_p^*$ at random. It then sets $s_k = z_1z_2 + s'_k$ and for every $i \neq k$, sets $s_i = s'_i$. Finally it sets $s = \sum_{i=1}^{n} s_i = z_1z_2 + \sum_{i=1}^{n} s'_i$.

The D_0 component of the secret key can be computed as

$$D_0 = g^{w-s} = g^{z_1 z_2 - s} = g^{-\sum_{i=1}^{n} s'_i}.$$

For k, $\mathcal{B}$ computes the components $[D_{k,0}, D_{k,1}, D_{k,2}] = [g^{s_k}(A_{k,t_k})^{a_{k,t_k} b_{k,t_k} \lambda_k}, g^{a_{k,t_k}\lambda_k}, g^{b_{k,t_k}\lambda_k}]$ as follows:

$$\begin{aligned} D_{k,0} &= g^{s_k}(A_{k,t_k})^{a_{k,t_k} b_{k,t_k} \lambda_k} \\ &= g^{z_1 z_2 + s'_k}(A_{k,t_k})^{a_{k,t_k} b_{k,t_k} \lambda_k} \\ &= g^{z_1 z_2 + s'_k}(g^{z_1 \alpha_{k,t_k}})^{a_{k,t_k} b_{k,t_k} \lambda_k} \\ &= g^{s'_k}(g^{z_1 \alpha_{k,t_k}})^{a_{k,t_k} b_{k,t_k} \lambda'_k} \end{aligned}$$

where λ_k is chosen at random such that

$$\lambda_k = -\frac{z_2}{\alpha_{k,t_k} a_{k,t_k} b_{k,t_k}} + \lambda'_k$$

and random λ'_k is known to $\mathcal{B}$.
$\mathcal{B}$ can compute the components $[D_{k,1}, D_{k,2}]$ easily.
For $i \neq k$, $\mathcal{B}$ can also compute $[D_{i,0}, D_{i,1}, D_{i,2}]$ easily.

Challenge: $\mathcal{A}$ submits two challenge messages M_0 and M_1. $\mathcal{B}$ sets $\widetilde{C} = M_b Z$ and $C_0 = g^{z_3}$ which implies $r = z_3$ and generates, for W_b, the ciphertext $\langle \widetilde{C}, C_0, \{\{C_{i,t,1}, C_{i,t,2}\}_{1 \le t \le n_i}\}_{1 \le i \le n}\rangle$. When $v_{i,t} \in W_{b,i}$, $\mathcal{B}$ can generate $\{C_{i,t,1}, C_{i,t,2}\}$ correctly because $A_{i,t}$ does not contain unknown z_1 and when $v_{i,t} \notin W_{b,i}$, $\{C_{i,t,1}, C_{i,t,2}\}$ can be simply chosen at random.

Phase 2: Phase 1 is repeated.

Guess: $\mathcal{A}$ outputs a guess b' of b. If $b' = b$, $\mathcal{B}$ outputs 1 and otherwise outputs 0. By our assumption, the probability that $\mathcal{A}$ guesses b correctly in game G has a non-negligible ϵ difference from that of it guessing b correctly in G_0. When $Z = \boldsymbol{e}(g, g)^{z_1 z_2 z_3}$, $\mathcal{A}$ is in game G and when Z is random, $\mathcal{A}$ is in game G_0. Therefore the simulator $\mathcal{B}$ has advantage ϵ in the DBDH game.

□

C.2 Proof of Lemma 2

Proof. We prove our lemma by assuming that a polynomial adversary $\mathcal{A}$ has non-negligible difference ϵ between its advantage in game $G_{\ell-1}$ and its advantage in game G_ℓ. We build a simulator $\mathcal{B}$ that can play the D-Linear game with advantage ϵ.

Given a D-Linear challenge $[g, g^{z_1}, g^{z_2}, Z, g^{z_2 z_4}, g^{z_3 + z_4}]$ by the challenger where Z is either $g^{z_1 z_3}$ or random with equal probability, the simulator $\mathcal{B}$ creates the simulation. Note that this D-Linear assumption is equivalent to that of Sect. 2.2.2.

As mentioned in Sect. 4, in $G_{\ell-1}$, the ciphertext components $\{C_{i_\ell,t_\ell,1}, C_{i_\ell,t_\ell,2}\}$ are generated as in the real scheme, whereas, in G_ℓ, the components are random regardless of the random coin b and we assume that $(v_{i_\ell,t_\ell} \in W_{1,i_\ell} \wedge v_{i_\ell,t_\ell} \notin W_{0,i_\ell})$ without loss of generality.

