









Research Article

A Fully Secure KP-ABE Scheme on Prime-Order Bilinear Groups through Selective Techniques

Isaac Amankona Obiri ^{1,2,3}, Qi Xia ^{1,2,3}, Hu Xia ¹,
Kwame Opuni-Boachie Obour Agyekum ^{1,2,3}, Kwame Omono Asamoah ^{1,2,3},
Emmanuel Boateng Sifah ^{1,2,3}, Xiaosong Zhang ¹ and Jianbin Gao ^{2,3,4}

¹School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China

²UESTC- CDFH Joint Institute of Blockchain, Chengdu, China

³Chengdu Jiaozhi Financial Holding Group Co. LTD., Chengdu, China

⁴School of Resources and Environment, University of Electronic Science and Technology of China, Chengdu, China

Correspondence should be addressed to Jianbin Gao; gaojb@uestc.edu.cn

Received 8 March 2020; Revised 16 September 2020; Accepted 5 October 2020; Published 29 November 2020

Academic Editor: Savio Sciancalepore

Copyright © 2020 Isaac Amankona Obiri et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Key-policy attribute-based encryption (KP-ABE) is the cryptographic primitive which enables fine grained access control while still providing end-to-end encryption. Although traditional encryption schemes can provide end-to-end encryption, users have to either share the same decryption keys or the data have to be stored in multiple instances which are encrypted with different keys. Both of these options are undesirable. However, KP-ABE can provide less key overhead compared to the traditional encryption schemes. While there are a lot of KP-ABE schemes, none of them simultaneously supports multiuse of attributes, adaptive security, monotone span programs, and static security assumption. Hence, we propose a fully secure KP-ABE scheme for monotone span programs in prime-order group. This scheme uses selective security proof techniques to obtain the requisite ingredients for full security proof. This strengthens the correlation between selective and full security models and enables the transition of the best qualities in selective security models to fully secure systems. The security proof is based on decisional linear assumption and three-party Diffie–Hellman assumption.

1. Introduction

Attribute-based encryption (ABE) is a public key cryptosystem which yields fine grained access control over ciphertext. Succinctly put, the ABE system allows ciphertext and key to be linked to a set of attributes such that the decryption of a particular ciphertext is feasible only if the set of attributes of a user's secret key satisfies the attributes of the ciphertext. In key-policy ABE (KP-ABE) construction for instance, a message is encrypted over attribute set such as “profession: nurse, sex: female, and institution: hospital A,” and keys are generated over access policy like “profession: nurse \wedge sex: female.” The decryption of a given ciphertext is feasible only if the attributes satisfy the access policy. Ciphertext-policy ABE (CP-ABE) construction is a dual

version of KP-ABE scheme with the ciphertext and key attached to access policy and attributes, respectively [1–3].

ABE is useful cryptographic primitive when data are outsourced in untrusted repositories such as third-party cloud servers. ABE provides an efficient mechanism to share the outsourced data with multiple users based on the user's roles or attributes. While traditional encryption methods can provide end-to-end encryption, users have to either share the same decryption keys or the data have to be stored in multiple instances which are encrypted with different keys. Both of these options are inappropriate. However, ABE can provide less key overhead compared to traditional encryption methods. ABE offers fine grained access control while still providing end-to-end encryption. In the public repository, a malicious user can obtain the stored encrypted

data which do not match his/her attributes' secret key; however, the user cannot access the content of the data without decipherable keys.

Suppose that a patient encrypts his or her personal health data with the attributes $\{Hospital A, Hospital B, doctor, nurse\}$ and the access control policy is $\{\{Hospital A\} \text{ and } \{nurse \text{ or } doctor\}\} \text{ or } \{\{Hospital B \text{ and } \{nurse \text{ or } doctor\}\}\}$. With this access policy, any nurse or doctor in one of the hospitals satisfies the access policy requirement and can access the encrypted data. This example is visualized in Figure 1.

However, in this scenario, the challenge is that the attributes "nurse" and "doctor" have been used in the access policy multiple times. An ABE scheme with a single-use of attributes restriction requires that an attribute must appear only once in the access structure. [3]. One caveat way to overcome this single use of restriction is to fix multiple attributes for each use of the attributes such as $nurse-1, doctor-1, nurse-2, doctor-2$, and so on, in advance. However, there are two problems with this solution. The first is that the maximum number of similar attributes appearing in the access policy should be determined at the setup stage. Hence, the access policy supported by the scheme become restricted. The second one is that, in KP-ABE, for example, this solution blows up the size of ciphertext relative to the maximum number an attribute is reused, which yields reduction in performance. Conversely, in a KP-ABE construction that supports multiple attributes usage, policies are not constrained and any combination of attributes can be rendered arbitrarily to create a policy. So, in this KP-ABE construction, the size of ciphertext becomes policy-independent and compact.

The expressiveness of access policy ensures rich structure of keys and ciphertexts of ABE construction [3]. The more expressive an ABE scheme is, the more space is required for the potential access policies and attributes. This raises substantial obstacles, when proving the security of ABE schemes, since the standard notion of security should impose collusion resistance. Thus, a coalition of unqualified users must not have the ability to aggregate their secret keys to decrypt a ciphertext in which not one of them is approved to decrypt. Therefore, the security proof must consider an adversary who is capable of collecting different keys, not only the private key formally assigned to him/her. This necessitates security reductions to strike a balance between two conflicting objectives: the simulator should be sufficiently strong to provide an adversary with the numerous keys he/she requests adaptively; however, the simulator should also lack vital information about the strategies of the adversary which enables him/her to achieve success. Thus, the procedures of the adversary should be hidden from the simulator. The foremost security proof in the standard model for ABE constructions in [4, 5] adopts a strategy, known as "partitioning," to reconcile these two objectives. The partitioning proof strategy was formerly employed in the settings of identity-based encryption (IBE) [6–10].

With the partitioning proof, the simulator configures the system in such a way that every possible private key is in one of the two spaces: keys which the simulator has the ability to

create and which she/he cannot [3]. In order to guarantee that the keys that the adversary queries are within the set of keys that the simulator can produce, the previous works [4, 5, 11] resorted to a weaker security model referred to as "selective security." Unfortunately, under this model, the adversary must announce the access structure to be attacked before giving the public parameters of the system. This does not seem to be suitable security notion in practice for the high security requirement in the real world applications. At the intermediate phase, this concept of selective security is quite useful, but unsuitable as an ultimate goal. In the settings of IBE, the drawback of selective security was eliminated by giving the simulator the ability to "guess" a partition and terminate whenever the adversary exceeds its limit [10]. Nevertheless, if this approach is used in the ABE schemes, it will lead to exponential loss of security because the ABE scheme has a highly expressive access policy, which makes it difficult to identify a partition that is consistent with the partial power ordering of each key. Moreover, CP-ABE's selective security is still a challenge and the state-of-the-art approach in [5] introduced "q-type" assumption into the fully secure ABE constructions. The q-type assumption in [5] resulted from the need to encode small public parameters with a potentially large access policy. However, since this assumption is extremely complex to understand and also vulnerable to Cheon attack [12], it leads us to seek for KP-ABE construction which is proven fully secure under simple static assumptions such as three-party Diffie–Hellman assumption and decisional linear assumption.

Dual system of encryption was introduced by Waters [1] to solve the constraints imposed by the partitioning model. In the proof of the dual system of encryption, the simulator is incessantly configured to produce every key and the challenge ciphertext the adversary requests. The principal idea of the technique is that there are two categories of keys and ciphertexts, namely, "normal" and "semifunctional," which the simulator can produce [1]. A ciphertext can be decrypted with a key when both the key and the ciphertext are not semifunctional. The combination of ciphertext and a key, both semifunctional, triggers failed decryption because in the semifunctional space, the hidden objects are not cancelled. The semifunctional keys and ciphertext are used in hybrid proof while the normal key and ciphertext are used in the actual system. In the hybrid proof, the adversary is given either normal or semifunctional ciphertext and the secret key that is progressively given to him/her is converted to semifunctional one after the other until it gets to a point where the simulator only issues semifunctional key in the security game. At this point, it becomes easy to prove the security.

The critical step in the hybrid proof is when a key becomes semifunctional. At this stage, depending on the proposition that the key (now semifunctional) cannot correctly decrypt the challenge ciphertext, we indicate that the adversary cannot recognize the subtle change in the key. However, since the simulator does not need a partition, she/he cannot be restricted from generating a key and testing the workability of the key by himself/herself via decrypting a semifunctional ciphertext with the key. This challenge in

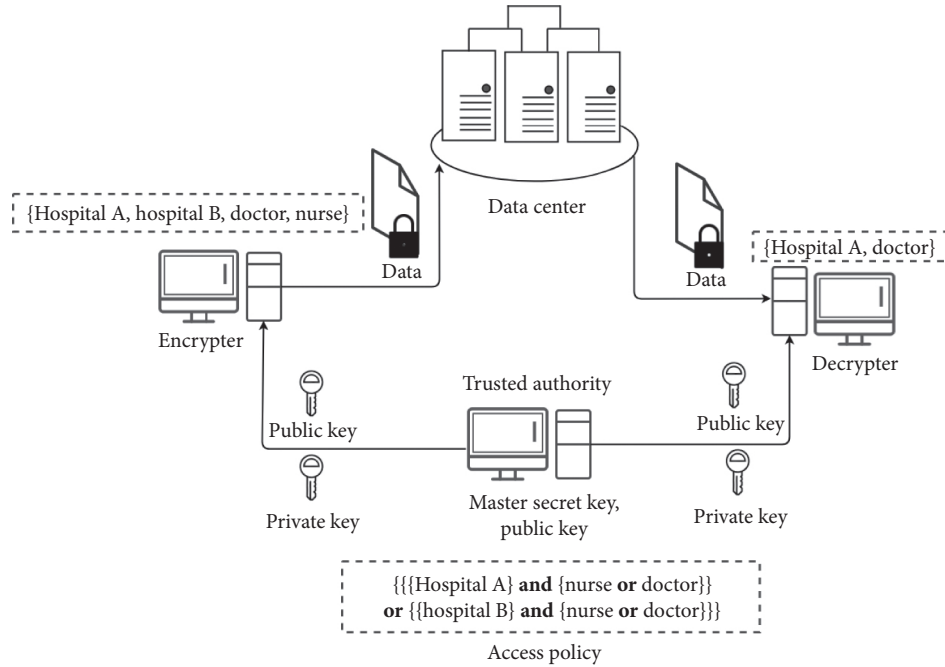


FIGURE 1: Example of using ABE in sharing personal health data.

dual encryption proof was resolved in [13] by guaranteeing that the simulator generates only a key decipherable with a semifunctional ciphertext so that the decryption is not hindered from the perspective of the simulator irrespective of whether the semifunctional object exists or not. This type of key formed by the simulator is known as “nominal semifunctional key.” Conversely, in the perspective of the adversary who is restricted from requesting a qualified key, the decryption of the “semifunctional ciphertext” with the “nominal semifunctional key” is hindered. The correlated factor within the nominal semifunctional key and the semifunctional ciphertext which guarantees successful decryption is information-theoretic concealed from the adversary [13]. This presented the first proof of full security in the standard model.

Our dual encryption system is constructed over semifunctional and normal space. The semifunctional components of ciphertext and keys are much like the normal component of the actual system except that they are decoupled from the public parameters. This gives us the chance to obtain related parameters in the semifunctional space to create relevant variables during the course of simulation rather than to have all the parameters fixed up in the setup phase. The semifunctional space can supply fresh parameters to the simulator for key isolation mechanism; this implies that every semifunctional key should have unique distribution through the use of fresh parameters in the semifunctional space. When the simulator first issues a secret key, then the challenge access policy is known before the semifunctional parameters are defined. Based on the known access policy, the simulator can embed a difficulty in the secret key from the semifunctional space and later annul this difficulty in the ciphertext. On the other hand, if a ciphertext is first issued, then the attribute set of the challenge ciphertext is known before semifunctional

parameters are specified. Based on the known attributes, the simulator can also embed a difficulty in the ciphertext from the semifunctional space and later annul this difficulty in the key. The difficulty is random variables chosen in the semifunctional space with their attachment to either the secret key or the ciphertext rendering invalid decryption unless those variables are cancelled out. The difficulty which is embedded in either the key or the ciphertext requires a complete set of key’s component to cancel it out. However, based on the restriction that the adversary cannot obtain complete components of a secret key, computationally this difficulty is unknown to him/her. Therefore, the two selective ways of embedding difficulties to prevent correct decryption of ciphertext can be combined to attain full security in the standard model.

1.1. Our Contribution. We present a KP-ABE scheme in the prime-order settings that supports monotone span program for access policies. The construction achieves full security through selective techniques. Our scheme is based on simple security assumption such as three-party Diffie–Hellman assumption and decisional linear assumption. In summary, our KP-ABE scheme simultaneously achieves the following results:

- (1) It enables arbitrary usage of attributes in the access policy.
- (2) It achieves adaptive security through selective techniques in the standard model.
- (3) It depends on static assumption in the standard model.
- (4) It supports span program matrix.
- (5) It has compact ciphertext size.

2. Related Works

ABE, which evolved from IBE, was initially designed by Shamir [14] and then constructed in [6, 15]. Horwitz and Lynn [16] expanded this idea to hierarchical identity-based encryption (HIBE) which was firstly constructed by Gentry and Silverberg [17]. In the standard model, some earlier ABE constructions [8–10, 18] were proven to be selectively secure. Also, ABE construction has been proven to be secure in the generic group model [19]. Dual encryption proofing techniques were also further explored in the study of [20, 21] and applied to attain leakage resilience in [22–24] and applied directly to computational assumptions in [25]. Lewko and Waters [3] developed new methodology to prove full security by integrating selective techniques used to prove selective security for KP-ABE and CP-ABE constructions. However, they used “q-type assumption” in the security proof which is susceptible to attack [12]. From the studies in [12], it can be inferred that as “q” grows larger, the “q-type” assumption becomes stronger and the scheme which requires it becomes vulnerable, particularly if “q” scales in a predictable way. Recently, Tomida et al. [26] and Kowalczyk et al. [27] proposed ABE schemes that address the problem of one-use restriction of attributes in access policy. The schemes were built on a piecewise guessing framework developed in [28], and they proved adaptive security of the ABE constructions with some polynomial security losses. Nonetheless, these schemes do not directly support span programs (linear secret sharing scheme matrix) to express policies. Song et al. [29] proposed attribute-based encryption which enables users to request for their attribute private keys without revealing their attributes to the key generator. Even though this scheme ensures users’ attributes privacy, it is not fully secure and it does not ensure multiuse of attributes. Recently, Khan et al. [30] proposed efficient attribute-based encryption with repeated attribute optimization. The authors employed “RAO” algorithm to remove repeated redundant attribute shares in encryption operations to reduce ciphertext size and computational cost. However, the limitation of this scheme is that the security is proven in selective model. Also, the scheme is not proven secure against chosen ciphertext attacks but rather chosen plaintext attacks under generic bilinear group model. Hence, the scheme does not achieve full security.

Table 1 shows the comparison of our scheme with other KP-ABE schemes which satisfy full security (adaptive security) notion. Note that adaptive security just refers to an adversary who does not execute all the queries at once (batch queries), but rather adapts her/his queries from previous results (see Definition 10). The last row describes our scheme in Section 4. From Table 1, it can be seen that JL [26] scheme possesses most of the needed properties of ABE construction. However, this scheme inherits the problem of polynomial security losses from the “piecewise guessing framework” that was used for its construction. Also, the security of schemes JL [26] and LK [27] was proven in random oracle model. Therefore, from the authors’ point of view, there is no scheme that simultaneously achieves the

properties listed in Section 1.1. and that is still able to retain the efficiency of selective security in the standard model.

2.1. Organization. In Section 3, we revise the important concepts on KP-ABE systems in prime-order bilinear groups, along with formal definitions of the complexity assumptions. In Section 4, we provide the construction of our scheme and demonstrate its correctness. Section 5 shows security proof of the scheme. In Section 6, we provide implementation and evaluation of the proposed scheme and other related schemes. Finally, in Section 7, we conclude the work.

