Practical Attribute Based Inner Product Functional Encryption from Simple Assumptions

Yuechen Chen, Linru Zhang, and Siu-Ming Yiu

Department of Computer Science, The University of Hong Kong,
Pokfulam Road, HKSAR, China
Email: {ycchen,lrzhang,smyiu}@cs.hku.hk

Abstract. Functional encryption (FE) that bases on user attributes has many useful practical applications. For example, a company may only authorize department heads of other sections to query the average sale figures of the sales department from the encrypted sales amounts of all sales. However, FE schemes that can solve this problem are based on new, but not well-studied assumptions (such as indistinguishable obfuscation or multilinear maps). It is not clear if these FE schemes are secure. In this paper, we develop the first functional encryption scheme (ABFE) from simple and well-studied assumptions that can authorize a user base on the user's attributes to obtain a functional value of the encrypted data.

Keywords: Functional Encryption, Attribute Based Encryption, Inner Product, Duel System Encryption,

1 Introduction

Data privacy has become an important issue. For example, in a company, employees may not want to disclose their salaries (or sale figures) to even some senior staff, e.g. department heads of other sections. We can store their salaries in encrypted form and further require that some department heads are only authorized to query certain statistics (e.g. the average/highest salary or sale figure) of the employees in another team, but not the exact figure of individual employee. To solve this problem in a practical way, we need an encryption scheme that can produce decryption keys which only allow authorized users (e.g. based on user's attributes) to obtain a functional value of the encrypted data.

At first glance, functional encryption (FE) (e.g. [ONe10; BSW11]) seems to be able to solve our problem as it can provide a decryption key that allows a user to obtain only a functional value of the encrypted data. More precisely, the $function$ can be modeled as a Turing Machine $F(\cdot,\cdot)$. The user who has a secret key sk_k can compute the function $F(k,x)$ on an encryption of x. However, FE schemes that can solve our problem (i.e., the ones that can consider the identities (or attributes) of the users in its decryption, e.g. [BCP14; Gar+16; Ber+13; Wat15; AS16]) are based on new, but not well-studied assumptions (such as indistinguishable obfuscation or multilinear maps). Attacks were identified for some constructions on indistinguishable obfuscation and multilinear maps [Apo+17; CGH17; Che+15; Cor+16]. Hence, it is not clear whether these FE schemes are secure.

For FE based on well-studied assumptions, the functionality is limited. Recently, a line of work called Functional Encryption for Inner Product (IPFE) started by Abdalla et al. [Abd+15] aims at building Functional Encryption constructions based on standard assumptions such as the plain decisional Diffie-Hellman assumption rather than multi-linear map or Indistinguishable Obfuscation(IO). More precisely, A user could store an encrypted vector $\boldsymbol{y}$ on an untrusted remote server. The authority can generate a series of secret keys $\{sk_i\}$ corresponding to different vectors $\{\boldsymbol{x_i}\}$. These keys can be sent to the server for decrypting the message to get inner product $\langle \boldsymbol{x_i}, \boldsymbol{y} \rangle$, while ensuring no more leakage of information than the computation result. Abdalla et al. [Abd+15] first proposed a framework to construct IND-secure IPFE scheme with selective security . Continued by [ALS16], the security of IPFE are improved to fully secure, also from standard assumptions. A generic construction of IPFE is given in [Abd+16], as well as three instantiations from Decisional Diffie-Hellman assumption (DDH), Decisional Composite Residuosity assumption (DCR) and Learning with errors assumption (LWE), respectively. [Bal+17] realizes Functional Encryption for quadratic functions, with linear size of ciphertext. However, these FE schemes reveals the decryption result to all users, and all users get the same decryption result, which is not enough for allowing only authorized users to obtain functional values based on their attributes.

While Attribute-Based Encryption (ABE) offers fine-grained decryption policy such that users can do decryption if their attributes satisfy the policy, users with different attributes will get different decryption values based on their at-

tributes. ABE is first introduced in[SW05]. It can be divided into Key-Policy ABE (KP-ABE) and Ciphertext-Policy ABE (CP-ABE). In KP-ABE [SW05; Goy+06; OSW07; OT10], secret key is associated with access policy P, saying that users with what attributes can decrypt the data. And ciphertext is associated with user's attribute set S. The secret key can decrypt the ciphertext if the attribute set S satisfies the policy P. In CP-ABE, ciphertext is associated with policy P, while secret key is associated with user's attribute set S [BSW07; CN07; Wat11; Yam+11]. The property of ABE inspires us to use ABE to solve our problem in FE.

This motivates us to consider the following research problem:

To build a practical crypto system from simple and well-studied assumptions that can authorize a user base on the user's attributes to obtain a functional value of the encrypted data.

1.1 Our Contributions

We provide the first solution, an attribute-based functional encryption scheme (ABFE), that is based on simple assumptions for solving the above problem[1]. ABFE is an improvement of traditional ABE (attribute-based encryption) as we can set the output of the decryption to be the original message, instead of a functional value of it.

We show the details of our ABFE definitions in Section 3, where we use CP-ABE to realize data access control in FE. In section 4, we show a construction of ABFE, which is named as attribute-based inner product functional encryption (ABIPFE) since we use inner product functionality. The security proof of ABIPFE is given in Section 5. When considering the functionality in ABFE, we start from the *inner product*, which is simple but useful[2]. There are quite a number of practical applications for inner product, e.g. computing the weighted mean.

Technically, both of our schemes are built from three decisional problems in composite order bilinear groups. The order of the groups is $N = p_1p_2p_3$, where p_1, p_2, p_3 are three distinct primes. We use the dual system encryption [Wat09] to prove the security of the schemes. This is the first attempt to introduce the dual system encryption into the line of work that builds functional encryption schemes from simple assumptions. More precisely, the ciphertexts are built in the subgroup $\mathbb{G}_{p_1}$ and the secret keys are built in the subgroup $\mathbb{G}_{p_1p_3}$. When we apply a bilinear map $\hat{e}$, which takes the ciphertext and the secret key as inputs, the $\mathbb{G}_{p_3}$ parts of the secret key was orthogonal to the ciphertext and will not affect the final result. In the dual system encryption, random elements

[1] Note that Boyen's work [Boy13] which has a similar name with our work, is totally different with our work. It realizes attribute based encryption in Lattices.

[2] In fact, after realising that most of indistinguishable obfuscation and multilinear map schemes may be insecure, researchers start to focus on functional encryption schemes from standard assumptions [Abd+15; ALS16; Abd+17; Abd+18]. Inner product is also selected as a starting point in these studies.

from $\mathbb{G}_{p_2}$ will be added to both the semi-functional ciphertext and the semi-functional secret key, then if we use the semi-functional secret key to decrypt the semi-functional ciphertext, the result will be blinded by a factor in $\mathbb{G}_{p_2}$. And otherwise if we use the semi-functional key to decrypt the normal ciphertext or we use the normal key to decrypt the semi-functional ciphertext, the $\mathbb{G}_{p_2}$ parts of the semi-functional components will not affect the result.