Init: The simulator $\mathcal{B}$ runs $\mathcal{A}$. $\mathcal{A}$ gives $\mathcal{B}$ two challenge chiphertext policies $W_0 = [W_{0,1}, \ldots, W_{0,n}], W_1 = [W_{1,1}, \ldots, W_{1,n}]$. Then $\mathcal{B}$ flips a random coin $b \in \{0, 1\}$. If $b = 0$, $\mathcal{B}$ aborts and takes a random guess. The reason for this is by our definition if $b = 0$ where $(v_{i_\ell,t_\ell} \in W_{1,i} \wedge v_{i_\ell,t_\ell} \notin W_{0,i})$, we have $G_{\ell-1} = G_\ell$ because the distribution of the challenge ciphertext in game $G_{\ell-1}$ is the same as that of game G_ℓ, so there is no difference of advantage of $\mathcal{A}$ in $G_{\ell-1}$ and G_ℓ. We proceeds assuming $b = 1$.

Setup: To provide a public key PK to $\mathcal{A}$, $\mathcal{B}$ sets Y to $\boldsymbol{e}(g, g)^w$ where w is known to $\mathcal{B}$. For each attribute i where $1 \leq i \leq n$, $\mathcal{B}$ generates $\{A_{i,t}\}_{1 \leq t \leq n_i}$ such that $A_{i,t} = g^{\alpha_{i,t}}$ if $v_{i,t} \in W_{b,i}$ and $A_{i,t} = g^{z_1 \alpha_{i,t}}$ if $v_{i,t} \notin W_{b,i}$ where $\{\alpha_{i,t} \in \mathbb{Z}_p^*\}_{1 \leq t \leq n_i}$ are random. Then $\mathcal{B}$ publishes public parameters as in the real scheme by picking up $\{a_{i,t}, b_{i,t}\}_{1 \leq t \leq n_i}$ at random for $1 \leq i \leq n$ with the exception that, for a_{i_ℓ,t_ℓ} and b_{i_ℓ,t_ℓ}, $\mathcal{B}$ sets $a_{i_\ell,t_\ell} = z_1$ and $b_{i_\ell,t_\ell} = z_2$ and can compute $A_{i_\ell,t_\ell}^{a_{i_\ell,t_\ell}} = g^{\alpha_{i_\ell,t_\ell} a_{i_\ell,t_\ell}}$ and $A_{i_\ell,t_\ell}^{b_{i_\ell,t_\ell}} = g^{\alpha_{i_\ell,t_\ell} b_{i_\ell,t_\ell}}$ without knowing z_1, z_2.

Phase 1: $\mathcal{A}$ submits an attribute list $L = [L_1, \ldots, L_n]$ in a secret key query. If $L_{i_\ell} \neq v_{i_\ell,t_\ell}$, $\mathcal{B}$ can generate the corresponding secret key easily.
Let's assume $L_{i_\ell} = v_{i_\ell,t_\ell}$. $\mathcal{B}$ needs to compute the secret key components $[D_{i_\ell,0}, D_{i_\ell,1}, D_{i_\ell,2}] = [g^{s_{i_\ell}} (A_{i_\ell,t_\ell})^{a_{i_\ell,t_\ell} b_{i_\ell,t_\ell} \lambda_{i_\ell}}, g^{a_{i_\ell,t_\ell} \lambda_{i_\ell}}, g^{b_{i_\ell,t_\ell} \lambda_{i_\ell}}]$ where $a_{i_\ell,t_\ell} = z_1, b_{i_\ell,t_\ell} = z_2$.
$\mathcal{B}$ can compute $D_{i_\ell,0}$ as

$$\begin{aligned} D_{i_\ell,0} &= g^{s_{i_\ell}} (A_{i_\ell,t_\ell})^{a_{i_\ell,t_\ell} b_{i_\ell,t_\ell} \lambda_{i_\ell}} \\ &= g^{s_{i_\ell}} (A_{i_\ell,t_\ell})^{z_1 z_2 \lambda_{i_\ell}} \\ &= g^{s_{i_\ell}} (g^{\alpha_{i_\ell,t_\ell}})^{z_1 z_2 \lambda_{i_\ell}} \\ &= g^{s'_{i_\ell}} \end{aligned}$$

where s_{i_ℓ} is chosen at random such that

$$s_{i_\ell} = s'_{i_\ell} - \alpha_{i_\ell,t_\ell} z_1 z_2 \lambda_{i_\ell}$$

and random s'_{i_ℓ} is known to $\mathcal{B}$. $\mathcal{B}$ can compute the components $[D_{i_\ell,1}, D_{i_\ell,2}]$ easily without knowing z_1, z_2.
Here we can assume $L \not\models W_0 \wedge L \not\models W_1$ because $L_{i_\ell} = v_{i_\ell,t_\ell} \wedge v_{i_\ell,t_\ell} \notin W_{1-b,i_\ell}$. That is, we have $L \not\models W_{1-b}$ and therefore $L \not\models W_b$, so there must be $k \in \{1, \ldots, n\}$ such that $L_k (= v_{k,t_k}) \notin W_{b,k}$. Then $\mathcal{B}$ generates $[D_{k,0}, D_{k,1}, D_{k,2}]$ as follows: $\mathcal{B}$ sets $s_k = s'_k + \alpha_{i_\ell,t_\ell} z_1 z_2 \lambda_{i_\ell}$ where s'_k is random and known to $\mathcal{B}$ and computes