3. Preliminaries

Definition 1 (access structure) [33]. Let $P = \{\rho_1, \rho_2, \dots, \rho_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^P$ is monotone if $B \in \mathbb{A}$ and $B \subseteq C$ imply that $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection of nonempty subsets of P . The sets in \mathbb{A} are authorized sets, and the sets not in \mathbb{A} are nonauthorized sets.

Definition 2 (linear secret sharing scheme). A linear secret sharing scheme is made of two algorithms, share and reconstruct. To distribute a secret $s \in \mathbb{Z}_p$ among n parties, the share algorithm sets $r_1 = s$, randomly selects $(r_1, r_2, \dots, r_t) \in \mathbb{Z}_p^t$, and computes $\rho(i) = \sum_{k=1}^t r_k i^k$ for all $1 \leq i \leq n$. The shadows or shares $\rho(i) = \lambda_i$ are distributed to n distinct parties. Since the secret is the constant term $s = r_1 = \rho(1)$, the reconstruct algorithm recovers the secret from any t shares λ_i , for the attribute set $S = \mathbb{A}$ and $I = \{i | \rho(i) \in S\}$, by computing the linear function of the shares as $\sum_{i \in I} \omega_i \lambda_i = s$, where each constant $\omega_i = \prod_{j \in I, j \neq i} (i / (j - 1))$ can be obtained efficiently in the polynomial time.

Definition 3 (monotone span program (msp)) [34]. A msp is a linear algebraic model for computing monotone functions. Let \mathbb{Z} be a field and $\{w_1, \dots, w_n\}$ be variables. A msp is a tuple $\Delta = (M, \rho)$ where $M \in \mathbb{Z}^{t \times n}$ is a matrix and $\rho: \{1, 2, \dots, \rho_t\} = \{w_1, \dots, w_n\}$ is labelling function. The msp Δ actualizes the monotone access structure $\mathbb{A} \subseteq 2^P$ where $B \in \mathbb{A}$ if and only if n is spanned by the rows of the matrix M whose labels belong to B . The size of Δ is t , and the number of rows in M . With regard to secret sharing, the size of the msp is the total number of shares that are given to all parties in P .

Definition 4 (bilinear groups). A group generating algorithm \mathcal{G} takes a secret parameter λ and returns a description of a group $\mathcal{G} \rightarrow (p, \mathbb{G}, \mathbb{G}_T, g, e)$, where p is a prime number, \mathbb{G} and \mathbb{G}_T are cyclic groups of order p , $g \in \mathbb{G}$ is a generator, and $e: \mathbb{G}^2 \rightarrow \mathbb{G}_T$ is a bilinear map, which has two properties:

- (1) Bilinearity:
 $\forall g, h \in \mathbb{G}, z, y \in \mathbb{Z}_p, e(g^z, h^y) = e(g, h)^{zy}$.
- (2) Nondegeneracy: for generators g and h , $e(g, h) \neq 1$.

TABLE 1: Comparison of KP-ABE schemes which have adaptive security.

Scheme	Msp	Assumption	Compact	Multiuse	Full (adaptive) security
GSW [31]	✓	Static ✓	✓		
Lw [3]	✓	q-type	✓	✓	✓
CGK [32]	✓	k-Lin ✓			✓
JL [26]		Static ✓	✓	✓	✓
LK [27]		Static ✓		✓	✓
Ours	✓	Static ✓	✓	✓	✓

Definition 5 (the decisional linear (DLIN) assumption). With a given group generating algorithm \mathcal{G} , we define the following distribution:

$$\begin{aligned} \mathbb{G} := & (G, p, G_T, e) \xleftarrow{\mathcal{G}}, \\ & g, b, d \xleftarrow{\mathbb{S}} G, a_1, a_2 \xleftarrow{\mathbb{S}} \mathbb{Z}_p, \\ & D \leftarrow (G, g, b, d, b^{a_1}, d^{a_2}), \\ & T_0 \leftarrow g^{a_1 + a_2}, \\ & T_1 \leftarrow \mathbb{G}. \end{aligned} \quad (1)$$

The advantage an algorithm \mathcal{A} has in breaking this assumption is

$$\text{Adv}_{\mathcal{G}, \mathcal{A}}^{\text{dL}}(\lambda) := |P[\mathcal{A}(D, T_0) = 1] - P[\mathcal{A}(D, T_1) = 1]|. \quad (2)$$

We declare that DLIN assumption is satisfied by \mathcal{G} , if for any probabilistic polynomial time (PPT) algorithm \mathcal{A} , $\text{Adv}_{\mathcal{G}, \mathcal{A}}^{\text{dL}}(\lambda)$ is negligible.

Definition 6 (the three-party Diffie–Hellman (TPDH) assumption). With a given group generating algorithm \mathcal{G} , we define the following distribution:

$$\begin{aligned} \mathbb{G} := & (G, p, G_T, e) \xleftarrow{\mathcal{G}}, \\ & g \xleftarrow{\mathbb{S}} G, x, y, z \xleftarrow{\mathbb{S}} \mathbb{Z}_p, \\ & D \leftarrow (G, g^x, g^y, g^z), \\ & T_0 \leftarrow g^{xyz}, \\ & T_1 \leftarrow G. \end{aligned} \quad (3)$$

The advantage an algorithm \mathcal{A} has in breaking this assumption is

$$\text{Adv}_{\mathcal{G}, \mathcal{A}}^{\text{3PDH}}(\lambda) := |P[\mathcal{A}(D, T_0) = 1] - P[\mathcal{A}(D, T_1) = 1]|. \quad (4)$$

We declare that TPDH assumption is satisfied by \mathcal{G} , if for any probabilistic polynomial time (PPT) algorithm \mathcal{A} , $\text{Adv}_{\mathcal{G}, \mathcal{A}}^{\text{3PDH}}(\lambda)$ is negligible.

Definition 7 (dual pairing vector spaces). We follow the definition of double vector pairing spaces in [35, 36]. For $\vec{u} = (u_1, \dots, u_n) \in \mathbb{Z}_p^n$ and $g \in \mathbb{G}$, we write $g^{\vec{u}}$ to represent the n -tuple of elements of \mathbb{G} :

$$g^{\vec{u}} := (g^{u_1}, \dots, g^{u_n}). \quad (5)$$

We can execute scalar product and exponentiation in the exponent. For any $a \in \mathbb{Z}_p$ and $\vec{u}, \vec{v} \in \mathbb{Z}_p^n$, we have

$$\begin{aligned} g^{a\vec{u}} &:= (g^{au_1}, \dots, g^{au_n}), \\ g^{\vec{u}+\vec{v}} &:= (g^{\vec{u}_1+\vec{v}_1}, \dots, g^{\vec{u}_n+\vec{v}_n}). \end{aligned} \quad (6)$$

We define a bilinear map e_n to represent the product of the componentwise pairings:

$$e_n(g^{\vec{u}}, g^{\vec{v}}) := \prod_{i=1}^n (g^{\vec{u}_i}, g^{\vec{v}_i}) = e(g, g)^{\vec{u} \cdot \vec{v}}. \quad (7)$$

Here, the dot product is executed using modulo p . We select two random sets of vectors: $\mathbb{B} := \{\vec{b}_1, \vec{b}_2, \dots, \vec{b}_n\}$ and $\mathbb{B}^* := \{\vec{b}_1^*, \vec{b}_2^*, \dots, \vec{b}_n^*\}$ of \mathbb{Z}_p^n subject to the following constraints:

- (1) The basis \mathbb{B} with the family \vec{b}_i and dual basis \mathbb{B}^* with the family \vec{b}_i^* are dual orthonormal when $\vec{b}_i \cdot \vec{b}_j^* = 0 \pmod{p}$, for $i, j = \{1, \dots, n\}$, whenever $i \neq j$. Therefore, the two vectors are perpendicular to each other. As a consequence, their dot product yields zero.
- (2) Conversely, \vec{b}_i is orthonormal to \vec{b}_i^* when they have the same index, i.e., $\vec{b}_i \cdot \vec{b}_j^* = \delta \pmod{p}$, for $i, j = 1, \dots, n$ whenever $i = j$, where δ denotes nonzero element of \mathbb{Z}_p . Here, it can be seen that we have abused the terminology “orthonormal,” since δ is not constrained to 1.

Note that the random selection of $(\mathbb{B}, \mathbb{B}^*)$ from the sets that satisfy requirements of dual orthonormality can be done by selecting a set of n vectors (i.e., $\{\vec{b}_1, \vec{b}_2, \dots, \vec{b}_n\} \in \mathbb{B}$) at uniformly random from \mathbb{Z}_p^n . Then, each vector \mathbb{B}^* is determined from the orthonormality constraint such that under high probability, the vectors $(\mathbb{B}, \mathbb{B}^*)$ are linearly independent.

Definition 8 (the subspace assumption). With a given group generating algorithm \mathcal{G} , we define the subspace assumption as

$$\begin{aligned}
\mathbb{G} &:= (G, p, G_T, e) \xleftarrow{\$} \mathcal{G}, g \xleftarrow{\$} G, \delta, \eta, \beta, \tau_1, \tau_2, \tau_3, \mu_1, \mu_2, \mu_3 \xleftarrow{\$} \mathbb{Z}_p, \\
&\cdot (\mathbb{B}_1, \mathbb{B}_1^*) \xleftarrow{\$} \text{Dual}(\mathbb{Z}_p^{n_1}, \delta), \dots, (\mathbb{B}_m, \mathbb{B}_m^*) \xleftarrow{\$} \text{Dual}(\mathbb{Z}_p^{n_m}, \delta), d, \\
\mathcal{U}_{1,i} &:= g^{\mu_1 \gamma_1 \bar{b}_{1,i} + \mu_2 \gamma_2 \bar{b}_{k_i+1,i} + \mu_3 \gamma_3 \bar{b}_{2k_i+1,i}}, \\
\mathcal{U}_{2,i} &:= g^{\mu_1 \gamma_1 \bar{b}_{2,i} + \mu_2 \gamma_2 \bar{b}_{k_i+2,i} + \mu_3 \gamma_3 \bar{b}_{2k_i+2,i}}, \\
&\dots, \\
\mathcal{U}_{k_i,i} &:= g^{\mu_1 \gamma_1 \bar{b}_{k_i,i} + \mu_2 \gamma_2 \bar{b}_{k_i+2,i} + \mu_3 \gamma_3 \bar{b}_{2k_i,i+2,i}}, \quad \forall i \in [m], \\
V_{1,i} &:= g^{\tau_1 \eta \bar{b}_{1,i}^* + \tau_2 \beta \bar{b}_{k_i+1,i}^*}, \\
V_{2,i} &:= g^{\tau_1 \eta \bar{b}_{2,i}^* + \tau_2 \beta \bar{b}_{k_i+2,i}^*}, \\
&\dots, \\
V_{k_i,i} &:= g^{\tau_1 \eta \bar{b}_{k_i,i}^* + \tau_2 \beta \bar{b}_{2k_i,i}^*}, \quad \forall i \in [m], \\
W_{1,i} &:= g^{\tau_1 \eta \bar{b}_{1,i}^* + \tau_2 \beta \bar{b}_{k_i+1,i}^* + \tau_3 \bar{b}_{k_i+1,i}^*}, \\
W_{2,i} &:= g^{\tau_1 \eta \bar{b}_{2,i}^* + \tau_2 \beta \bar{b}_{k_i+2,i}^* + \tau_3 \bar{b}_{2k_i+2,i}^*}, \\
&\dots, \\
W_{k_i,i} &:= g^{\tau_1 \eta \bar{b}_{k_i,i}^* + \tau_2 \beta \bar{b}_{2k_i,i}^* + \tau_3 \bar{b}_{3k_i,i}^*}, \quad \forall i \in [m], \\
D &:= \left(\mathbb{G}, g, \left\{ g^{\bar{b}_{1,i}}, g^{\bar{b}_{2,i}}, \dots, g^{\bar{b}_{2k_i,i}}, g^{\bar{b}_{3k_i+1,i}}, \dots, g^{\bar{b}_{n_i,i}}, g^{\eta \bar{b}_{1,i}^*}, \dots, g^{\eta \bar{b}_{k_i,i}^*}, g^{\beta \bar{b}_{k_i+1,i}^*}, \right. \right. \\
&\quad \left. \left. \dots, g^{\beta \bar{b}_{2k_i,i}^*}, g^{\beta \bar{b}_{2k_i+1,i}^*}, \dots, g^{\bar{b}_{n_i,i}^*}, \mathcal{U}_{1,i}, \mathcal{U}_{2,i}, \dots, \mathcal{U}_{k_i,i} \right\}_{i=1}^m, \mu_3 \right). \tag{8}
\end{aligned}$$

We assert that for any PPT algorithm \mathcal{A} which returns a value in $\{0, 1\}$,

$$\begin{aligned}
\text{Adv}_{\mathcal{G}} \mathcal{A} &:= \left| P[\mathcal{A}(D, \{V_{1,i}, \dots, V_{k_i,i}\}_{i=1}^m) = 1] \right. \\
&\quad \left. - P[\mathcal{A}(D, \{W_{1,i}, \dots, W_{k_i,i}\}_{i=1}^m) = 1] \right| \tag{9}
\end{aligned}$$

is negligible in the security parameter λ . The subspace assumption is the application of the DLIN assumption with vectors. The proof of this assumption can be found in pages 37-38 [3].

Definition 9 (KP-ABE scheme). Under standard definition, a key-policy attribute-based encryption scheme has a quintuple algorithm (setup, Enc, KeyGen, Dec):

- (1) Setup($\mathcal{U}, 1^\lambda$) \rightarrow (msk, pp): it takes attribute universe description \mathcal{U} and a security parameter 1^λ . It returns a master secret key msk and public parameters pp.
- (2) Enc(msg, pp, S) \rightarrow (ct): it takes message msg, public parameters pp, and the set of attributes S. It outputs the ciphertext ct.
- (3) KeyGen(pp, msk, \mathbb{A}) \rightarrow $\text{sk}_{\mathbb{A}}$: it takes public parameters pp, master secret key msk, and an access structure \mathbb{A} . It outputs a secret key $\text{sk}_{\mathbb{A}}$.

- (4) Dec(pp, ct, $\text{sk}_{\mathbb{A}}$) \rightarrow msg: it takes public parameters pp, ciphertext ct, and secret key sk and returns a message msg or \perp .

3.1. Correctness. KP-ABE construction is correct if it meets the following requirements. With a given ciphertext and a secret key, if the ciphertext attribute's set matches the key's access structure, then for any msg \in Msg, we have

$$\left[\begin{array}{l} \text{pp, msk} \leftarrow \text{Setup}(1^\lambda, \mathcal{U}) \\ \text{ct} \leftarrow \text{Enc}(\text{pp}, \text{msg}, S) \\ \text{sk}_{\mathbb{A}} \leftarrow \text{KeyGen}(\text{pp}, \text{msk}, \mathbb{A}) \\ \text{msg}^* \leftarrow \text{Dec}(\text{pp}, \text{ct}, \text{sk}_{\mathbb{A}}) \end{array} \right] \text{msg} = \text{msg}^* \tag{10}$$

Definition 10 (KP-ABE full security model). The security game between the challenger C and adversary A proceeds as follows.