2 Background

2.1 Functional Encryption (FE)

Following [BSW11], we define the functionality $\mathcal{F}$, then the FE scheme for $\mathcal{F}$.

Definition 1 (Functionality). *A functionality $\mathcal{F}$ defined over $(K \times X)$ is a function $F : K \times X \to \Sigma \cap \{\perp\}$, where K is the key space, X is the message space and Σ is the output space and $\perp$ is a special string not contained in Σ.*

Definition 2 (FE scheme). *A functional encryption scheme FE for a functionality $\mathcal{F}$ is a tuple $FE = (Setup, KeyGen, Encrypt, Decrypt)$ of 4 algorithms:*

1. *$Setup(1^\lambda)$, on input a security parameter lambda, outputs both public key and master secret keys (mpk, msk).*
2. *$KeyGen(msk, k)$, on input a master secret key msk and key $k \in K$, outputs the secret key sk_k.*
3. *$Encrypt(mpk, x)$, on input public key mpk and message $x \in X$, outputs the ciphertext Ct.*
4. *$Decrypt(mpk, Ct, sk_k)$ outputs $y \in \Sigma \times \{\perp\}$.*

The correctness requirement: for all $(mpk, msk) \leftarrow Setup(1^\lambda)$, all $k \in K$, $m \in M$, for $sk_k \leftarrow KeyGen(msk, k)$ and $Ct \leftarrow Encrypt(mpk, m)$, we have $Decrypt(mpk, Ct, sk_k) = F(k, m)$ whenever $F(k, m) \neq \perp$, except with negligible probability.

2.2 Bilinear Maps

We review some facts related to groups with efficiently computable bilinear maps in [Wat11] and then give our number theoretic assumptions. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}$ and e be a bilinear map, $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The bilinear map e has the following properties:

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

We say that $\mathbb{G}$ is a bilinear group if the group operation in $\mathbb{G}$ and the bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ are both efficiently computable. Notice that the map e is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

2.3 Decisional Parallel BDHE Assumption

We review the definition of decisional $q-parallel$ Bilinear Diffie-Hellman Exponent problem in [Wat11] as follows. Choose a group $\mathbb{G}$ of prime order p according to the security parameter λ. Let $a, s, b_1, \ldots, b_q \in \mathbb{Z}_p$ be chosen at random and g be a generator of $\mathbb{G}$. If an adversary is given $\boldsymbol{y} =$

$$\begin{array}{c} g, g^s, g^a, \ldots, g^{a^q}, , g^{a^{q+2}}, \ldots, g^{a^{2q}} \\ \forall_{1\leq j\leq q}\ g^{s\cdot b_j}, g^{a/b_j}, \ldots, g^{a^q/b_j}, , g^{a^{q+2}/b_j}, \ldots, g^{a^{2q}/b_j} \\ \forall_{1\leq j\leq q, k\neq j} g^{a\cdot s\cdot b_k/b_j}, \ldots, g^{a^q\cdot s\cdot b_k/b_j}, \end{array}$$

it is hard to distinguish $e(g,g)^{a^{q+1}s} \in \mathbb{G}_T$ from a random element R in $\mathbb{G}_T$.

An algorithm $\mathcal{B}$ that outputs $z \in \{0,1\}$ has advantage ϵ in solving decisional *q-parallel* BDHE in $\mathbb{G}$ if

$$|Pr[\mathcal{B}(\boldsymbol{y}, T = e(g,g)^{a^{q+1}s}) = 0] - Pr[\mathcal{B}(\boldsymbol{y}, T = R) = 0]| \geq \epsilon$$

Definition 3. *We say that the (decision) q-parallel-BDHE assumption holds if no polynomial time algorithm $\mathcal{B}$ has a non-negligible advantage in solving the decisional q-parallel BDHE problem.*

2.4 Linear Secret Sharing Schemes

We review the definition of linear secret sharing scheme (LSSS) in [Wat11] as follows.

Definition 4. *(Linear Secret-Sharing Schemes (LSSS)) A secret-sharing scheme over a set of parties $\mathcal{P}$ is called linear (over $\mathbb{Z}_p$) if*

1. *The shares for each party form a vector over $\mathbb{Z}_p$.*
2. *There exists a matrix an M with ℓ rows and n columns called the share-generating matrix for Π. For all $i = 1, \ldots, \ell$, the i'th row of M, we let the function ρ defined the party labelling row i as $\rho(i)$. When we consider the column vector $v = (s, r_2, \ldots, r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $r_2, \ldots, r_n \in \mathbb{Z}_p$ are randomly chosen, then Mv is the vector of ℓ shares of the secret s according to Π. The share $(Mv)_i$ belongs to party $\rho(i)$.*

It is shown in [Wat11] that every LSSS according to the above definition also enjoys the *linear reconstruction* property, defined as follows: Suppose that Π is an LSSS for the access structure $\mathbb{A}$. Let $S \in \mathbb{A}$ be any authorized set, and let $I \subset \{1, 2, \ldots, \ell\}$ be defined as $I = \{i : \rho(i) \in S\}$. Then, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i\in I}$ such that, if $\{\lambda_i\}$ are valid shares of any secret s according to Π, then $\sum_{i\in I} \omega_i \lambda_i = s$. Furthermore, it is shown in [Wat11] that these constants ω_i can be found in time polynomial in the size of the share-generating matrix M.

2.5 Ciphertext-policy attribute based encryption

We review the definition of CP-ABE in [Wat11] here. Let S represent a set of attributes, U denotes the attribute universe, and $\mathbb{A}$ an access structure. We defines $\mathbb{A}$ and S as the inputs to the encryption and key generation algorithm, and the function $f(S, \mathbb{A})$ outputs 1 iff S satisfies $\mathbb{A}$, respectively.

Definition 5. *A CP-ABE scheme consists of four algorithms: Setup, Encrypt, KeyGen and Decrypt.*

$(PK, MSK) \leftarrow \mathsf{Setup}(\lambda, U)$. The setup algorithm takes security parameter λ and attribute universe description U as input. It outputs the public parameters PK and a master key MSK.

$CT \leftarrow \mathsf{Encrypt}(PK, m, \mathbb{A})$. The encryption algorithm takes as input the public parameters PK, a message m, and an access structure $\mathbb{A}$. It outputs the ciphertext CT.