$$\begin{aligned} D_{k,0} &= g^{s_k} (A_{k,t_k})^{a_{k,t_k} b_{k,t_k} \lambda_k} \\ &= g^{s'_k + \alpha_{i_\ell,t_\ell} z_1 z_2 \lambda_{i_\ell}} (g^{z_1 \alpha_{k,t_k}})^{a_{k,t_k} b_{k,t_k} \lambda_k} \\ &= g^{s'_k} (g^{z_1 \alpha_{k,t_k}})^{a_{k,t_k} b_{k,t_k} \lambda'_k} \end{aligned}$$

where λ_k is chosen at random such that

$$\lambda_k = \lambda'_k - \frac{\alpha_{i_\ell,t_\ell} z_2 \lambda_{i_\ell}}{\alpha_{k,t_k} a_{k,t_k} b_{k,t_k}}$$

and random λ'_k is known to $\mathcal{B}$. $\mathcal{B}$ can compute the components $[D_{k,1}, D_{k,2}]$ easily without knowing z_2.
Also, for $i \neq i_\ell, k$, $\mathcal{B}$ can compute $[D_{i,0}, D_{i,1}, D_{i,2}]$ easily.
Finally by computing

$$\begin{aligned} s &= \sum_{i=1}^{n} s_i \\ &= s_{i_\ell} + s_k + \sum_{i \neq i_\ell, k} s_i \\ &= s'_{i_\ell} - \alpha_{i_\ell,t_\ell} z_1 z_2 \lambda_{i_\ell} + s'_k + \alpha_{i_\ell,t_\ell} z_1 z_2 \lambda_{i_\ell} + \sum_{i \neq i_\ell, k} s_i \\ &= s'_{i_\ell} + s'_k + \sum_{i \neq i_\ell, k} s_i, \end{aligned}$$

the component $D_0 = g^{w-s}$ of the secret key can be computed.

Challenge: $\mathcal{A}$ submits two challenge messages M_0 and M_1. $\mathcal{B}$ sets $C_0 = g^{z_3+z_4}$ which implies $r = z_3 + z_4$. If $L \not\models W_0 \wedge L \not\models W_1$ for every queried L, $\mathcal{B}$ sets $\widetilde{C}$ to be random and otherwise sets $\widetilde{C} = M_b \boldsymbol{e}(g, g^{z_3+z_4})^w$. $\mathcal{B}$ generates, for W_b, the ciphertext components $\{\{C_{i,t,1}, C_{i,t,2}\}_{1 \leq t \leq n_i}\}_{1 \leq i \leq n}$ as in $G_{\ell-1}$ with the exception that the components $\{C_{i_\ell,t_\ell,1}, C_{i_\ell,t_\ell,2}\}$ are computed as

$$C_{i_\ell,t_\ell,1} = (A_{i_\ell,t_\ell}^{b_{i_\ell,t_\ell}})^{r_{i_\ell,t_\ell}} = (A_{i_\ell,t_\ell}^{z_2})^{z_4} = (g^{\alpha_{i_\ell,t_\ell} z_2})^{z_4},$$

$$C_{i_\ell,t_\ell,2} = (A_{i_\ell}^{a_{i_\ell,t_\ell}})^{r - r_{i_\ell,t_\ell}} = (g^{\alpha_{i_\ell,t_\ell} z_1})^{z_3} = Z^{\alpha_{i_\ell,t_\ell}}$$

without knowing $z_2 z_4, z_1 z_3$. This implies that $r_{i_\ell,t_\ell} = z_4$ and $Z = g^{z_1 z_3}$ and if $Z = g^{z_1 z_3}$, the components are well-formed and $\mathcal{A}$ is in game $G_{\ell-1}$.

Phase 2: Phase 1 is repeated.

Guess: $\mathcal{A}$ outputs a guess b' of b. If $b' = b$, $\mathcal{B}$ outputs 1 and otherwise outputs 0. By our assumption, the probability that $\mathcal{A}$ guesses b correctly in game $G_{\ell-1}$ has a non-negligible ϵ difference from that of it guessing b correctly in G_ℓ. When $Z = g^{z_1 z_3}$, $\mathcal{A}$ is in game $G_{\ell-1}$ and when Z is random, $\mathcal{A}$ is in game G_ℓ. Therefore the simulator $\mathcal{B}$ has advantage ϵ in the D-Linear game. $\square$