The security definition for fully secure KP-ABE depends on indistinguishable game with PPT-chosen plaintext attacker. The game proceeds as follows:

- (1) Setup: the challenger \mathcal{C} executes Setup($1^\lambda, \mathcal{U}$) \longrightarrow (pp, msk) and submits pp to adversary \mathcal{A} .
- (2) Phase 1: \mathcal{A} adaptively queries \mathcal{C} for the secret keys corresponding to a set of access structures $\{\mathbb{A}_1, \dots, \mathbb{A}_Q\}$. For each time, it obtains $\text{sk}_{\mathbb{A}} \leftarrow \text{KeyGen}(\text{pp}, \text{msk}, \mathbb{A}_k)$ from \mathcal{C} .
- (3) Challenge: \mathcal{A} sends two messages $\{\text{msg}_0^*, \text{msg}_1^*\}$ of equal size together with the set of attributes S^* to \mathcal{C} . Then, \mathcal{C} tosses a binary coin b and executes $\text{ct}^* \leftarrow \text{Enc}(\text{pp}, \text{msg}_b^*, S^*)$ and gives ct^* to \mathcal{A} on a condition that S^* does not satisfy any of the access structures queried in phase 1.
- (4) Phase 2: \mathcal{A} adaptively queries \mathcal{C} for the secret keys corresponding to the set of access structures $\{\mathbb{A}_{Q+1}, \dots, \mathbb{A}_Q\}$ with the condition that none of these satisfy S^* . For each time, it obtains $\text{sk}_{\mathbb{A}} \leftarrow \text{KeyGen}(\text{pp}, \text{msk}, \mathbb{A}_k)$ from \mathcal{C} .
- (5) At the end, \mathcal{A} returns b^* as a guess for b , and the adversary \mathcal{A} is a winner if $b = b^*$. The advantage of \mathcal{A} for this indistinguishable game is defined as

$$\text{Adv}^{\mathcal{A}} \text{Game}_{\text{real}} = \left| P[b = b^*] - \frac{1}{2} \right|. \quad (11)$$

Definition 11. A KP-ABE construction is fully secure if PPT algorithm \mathcal{A} has negligible advantage in the above security game.

Note that with the selective security game, the adversary must announce \mathbb{A} before viewing the pp. Henceforth, the term semifunctional will be denoted as SF.

4. Prime-Order KP-ABE Construction

We use the dual framework of data encryption proof technique in prime-order settings, where orthogonal subspaces within the exponents perform the role of both normal and SF components. Since SF vectors are never published, they can serve as “hidden parameters” which create new randomness even with a fixed size of public parameters. We provide fresh pair of vectors for each attribute to produce enough randomness to ensure an information-theoretic transition from a nominal SF key (one with SF components but still capable of correctly decrypting SF ciphertext) to a real SF one (a key which is incapable of decrypting SF ciphertext). Again, we denote the attribute universe $[\mathbb{U}] = \{1, 2, \dots, \mathbb{U}\}$ as the complete number of attributes within the system. The scheme is constructed as follows:

- (1) Setup($1^\lambda, \mathcal{U}$) \longrightarrow pp, msk: this algorithm picks a bilinear group G of prime order p and a generator g . It picks at random two pairs of dual orthonormal bases $(\mathbb{B}, \mathbb{B}^*)$, $(\mathbb{B}_0, \mathbb{B}_0^*)$ of dimension 3 and \mathcal{U} pairs of dual orthonormal bases $(\mathbb{B}_1, \mathbb{B}_1^*), \dots, (\mathbb{B}_{\mathcal{U}}, \mathbb{B}_{\mathcal{U}}^*)$ of dimension 6, bound to the restriction that they all

hold the same value δ . We let \vec{b}_i, \vec{b}_i^* denote the family vectors of $(\mathbb{B}, \mathbb{B}^*)$, and $\vec{b}_{i,j}, \vec{b}_{i,j}^*$ are the basis vectors of $(\mathbb{B}_j, \mathbb{B}_j^*)$ for each j from 0 to \mathcal{U} . The setup algorithm also picks a quadruple of random exponents $\{\alpha_1, \alpha_2, r_1, r_2\} \xleftarrow{\$} \mathbb{Z}_p$ and another quadruple of random exponents $\{y_1, y_2\} \xleftarrow{\$} \mathbb{Z}_p$, $\{y_1^*, y_2^*\} \xleftarrow{\$} \mathbb{Z}$ with the restriction that $\forall i = j, y_i \cdot y_i^* = 1$. The public parameters comprise

$$\text{pp} = \left\{ \begin{array}{l} G, p, g^{y_1 \vec{b}_{1,i}}, g^{r_1 \vec{b}_{1,i}}, g^{y_2 \vec{b}_{2,i}}, g^{r_2 \vec{b}_{2,i}}, g^{r_1 \vec{b}_3}, \\ g^{r_2 \vec{b}_4}, e(g, g)^{\alpha_1 \delta}, e(g, g)^{\alpha_2 \delta} \end{array} \right\}, \quad \forall i \in [\mathbb{U}]. \quad (12)$$

Additionally, the master secret key msk is

$$\text{msk} = \left\{ \alpha_1, \alpha_2, g^{y_1^* \vec{b}_{1,i}}, g^{r_1^* \vec{b}_{1,i}}, g^{y_2^* \vec{b}_{2,i}}, g^{r_2^* \vec{b}_{2,i}}, g^{r_1^* \vec{b}_3}, g^{r_2^* \vec{b}_4} \right\}, \quad \forall i \in [\mathbb{U}]. \quad (13)$$

- (2) KeyGen(pp, msk, $\mathbb{A} = (M, \rho)$) \longrightarrow $\text{sk}_{\mathbb{A}}$: the algorithm gets the public key pp, a master key msk, and access structure $\mathbb{A} = (M, \rho)$, and the algorithm picks randomly $\{(z_1, z_2, \dots, z_t), (r_1, r_2, \dots, r_t)\} \in \mathbb{Z}_p^t$. Then, set $z_1 = \alpha_1, r_1 = \alpha_2$ and compute the shares $\lambda_i = \sum_{k=1}^t z_k i^k$ and $\omega_i = \sum_{k=1}^t r_k i^k$ for all $1 \leq i \leq n$, where (i^1, i^2, \dots, i^t) is the vector of $M_i \in M$ which corresponds to the i -th row of M . It then picks randomly $a_1, a_2 \in \mathbb{Z}_p$ and outputs

$$\text{sk}_{\mathbb{A}} = \left\{ \begin{array}{l} K_{1,i} = g^{\lambda_i y_1^* \vec{b}_{1,i} + a_1 r_1 y_1^* \vec{b}_{1,i}} \cdot g^{\omega_i y_2^* \vec{b}_{2,i} + a_2 r_2 y_2^* \vec{b}_{2,i}} \\ K_2 = g^{a_1 r_1 \vec{b}_3 + a_2 r_2 \vec{b}_4} \end{array} \right\}, \quad i \in \rho(i). \quad (14)$$

- (3) Enc(msg, S, pp) \longrightarrow ct: the algorithm gets the message msg, attribute sets S, and public parameter pp, picks randomly $s_1, s_2 \xleftarrow{\$} \mathbb{Z}_p$, and outputs

$$\text{ct} = \left\{ \begin{array}{l} C_0 = \text{msg} \cdot e(g, g)^{\alpha_1 s_1 \delta + \alpha_2 s_2 \delta}, C_{1,i} = g^{s_1 y_1 \vec{b}_{1,i} + s_2 y_2 \vec{b}_{2,i}} \\ C_2 = g^{s_1 \vec{b}_3 + s_2 \vec{b}_4} \end{array} \right\}, \quad \forall i \in S. \quad (15)$$

- (4) Dec(ct, $\text{sk}_{\mathbb{A}}$) \longrightarrow msg: let S^* correspond to the set of attributes associated to ciphertext ct and M be the policy matrix. If S^* satisfies \mathbb{A} , the decryption algorithm computes $\alpha_1 = \sum_{i \in S^*} \lambda_i \sigma_i$ and

$\alpha_2 = \sum_{i \in S^*} \omega_i \sigma_i$, where each constant $\sigma_i = \prod_{j \in S^*, i \neq j} (i/(j-1))$ can be obtained efficiently in the polynomial time. It then computes

$$\begin{aligned}
\Omega &= \prod_{i \in S^*} \left(\frac{e_6(K_{1,i}, C_{1,i})}{e_3(K_2, C_2)} \right) \\
&= \prod_{i \in S^*} \left(\frac{e_6 \left(g^{\lambda_i y_1^* \bar{b}_{1,i} + a_1 r_1 y_1^* \bar{b}_{1,i}} \cdot g^{\omega_i y_2^* \bar{b}_{2,i} + a_2 r_2 y_2^* \bar{b}_{2,i}}, g^{s_1 y_1 \bar{b}_{1,i} + s_2 y_2 \bar{b}_{2,i}} \right)}{e_3 \left(g^{a_1 r_1 \bar{b}_3 + a_2 r_2 \bar{b}_4}, g^{s_1 \bar{b}_3 + s_2 \bar{b}_4} \right)} \right)^{\sigma_i} \\
&= \prod_{i \in S^*} \left(\frac{e_6 \left(g^{\lambda_i y_1^* \bar{b}_{1,i} + a_1 r_1 y_1^* \bar{b}_{1,i}}, g^{s_1 y_1 \bar{b}_{1,i} + s_2 y_2 \bar{b}_{2,i}} \right) e_6 \left(g^{\omega_i y_2^* \bar{b}_{2,i} + a_2 r_2 y_2^* \bar{b}_{2,i}}, g^{s_1 y_1 \bar{b}_{1,i} + s_2 y_2 \bar{b}_{2,i}} \right)}{e_3 \left(g^{a_1 r_1 \bar{b}_3 + a_2 r_2 \bar{b}_4}, g^{s_1 \bar{b}_3 + s_2 \bar{b}_4} \right)} \right)^{\sigma_i} \\
&= \prod_{i \in S^*} \left(\frac{e_6(g, g)^{\left(\lambda_i y_1^* \bar{b}_{1,i} + a_1 r_1 y_1^* \bar{b}_{1,i} \right) \left(s_1 y_1 \bar{b}_{1,i} + s_2 y_2 \bar{b}_{2,i} \right)} e_6(g, g)^{\left(\omega_i y_2^* \bar{b}_{2,i} + a_2 r_2 y_2^* \bar{b}_{2,i} \right) \left(s_1 y_1 \bar{b}_{1,i} + s_2 y_2 \bar{b}_{2,i} \right)}}{e_3(g, g)^{\left(a_1 r_1 \bar{b}_3 + a_2 r_2 \bar{b}_4 \right) \left(s_1 \bar{b}_3 + s_2 \bar{b}_4 \right)}} \right)^{\sigma_i} \\
&= \prod_{i \in S^*} \left(\frac{e_6(g, g)^{\left(\lambda_i s_1 y_1 y_1^* \bar{b}_{1,i} + a_1 r_1 s_1 y_1 y_1^* \bar{b}_{1,i} \right) + \left(a_1 r_1 s_1 y_1 y_1^* \bar{b}_{1,i} \bar{b}_{1,i} \right)} e_6(g, g)^{\left(\omega_i s_2 y_2 y_2^* \bar{b}_{2,i} + a_2 r_2 s_2 y_2 y_2^* \bar{b}_{2,i} \right) + \left(a_2 s_2 y_2 r_2 y_2^* \bar{b}_{2,i} \bar{b}_{2,i} \right)}}{e_3(g, g)^{\left(a_1 r_1 s_1 \bar{b}_3 \bar{b}_3 \right) + \left(a_2 r_2 s_2 \bar{b}_4 \bar{b}_4 \right)}} \right)^{\sigma_i} \\
&= \prod_{i \in S^*} \left(\frac{e(g, g)^{\left(\lambda_i s_1 \right) \delta + \left(a_1 r_1 s_1 \right) \delta} e(g, g)^{\left(\omega_i s_2 \right) \delta + \left(a_2 r_2 s_2 \right) \delta}}{e(g, g)^{\left(a_1 r_1 s_1 \right) \delta + \left(a_2 r_2 s_2 \right) \delta}} \right)^{\sigma_i} \\
&= e(g, g)^{\sum_{i \in S^*} \lambda_i \sigma_i s_1 \delta} e(g, g)^{\sum_{i \in S^*} \omega_i \sigma_i s_2 \delta}, \\
\Omega &= e(g, g)^{\alpha_1 s_1 \delta} e(g, g)^{\alpha_2 s_2 \delta}.
\end{aligned} \tag{16}$$

Then, the message is retrieved as

$$\begin{aligned}
\text{msg} &= \frac{C_0}{\Omega}, \\
\text{msg} &= \frac{\text{msg} \cdot e(g, g)^{\alpha_1 s_1 \delta} e(g, g)^{\alpha_2 s_2 \delta}}{e(g, g)^{\alpha_1 s_1 \delta} e(g, g)^{\alpha_2 s_2 \delta}}.
\end{aligned} \tag{17}$$

5. Security Proof

Theorem 1. *Under the DLIN assumption and TPDH assumption defined in Section 3, our KP-ABE construction is fully secure (i.e., see Definition 11).*

The security proof for our construction depends on hybrid argument over series of games. We will define the set

of keys and ciphertext that will be used in the games. To commence the security game, first the challenger generates $(g^{b_{5,i}}, g^{b_{5,i}}) \leftarrow (\mathbb{B}_i, \mathbb{B}_i^*) \leftarrow \text{Dual}(\mathbb{Z}_p^6, \delta)$ and $(g^{\bar{b}_6}, g^{\bar{b}_6}) \leftarrow (\mathbb{B}, \mathbb{B}) \leftarrow \text{Dual}(\mathbb{Z}_p^3, \delta)$ as the related parameters that will be used in the security proof.

5.1. SF Ciphertext. To create this ciphertext for a set of attributes S , firstly, we execute the normal encryption algorithm in equation (15). The ciphertext is made up of the following components: $C_0, \{C_{1,i}\}, \forall i \in S, C_2$. Then, we pick random values $(\mathbb{B}, \mathbb{B}^*) \leftarrow \text{Dual}(\mathbb{Z}_p^3, \delta) S_3 \in \mathbb{Z}_p$ and multiply $C_{1,i}$ by $g^{s_3 \bar{b}_{5,i}}$. Also, we multiply C_2 by $g^{s_3 \bar{b}_6}$. The other component of the ciphertext stays unaltered as shown below.

$$\text{ct} = \left\{ \begin{array}{l} C_0 = \text{msg} \cdot e(g, g)^{\alpha_1 s_1 \delta + \alpha_2 s_2 \delta}, C_{1,i} = g^{s_1 y_1 \bar{b}_{1,i} + s_2 y_2 \bar{b}_{2,i}} g^{s_3 \bar{b}_{5,i}} \\ C_2 = g^{s_1 \bar{b}_3 + s_2 \bar{b}_4} g^{s_3 \bar{b}_6} \end{array} \right\}, \quad \forall i \in S. \tag{18}$$

5.2. *SF Keys.* To generate these keys for an $\text{msp}(M, \rho)$, we first execute the normal key generation algorithm in equation (14) to get a normal key made up of the following components of $\{K_{1,i}\} \in \rho(i), K_2$. We then pick random

secret values $a_3, \alpha_3 \in \mathbb{Z}_p$ and a random vector $\nu = (r_1, \dots, r_t) \in \mathbb{Z}_p^t$ and set the index $r_1 = \alpha_3$. We produce shares for the secret as $\Phi_i = M_i \cdot \nu^T$, where M_i is the row vector in M with the label $\rho(i)$. The SF key is output as

$$\text{sk}_{\mathbb{A}} = \left\{ \begin{array}{l} K_{1,i} = g^{\lambda_i y_1^* \bar{b}_{1,i} + a_1 r_1 y_1^* \bar{b}_{1,i}} \cdot g^{\omega_i y_2^* \bar{b}_{2,i} + a_2 r_2 y_2^* \bar{b}_{2,i}} \cdot g^{\Phi_i \bar{b}_{5,i} + a_3 \bar{b}_{5,i}} \\ K_2 = g^{a_1 r_1 \bar{b}_3 + a_2 r_2 \bar{b}_4} \cdot g^{a_3 \bar{b}_6} \end{array} \right\}, \quad i \in \rho(i). \quad (19)$$

Recall that we do not put a partition on a simulator with a nominal SF key. Therefore, the nominal SF key correlates correctly with the SF ciphertext to allow decryption, regardless of the presence or absence of SF components. This happens because the share of the secret α_3 in the SF space is zero.