$SK \leftarrow \mathsf{KeyGen}(MSK, S)$. The key generation algorithm takes as input the master key MSK and an attribute set S and outputs the secret key SK.

$m/\perp \leftarrow \mathsf{Decrypt}(SK, CT)$. The decrypt algorithm takes as input SK for S and CT that was originally encrypted under $\mathbb{A}$. It outputs the message m if $f(S, \mathbb{A}) = 1$ and the error symbol $\perp$ otherwise.

We now describe a security model for ciphertext-policy ABE schemes. The security model allows the adversary to query for any private keys that cannot be used to decrypt the challenge ciphertext. In CP-ABE the ciphertexts are identified with access structures and the private keys with attributes. It follows that in our security definition the adversary will choose to be challenged on an encryption to an access structure $\mathbb{A}$ and can ask for any private key S such that S does not satisfy $\mathbb{A}$. We now give the formal security game.

Security model for CP-ABE

Setup. The challenger $\mathcal{C}$ runs the setup algorithm and gives the public parameters, PK, to the adversary $\mathcal{A}$.

Phase 1. The adversary $\mathcal{A}$ makes repeated private keys corresponding to the sets of attributes $S_1, \ldots, S_{q_1}$.

Challenge. The adversary $\mathcal{A}$ submits two equal-length messages m_0 and m_1. In addition the adversary $\mathcal{A}$ gives a challenge access structure $\mathbb{A}^*$ such that none of the sets $S_1, \ldots, S_{q_1}$ from Phase 1 satisfy the access structure. The challenger $\mathcal{C}$ flips a random coin $b \in \{0, 1\}$, and encrypts M_b under $\mathbb{A}^*$. The ciphertext CT^* is given to the adversary $\mathcal{A}$.

Phase 2. Phase 1 is repeated with the restriction that none of the sets of attributes $S_{q_1+1}, \ldots, S_q$ satisfies the access structure corresponding to the challenge.

Guess. The adversary $\mathcal{A}$ outputs a guess b' of b.

The advantage of an adversary $\mathcal{A}$ in this game is defined as $Adv = Pr[b' = b] - \frac{1}{2}$. We note that the model can easily be extended to handle chosen-ciphertext attacks by allowing for decryption queries in Phase 1 and Phase 2.

Definition 6. *A ciphertext-policy attribute-based encryption scheme is secure if all polynomial time adversaries have at most a negligible advantage in the above game.*

2.6 Composite order bilinear Groups

We will use composite order bilinear groups G, which enjoys the orthogonality property of the paring operations of two elements in the subgroups of G, respectively. Define cyclic groups $G = \langle \mathcal{G} \rangle$ and G_T of order $N = p_1 p_2 p_3$, where p_1, p_2 and p_3 are distinct primes, and $e : G \times G \to G_T$ is a map such that:

1. (Bilinear) $\forall g, h \in G, a, b \in \mathbb{Z}_N, e(g^a, h^b) = e(g, h)^{ab}$.
2. (Non-degenerate) $\exists g \in G$ such that $e(g, g)$ has order N in G_T.

We assume that the group operations in G and G_T as well as the bilinear map e are computable in polynomial time with respect to λ and that the group descriptions of G and G_T include generators of the respective cyclic groups. Let G_{p_1}, $G_{p_2}$and G_{p_3} denote the subgroups of order p_1, p_2 and p_3 in G, respectively. Note that when $h_i \in G_{p_i}$ and $h_j \in G_{p_j}$ for $i \neq j$, $e(h_i, h_j)$ is the identity element in G_T. This orthogonality property of G_{p_1}, G_{p_2}, G_{p_3} will be used to implement semi-functionality in our construction.

To see this, suppose $h_1 \in G_{p_1}$ and $h_2 \in G_{p_2}$. Let g denote a generator of G. Then, $g^{p_1 p_2}$ generates G_{p_3}, $g^{p_1 p_3}$generates G_{p_2}, and $g^{p_2 p_3}$generates G_{p_1}. Hence, for some α_1 , $\alpha_2, h_1 = (g^{p_2 p_3})^(\alpha_1)$ and $h_2 = (g_1^p p_3)^{\alpha_2}$. Then:

$$e(h_1, h_2) = e(g^{p_2 p_3 \alpha_1}, g^{p_1 p_3 \alpha_2}) = e(g^{\alpha_1}, g^{p_3 \alpha_2})^{p_1 p_2 p_3} = 1$$

We now state the complexity assumptions that we will rely on to prove security of our systems. In the assumptions below, we let $G_{p_1 p_2}$, e.g., denote the subgroup of order $p_1 p_2$ in G.

With permission, we have reproduced the proof of the fact that these assumptions hold in the generic group model. We note that all three assumptions are static (constant size) and the first assumption is just the subgroup decision problem in the case where the group order is a product of three primes.

Assumption 1. *(Subgroup decision problem for 3 primes) Given a group generator $\mathcal{G}$, we define the following distribution:*

$$\begin{gathered} \mathbb{G} = (N = P_1 P_2 P_3, G, G_T, e) \xleftarrow{R} \mathcal{G}, \\ g \xleftarrow{R} G_{p_1}, X_3 \xleftarrow{R} G_{p_3}, \\ D = (\mathbb{G}, g, X_3), \\ T_1 \xleftarrow{R} G_{p_1 p_2}, T_2 \xleftarrow{R} G_{p_1}. \end{gathered}$$

We define the advantage of an algorithm $\mathcal{A}$ in breaking **Assumption 1** to be:

$$\mathbf{Adv1}_{\mathcal{G},\mathcal{A}}(\lambda) := |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|.$$

We note that T_1 can be written (uniquely) as the product of an element of G_{p_1} and an element of G_{p_2}. We refer to these elements as the "G_{p_1} part of T_1"and the "G_{p_2} part of T_1" respectively. We will use this terminology in our proofs.

Definition 7. *We say that $\mathcal{G}$ satisfies* **Assumption 1** *if* $\mathbf{Adv1}_{\mathcal{G},\mathcal{A}}(\lambda)$ *is a negligible function of λ for any polynomial time algorithm $\mathcal{A}$.*

Assumption 2. *Given a group generator $\mathcal{G}$, we define the following distribution:*

$$\begin{gathered}\mathbb{G} = (N = P_1P_2P_3, G, G_T, e) \xleftarrow{R} \mathcal{G},\\ g, X_1 \xleftarrow{R} G_{p_1}, X_2, Y_2 \xleftarrow{R} G_{p_2}, X_3, Y_3 \xleftarrow{R} G_{p_3},\\ D = (\mathbb{G}, g, X_1X_2, X_3, Y_2Y_3),\\ T_1 \xleftarrow{R} G, T_2 \xleftarrow{R} G_{p_1p_3}.\end{gathered}$$

We define the advantage of an algorithm $\mathcal{A}$ in breaking **Assumption 2** to be:

$$\mathbf{Adv2}_{\mathcal{G},\mathcal{A}}(\lambda) := |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|.$$

We use $G_{p_1p_3}$ to denote the subgroup of order p_1p_3 in G. We note that T_1 can be (uniquely) written as the product of an element of G_{p_1}, an element of G_{p_2}, and an element of G_{p_3}. We refer to these as the "G_{p_1} part of T_1", the "G_{p_2} part of T_1", and the "G_{p_3} part of T_1", respectively. T_2 can similarly be written as the product of an element of G_{p_1} and an element of G_{p_3}.

Definition 8. *We say that $\mathcal{G}$ satisfies* **Assumption 2** *if* $\mathbf{Adv2}_{\mathcal{G},\mathcal{A}}(\lambda)$ *is a negligible function of λ for any polynomial time algorithm $\mathcal{A}$.*

Assumption 3. *Given a group generator $\mathcal{G}$, we define the following distribution:*

$$\begin{gathered}\mathbb{G} = (N = P_1P_2P_3, G, G_T, e) \xleftarrow{R} \mathcal{G}, \alpha, s \xleftarrow{R} \mathbb{Z}_N,\\ g \xleftarrow{R} G_{p_1}, X_2, Y_2, Z_2 \xleftarrow{R} G_{p_2}, X_3 \xleftarrow{R} G_{p_3},\\ D = (\mathbb{G}, g, g^{\alpha}X_2, X_3, g^{s}Y_2, Z_2),\\ T_1 = e(g, g)^{\alpha s}, T_2 \xleftarrow{R} G_T.\end{gathered}$$

We define the advantage of an algorithm $\mathcal{A}$ in breaking **Assumption 3** to be:

$$\mathbf{Adv3}_{\mathcal{G},\mathcal{A}}(\lambda) := |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|.$$

Definition 9. *We say that $\mathcal{G}$ satisfies* **Assumption 3** *if* $\mathbf{Adv3}_{\mathcal{G},\mathcal{A}}(\lambda)$ *is a negligible function of λ for any polynomial time algorithm $\mathcal{A}$.*

2.7 Dual system encryption[LW10; Wat09]

In a dual system encryption, both ciphertexts and keys can act as two forms: normal or semi-functional. Normal ciphertexts and keys are used in both the scheme and the security proof, while semi-functional ciphertexts and keys are only used in the security proof. A normal secret key can decrypt both normal ciphertext and semi-functional ciphertext. A semi-functional secret key can decrypt normal ciphertexts. When a semi-functional secret key is used to decrypt a semi-functional ciphertext, decryption will fail.

The security of a dual system encryption is proved by a sequence of security games. Generally speaking, the first game is the real security game in the security definition, and the second game is the same as the first game except that the challenge ciphertext is semi-functional. Let q denote the number of queries the adversary makes. There are q games following the second game. In game k, the challenge ciphertext is semi-functional and the first k secret keys generated by Key Generation queries are also semi-functional. So in game q, all secret keys in the query phase are semi-functional. Finally, the last game is the same as game q except that the challenge ciphertext is a semi-functional ciphertext of a random message, not one of the two challenge messages. So, in the last game, the correct answer to the challenge b is information theoretically hidden from the adversary and the proof is done.

3 Definitions for ABFE

We use $\mathcal{D}, \mathcal{D}'$ to denote a description of the attributes about users. An attribute-based functional encryption scheme (ABFE) consists of 4 algorithms:
Setup: On input a security parameter, it generates a master public key mpk and a master secret key msk.
Encrypt(mpk,$\mathcal{D}$,M): On input the master public key mpk, a message M from the message space and a description of users who can use M to do the computation, it generates a ciphertext $ct_{(\mathcal{D},M)}$.
KeyGen(msk,$\mathcal{D}'$,f): On input the master secret key msk, a function f from the function space and a description of user who can compute the function value $f(\cdot)$, it generates a secret key $sk_{(\mathcal{D}',f)}$
Decrypt(mpk,$ct_{(\mathcal{D},M)}$,$sk_{(\mathcal{D}',f)}$): On input the master public key mpk, a ciphertext $ct_{(\mathcal{D},M)}$ and a secret key $sk_{(\mathcal{D}',f)}$, it outputs a function value $f(M)$ if $\mathcal{D}$ is accepted by $\mathcal{D}'$ and $\perp$ otherwise.

Actually, an ABFE scheme can be viewed as a kind of functional encryption scheme. We can set the plaintext as a pair (ind, M) and the function $F(\cdot,\cdot,\cdot,\cdot)$ is defined as $F(k, ind, f, M) = f(M)$ if and only if ind is accepted by k. So our instantiation scheme in Section 4 can be viewed as an implementation of a new kind of functional encryption from standard assumptions. (Notice that only functional encryption for inner product and quadratic functions from standard assumptions were implemented up to now [Abd+15; ALS16; Abd+17; Abd+18; Bal+17]).

To define the security of an ABFE against an adaptive adversary $\mathcal{A}$ that repeatedly asks for secret keys $sk_{(\mathcal{D}',f)}$, the problem is how to restrict the adversary's power in the security game. Once the adversary gets all the secret keys it wants, it will output two challenge messages m_0, m_1 and a challenge description of users $\mathcal{D}^*$. The challenger will return an encryption $ct_{(\mathcal{D}^*,m_b)}$ to the adversary. Clearly, if the adversary has a secret key $sk_{(\mathcal{D}',f)}$ such that 1). the description $\mathcal{D}^*$ is accepted by a description $\mathcal{D}'$ 2). $f(m_0) \neq f(m_1)$. Then the adversary can easily distinguish whether $b = 0$ or 1 by comparing $\mathbf{Decrypt}(ct_{((D^*,m_b))}, sk_{(\mathcal{D}',f)})$ with $f(m_0)$. So, we must restrict the adversary's choice of challenge messages and challenge description of users.

Inspired by IBE and ABE, one possible approach is to require that the challenge description of users $\mathcal{D}^*$ cannot be accepted by any description $\{\mathcal{D}'\}$ that has been queried. [3] After adding this restriction, the adversary cannot decide the value of b from $\mathbf{Decrypt}(ct_{((D^*,m_b))}, sk_{(\mathcal{D}',f)})$, where $sk_{(\mathcal{D}',f)}$ is any of the secret keys it has. In applications, this restriction means that the description of users is kept secret from the adversary. So the adversary can fabricate many 'fake' descriptions and asks for the secret keys, but it will never know the true description. This can be a reasonable assumption. We use this restriction to create our security definition as follows.