5.3. *Ephemeral SF Keys.* These keys are indistinguishable to nominal keys, with the exception that SF components attach to either $K_{1,i}$ or K_2 which is now being randomized (which prevent accurate SF ciphertext decryption). Concretely, to

create an ephemeral SF key for the access matrix M , we first execute the normal key generation algorithm in equation (15) to get a normal key made up of the following components of $\{K_{1,i}\} \in \rho(i), K_2$. We then pick random secret values $a_3, a_4, a_5, \alpha_3 \in \mathbb{Z}_p$ and a random vector $\nu = (r_1, \dots, r_t) \in \mathbb{Z}_p^t$ and set the index $r_1 = \alpha_3$. Note that value of the secret α_3 in the SF space is zero. We produce shares for the secret as $\Phi_i = M_i \cdot \nu^T$, where M_i is the row vector in M with the label $\rho(i)$. The SF key is output as

$$\text{sk}_{\mathbb{A}} = \left\{ \begin{array}{l} K_{1,i} = g^{\lambda_i y_1^* \bar{b}_{1,i} + a_1 r_1 y_1^* \bar{b}_{1,i}} \cdot g^{\omega_i y_2^* \bar{b}_{2,i} + a_2 r_2 y_2^* \bar{b}_{2,i}} \cdot g^{a_4 \Phi_i \bar{b}_{5,i} + a_3 \bar{b}_{5,i}} \\ K_2 = g^{a_1 r_1 \bar{b}_3 + a_2 r_2 \bar{b}_4} \cdot g^{a_3 \bar{b}_6} \end{array} \right\}, \quad i \in \rho(i). \quad (20)$$

5.4. *Proof Structure.* The hybrid proof is executed over a series of games. Denoting Q as the total number of key requested by adversary, we define the series of games as follows: $\text{Game}_{\text{real}}$ is the real security game as in Section 3 (see Definition 10). In Game_k , the ciphertext submitted to the adversary is SF, as are the first k keys. The rest of the keys are normal. Game_k^N is similar to Game_k , besides the fact that the k -th key delivered to the adversary is nominal SF key. The first $k-1$ keys are SF, whereas the rest of the keys are normal. Game_k^T is similar to Game_k , besides the fact that the k -th key delivered to the adversary is an ephemeral SF. The first $k-1$ keys are SF, whereas the rest of the keys are normal. $\text{Game}_{\text{final}}$ is analogous to Game_Q , besides the fact that the SF ciphertext delivered to the adversary is encryption of random message.

The layout of our hybrid argument will be as follows. Firstly, we move from $\text{Game}_{\text{real}}$ to Game_0 , then to Game_1 , next to Game_2 , and so on. Eventually, we get to Game_Q , where all of the keys and the ciphertext delivered to the adversary are SF. Then, we move to $\text{Game}_{\text{final}}$ and this completes our security proof since any adversary in the final game has negligible advantage. The transition from $\text{Game}_{\text{real}}$ to Game_0 and from Game_Q to $\text{Game}_{\text{final}}$ is not complicated and can be done with the help of the computational assumptions. However, the transition from Game_{k-1} to Game_k is a bit complicated and requires other steps. For these steps, we will consider making transition between two

phases. Phase 1 is when the adversary requests a challenge ciphertext after obtaining the secret key. In phase 2, the adversary requests a secret key after obtaining the challenge ciphertext. Therefore, in order to get from Game_{k-1} to Game_k , we will transition first from Game_{k-1} to Game_k^N , then to Game_k^T , and finally to Game_k . We let Q_1 represent the number of queries in phase 1, and we will tackle this transition independently for $k \leq Q_1$ and $k \geq Q_1$. The security proof for phase 1 queries and phase 2 queries is similar to the selective security proof in the KP-ABE settings and CP-ABE settings, respectively.

Lemma 1. *Under the subspace assumption, no PPT adversary can achieve a non-negligible advantage in distinguishing $\text{Game}_{\text{real}}$ and Game_0 .*

Proof: Suppose a PPT algorithm \mathcal{A} achieves a non-negligible advantage in distinguishing $\text{Game}_{\text{real}}$ from Game_0 , then we will construct a PPT algorithm \mathcal{C} to break the subspace assumption. We will set the parameters $n_i = 3$, $m = \mathcal{U} + 2$, $k_i = 1$, for two values of i and $n_i = 6$, $k_i = 2$ for the remaining values of i in the subspace assumption. To correctly align the assumption notation with our scheme notation, we hereby designate the bases of the assumption as $(\mathbb{B}, \mathbb{B}^*)$, $(\mathbb{B}_0, \mathbb{B}_0^*) \in \text{Dual}(\mathbb{Z}_p^3, \delta)$ and $(\mathbb{B}_1, \mathbb{B}_1^*), \dots, (\mathbb{B}_{\mathcal{U}}, \mathbb{B}_{\mathcal{U}}^*) \in \text{Dual}(\mathbb{Z}_p^6, \delta)$. We will exclude the μ_3 term

because it is not applicable here. The procedure for simulating $\text{Game}_{\text{real}}$ and Game_0 is described as follows :

- (1) $(G, p, g^{\vec{d}_3}, g^{\vec{d}_4}, g^{\eta \vec{d}_3}, g^{\beta \vec{d}_4}, \{g^{\vec{d}_{1,i}}, g^{\vec{d}_{2,i}}, g^{\eta \vec{d}_{1,i}}, g^{\beta \vec{d}_{2,i}}\}_{\forall i \in [U]}, \Gamma) \leftarrow \mathcal{C}$;
- (2) $b \leftarrow \{0, 1\}$;
- (3) $\text{KeyGen}(\text{Gen})\{$
 - (3.1) let $\gamma_1^*, \gamma_2^*, \tau_1, \tau_2, \tau_3 \xleftarrow{\$} \mathbb{Z}_p$;
 - (3.2) let $\left\{ \begin{array}{l} \Gamma_0 := (T_{1,i} \leftarrow g^{\gamma_1^* \tau_1 \eta \vec{d}_{1,i} + \gamma_2^* \tau_2 \beta \vec{d}_{2,i}}, T_2 \leftarrow g^{\gamma_1^* \tau_1 \eta \vec{d}_{1,i} + \gamma_2^* \tau_2 \beta \vec{d}_{2,i}}), \\ \Gamma_1 := (T_{1,i} \leftarrow g^{\gamma_1^* \tau_1 \eta \vec{d}_{1,i} + \gamma_2^* \tau_2 \beta \vec{d}_{2,i} + \tau_3 \vec{d}_{1,i}}, T_2 \leftarrow g^{\gamma_1^* \tau_1 \eta \vec{d}_{1,i} + \gamma_2^* \tau_2 \beta \vec{d}_{2,i} + \tau_3 \vec{d}_{1,i}}) \end{array} \right\}_{\forall i \in [U]}$;
 - (3.3) return $(\mathcal{C} \leftarrow \{\Gamma_b\})$;
 - (3.4) }
- (4) $\alpha_1, \alpha_2 \xleftarrow{\$} \mathbb{Z}_p$;
- (5) $\alpha_1 \leftarrow \eta \alpha_1, \alpha_2 \leftarrow \beta \alpha_2$;
- (6) let $\text{pp} = \{G, p, g^{\vec{d}_{1,i}}, g^{\vec{d}_{2,i}}, g^{\vec{d}_3}, g^{\vec{d}_4}\}_{\forall i \in [U]}, e_g(g^{\vec{d}_{1,i}}, g^{\eta \vec{d}_{1,i}})^{\alpha_1}, e_g(g^{\vec{d}_{2,i}}, g^{\beta \vec{d}_{2,i}})^{\alpha_2}$;
- (7) Output $\text{pp} \rightarrow \mathcal{A}$;
- (8) $(K_{1,[1,\dots,Q_1]}, K_2) \leftarrow \mathcal{A}(\mathbb{A}_1[1, \dots, Q_1])$ with;
 - (8.1) $\gamma_1^*, \alpha_3 \xleftarrow{\$} \mathbb{Z}_p$;
 - (8.2) $\mathcal{C}(\text{pp}, \alpha_1, \alpha_2, \alpha_3, \Gamma_0, \Gamma_1)$
 - (8.3) let $\vec{v}_1, \vec{v}_2, \vec{v}_3 \xleftarrow{\$} \mathbb{Z}_p^t$;
 - (8.4) let $\lambda_i \leftarrow \mathbb{M}_1^T * \vec{v}_1$, where α_1 is the secret, $\mathbb{A}_1 \equiv \mathbb{M}_1$;
 - (8.5) let $\omega_i \leftarrow \mathbb{M}_1^T * \vec{v}_2$, where α_2 is the secret;
 - (8.6) let $\Phi_i \leftarrow \mathbb{M}_1^T * \vec{v}_3$, where α_3 is the secret;
 - (8.7) let $K_2 = T_2$;
 - (8.8) for $i = 1$ to Q_1 ;
 - (8.8.1) let $\vec{d}_{1,i}^* \rightarrow \vec{d}_{1,i} \lambda_i$;
 - (8.8.2) let $\vec{d}_{2,i}^* \rightarrow \vec{d}_{2,i} \omega_i$;
 - (8.8.3) let $\vec{d}_{5,i}^* \rightarrow \vec{d}_{5,i} \Phi_i$;
 - (8.8.4) write $K_{1,i} = g^{\gamma_1^* \eta \vec{d}_{1,i}^*} T_{1,i}$;
 - (8.8.5) Output $\{(K_{1,i}, K_2) \forall i \in \mathbb{A}_1\} \rightarrow \mathcal{A}$;
- (9) }
- (10) $(C_0, C_{1,i}, C_2) \leftarrow \mathcal{A}(\text{msg}_0^*, \text{msg}_1^*, S^*)$ with;
- (11) $\mathcal{C}(\text{pp}, \text{msg}_0^*, \text{msg}_1^*, S^*)\{$
 - (12) let $\vec{s}_1, \vec{s}_2, \gamma_1, \gamma_2 \xleftarrow{\$} \mathbb{Z}_p$
 - (13) let $C_0 = \text{msg}_b \cdot e(g, g)^{\alpha_1 \vec{s}_1 \delta + \alpha_2 \vec{s}_2 \delta}$;
 - (14) let $C_2 = g^{\vec{s}_1 \vec{b}_3 + \vec{s}_2 \vec{b}_4} = (g^{\vec{d}_3})^{\vec{s}_1} \cdot (g^{\vec{d}_4})^{\vec{s}_2}$;
 - (15) for i to $|S^*|$
 - (15.1) $\text{guide}(S^* \neq \mathbb{A}_1)$;
 - (15.2) Write $g^{\vec{s}_1 \gamma_1 \vec{b}_{1,i} + \vec{s}_2 \gamma_2 \vec{b}_{2,i}} = (g^{\vec{d}_{1,i}})^{\vec{s}_1 + \gamma_1} (g^{\vec{d}_{2,i}})^{\vec{s}_2 + \gamma_2}$;
 - (15.3) Output $\{(C_0, C_{1,i}, C_2) \forall i \in S^*\} \rightarrow \mathcal{A}$;
- (16) }
- (17) $(K_{1,[Q_1+1,\dots,Q]}, K_2) \leftarrow \mathcal{A}(\mathbb{A}_1[Q_1 + 1, \dots, Q])$ with;
 - (17.1) $\gamma_1^*, \alpha_3 \xleftarrow{\$} \mathbb{Z}_p$;
 - (17.2) $\mathcal{C}(\text{pp}, \alpha_1, \alpha_2, \alpha_3, \Gamma_0, \Gamma_1)$
 - (17.3) let $\vec{v}_1, \vec{v}_2, \vec{v}_3 \xleftarrow{\$} \mathbb{Z}_p^t$;

- (17.4) let $\lambda_i \leftarrow \mathbb{M}_1^T * \vec{v}_1$, where α_1 is the secret, $\mathbb{A}_1 \equiv \mathbb{M}_1$;
- (17.5) let $\omega_i \leftarrow \mathbb{M}_1^T * \vec{v}_2$, where α_2 is the secret;
- (17.6) let $\Phi_i \leftarrow \mathbb{M}_1^T * \vec{v}_3$, where α_3 is the secret;
- (17.7) let $K_2 = T_2$;
- (17.8) for $i = Q_1 + 1$ to Q

(17.8.1) guard $(\mathbb{A}_1 \notin S^*)$

(17.8.2) let $\vec{d}_{1,i}^* \rightarrow \vec{d}_{1,i} \lambda_i$;

(17.8.3) let $\vec{d}_{2,i}^* \rightarrow \vec{d}_{2,i} \omega_i$;

(17.8.4) let $\vec{d}_{5,i}^* \rightarrow \vec{d}_{5,i} \Phi_i$;

(17.8.5) write $K_{1,i} = g^{\gamma_1^* \eta \vec{d}_{1,i}^*} T_{1,i}$

(17.8.6) Output $\{(K_{1,i}, K_2) \forall i \in \mathbb{A}_1\} \rightarrow \mathcal{A}$

(18) $b = b'$

To simulate either $\text{Game}_{\text{real}}$ or Game_0 , Algorithm \mathcal{C} sets the bases for the construction as

$$\left\{ \vec{b}_{1,i} = \eta \vec{b}_{1,i}, \vec{b}_{2,i} = \beta \vec{b}_{2,i}, \vec{b}_3 = \eta \vec{b}_3, \vec{b}_4 = \beta \vec{b}_4, \vec{b}_{5,i} = \vec{b}_{5,i}, \vec{b}_6 = \vec{b}_6 \right\}_{\forall i \in [U]},$$

$$\left\{ \vec{b}_{1,i}^* = \eta^{-1} \vec{d}_{1,i}^*, \vec{b}_{2,i}^* = \beta^{-1} \vec{d}_{2,i}^*, \vec{b}_3^* = \eta^{-1} \vec{d}_3^*, \vec{b}_4^* = \beta^{-1} \vec{d}_4^*, \vec{b}_{5,i}^* = \vec{d}_{5,i}^*, \vec{b}_6^* = \vec{d}_6^* \right\}_{\forall i \in [U]}.$$

(21)

We assert that these are well distributed because $(\mathbb{D}^*, \mathbb{D})$, $(\mathbb{D}_0^*, \mathbb{D}_0)$, etc., are chosen randomly up to sharing the same value δ . Implicitly, \mathcal{C} selects $\alpha_1, \alpha_2 \in \mathbb{Z}_p$ and sets $\alpha_1 = \eta \alpha_1, \alpha_2 = \beta \alpha_2$. Then, C produces

$$e(g, g)^{\alpha_1 s_1} = e\left(g^{\vec{d}_{1,i}}, g^{\eta \vec{d}_{1,i}}\right)^{\alpha_1},$$

$$e(g, g)^{\alpha_2 s_2 \delta} = e\left(g^{\vec{d}_{2,i}}, g^{\beta \eta \vec{d}_{2,i}}\right)^{\alpha_2}.$$

(22)

In line 1, the subspace assumption adversary \mathcal{C} is given the public parameters of the system and its challenge Γ . In line 8, \mathcal{A} requests for its private keys, which \mathcal{C} replies correctly to. In line 10, \mathcal{A} sends two messages $\{\text{msg}_0^*, \text{msg}_1^*\}$ of the same length with the attribute S^* to \mathcal{C} and requests for the challenge ciphertext. In response, \mathcal{C} outputs the correct ciphertext tuple $\{(C_0, C_{1,i}, C_2) \forall i \in S^*\}$ to \mathcal{A} with the restriction that the attribute S^* does not satisfy the access structure which is enforced by the guard. In line 17, \mathcal{A} requests for the private key $\{(K_{1,i}, K_2) \forall i \in \mathbb{A}_1\}$ for the second time. \mathcal{C} outputs the correct private key to \mathcal{A} with the restriction that the access structure \mathbb{A}_1 does not satisfy the attribute set S^* of the previously queried ciphertext. Eventually, \mathcal{C} outputs \mathcal{A} 's guess as its own guess. By analysing this game, when the τ_3 terms are absent in the private key, then \mathcal{C} correctly simulates $\text{Game}_{\text{Real}}$. In this instance, Γ_0 is used in generating the private key. When τ_3 terms are present, then \mathcal{C} correctly simulates Game_0 . In this case, Γ_1 is used in generating the private key. Therefore, \mathcal{C} can capitalize on algorithm \mathcal{A} 's non-negligible advantage in

distinguishing between these two games to obtain a non-negligible advantage against subspace assumption.