Setup

The challenger $\mathcal{C}$ runs the **ABFE.Setup** algorithm to generate a master public key mpk and a master secret key msk. Then the challenger sends mpk to the adversary $\mathcal{A}$.

Query 1

The adversary can make Key Generation queries. To make a Key Generation query, The adversary chooses a description $\mathcal{D}'$ and a function f from the function space $\mathcal{F}$, and sends to the challenger. The challenger runs **ABFE.KeyGen** algorithm to generate a secret key $sk_{(\mathcal{D}',f)}$ and returns it to the adversary.

Challenge

The adversary sends two message m_0 and m_1 with a challenger description $\mathcal{D}^*$ to the challenger. The description $\mathcal{D}^*$ must not be accepted by any description in **Query 1**. The challenger picks a bit $b \in \{0, 1\}$ randomly and runs **ABFE.Encrypt** algorithm to create the ciphertext $ct_{(\mathcal{D}^*,m_b)}$. The challenger returns the ciphertext to the adversary.

Query 2

This is the same as **Query 1** with the condition that the adversary cannot make a query for description $\mathcal{D}'$ such that $\mathcal{D}^*$ is accepted by $\mathcal{D}'$.

Guess

The adversary outputs a guess b' for b.

The advantage of the adversary $\mathcal{A}$ is defined to be $Pr[b' = b] - \frac{1}{2}$.

[3] In IBE, the security definition require that the challenge identity ID^* cannot be queried. In ABE, the security definition require that the attribute γ used in encrypting the challenge message cannot be queried.

Definition 10. *An ABFE scheme is said to be IND-secure if all polynomial time adversaries achieve at most negligible advantage in the above security game.*

The above definitions and the security game can be easily modified (simplified) if we use identities to replace the description of the attributes. Our functionality for inner product is defined as follows.

Definition 11. *The inner product functionality [Abd+15] $\mathcal{F}$ defined over (K, X) is a function $F : K \times X \to \Sigma \cup \{\perp\}$ where K is the key space, X is the message space and Σ is the output space. For an input $\boldsymbol{y} \in K$ and $\boldsymbol{x} \in X$, $F(\boldsymbol{y}, \boldsymbol{x}) = \langle \boldsymbol{y}, \boldsymbol{x} \rangle$.*

4 Construction

Here we show our construction of our Attribute Based Inner Product Functional Encryption (ABIPFE) scheme. Our construction uses Inner product functionality, and using CP-ABE as the access control of the decryption results.

Setup (1^λ,U)

The setup algorithm takes the attribute universe U and the security parameter λ as input. It runs a bilinear group generator $\mathcal{G}(1^\lambda)$ to get ($N = p_1p_2p_3$, $\mathbb{G}$, $\mathbb{G}_T$, e) as defined in Section 2. We let $\mathbb{G}_{p_i}$ denote the subgroup of order p_i in $\mathbb{G}$, and g is the generator of $\mathbb{G}_{p_1}$. For each attribute $i \in U$, it chooses a random value $s_i \in \mathbb{Z}_N$. Use n to denote the length of encrypted vectors, it then chooses random exponents $\alpha_j, a_j \in \mathbb{Z}_N$, and random group element $g_j \in G_{p_1}$, for all $j \in [n]$. The public parameters PK are N, g, $\{g_j\}$, $\{g_j^{a_j}\}$, $\{e(g_j, g_j)^{\alpha_j}\}$, $\{T_{i,j}\} = \{g_j^{s_i}\}, \forall i, j$. The master secret key MSK is $\{\alpha_j\}$ and a generator X_3 of G_{p_3}.

KeyGen($MSK, S, \boldsymbol{y}, PK$)

The key generation algorithm takes the master secret key MSK, an attribute set S of a user, a vector $\boldsymbol{y}$ and the public parameters PK as input.It chooses a random $t_j \in \mathbb{Z}_N, \forall j \in [n]$, and random elements $R_{0,j}, R'_{0,j}$, $R_{i,j} \in G_{p_3}$. The secret key is:

$S, H_j = g_j^{\alpha_j y_j} g_j^{a_j t_j} R_{0,j}$, $L_j = g_j^{t_j} R'_{0,j}$, $K_{i,j} = T_{i,j}^{t_j} R_{i,j} \forall i \in S$, and $j \in [n]$.

Encrypt ((M,ρ),PK,$\boldsymbol{x}$)

The Encrypt algorithm takes an access structure (M,ρ), a vector $x = (x_0, x_1, \ldots, x_n)$ and the public parameters PK as input, where M is an $l \times n$ matrix and ρ is map from each row M_x of M to an attribute $\rho(x)$. The encryption algorithm chooses random vectors $\boldsymbol{v}_j \in \mathbb{Z}_N^n$, $\forall j \in [n]$, denoted $\boldsymbol{v}_j = (k_j, v_{2,j}, \ldots, v_{n,j})$. For each row M_x of M and each $j \in [n]$, it chooses a random $r_{x,j} \in \mathbb{Z}_N$. The ciphertext is (we also include (M, rho) in the ciphertext, though we do not write it below):

$A_j = e(g, g)^{x_j} e(g_j, g_j)^{\alpha_j k_j}, B_j = g_j^{k_j}$, $C_{x,j} = g_j^{a_j M_x \cdot \boldsymbol{v_j}} T_{\rho(x)}^{-r_{x,j}}, D_{x,j} = g_j^{r_{x,j}} \forall x$ and $\forall j \in [n]$.

Decrypt(CT, PK, SK)

The decryption algorithm takes a ciphertext CT, the public parameters PK, and a secret key SK as input. It can find constants $\omega_x \in \mathbb{Z}_N$ such that $\Sigma_{\rho(x) \in S}$ $\omega_x M_x = (1, 0, ..., 0)$. It then computes:

$$\frac{\Pi_{j\in[n]}e(B_j,H_j)}{\Pi_{j\in[n]}(e(C_{x,j},L_j)e(D_{x,j},K_{\rho(x),j}))^{\omega_x}} = \Pi_{j\in[n]}e(g_j,g_j)^{\alpha_j k_j y_j},$$

Then compute

$$\frac{\Pi_{j\in[n]}(A_j)^{y_j}}{\Pi_{j\in[n]}e(g_j,g_j)^{\Pi\alpha_j k_j y_j}} = e(g,g)^{\langle x,y\rangle}.$$