Lemma 2. *Under the subspace assumption, no PPT adversary can have a non-negligible advantage in distinguishing between Game_{k-1} and Game_k^N for any k from 1 to Q .*

Proof: Suppose a PPT algorithm \mathcal{A} achieves a non-negligible advantage in distinguishing Game_{K-1} from Game_k^N , then we will construct a PPT algorithm \mathcal{E} to break the subspace assumption. We will set the parameters $n_i = 3, m = \mathcal{U} + 2, k_i = 1$, for two values of i and $n_i = 6, k_i = 2$ for the remaining values of i in the subspace assumption. To correctly align the assumption notation with our scheme notation, we hereby designate the bases of the assumption as $(\mathbb{B}, \mathbb{B}^*), (\mathbb{B}_0, \mathbb{B}_0^*) \in \text{Dual}(\mathbb{Z}_p^3, \delta)$ and $(\mathbb{B}_1, \mathbb{B}_1^*), \dots, (\mathbb{B}_{\mathcal{U}}, \mathbb{B}_{\mathcal{U}}^*) \in \text{Dual}(\mathbb{Z}_p^6, \delta)$. We will exclude the μ_3 term because it is not applicable here. The procedure for simulating Game_{k-1} and Game_k^N is described as follows:

- (1) $(G, p, g^{\vec{d}_3}, g^{\eta \vec{d}_3}, g^{\beta \vec{d}_4}, \{g^{\vec{d}_{1,i}}, g^{\vec{d}_{2,i}}, g^{\eta \vec{d}_{1,i}}, g^{\beta \vec{d}_{2,i}}\}_{\forall i \in [\mathcal{U}]}, \Gamma, \mathcal{U}) \leftarrow \mathcal{E}$;
- (2) $b \leftarrow \{0, 1\}$;
- (3) $\text{KeyGen}(\text{Gen})\{$
 - (3.1) let $\gamma_1^*, \gamma_2^*, \tau_1, \tau_2, \tau_3 \xleftarrow{\$} \mathbb{Z}_p$;
 - (3.2) let $\left\{ \Gamma_0 := (T_{1,i} \leftarrow g^{\gamma_1^* \tau_1 \eta \vec{d}_{1,i} + \gamma_2^* \tau_2 \beta \vec{d}_{2,i}}, T_2 \leftarrow g^{\gamma_1^* \tau_1 \eta \vec{d}_3 + \gamma_2^* \tau_2 \beta \vec{d}_4}, \Gamma_1 := (T_{1,i} \leftarrow g^{\gamma_1^* \tau_1 \eta \vec{d}_{1,i}} + \gamma_2^* \tau_2 \beta \vec{d}_{2,i}}, T_{2,i} \leftarrow g^{\gamma_1^* \tau_1 \eta \vec{d}_3 + \gamma_2^* \tau_2 \beta \vec{d}_4 + \tau_3 \vec{d}_6}) \right\}_{\forall i \in [\mathcal{U}]}$;
 - (3.3) return $(\mathcal{E} \leftarrow \{\Gamma_b\})$;
 - (3.4) }
- (4) $\text{Enc}(\text{Gen})\{$
 - (4.1) let $\mu_1, \mu_2, \mu_3, \gamma_1, \gamma_2 \xleftarrow{\$} \mathbb{Z}_p$;
 - (4.2) let $\left\{ \Gamma_0^* = (\mathcal{U}_{1,i} \leftarrow g^{\mu_1 \gamma_1 \vec{b}_{1,i} + \mu_2 \gamma_2 \vec{b}_{2,i}}, \mathcal{U}_2 \leftarrow g^{\mu_1 \gamma_1 \vec{b}_3 + \mu_2 \gamma_2 \vec{b}_4}) \Gamma_1^* = (\mathcal{U}_{1,i} \leftarrow g^{\mu_1 \gamma_1 \vec{b}_{1,i} + \mu_2 \gamma_2 \vec{b}_{2,i}} + \mu_3 \vec{b}_{5,i}, \mathcal{U}_2 \leftarrow g^{\mu_1 \gamma_1 \vec{b}_3 + \mu_2 \gamma_2 \vec{b}_4 + \mu_3 \vec{b}_{5,i}}) \right\}_{\forall i \in [\mathcal{U}]}$;
 - (4.3) return $(\mathcal{E} \leftarrow \{\Gamma_b^*\})$;
 - (4.4) }
- (5) $\alpha_1, \alpha_2 \xleftarrow{\$} \mathbb{Z}_p$;
- (6) let $\text{pp} = \left\{ \left\{ G, p, g^{\vec{d}_{1,i}}, g^{\vec{d}_{2,i}}, g^{\vec{d}_3}, g^{\vec{d}_4} \right\}_{\forall i \in [\mathcal{U}]}, \text{en}(g^{\vec{d}_1}, g^{\eta \vec{d}_1})^{\alpha_1}, \text{en}(g^{\vec{d}_1}, g^{\beta \vec{d}_2})^{\alpha_2} \right\}$;
- (7) Output $\text{pp} \rightarrow \mathcal{A}$;
- (8) $(K_{1,[1,\dots,Q_1]}, K_2) \leftarrow \mathcal{A}(\mathbb{A}_1[1, \dots, Q_1])$ with;
 - (8.1) $\gamma_1^*, \alpha_3, \xleftarrow{\$} \mathbb{Z}_p$;
 - (8.2) $\mathcal{E}(\text{pp}, \alpha_1, \alpha_2, \alpha_3, \Gamma_0, \Gamma_1)\{$
 - (8.3) let $\vec{v}_1, \vec{v}_2, \vec{v}_3 \xleftarrow{\mathbb{Z}_p^t}$;
 - (8.4) let $\lambda_i \leftarrow \mathbb{M}_1^T * \vec{v}_1$, where α_1 is the secret, $\mathbb{A}_1 \equiv \mathbb{M}_1$;
 - (8.5) let $\omega_i \leftarrow \mathbb{M}_1^T * \vec{v}_2$, where α_2 is the secret;
 - (8.6) let $\Phi_i \leftarrow \mathbb{M}_1^T * \vec{v}_3$, where α_3 is the secret;
 - (8.7) let $K_2 = T_2$;

(8.8) for $i = 1$ to Q_1

$$(8.8.1) \text{ let } \vec{d}_{1,i}^* \xrightarrow{\$} \vec{d}_{1,i}^* \lambda_i;$$

$$(8.8.2) \text{ let } \vec{d}_{2,i}^* \xrightarrow{\$} \vec{d}_{2,i}^* \omega_i;$$

$$(8.8.3) \text{ let } \vec{d}_{2,i}^* \xrightarrow{\$} \vec{d}_{2,i}^* \omega_i;$$

$$(8.8.4) \text{ write } K_{1,i} = g^{\lambda_i \gamma_1^* \vec{b}_{1,i}} T_{1,i};$$

$$(8.8.5) \text{ Output } \{(K_{1,i}, K_2) \forall i \in \mathbb{A}_1\} \rightarrow \mathcal{A};$$

(9) }

$$(10) (C_0, C_{1,i}, C_2) \leftarrow \mathcal{A}(\text{msg}_0^*, \text{msg}_1^*, S^*) \text{ with;}$$

$$(11) \mathcal{E}(\text{pp}, \text{msg}_0^*, \text{msg}_1^*, S^*)\{$$

$$(12) \text{ let } \vec{s}_1, \vec{s}_2, \vec{s}_3 \xleftarrow{\$} \mathbb{Z}_p;$$

$$(13) \text{ let } \vec{s}_1 \leftarrow \mu_1, \vec{s}_2 \leftarrow \mu_2, \vec{s}_3 \leftarrow \mu_3;$$

$$(14) \text{ let } C_0 = \text{msg}_b \cdot e(g, g)^{\alpha_1 \vec{s}_1 \delta + \alpha_2 \vec{s}_2 \delta};$$

$$(15) \text{ let } C_2 = \mathcal{U}_2;$$

$$(16) \text{ for } i \text{ to } |S^*|$$

$$(16.1) \text{ guide}(S^* \neq \mathbb{A}_1);$$

$$(16.2) \text{ write } C_{1,i} = \mathcal{U}_{1,i};$$

$$(16.3) \text{ Output } \{(C_0, C_{1,i}, C_2) \forall i \in S^*\} \rightarrow \mathcal{A};$$

(17) }

$$(18) (K_{1,[Q_1+1,\dots,Q]}, K_2) \leftarrow \mathcal{A}(\mathbb{A}_1[Q_1 + 1, \dots, Q]) \text{ with;}$$

$$(18.1) \gamma_1^*, \alpha_3, \xleftarrow{\$} \mathbb{Z}_p;$$

$$(18.2) \mathcal{E}(\text{pp}, \alpha_1, \alpha_2, \alpha_3, \Gamma_0, \Gamma_1)\{$$

$$(18.3) \text{ let } \vec{v}_1, \vec{v}_2, \vec{v}_3 \xleftarrow{\mathbb{Z}_p^t};$$

$$(18.4) \text{ let } \lambda_i \leftarrow \mathbb{M}_1^T * \vec{v}_1, \text{ where } \alpha_1 \text{ is the secret, } \mathbb{A}_1 \equiv \mathbb{M}_1;$$

$$(18.5) \text{ let } \omega_i \leftarrow \mathbb{M}_1^T * \vec{v}_2, \text{ where } \alpha_2 \text{ is the secret;}$$

$$(18.6) \text{ let } \Phi_i \leftarrow \mathbb{M}_1^T * \vec{v}_3, \text{ where } \alpha_3 \text{ is the secret;}$$

$$(18.7) \text{ let } K_2 = T_2;$$

$$(18.8) \text{ for } i = Q_1 + 1 \text{ to } Q$$

$$(18.8.1) \text{ guard } (\mathbb{A}_1 \neq S^*)$$

$$(18.8.2) \text{ let } \vec{d}_{1,i}^* \xrightarrow{\$} \vec{d}_{1,i}^* \lambda_i;$$

$$(18.8.3) \text{ let } \vec{d}_{2,i}^* \xrightarrow{\$} \vec{d}_{2,i}^* \omega_i;$$

$$(18.8.4) \text{ let } \vec{d}_{5,i}^* \xrightarrow{\$} \vec{d}_{5,i}^* \Phi_i;$$

$$(18.8.5) \text{ write } K_{1,i} = g^{\lambda_i \gamma_1^* \vec{d}_{1,i}} T_{1,i};$$

$$(18.8.6) \text{ Output } \{(K_{1,i}, K_2) \forall i \in \mathbb{A}_1\} \rightarrow \mathcal{A};$$

(19) $b = b'$;

To simulate either Game_{K-1} or Game_K^N , algorithm \mathcal{E} sets the bases for the construction as

$$\begin{aligned} & \left\{ \vec{b}_{1,i} = \eta \vec{b}_{1,i}, \vec{b}_{2,i} = \beta \vec{b}_{2,i}, \vec{b}_3 = \eta \vec{b}_3, \vec{b}_4 = \beta \vec{b}_4, \right. \\ & \left. \vec{b}_{5,i} = \vec{b}_{5,i}, \vec{b}_6 = \vec{b}_6 \right\}_{\forall i \in [\mathcal{U}]}, \\ & \left\{ \vec{b}_{1,i}^* = \eta^{-1} \vec{d}_{1,i}^*, \vec{b}_{2,i}^* = \beta^{-1} \vec{d}_{2,i}^*, \vec{b}_3^* = \eta^{-1} \vec{d}_3^*, \right. \\ & \left. \vec{b}_4^* = \beta^{-1} \vec{d}_4^*, \vec{b}_{5,i}^* = \vec{d}_{5,i}^*, \vec{b}_6^* = \vec{d}_6^* \right\}_{\forall i \in [\mathcal{U}]}. \end{aligned} \quad (23)$$

We assert that these are well distributed because $(\mathbb{D}^*, \mathbb{D})$, $(\mathbb{D}_0^*, \mathbb{D}_0)$, etc., are chosen randomly up to sharing the same value δ . Implicitly, \mathcal{C} selects $\alpha_1, \alpha_2 \in \mathbb{Z}_p$ and sets $\alpha_1 = \eta\alpha_1, \alpha_2 = \beta\alpha_2$. Then, C produces

$$\begin{aligned} e(g, g)^{\alpha_1 s_1} &= \text{en} \left(g^{\bar{d}_1}, g^{\eta \bar{d}_1^*} \right)^{\alpha_1}, \\ e(g, g)^{\alpha_2 s_2 \delta} &= \text{en} \left(g^{\bar{d}_1}, g^{\beta \eta \bar{d}_2^*} \right)^{\alpha_2}. \end{aligned} \quad (24)$$

In line 1, the subspace assumption adversary \mathcal{C} is given the public parameters of the system and its challenge Γ, \mathcal{U} . In line 8, \mathcal{A} requests for its private keys, which \mathcal{C} replies correctly to. In line 10, \mathcal{A} sends two messages $\{\text{msg}_0^*, \text{msg}_1^*\}$ of the same length with the attribute S^* to \mathcal{C} and requests for the challenge ciphertext. In response, \mathcal{C} outputs the correct ciphertext tuple $\{(C_0, C_{1,i}, C_2) \forall i \in S^*\}$ to \mathcal{A} with the restriction that the attribute S^* does not satisfy the access structure \mathbb{A}_1 which is enforced by the guard. In line 18, \mathcal{A} requests for the private key $\{(K_{1,i}, K_2) \forall i \in \mathbb{A}_1\}$ for the second time. \mathcal{C} outputs the correct private key to \mathcal{A} with the restriction that the access structure \mathbb{A}_1 does not satisfy the attribute set S^* of the previously queried ciphertext. Eventually, \mathcal{C} outputs \mathcal{A} 's guess as its own guess. By analysing this game, when the extra components $\mu_3 b_{5,i}$ and $\mu_3 b_6$ on $C_{1,i}$ and C_2 , respectively, are present, then the ciphertext is SF ciphertext (i.e., $\Gamma^* = \Gamma_1^*$); otherwise, it is normal ciphertext (i.e., $\Gamma^* = \Gamma_0^*$). Hence, \mathcal{C} is capable to simulate normal and SF ciphertext. When the τ_3 terms are absent in the private key, then \mathcal{C} correctly simulates Game_{k-1} . In this instance, Γ_0 is used in generating the private key. When τ_3 terms are present, then \mathcal{C} correctly simulates Game_k^N . In this case, Γ_1 is used in generating the private key. Therefore, \mathcal{C} can capitalize on algorithm A 's non-negligible advantage in distinguishing between these two games to obtain a non-negligible advantage against subspace assumption.

Lemma 3. *Under the TPDH assumption, no PPT adversary can have a non-negligible advantage in distinguishing between Game_k^N and Game_k^T for any k from 1 to Q_1 (note that these are phase 1 queries).*

Proof: Suppose a PPT algorithm \mathcal{A} achieves a non-negligible advantage in distinguishing Game_k^N from Game_k^T , then we will construct a PPT algorithm \mathcal{C} to break the TPDH assumption. \mathcal{C} gets g, g^x, g^y, g^z where T is either g^{xyz} or random element of G . Algorithm \mathcal{C} simulates either Game_k^N from Game_k^T based on the nature of T . \mathcal{C} picks a random dual orthonormal bases $(\mathbb{B}, \mathbb{B}^*)$, $(\mathbb{B}_0, \mathbb{B}_0^*)$ of 3 dimensions and $(\mathbb{B}_1, \mathbb{B}_1^*), \dots, (\mathbb{B}_{\mathcal{U}}, \mathbb{B}_{\mathcal{U}}^*)$ of 6 dimensions n , all with the same value δ . The procedure for simulating Game_k^N and Game_k^T is described as follows:

- (1) $(G, p, g^{\bar{b}_3}, g^{\bar{b}_3^*}, g^{\bar{b}_4}, \{g^{\bar{b}_{1,i}}, g^{\bar{b}_{2,i}}, g^{\bar{b}_{1,i}^*}, g^{\bar{b}_{2,i}^*}\}_{\forall i \in [\mathcal{U}]}, \Gamma) \leftarrow \mathcal{C}$;
- (2) $b \leftarrow \{0, 1\}$;
- (3) KeyGen (Gen){
 - (3.1) let $x, y, z \leftarrow \mathbb{Z}_p$;

- (3.2) let $\bar{b}_{5,i}^* \leftarrow \text{Dual}(\mathbb{Z}_p^6, \delta), \bar{b}_6^* \leftarrow \text{Dual}(\mathbb{Z}_p^3, \delta)$;
- (3.3) $\mathcal{C} \leftarrow \{g, g^x, g^y, g^z, T\}$;

- (3.4) let $\{g, g^x, g^y, T\}$;

- (3.5) return $\left(\left\{ \bar{b}_{5,i}^* \right\}_{\forall i \in [\mathcal{U}]}, \bar{b}_6^* \right)$;
- (3.6) $\{ b? (T \leftarrow g^{xyz}): (T \leftarrow G_1) \}$

- (4) $\alpha_1, \alpha_2 \xleftarrow{\$} \mathbb{Z}_p$;

- (5) let $\text{pp} = \left\{ \left\{ G, p, g^{\bar{b}_{1,i}}, g^{\bar{b}_{2,i}}, g^{\bar{b}_3}, g^{\bar{b}_4} \right\}_{\forall i \in [\mathcal{U}]}, e(g, g)^{\alpha_1}, e(g, g)^{\alpha_2} \right\}$;

- (6) Output $\text{pp} \rightarrow \mathcal{A}$;

- (7) $(K_{1, [Q_1+1, \dots, Q]}, K_2) \leftarrow \mathcal{A}(\mathbb{A}_1 [Q_1 + 1, \dots, Q])$ with;

- (7.1) $\mathcal{C}(\text{pp}, \alpha_1, \alpha_2, \alpha_3, \Gamma_0, \Gamma_1) \{$

- (7.2) $y_1^*, y_2^*, \alpha_3, r_1, r_2, a_1, a_2, a_3, a_4 \xleftarrow{\$} \mathbb{Z}_p$;

- (7.3) let $a_4 = z^{-1}, \alpha_3 \leftarrow 0 \in \mathbb{Z}_p$;

- (7.4) let $\bar{v}_1, \bar{v}_2, \bar{v}_3 \leftarrow \mathbb{Z}_p^t$;

- (7.5) let $\lambda_i \leftarrow \mathbb{M}_1^T * \bar{v}_1$, where α_1 is the secret, $\mathbb{A}_1 \equiv \mathbb{M}_1$;

- (7.6) let $\omega_i \leftarrow \mathbb{M}_1^T * \bar{v}_2$, where α_2 is the secret, $\mathbb{A}_1 \equiv \mathbb{M}_1$;

- (7.7) let $\Phi_i \leftarrow \mathbb{M}_1^T * \bar{v}_3$, where α_3 is the secret, $\mathbb{A}_1 \equiv \mathbb{M}_1$;

- (7.8) let $K_2 = g^{a_1 r_1 \bar{b}_3 + a_2 r_2 \bar{b}_4 + a_3 c \bar{b}_6}$;

- (7.9) for $i = Q_1 + 1$ to Q

- (7.9.1) write $K_{1,i} = K_{1,i} = g^{\lambda_i y_1^* \bar{b}_{1,i} + a_1 r_1 y_1^* \bar{b}_{1,i} + \omega_i y_2^* \bar{b}_{2,i} + a_2 r_2 y_2^* \bar{b}_{2,i} + \Phi_i a_4 \bar{b}_{5,i} + a_3 \bar{b}_{5,i}}$;

- (7.9.2) Output $\{(K_{1,i}, K_2) \forall i \in \mathbb{A}_1\} \rightarrow \mathcal{A}$;

- (8) }

- (9) $(C_0, C_{1,i}, C_2) \leftarrow \mathcal{A}(\text{msg}_0^*, \text{msg}_1^*, S^*)$ with;

- (10) $\mathcal{C}(\text{pp}, \text{msg}_0^*, \text{msg}_1^*, S^*) \{$

- (11) let $\tilde{s}_1, \tilde{s}_2, \tilde{s}_3 \leftarrow \mathbb{Z}_p$;

- (12) let $C_0 = \text{msg}_b \cdot e(g, g)^{\alpha_1 \tilde{s}_1 \delta + \alpha_2 \tilde{s}_2 \delta}$;

- (13) let $C_2 = g^{\tilde{s}_1 b_3 + \tilde{s}_2 b_4 + \tilde{s}_3 b_6}$;

- (14) for i to $|S^*|$

- (14.1) guide $(S^* \not\in \mathbb{A}_1)$;

- (14.2) let $\bar{d}_{5,i} \leftarrow \text{Dual}(\mathbb{Z}_p^6, \delta)$;

- (14.3) write $C_{1,i} = g^{s_1 y_1 \bar{b}_{1,i} + s_2 y_2 \bar{b}_{2,i} + \tilde{s}_3 \bar{b}_{5,i}}$;

- (14.4) Output $\{(C_0, C_{1,i}, C_2) \forall i \in S^*\} \rightarrow \mathcal{A}$;

- (15) }

- (16) $b = b'$;

To simulate either Game_k^N or Game_k^T , algorithm \mathcal{C} sets the bases for the construction as

$$\begin{aligned} \left\{ \bar{b}_{1,i} = \bar{d}_{1,i}, \bar{b}_{2,i} = \bar{d}_{2,i}, \bar{b}_3 = \bar{d}_3, \bar{b}_4 = \bar{d}_4, \right. \\ \left. \bar{b}_{5,i} = \bar{d}_{5,i}, \bar{b}_6 = \bar{d}_6 \right\}_{\forall i \in [\mathcal{U}]}, \\ \left\{ \bar{b}_{1,i}^* = \bar{d}_{1,i}^*, \bar{b}_{2,i}^* = \bar{d}_{2,i}^*, \bar{b}_3^* = \bar{d}_3^*, \bar{b}_4^* = \bar{d}_4^*, \right. \\ \left. \bar{b}_{5,i}^* = (xy)^{-1} \bar{d}_{5,i}^*, \bar{b}_6^* = \bar{d}_6^* \right\}_{\forall i \in [\mathcal{U}]} \end{aligned} \quad (25)$$

We assert that these are well distributed because $(\mathbb{B}^*, \mathbb{B}), (\mathbb{B}_0^*, \mathbb{B}_0)$, etc., are chosen randomly up to sharing the same value δ . The SF components $\{\bar{b}_{5,i}^*, \bar{b}_6^*\}$ and $\{\bar{b}_{5,i}^*, \bar{b}_6^*\}$ can supply fresh parameters to randomize the ciphertext and the private key, respectively.

In line 1, the TPDH assumption adversary \mathcal{C} is given the public parameters of the system and its challenge T . In line 7, \mathcal{A} requests for its private keys, which \mathcal{C} replies correctly to. In line 9, \mathcal{A} sends two messages $\{\text{msg}_0^*, \text{msg}_1^*\}$ of the same length with the attribute S^* to \mathcal{C} and requests for the challenge ciphertext. In response, \mathcal{C} outputs the correct ciphertext tuple $\{(C_0, C_{1,i}, C_2) \forall i \in S^*\}$ to \mathcal{A} with the restriction that the attribute S^* does not satisfy the access structure \mathbb{A}_1 which is enforced by the guard. Eventually, \mathcal{C} outputs \mathcal{A} 's guess as its own guess. By analysing this game, if $T = g^{xyz}$, then the power vector becomes $xyz(xyz)^{-1} \Phi_i \bar{d}_{5,i}^* = \Phi_i \bar{b}_{5,i}^*$ as needed for the nominal SF key. Alternatively, this power vector is distributed as random multiples of $\bar{b}_{5,i}^*$, which is required for an ephemeral SF key. Hence, when $T = g^{xyz}$, then \mathcal{C} has successfully simulated Game_k^N , and if T is a random group element, then \mathcal{C} has successfully simulated Game_k^T . Therefore, \mathcal{C} can capitalize on \mathcal{A} 's non-negligible advantage in distinguishing between these two games to obtain a non-negligible advantage against the TPDH assumption.

Lemma 4. *Under the TPDH assumption, no PPT adversary can have a non-negligible advantage in distinguishing between Game_k^N and Game_k^T for any $k > Q_1$ (note that these are phase 2 queries).*

Proof: Suppose a PPT algorithm \mathcal{A} achieves a non-negligible advantage in distinguishing Game_{k_1} and Game_k^N for some k such that $Q_1 < k \leq Q$. We will construct a PPT algorithm \mathcal{C} to break the TPDH assumption. \mathcal{C} gets g, g^x, g^y, g^z, T where T is either g^{xyz} or a random element of \mathbb{B}^* . \mathcal{C} will simulate either Game_k^N or Game_k^T with algorithm \mathcal{A} based on T . \mathcal{C} picks a random dual orthonormal bases $(\mathbb{B}, \mathbb{B}^*), (\mathbb{B}_0, \mathbb{B}_0^*)$ of 3 dimensions and $(\mathbb{B}_1, \mathbb{B}_1^*), \dots, (\mathbb{B}_{Q_2}, \mathbb{B}_{Q_2}^*)$ of 6 dimensions, all with the same value δ . The procedure for simulating Game_k^N and Game_k^T is described as follows:

- (1) $(G, p, g^{\bar{b}_3}, g^{\bar{b}_3^*}, g^{\bar{b}_4}, \{g^{\bar{b}_{1,i}}, g^{\bar{b}_{2,i}}, g^{\bar{b}_{1,i}^*}, g^{\bar{b}_{2,i}^*}\}_{\forall i \in [U]} \Gamma) \leftarrow \mathcal{C}$;
- (2) $b \leftarrow \{0, 1\}$;
- (3) $\text{KeyGen}(\text{Gen})\{$
 - (3.1) let $x, y, z, \{r_i\}_{\forall i \in S} \xleftarrow{\$} \mathbb{Z}_p$
 - (3.2) let $\bar{b}_{5,i}^* \xleftarrow{\text{Dual}}(\mathbb{Z}_p^6, \delta), \bar{b}_6^* \xleftarrow{\text{Dual}}(\mathbb{Z}_p^3, \delta)$;
 - (3.3) $\bar{b}_{5,i}^* \rightarrow \{x^{-1} r_i \bar{d}_{5,i}^*\}_{\forall i \in S}$;
 - (3.4) $\mathcal{C} \leftarrow \{g, g^x, g^y, g^z, T\}$;
 - (3.5) let $\{g, g^x, g^y, T\}$;

$$(3.6) \text{ return } \left(\left\{ r_i, \bar{b}_{5,i}^* \right\}_{\forall i \in [U]}, \bar{b}_6^* \right);$$

$$(3.7) \} b? (T \leftarrow g^{xyz}): (T \leftarrow G_1)$$

$$(4) \alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p;$$

$$(5) \text{ let } \text{pp} = \left\{ \left\{ G, p, g^{\bar{b}_{1,i}}, g^{\bar{b}_{2,i}}, g^{\bar{b}_3}, g^{\bar{b}_4} \right\}_{\forall i \in [U]}, e(g, g)^{\alpha_1}, e(g, g)^{\alpha_2} \right\};$$

$$(6) \text{ Output } \text{pp} \rightarrow \mathcal{A}$$

$$(7) (C_0, C_{1,i}, C_2) \leftarrow \mathcal{A}(\text{msg}_0^*, \text{msg}_1^*, S^*) \text{ with};$$

$$(8) \mathcal{C}(\text{pp}, \text{msg}_0^*, \text{msg}_1^*, S^*)\{$$

$$(9) \text{ let } \tilde{s}_1, \tilde{s}_2, \tilde{s}_3 \leftarrow \mathbb{Z}_p;$$

$$(10) \text{ let } C_0 = \text{msg}_b \cdot e(g, g)^{\alpha_1 \tilde{s}_1 \delta + \alpha_2 \tilde{s}_2 \delta};$$

$$(11) \text{ let } \bar{b}_6 \leftarrow \text{Dual}(\mathbb{Z}_p^3, \delta);$$

$$(12) \text{ let } C_2 = g^{\tilde{s}_1 \bar{b}_3 + \tilde{s}_2 \bar{b}_4 + \tilde{s}_3 \bar{b}_6};$$

$$(13) \text{ for } i \text{ to } |S^*|$$

$$(13.1) \text{ let } \bar{b}_{5,i}^* \leftarrow \text{Dual}(\mathbb{Z}_p^6, \delta);$$

$$(13.2) \text{ write } C_{1,i} = g^{\tilde{s}_1 \gamma_1 \bar{b}_{1,i} + \tilde{s}_2 \gamma_2 \bar{b}_{2,i} + \tilde{s}_3 \bar{b}_{5,i}^*};$$

$$(13.3) \text{ Output } \{(C_0, C_{1,i}, C_2) \forall i \in S^*\} \rightarrow \mathcal{A};$$

$$(14) \}$$

$$(15) (K_{1, [Q_1+1, \dots, Q_1, \dots, Q_1]}, K_2) \leftarrow \mathcal{A}(\mathbb{A}_1 [Q_1 + 1, \dots, Q])$$

with;

$$(15.1) \mathcal{C}(\text{pp}, \alpha_1, \alpha_2, r_i, \bar{b}_{5,i}^*, \bar{b}_6^*, T)\{$$

$$(15.2) \gamma_1^*, \gamma_2^*, \alpha_3, r_1, r_2, a_4 \xleftarrow{\$} \mathbb{Z}_p'$$

$$(15.3) \text{ let } \alpha_3 \leftarrow 0 \in \mathbb{Z}_p;$$

$$(15.4) \text{ Let } \sum_{i=1}^n r_i \rightarrow \psi, \theta_i \leftarrow \{(1/r_i) \cdot (1/\psi)\}_{\forall i \in S^*};$$

$$(15.5) \text{ let } a_4 \leftarrow y^{-1}, z = \psi;$$

$$(15.6) \text{ let } \bar{v}_1, \bar{v}_2, \bar{v}_3 \xleftarrow{\$} \mathbb{Z}_p^T$$

$$(15.7) \text{ let } \lambda_i \leftarrow \mathbb{M}_1^T * \bar{v}_1, \text{ where } \alpha_1 \text{ is the secret, } \mathbb{A}_1 \equiv \mathbb{M}_1;$$

$$(15.8) \text{ let } \omega_i \leftarrow \mathbb{M}_1^T * \bar{v}_2, \text{ where } \alpha_2 \text{ is the secret, } \mathbb{A}_1 \equiv \mathbb{M}_1;$$

$$(15.9) \text{ let } \Phi_i \leftarrow \mathbb{M}_1^T * \bar{v}_3, \text{ where } \alpha_3 \text{ is the secret, } \mathbb{A}_1 \equiv \mathbb{M}_1;$$

$$(15.10) \text{ guide}(S^* \notin \mathbb{A}_1);$$

$$(15.11) \text{ let } K_2 = g^{a_1 r_1 \bar{b}_3 + a_2 r_2 \bar{b}_4 + a_3 c \bar{b}_6};$$

$$(15.12) \text{ for } i = Q_1 + 1 \text{ to } Q$$

$$(15.12.1) \text{ let } \tilde{\Phi}_i \leftarrow \Phi_i \theta_i;$$

$$(15.12.2) \text{ write } K_{1,i} = g^{\lambda_i \gamma_1^* \bar{b}_{1,i} + a_1 r_1}$$

$$\gamma_1^* \bar{b}_{1,i} + \omega_i \gamma_2^* \bar{b}_{2,i} + a_2 r_2 \gamma_2^* \bar{b}_{2,i} T^{a_4}$$

$$\tilde{\Phi}_i \bar{b}_{5,i} + a_3 \bar{b}_{5,i};$$

$$(15.12.3)$$

$$\text{Output } \{(K_{1,i}, K_2) \forall i \in \mathbb{A}_1\} \rightarrow \mathcal{A};$$

$$(16) b = b';$$

To simulate either Game_k^N or Game_k^T , algorithm \mathcal{C} sets the bases for the construction as

$$\{\bar{b}_{1,i} = (xy)^{-1} \bar{d}_{1,i}, \bar{b}_{2,i} = \bar{d}_{2,i}, \bar{b}_3 = \bar{d}_3, \bar{b}_4 = \bar{d}_4\}_{\forall i \in [U]},$$

$$\{\bar{b}_{1,i}^* = xy \bar{d}_{1,i}^*, \bar{b}_{2,i}^* = \bar{d}_{2,i}^*, \bar{b}_3^* = \bar{d}_3^*, \bar{b}_4^* = \bar{d}_4^*\}_{\forall i \in [U]}.$$

(26)

We assert that these are well distributed because $(\mathbb{B}^*, \mathbb{B})$, $(\mathbb{B}_0^*, \mathbb{B}_0)$, etc., are chosen randomly up to sharing the same value δ . The SF components $\{\bar{b}_{5,i}, \bar{b}_6\}$ and $\{\bar{b}_{5,i}^*, \bar{b}_6^*\}$ which will be set later can supply fresh parameters to randomize the ciphertext and the private key, respectively.