Correctness Analysis. We give the correctness analysis here. If the attribute set S in the secret key satisfies the access structure (M,ρ) in the ciphertext, we have:

$$\begin{aligned}
&\frac{\Pi_{j\in[n]}e(B_j,H_j)}{\Pi_{j\in[n]}(e(C_{x,j},L_j)e(D_{x,j},K_{\rho(x),j}))^{\omega_x}}\\
&= \frac{\Pi_{j\in[n]}e(g_j^{k_j},g_j^{\alpha_j y_j}g_j^{a_j t_j}R_{0,j})}{\Pi_{j\in[n]}(e(g_j^{a_j M_x\cdot\boldsymbol{v_j}}T_{\rho(x),j}^{-r_{x,j}},g_j^{t_j}R'_{0,j})e(g_j^{r_{x,j}},T_{\rho(x),j}^{t_j}R_{i,j}))^{\omega_x}}\\
&= \frac{\Pi_{j\in[n]}e(g_j,g_j)^{\alpha_j y_j k_j}e(g_j^{k_j},g_j^{a_j t_j}R_{0,j})}{\Pi_{j\in[n]}(e(g_j^{a_j k_j}g_j^{-s_{\rho(x)}r_{x,j}},g_j^{t_j}R'_{0,j})e(g_j^{r_{x,j}},g_j^{-s_{\rho(x)}t_j}R_{i,j}))}\\
&= \frac{\Pi_{j\in[n]}e(g_j,g_j)^{\alpha_j y_j k_j}e(g_j,g_j)^{k_j a_j t_j}}{\Pi_{j\in[n]}(e(g_j,g_j)^{k_j a_j t_j}e(g_j,g_j)^{-s_{\rho(x)}r_{x,j}t_j}e(g_j,g_j)^{s_{\rho(x)}r_{x,j}t_j})}\\
&= \Pi_{j\in[n]}e(g_j,g_j)^{\alpha_j k_j y_j},
\end{aligned}$$

$$\frac{\Pi_{j\in[n]}(A_j)^{y_j}}{\Pi_{j\in[n]}e(g_j,g_j)^{\Pi\alpha_j k_j y_j}} = \frac{\Pi_{j\in[n]}(e(g,g)^{x_j}e(g_j,g_j)^{\alpha_j k_j})^{y_j}}{\Pi_{j\in[n]}e(g_j,g_j)^{\Pi\alpha_j k_j y_j}} = e(g,g)^{\langle x,y\rangle}.$$

5 Security Proof

Before we give our proof of security, we need to define two additional structures: semi-functional ciphertexts and keys. These will not be used in the real system, but will be needed in our proof.

Semi-functional Ciphertext

A semi-functional ciphertext is formed as follows. We let h denote a generator of G_{p_2}.For each attribute $i \in S$ and $j \in [n]$, we choose random exponent c_j modulo N, random values $z_{i,j} \in \mathbb{Z}_N$, random values $\gamma_{x,j} \in \mathbb{Z}_N$ associated to matrix rows x, and random vectors $u_j \in \mathbb{Z}_N^n$. Then:

$A_j = e(g,g)^{x_j}e(g_j,g_j)^{k_j}, B_j = g_j^{k_j}h^{c_j},$

$C_{x,j} = g_j^{a_j A_x\cdot\boldsymbol{v_j}}T_{\rho(x)}^{-r_{x,j}}h^{A_x\cdot u_j+\gamma_{x,j}k\rho(x)}, D_x = g_j^{r_{x,j}}h^{-\gamma_{x,j}}, \forall x$ and $\forall j \in [n]$.

Semi-functional Key

A semi-functional key will take on one of two forms. A semi-functional key of type 1 is formed as follows. For each $j \in [n]$, exponents $t_j, d_j, b_j \in \mathbb{Z}_N$ and elements $R_{0,j}, R'_{0,j}, R_{i,j} \in G_{p_3}$ are chosen randomly. The key is set as:

$S, H_j = g_j^{\alpha_j y_j}g_j^{a_j t_j}R_{0,j}h^{d_j}$, $L_j = g_j^{t_j}R'_{0,j}h^{b_j}$, $K_{i,j} = T_{i,j}^{t_j}R_{i,j}h^{b_j z_{i,j}}\forall i \in S$, and $j \in [n]$.

A semi-functional key of type 2 is formed without the terms g_2^b and $g_2^{bz_i}$ (one could also interpret this as setting $b = 0$):

$S, H_j = g_j^{\alpha_j y_j}g_j^{a_j t_j}R_{0,j}h^{d_j}$, $L_j = g_j^{t_j}R'_{0,j}$, $K_{i,j} = T_{i,j}^{t_j}R_{i,j}\forall i \in S$, and $j \in [n]$.

We note that when we use a semi-functional key to decrypt a semi-functional ciphertext, we are left with an additional term: $e(g,h)^{c_j d_j - b_j u_{j,1}}$, where $u_{j,1}$

denotes the first coordinate of u_j (i.e. $(1, 0, \ldots, 0) \cdot u_j$). We also note that these values $z_{i,j}$ are common to semi-functional ciphertexts and semi-functional keys of type 1. These $z_{i,j}$ terms always cancel when semi-functional keys are paired with semi-functional ciphertexts, so they do not hinder decryption. Instead, they are used as blinding factors to hide the value being shared in the G_{p_2} subgroup of a semi-functional ciphertext (the value $u_{j,1}$) from an attacker who cannot decrypt. This is where our one-use restriction is crucial: an attacker with a single semi-functional key of type 1 which cannot decrypt the challenge ciphertext should only be able to gain very limited information-theoretic knowledge of the $z_{i,j}$ values. If attributes are used multiple times, too many $z_{i,j}$ values may be exposed to the attacker. In each of the games we define below, at most one key is semi-functional of type 1 and all other semi-functional keys are type 2. This is to avoid information-theoretically leaking the $z_{i,j}$ values by using them in multiple keys at once.

We call a semi-functional key of type 1 *nominally* semi-functional if $c_j d_j - b_j u_{j,1} = 0$. Notice that when such a key is used to decrypt a corresponding semi-functional ciphertext, decryption will succeed.

We will prove the security of our system from Assumptions 1, 2, and 3 using a hybrid argument over a sequence of games. The first game, $Game_{Real}$, is the real security game (the ciphertext and all the keys are normal). In the next game, $Game_0$, all of the keys will be normal, but the challenge ciphertext will be semi-functional. We let q denote the number of key queries made by the attacker. For k from 1 to q, we define:

$Game_{k,1}$ In this game, the challenge ciphertext is semi-functional, the first $k-1$ keys are semi-functional of type 2, the k^{th} key is semi-functional of type 1, and the remaining keys are normal.

$Game_{k,2}$ In this game, the challenge ciphertext is semi-functional, the first k keys are semi- functional of type 2, and the remaining keys are normal.

We note that in $Game_{q,2}$, all of the keys are semi-functional of type 2. In the final game, $Game_{Final}$, all keys are semi-functional of type 2 and the ciphertext is a semi-functional encryption of a random message, independent of the two messages provided by the attacker. In $Game_{Final}$, the attacker's advantage is 0. We will prove these games are indistinguishable in the following four lemmas. For notational purposes in the lemmas below, we think of $Game_{0,2}$ as another way of denoting $Game_0$.

Lemma 1 Suppose there exists a polynomial time algorithm $\mathcal{A}$ such that $Game_{Real}Adv_{\mathcal{A}} - Game_0Adv_{\mathcal{A}} = \epsilon$. Then we can construct a polynomial time algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 1.

Lemma 2. Suppose there exists a polynomial time algorithm $\mathcal{A}$ such that $Game_{k-1,2}Adv_{\mathcal{A}} - Game_{k,1}Adv_{\mathcal{A}} = \epsilon$. Then we can construct a polynomial time algorithm $\mathcal{B}$ with advantage negligibly close to ϵ in breaking Assumption 2.

Lemma 3 Suppose there exists a polynomial time algorithm $\mathcal{A}$ such that $Game_{k,1}Adv_{\mathcal{A}} - Game_{k,2}Adv_{\mathcal{A}} = \epsilon$. Then we can construct a polynomial time algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 2.

Lemma 4 Suppose there exists a polynomial time algorithm $\mathcal{A}$ such that $Game_{q,2}Adv_{\mathcal{A}} - Game_{Final}Adv_{\mathcal{A}} = \epsilon$. Then we can construct a polynomial time algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 3.

Considering the page limit, the proof of **Lemma 1-4** can be found in **Appendix A**.

We have now proven the following theorem:

Theorem 1 If Assumptions 1, 2, and 3 hold, then our CP-ABE system is secure.

Proof. If Assumptions 1, 2, and 3 hold, then we have shown by the previous lemmas that the real security game is indistinguishable from $Game_{Final}$, in which the value of β is information- theoretically hidden from the attacker. Hence the attacker cannot attain a non-negligible advantage in breaking the CP-ABE system. □

6 Conclusions

In view of the growing demand of secure data sharing and computations on sensitive data, improving functionality and fine grained data access becomes a significant question in Functional Encryption. In this paper, we mainly consider the problem of enhancing the functionality of functional encryption such that the decryption can base on the attributes of a user, i.e., users with different attributes will get different decryption results. We defines Attribute Based Functional Encryption (ABFE) scheme, and provide the first Attribute-Based Inner Product Functional Encryption (ABIPFE) scheme that is based on simple assumptions, as well as an instantiation of Attribute Based Inner Product functional Encryption, which is a practical application of Functional Encryption.

In the future research of ABFE, constructions for more computations other than the inner product is expected. Other future problems includes improving the access control policy and realizing multi-input ABFE.

References

[SW05] Amit Sahai and Brent Waters. "Fuzzy identity-based encryption". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2005, pp. 457–473.

[Goy+06] Vipul Goyal et al. "Attribute-based encryption for fine-grained access control of encrypted data". In: *Proceedings of the 13th ACM conference on Computer and communications security*. Acm. 2006, pp. 89–98.

[BSW07] John Bethencourt, Amit Sahai, and Brent Waters. "Ciphertext-policy attribute-based encryption". In: *2007 IEEE symposium on security and privacy (SP'07)*. IEEE. 2007, pp. 321–334.

[CN07] Ling Cheung and Calvin Newport. "Provably secure ciphertext policy ABE". In: *Proceedings of the 14th ACM conference on Computer and communications security*. ACM. 2007, pp. 456–465.

[OSW07] Rafail Ostrovsky, Amit Sahai, and Brent Waters. "Attribute-based encryption with non-monotonic access structures". In: *Proceedings of the 14th ACM conference on Computer and communications security*. ACM. 2007, pp. 195–203.

[Wat09] Brent Waters. "Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions". In: *Advances in Cryptology-CRYPTO 2009*. Springer, 2009, pp. 619–636.

[LW10] Allison Lewko and Brent Waters. "New techniques for dual system encryption and fully secure HIBE with short ciphertexts". In: *Theory of Cryptography Conference*. Springer. 2010, pp. 455–479.

[OT10] Tatsuaki Okamoto and Katsuyuki Takashima. "Fully secure functional encryption with general relations from the decisional linear assumption". In: *Annual cryptology conference*. Springer. 2010, pp. 191–208.

[ONe10] Adam O'Neill. "Definitional Issues in Functional Encryption." In: *IACR Cryptology ePrint Archive* 2010 (2010), p. 556.

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. "Functional encryption: Definitions and challenges". In: *Theory of Cryptography Conference*. Springer. 2011, pp. 253–273.

[Wat11] Brent Waters. "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization". In: *International Workshop on Public Key Cryptography*. Springer. 2011, pp. 53–70.

[Yam+11] Shota Yamada et al. "Generic constructions for chosen-ciphertext secure attribute based encryption". In: *International Workshop on Public Key Cryptography*. Springer. 2011, pp. 71–89.

[Ber+13] Daniel J Bernstein et al. "Candidate Indistinguishability Obfuscation and Functional Encryption for all circuits". In: (2013).

[Boy13] X. Boyen. "Attribute-based functional encryption in lattices". In: *Theory of Cryptography (TCC2013)*. 2013, pp. 122–142.

[BCP14] Elette Boyle, Kai-Min Chung, and Rafael Pass. "On extractability obfuscation". In: *Theory of Cryptography Conference*. Springer. 2014, pp. 52–73.

[Abd+15] Michel Abdalla et al. "Simple functional encryption schemes for inner products". In: *IACR International Workshop on Public Key Cryptography*. Springer. 2015, pp. 733–751.

[Che+15] Jung Hee Cheon et al. "Cryptanalysis of the multilinear map over the integers". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2015, pp. 3–12.

[Wat15] Brent Waters. "A punctured programming approach to adaptively secure functional encryption". In: *Annual Cryptology Conference*. Springer. 2015, pp. 678–697.

[Abd+16] Michel Abdalla et al. “Better Security for Functional Encryption for Inner Product Evaluations.” In: *IACR Cryptology ePrint Archive* 2016 (2016), p. 11.

[ALS16] Shweta Agrawal, Benoıt Libert, and Damien Stehlé. “Fully secure functional encryption for inner products, from standard assumptions”. In: *Annual Cryptology Conference*. Springer. 2016, pp. 333–362.

[AS16] Prabhanjan Ananth and Amit Sahai. “Functional encryption for turing machines”. In: *Theory of Cryptography Conference*. Springer. 2016, pp. 125–153.

[Cor+16] Jean-Sébastien Coron et al. “Cryptanalysis of GGH15 multilinear maps”. In: *Annual Cryptology Conference*. Springer. 2016, pp. 607–628.