In line 1, the TPDH assumption adversary \mathcal{C} is given the public parameters of the system and its challenge T . In line 7, \mathcal{A} sends two messages $\{\text{msg}_0^*, \text{msg}_1^*\}$ of the same length with the attribute S^* to \mathcal{C} and requests for the challenge ciphertext. In response, \mathcal{C} outputs the correct ciphertext tuple $\{(C_0, C_{1,i}, C_2) \forall i \in S^*\}$ to \mathcal{A} . In line 15, \mathcal{A} requests for the private key $\{(K_{1,i}, K_2) \forall i \in \mathbb{A}_1\}$. \mathcal{C} outputs the correct private key to \mathcal{A} with the restriction that the access structure \mathbb{A}_1 does not satisfy the attribute set S^* of the previously queried ciphertext. Eventually, \mathcal{C} outputs \mathcal{A} 's guess as its own guess. By analysing this game, if $T = g^{xyz}$, then the power vector becomes $xyz(r_i z)^{-1} \Phi_i x^{-1} y^{-1} r_i \bar{d}_{5,i}^* = \Phi_i \bar{b}_{5,i}^*$ as needed for the nominal SF key. Alternatively, this power vector is distributed as random multiples of $\bar{b}_{5,i}^*$, which is required for an ephemeral SF key. Hence, when $T = g^{xyz}$, then \mathcal{C} has successfully simulated Game_k^N , and if T is a random group element, then \mathcal{C} has successfully simulated Game_k^T . Therefore, \mathcal{C} can capitalize on \mathcal{A} 's non-negligible advantage in distinguishing between these two games to obtain a non-negligible advantage against the TPDH assumption.

Lemma 5. *Under the subspace assumption, no PPT adversary can achieve a non-negligible advantage in distinguishing between Game_k^T and Game_k for any k from 1 to Q .*

Proof: This proof is closely indistinguishable to the proof of Lemma 1, except that \mathcal{C} adds additional terms of $g^{\Phi_i b_{1,i} + a_3 b_{1,i}}$ attached to $K_{1,i}$, respectively, for the k -th key (in which it picks $a_3 \in \mathbb{Z}_p$ randomly). This guarantees that if the τ_3 terms are not present, the k -th key will be correctly distributed as SF key.

Lemma 6. *Under the subspace assumption, no PPT adversary can have a non-negligible advantage in distinguishing between Game_Q and $\text{Game}_{\text{final}}$.*

Proof: Suppose a PPT algorithm \mathcal{A} achieves a non-negligible advantage in distinguishing Game_Q from $\text{Game}_{\text{final}}$, then we will construct a PPT algorithm \mathcal{C} to break the subspace assumption. We will set the parameters $n_i = 3$, $m = \mathcal{U} + 2, k_i = 1$, for two values of i and $n_i = 6, k_i = 2$ for the remaining values of i in the subspace assumption. To correctly align the assumption notation with our scheme notation, we hereby designate the bases of the assumption

as $(\mathbb{B}, \mathbb{B}^*)$, $(\mathbb{B}_0, \mathbb{B}_0^*) \in \text{Dual}(\mathbb{Z}_p^3, \delta)$ and $(\mathbb{B}_1, \mathbb{B}_1^*), \dots, (\mathbb{B}_{\mathcal{U}}, \mathbb{B}_{\mathcal{U}}^*) \in \text{Dual}(\mathbb{Z}_p^6, \delta)$. We will exclude the μ_3 term because it is not applicable here. The procedure for simulating Game_Q and $\text{Game}_{\text{final}}$ is described as follows:

- (1) $(G, p, g^{\bar{d}_3}, g^{\bar{d}_4}, g^{\eta \bar{d}_3}, g^{\beta \bar{d}_4}, \{g^{\bar{d}_{1,i}}, g^{\bar{d}_{2,i}}, g^{\eta \bar{d}_{1,i}}, g^{\beta \bar{d}_{2,i}}\}_{\forall i \in [\mathcal{U}]}, \Gamma) \leftarrow \mathcal{C}$;
- (2) $b \leftarrow \{0, 1\}$;
- (3) $\text{KeyGen}(\text{Gen})\{$
 - (3.1) let $\gamma_1^*, \gamma_2^*, \tau_1, \tau_2, \tau_3 \leftarrow \mathbb{Z}_p$
 - (3.2) let $\Gamma_0 = (T_{1,i} \leftarrow g^{\gamma_1^* \tau_1 \eta \bar{d}_{1,i} + \gamma_2^* \tau_2 \beta \bar{d}_{2,i}}, T_2 \leftarrow g^{\gamma_1^* \tau_1 \eta \bar{d}_3 + \gamma_2^* \tau_2 \beta \bar{d}_4}, \Gamma_1 = (T_{1,i} \leftarrow g^{\gamma_1^* \tau_1 \eta \bar{d}_{1,i} + \gamma_2^* \tau_2 \beta \bar{d}_{2,i} + \tau_3 \bar{d}_{3,i}}, T_2 \leftarrow g^{\gamma_1^* \tau_1 \eta \bar{d}_3 + \gamma_2^* \tau_2 \beta \bar{d}_4 + \tau_3 \bar{d}_6})_{\forall i \in [\mathcal{U}]}$;
 - (3.3) return $(\mathcal{C} \leftarrow \{\Gamma_b\})$;
 - (3.4) }
- (4) $\text{Enc}(\text{Gen})\{$
 - (4.1) let $\mu_1, \mu_2, \gamma_1, \gamma_2 \leftarrow \mathbb{Z}_p$;
 - (4.2) let $\mathcal{U}_{1,i} \leftarrow g^{\mu_1 \gamma_1 \bar{d}_{1,i} + \mu_2 \gamma_2 \bar{d}_{2,i} + \mu_3 \gamma_3 \bar{d}_{5,i}}, \mathcal{U}_2 \leftarrow g^{\mu_1 \gamma_1 \bar{d}_3 + \mu_2 \gamma_2 \bar{d}_4 + \mu_3 \bar{d}_6}_{\forall i \in [\mathcal{U}]}$;
 - (4.3) return $(\mathcal{U}_{1,i}, \mathcal{U}_2)$
 - (4.4) }
- (5) $\alpha_1, \alpha_2 \leftarrow \mathbb{Z}_p$
- (6) let $\text{pp} = \{G, p, g^{\bar{d}_{1,i}}, g^{\bar{d}_{2,i}}, g^{\bar{d}_3}, g^{\bar{d}_4}\}_{\forall i \in [\mathcal{U}]}, e(g, g)^{\alpha_1 s_{18}}, e(g, g)^{\alpha_2 s_{28}}$;
- (7) Output $\text{pp} \rightarrow \mathcal{A}$;
- (8) $(K_{1,[1,\dots,Q]}, K_2) \leftarrow \mathcal{A}(\mathbb{A}_1[1, \dots, Q_1])$ with;
 - (8.1) $\gamma_1^*, \alpha_3, \leftarrow \mathbb{Z}_p$;
 - (8.2) $\mathcal{C}(\text{pp}, \alpha_1, \alpha_2, \gamma_1^*, \Gamma_0, \Gamma_1)\{$
 - (8.3) let $v_1, v_2, v_3 \leftarrow \mathbb{Z}_p^T$
 - (8.4) let $\lambda_i \leftarrow \mathbb{M}_1^T * v_1$, where α_1 is the secret, $\mathbb{A}_1 \equiv \mathbb{M}_1$;
 - (8.5) let $\omega_i \leftarrow \mathbb{M}_1^T * v_2$, where α_2 is the secret, $\mathbb{A}_1 \equiv \mathbb{M}_1$;
 - (8.6) let $\Phi_i \leftarrow \mathbb{M}_1^T * v_3$, where α_3 is the secret, $\mathbb{A}_1 \equiv \mathbb{M}_1$;
 - (8.7) let $\bar{d}_{1,i} \rightarrow \bar{d}_{1,i} \lambda_i$;
 - (8.8) let $\bar{d}_{2,i} \rightarrow \bar{d}_{2,i} \omega_i$;
 - (8.9) let $\bar{d}_{5,i} \rightarrow \bar{d}_{5,i} \Phi_i$;
 - (8.10) let $K_2 = T_2$;
 - (8.11) for $i = 1$ to Q_1 ;
 - (8.11.1) write $K_{1,i} = g^{\lambda_i \gamma_1^* \bar{b}_{1,i}^*} T_{1,i}$;
 - (8.11.2) Output $\{(K_{1,i}, K_2) \forall i \in \mathbb{A}_1\} \rightarrow \mathcal{A}$;
- (9) }
- (10) $(C_0, C_{1,i}, C_2) \leftarrow \mathcal{A}(\text{msg}_0^*, \text{msg}_1^*, S^*)$ with;

- (11) $\mathcal{C}(\text{pp}, \text{msg}_0^*, \text{msg}_1^*, S^*)\{$
- (12) let $\tilde{s}_1, \tilde{s}_2, \tilde{s}_3 \leftarrow \mathbb{Z}_p$;
- (13) let $\tilde{s}_1 \leftarrow \mu_1, \tilde{s}_2 \leftarrow \mu_2, \tilde{s}_3 \leftarrow \mu_3$;
- (14) let $C_0 = \text{msg}_b \cdot e_6(\mathcal{U}_{1,i}, T_{1,i})$;
- (15) let $C_2 = \mathcal{U}_2$;
- (16) for i to $|S^*|$

(16.1) $\text{guide}(S^* \neq \mathbb{A}_1)$;

- (16.2) write $C_{1,i} = (\vec{d}_{1,i})^{\mu_1} (\vec{d}_{2,i})^{\mu_2}$;
- (16.3) Output $\{(C_0, C_{1,i}, C_2) \forall i \in S^*\} \rightarrow \mathcal{A}$;

(17) }

(18) $b = b'$;

To simulate either Game_Q or $\text{Game}_{\text{final}}$, algorithm \mathcal{C} sets the bases for the construction as

$$\left\{ \vec{b}_{1,i} = \eta \vec{b}_{1,i}, \vec{b}_{2,i} = \beta \vec{b}_{2,i}, \vec{b}_3 = \eta \vec{b}_3, \vec{b}_4 = \beta \vec{b}_4, \vec{b}_{5,i} = \vec{b}_{5,i}, \vec{b}_6 = \vec{b}_6 \right\}_{\forall i \in [\mathbb{U}]} \quad (27)$$

$$\left\{ \vec{b}_{1,i} = \eta^{-1} \vec{d}_{1,i}^*, \vec{b}_{2,i} = \beta^{-1} \vec{d}_{2,i}^*, \vec{b}_3 = \eta^{-1} \vec{d}_3^*, \vec{b}_4 = \beta^{-1} \vec{d}_4^*, \vec{b}_{5,i} = \vec{d}_{5,i}^*, \vec{b}_6 = \vec{d}_6^* \right\}_{\forall i \in [\mathbb{U}]}$$

We assert that these are well distributed because $(\mathbb{B}^*, \mathbb{B}), (\mathbb{B}_0^*, \mathbb{B}_0)$, etc., are chosen randomly up to sharing the same value δ . Implicitly, \mathcal{C} selects $\alpha_1, \alpha_2 \in \mathbb{Z}_p$ and sets $\alpha_1 = \eta \tau_1, \alpha_2 = \beta \tau_2$. Then, C produces

$$e(g, g)^{\alpha_1 s_{1\delta}} = \prod_{i=1}^n e_6(T_{1,i}, g^{\gamma_1 \vec{d}_{1,i}})^{\alpha_1}, \quad (28)$$

$$e(g, g)^{\alpha_2 s_{2\delta}} = \prod_{i=1}^n e_6(T_{1,i}, g^{\gamma_2 \vec{d}_{2,i}})^{\alpha_2}.$$

In line 1, the subspace assumption adversary \mathcal{C} is given the public parameters of the system and its challenge Γ . In line 8, \mathcal{A} requests for its private keys, which \mathcal{C} replies correctly to. In line 10, \mathcal{A} sends two messages $\{\text{msg}_0^*, \text{msg}_1^*\}$ of the same length with the attribute S^* to \mathcal{C} and requests for the challenge ciphertext. In response, \mathcal{C} outputs the correct ciphertext tuple $\{(C_0, C_{1,i}, C_2) \forall i \in S^*\}$ to \mathcal{A} with the restriction that the attribute set S^* does not satisfy the access structure which is enforced by the guard. Eventually, \mathcal{C} outputs \mathcal{A} 's guess as its own guess. By analysing this game, if the exponent of $T_{1,i}$ is equal to $\gamma_1^* \tau_1 \eta \vec{d}_{1,i}^* + \gamma_2^* \tau_2 \beta \vec{d}_{2,i}^*$ and $\mathcal{U}_{1,i} = \mu_1 \gamma_1 \vec{b}_{1,i} + \mu_2 \gamma_2 \vec{b}_{2,i}$, then we have

$$\prod_{i=1}^n e_6(\mathcal{U}_{1,i}, T_{1,i}) = e(g, g)^{(\alpha_1 s_{1\delta} + \alpha_2 s_{2\delta})\delta}, \quad (29)$$

and hence we have a well-distributed SF encryption of msg_b^* , as required in Game_Q . In this instance, Γ_0 and Γ_0^* are used in generating the challenge ciphertext. If instead the power of $T_{1,i} = \gamma_1^* \tau_1 \eta \vec{d}_{1,i}^* + \gamma_2^* \tau_2 \beta \vec{d}_{2,i}^* + \tau_3 \vec{d}_{3,i}^*$, then we have

$$\prod_{i=1}^n e_6(\mathcal{U}_{1,i}, T_{1,i}) = e(g, g)^{(\alpha_1 s_{1\delta} + \alpha_2 s_{2\delta} + \alpha_3 s_{3\delta})\delta}. \quad (30)$$

As long as the τ_3 term remains hidden in the SF ciphertext, it provides a blinding factor required for encryption of random message in the $\text{Game}_{\text{final}}$. Consequently, C can capitalize on \mathcal{A} 's non-negligible advantage in distinguishing between these games to attain a non-negligible advantage against the subspace assumption.

6. Implementation and Evaluation

We implemented the automation proofs of our KP-ABE scheme in AutoG&P [37]. In all cases, the proof is discovered semiautomatically, with the lines of codes which involve manual hand-tuning steps. The implementation was executed on Intel i7 personal laptop with 2.2 GHz CPU and 8 GB RAM running on macOS High Sierra 10.13.6. The proof-generation time for all the hybrid games of our scheme (i.e. $\text{Game}_{\text{real}} \rightarrow \text{Game}_0, \text{Game}_{k-1} \rightarrow \text{Game}_k^N, \text{Game}_k^N \rightarrow \text{Game}_k^T, \text{Game}_k^T \rightarrow \text{Game}_k, \text{Game}_Q \rightarrow \text{Game}_{\text{final}}$) took 498 ms.

Additionally, we use python cryptographic library known as charm-crypto 0.43 [38] to implement our KP-ABE scheme and the ABE scheme by Lewko and Waters [3] (Lw), which are the only schemes which support dual vector subspace assumption and thus whose functionalities are close to our scheme among the known ABE schemes. We used SS512 elliptic curve with 512 bit base field and SHA-3 hash function. We set the number of attribute as 10 and increase by 10 number of attributes each time. The benchmarks of the experiments are shown in Figures 2–4.

As can be inferred from Figures 2 and 4, the computation cost for key generation and encryption algorithms increases with the increment in the size of attributes. Our scheme has less computation overhead as compared to Lw scheme. This is because our scheme has less exponentiation of computation of the group elements. Also, our scheme has less decryption computation cost which can be inferred from Figure 4. This is as a result of less number of pairing operations in decryption as compared with Lw scheme.

6.1. Theoretical Comparison. We provide theoretical comparisons with some KP-ABE schemes which are shown in Tables 2–5. To enable us to make comparison with JL [26] scheme which uses asymmetric elliptic curve, we adopted the approach in [39] to convert Lw [3], GSW [31], and our scheme from the symmetric setting unto asymmetric setting without having to compute isomorphisms between the

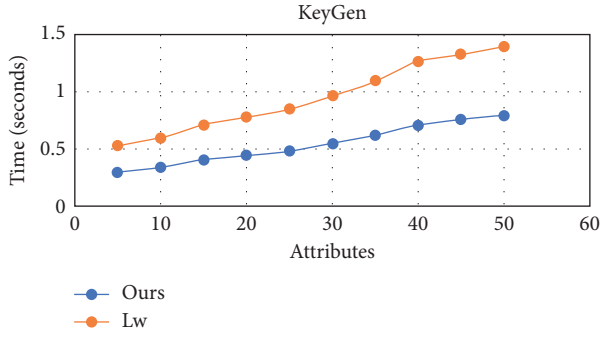


FIGURE 2: Computation cost for key generation.

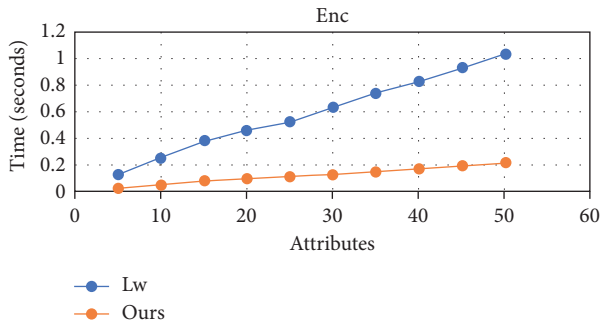


FIGURE 3: Computation cost for encryption.

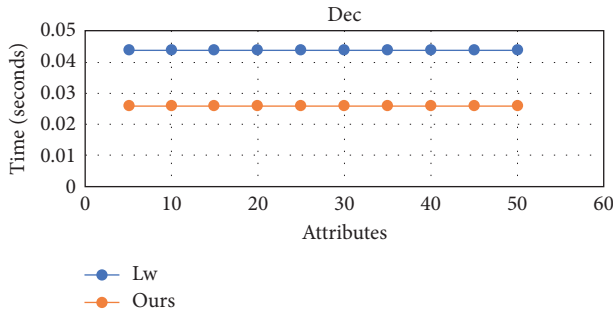


FIGURE 4: Computation cost for decryption.

TABLE 2: Comparison of key generation algorithm in KP-ABE schemes.

Schemes	Key generation			
	G_1		G_2	
	Exp	Hash	Exp	Hash
Ours	$4(6n_1) + 2(3)$	—	—	—
Lw [3]	$4(6n_1) + 4(3)$	—	—	—
JL [26]	$15n_t + 18n_f$	$12n'$	$3d$	—
GSW [31]	n_1	—	—	—

source groups. We use “MNT159” asymmetric curve with 159 bit base field from charm-crypto python library. Table 6 gives the cost of the computation operations. The parameters are set as follows:

TABLE 3: Comparison of encryption algorithm in KP-ABE schemes.

Schemes	Encryption			
	G_1		G_2	
	Exp	Hash	Exp	Hash
Ours	—	—	$2(6m) + 2(3)$	—
Lw [3]	—	—	$4(6m) + 4(3)$	—
JL [26]	$12m$	$12m$	3	—
GSW [31]	—	—	m	—

TABLE 4: Comparison of decryption algorithm in KP-ABE schemes. G_1, G_2, G_T

Schemes	Decryption			
	Exponentiation			Pairing
	G_1	G_2	G_T	
Ours	—	—	—	$6I + 3$
Lw [3]	—	—	—	$6I + 9$
JL [26]	$9I_f d$	—	—	$6d$
GSW [31]	—	—	—	I

We omit the multiplication cost in the groups (G_1, G_2, G_T) as they are insignificant when comparing with exponentiation and pairing.

TABLE 5: Size comparison of KP-ABE schemes.

Schemes	Key size		Ciphertext size	
	G_1	G_2	G_1	G_2
Ours	$6n_1 + 3$	—	—	$6m + 3$
Lw [3]	$6n_1 + 6$	—	—	$6m + 6$
JL [26]	$3(n_t + 2n_f)$	$3d$	$3m$	3
GSW [31]	—	n_1	—	m

TABLE 6: Running times of time-consuming operations.

Operation	Timing (seconds)
G_1 -exponentiation	0.011842
G_2 -exponentiation	0.037413
G_T -exponentiation	0.017396
Hashing to G_1	0.012839
Pairing	0.041798

- (i) m : the number of attributes.
- (ii) d : the maximum of multiple use of attributes.
- (iii) n' : the number of distinct labels ($n' \leq n$).
- (iv) n, n_t, n_f : the number of inputs and non-negated and negated inputs to a policy, respectively ($n = n_t + n_f$).
- (v) I, I_t, I_f : the number of attributes and non-negated and negated attributes in decryption, respectively ($I = I_t + I_f$).
- (vi) n_1 : the number of rows of a matrix for span programs.

As can be deduced from Tables 2–4, GSW is the most efficient scheme. However, it does not support the multiuse

and adaptive security properties. Although JL supports multiuse of attributes in the access policy, the computation cost of key generation increases with multiuse of attribute by the factor of d . However, since our scheme and Lw use the selective technique in generation of either the ciphertext or the key, the scheme performance is not affected by the multiuse of attribute. In terms of computation cost of decryption, the number of pairing operations and exponentiation increases with the factor of d in JL scheme when attributes are reused multiple times. However, our scheme and Lw scheme are not affected by multiuse of attribute. The computation cost only increases when there is an increment in the size of attributes.

From Table 6, we can infer that GSW has the least size of key and ciphertext. However, it does not support multiuse of attributes. Although JL supports multiuse of attributes, the size of the ciphertext increases by the factor of d . While the key and ciphertext sizes of Lw and our scheme are not affected by multiuse of attributes, comparatively our scheme has lesser key and ciphertext sizes than Lw scheme.

7. Conclusions

In the prime-order bilinear groups, we have introduced KP-ABE scheme which is fully secure and supports arbitrary usage of attributes in the access policy. This scheme attains full security under DLIN assumption and three-party assumption. This work removes high security loss that is involved in the reuse of attributes and enables the nonrestricted use of attributes. Our key point is inspired by the idea that the information-theoretical steps of the former dual system proof give the adversary excessive ground as if the computational arguments would be enough. So, we revived the earlier selective proofing techniques within the framework of dual system of encryption to gain enough ground to achieve full security proof.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This study was partially supported by the Sichuan Science and Technology Program (2018HH0102, 2019YFH0014, 2020YFH0030, and 2020YFSY0061).

References

- [1] B. Waters, "Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions," in *Proceedings of the 29th Annual International Cryptology Conference on Advances in Cryptology*, pp. 619–636, Santa Barbara, CA, USA, August 2009.
- [2] A. Lewko and B. Waters, "New Techniques for dual system encryption and fully secure HIBE with short ciphertexts," in *Proceedings of the 7th Theory of Cryptography Conference TCC 2010*, pp. 455–479, Zurich, Switzerland, February 2010.
- [3] A. Lewko and B. Waters, *New Proof Methods for Attribute-Based Encryption: Achieving Full Security through Selective Techniques*, Springer, Berlin, Germany, 2012.
- [4] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute based encryption for fine-grained access control of encrypted data," in *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 89–98, Alexandria, VI, USA, October 2006.
- [5] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *Proceedings of the Public Key Cryptography—PKC*, pp. 53–70, Taormina, Italy, March 2011.
- [6] D. Boneh and M. Franklin, "Identity based encryption from the Weil pairing," in *Proceedings of the CRYPTO*, pp. 213–229, Santa Barbara, CA, USA, August 2001.
- [7] R. Canetti, S. Halevi, and J. Katz, "A forward-secure public-key encryption scheme," in *Proceedings of the EUROCRYPT*, pp. 255–271, Warsaw, Poland, May 2003.
- [8] D. Boneh and X. Boyen, "Efficient selective-id secure identity based encryption without random oracles," in *Proceedings of the EUROCRYPT*, pp. 223–238, Interlaken, Switzerland, May 2004.
- [9] D. Boneh and X. Boyen, "Secure identity based encryption without random oracles," in *Proceedings of the CRYPTO*, pp. 443–459, Santa Barbara, CA, USA, August 2004.
- [10] B. Waters, "Efficient identity-based encryption without random oracles," in *Proceedings of the EUROCRYPT*, pp. 114–127, Aarhus, Denmark, May 2005.
- [11] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proceedings of the EUROCRYPT*, pp. 457–473, Aarhus, Denmark, May 2005.
- [12] J. H. Cheon, "Security analysis of the strong Diffie-Hellman problem," in *Proceedings of the EUROCRYPT*, pp. 1–11, St. Petersburg, Russia, May 2006.
- [13] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption," in *Proceedings of the EUROCRYPT*, pp. 62–91, French Riviera, France, May 2010.
- [14] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.
- [15] C. Cocks, "An identity based encryption scheme based on quadratic residues," in *Proceedings of the 8th International Conference on Cryptography and Coding*, pp. 26–28, Cirencester, UK, December 2001.
- [16] J. Horwitz and B. Lynn, "Toward hierarchical identity-based encryption," in *Proceedings of the EUROCRYPT*, pp. 466–481, Innsbruck, Austria, May 2002.
- [17] C. Gentry and A. Silverberg, "Hierarchical id-based cryptography," *Lecture Notes in Computer Science*, in *Proceedings of the ASIACRYPT*, pp. 548–566, Queenstown, New Zealand, December 2002.
- [18] V. Goyal, "Bounded ciphertext policy attribute based encryption," in *Proceedings of the International Colloquium on Automata, Languages, and Programming ICALP*, pp. 579–591, Reykjavik, Iceland, July 2008.
- [19] J. Bethencourt, S. Amit, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, USA, May 2007.
- [20] N. Attrapadung, "Dual system encryption via doubly selective security: framework, fully secure functional encryption for

- regular languages, and more,” in *Proceedings of the EUROCRYPT*, pp. 557–577, Copenhagen, Denmark, May 2014.
- [21] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, “Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption,” in *Proceedings of the EUROCRYPT 2010*, H. Gilbert, Ed., vol. 6110, pp. 62–91, Springer, French Riviera, France, May, 2010.
- [22] A. B. Lewko, M. Lewko, and B. Waters, “How to leak on key updates,” in *Proceedings of the STOC*, pp. 725–734, San Jose, CA, USA, June 2011.
- [23] A. Lewko, Y. Rouselakis, and B. Waters, “Achieving leakage resilience through dual system encryption,” in *Proceedings of the TCC*, pp. 70–88, Providence, RI, USA, March 2011.
- [24] Y. Dodis, A. B. Lewko, B. Waters, and D. Wichs, “Storing secrets on continually leaky devices,” in *Proceedings of the FOCS*, pp. 688–697, Palm Springs, CA, USA, October 2011.
- [25] M. Chase and S. Meiklejohn, “Déjà Q: using dual systems to revisit q -type assumptions,” in *Proceedings of the EUROCRYPT*, pp. 622–639, Copenhagen, Denmark, May 2014.
- [26] J. Tomida, Y. Kawahara, and R. Nishimaki, “Fast, compact, and expressive attribute-based encryption,” in *Proceedings of the Public-Key Cryptography*, p. 966, Beijing, China, April 2019.
- [27] L. Kowalczyk and H. Wee, “Compact adaptively secure ABE for NCs 1 from k-lin,” in *Proceedings of the EUROCRYPT 2019*, V. Rijmen and Y. Ishai, Eds., pp. 3–33, Springer, Darmstadt, Germany, May 2019.
- [28] Z. Jafargholi, C. Kamath, K. Klein, I. Komargodski, K. Pietrzak, and D. Wichs, “Be adaptive, avoid over-committing,” in *Proceedings of the CRYPTO 2017*, J. Katz and H. Shacham, Eds., vol. 10401, pp. 133–163, Springer, Santa Barbara, CA, USA, August 2017.
- [29] Y. Song, H. Wang, X. Wei, and L. Wu, “Efficient attribute-based encryption with privacy-preserving key generation and its application in industrial cloud,” *Security and Communication Networks*, vol. 2019, Article ID 3249726, 9 pages, 2019.
- [30] F. Khan, Y. H. Li, T. H. Zhang, H. Abbas, and T. Yaqoob, “Efficient attribute-based encryption with repeated attributes optimization,” *International Journal of Information Security*, pp. 1–14, 2020.
- [31] V. Goyal, O. Pandey, S. Amit, and B. Waters, “Attribute based encryption for fine-grained access control of encrypted data,” in *Proceedings of the ACM CCS 2006*, A. Juels, R. N. Wright, and S. D. C. di Vimercati, Eds., ACM Press, Alexandria, VA, USA, pp. 89–98, October 2006.
- [32] J. Chen, J. Gong, L. Kowalczyk, and H. Wee, “Unbounded ABE via bilinear entropy expansion, revisited,” in *Proceedings of the EUROCRYPT 2018*, J. B. Nielsen and V. Rijmen, Eds., vol. 10820, pp. 503–534, Springer, Tel Aviv, Israel, April 2018.
- [33] A. Beimel, *Secure schemes for secret sharing and key distribution*, Ph.D. thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.
- [34] M. Karchmer and A. Wigderson, “On span programs,” in *Proceedings of the Structure in Complexity*, San Diego, CA, USA, May 1993.
- [35] T. Okamoto and K. Takashima, “Hierarchical predicate encryption for inner-products,” in *Proceedings of the ASIA-CRYPT*, pp. 214–231, Tokyo, Japan, December 2009.
- [36] T. Okamoto and K. Takashima, “Fully secure functional encryption with general relations from the decisional linear assumption,” in *Proceedings of the CRYPTO*, pp. 191–208, Santa Barbara, CA, USA, August 2010.
- [37] AutoG&P tool, 2020, <https://github.com/ZooCrypt/AutoGnP>.
- [38] J. A. Akinyele, C. Garman, I. Miers et al., “Charm: a framework for rapidly prototyping cryptosystems,” *Journal of Cryptographic Engineering*, vol. 3, no. 2, pp. 111–128, 2013.
- [39] M. Abe, M. J. Groth, T. Ohkubo, and T. Tango, “Converting cryptographic schemes from symmetric to asymmetric bilinear groups,” *Advances in Cryptology—CRYPTO 2014*, Springer, Berlin, Germany, pp. 241–260, 2014.