[Gar+16] Sanjam Garg et al. “Candidate indistinguishability obfuscation and functional encryption for all circuits”. In: *SIAM Journal on Computing* 45.3 (2016), pp. 882–929.

[Abd+17] Michel Abdalla et al. “Multi-input inner-product functional encryption from pairings”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2017, pp. 601–626.

[Apo+17] Daniel Apon et al. “Cryptanalysis of indistinguishability obfuscations of circuits over GGH13”. In: *LIPIcs-Leibniz International Proceedings in Informatics*. Vol. 80. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2017.

[Bal+17] Carmen Elisabetta Zaira Baltico et al. “Practical functional encryption for quadratic functions with applications to predicate encryption”. In: *Annual International Cryptology Conference*. Springer. 2017, pp. 67–98.

[CGH17] Yilei Chen, Craig Gentry, and Shai Halevi. “Cryptanalyses of candidate branching program obfuscators”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2017, pp. 278–307.

[Abd+18] Michel Abdalla et al. “Multi-input functional encryption for inner products: function-hiding realizations and constructions without pairings”. In: *Annual International Cryptology Conference*. Springer. 2018, pp. 597–627.

Appendix A Proof of Lemmas

Proof of Lemma 1. $\mathcal{B}$ is given $g_j, X_{3,j}, \tau_j$. It will simulate $Game_{Real}$ or $Game_0$ with $\mathcal{A}$. $\mathcal{B}$ chooses random exponents $a_j, \alpha_j \in Z_N$ and a random exponent $s_i \in \mathbb{Z}_N$ for each attribute i in the system. It then sends $\mathcal{A}$ the public parameters:

$PK = \{N, \{g_j\}, \{g_j^{a_j}\}, \{e(g_j, g_j)^{\alpha_j}\}, \{T_{i,j}\} = \{g_j^{s_i}\}, \forall i, j\}$

It can generate normal keys in response to $\mathcal{A}$'s key requests by using the key generation algorithm, since it knows the $MSK = \alpha_j, X_3$.

$\mathcal{A}$ sends $\mathcal{B}$ two messages x_0, x_1 and an access matrix (A^*, ρ). To make the challenge ciphertext, $\mathcal{B}$ will implicitly set $g_j^{k_j}$ to be the G_{p_1} part of τ_j (we mean that τ_j is the product of $g_j^{k_j} \in G_{p_1}$ and possibly an element of G_{p_2}). It chooses a random $\beta \in 0, 1$ and sets:

$A_j = e(g,g)^{x_j} e(g_j, \tau_j), B_j = \tau_j,$

To form $C_{x,j}$ for each row x of A^*, $\mathcal{B}$ first chooses random values $v'_{2,j}, \ldots, v'_{n,j}$ $\in \mathbb{Z}_N$ and creates the vector $v'_j = (1, v'_{2,j}, \ldots, v'_{n,j})$. It also chooses a random $r'_{x,j} \in \mathbb{Z}_N$. It sets:

$C_{x,j} = \tau_j^{a_j A_x \cdot \boldsymbol{v'_j}} \tau_j^{-r'_{x,j} s_{\rho(x)}}$, $D_x = \tau_j^{r'_{x,j}} \forall x, j.$

We note that this implicitly sets $v_j = (k_j, sv'_{2,j}, \ldots, sv'_{n,j})$ and $r_{x,j} = r'_{x,j} k_j$. Modulo p_1, this v_j is a random vector with first coordinate k_j and $r_{x,j}$ is a random value. So if $\tau_j \in G_{p_1}$, this is a properly distributed normal ciphertext.

If $\tau_j \in G_{p_1 p_2}$, we let h^{c_j} denote the G_{p_2} part of τ_j (i.e. $\tau_j = g_j^{k_j} h^c$). We then have a semi- functional ciphertext with $u_j = c_j a_j v'_j$, $\gamma_{x,j} = -c r'_{x,j}$, and $z_{\rho(x)} = k_{\rho(x)}$. Though we are reusing values from the G_{p_1} parts here, this does not result in unwanted correlations. The values of $a_j, v'_{2,j}, \ldots, v'_{n,j}, r'_{x,j}, k_{\rho(x)}$ modulo p_2 are uncorrelated from their values modulo p_1 by the Chinese Remainder Theorem, so this is a properly distributed semi-functional ciphertext. Hence, $\mathcal{B}$ can use the output of $\mathcal{A}$ to break Assumption 1 with advantage ϵ. $\square$

Proof of Lemma 2. $\mathcal{B}$ is given $g_j, X_{1,j} X_{2,j}, X_{3,j}, Y_{2,j} Y_{3,j}, \tau_j$. It will simulate $Game_{k-1,2}$ or $Game_{k,1}$ with $\mathcal{A}$. It chooses random exponents $a_j, \alpha_j \in \mathbb{Z}_N$ and a random exponent $s_i \in \mathbb{Z}_N$ for each attribute i in the system. It then sends $\mathcal{A}$ the public parameters:

$PK = N, g_j, g_j^{a_j}, e(g,g)^{\alpha_j}, T_{i,j} = g_j^{s_i} \forall i, j.$

To make the first $k-1$ keys semi-functional of type 2, $\mathcal{B}$ responds to each key request by choosing random $t_j \in \mathbb{Z}_N$, random elements $R'_{0,j}, R_{i,j}$ of G_{p_3} , and setting:

$H_j = g_j^{\alpha_j} g_j^{a_j t_j} (Y_{2,j} Y_{3,j})^{t_j}, L_j = g_j^{t_j} R'_{0,j}, K_{i,j} = T_{i,j}^{t_j} R_{i,j} \forall i \in S$ and $j \in [n]$.

We note that H_j is properly distributed because the values of t_j modulo p_2 and p_3 are uncorrelated to its value modulo p_1. To make normal keys for requests $> k$, $\mathcal{B}$ can simply run the key generation algorithm since it knows the MSK.

To make key k, $\mathcal{B}$ will implicitly set $g_j^{t_j}$ equal to the G_{p_1} part of τ_j. $\mathcal{B}$ chooses random elements $R_{0,j}, R'_{0,j}, R_{i,j}$ in G_{p_3} and sets:

$H_j = g_j^{\alpha_j} \tau_j^{a_j} R_{0,j}, L_j = \tau_j R'_{0,j}, K_{i,j} = \tau_j^{s_i} R_{i,j} \forall i \in S$ and $j \in [n]$.

We note that if $\tau_j \in G_{p_1 p_3}$, this is a properly distributed normal key. If $\tau_j \in G$, this is a semi-functional key of type 1. In this case, we have implicitly set $z_{i,j} = s_i$. If we let h^{b_j} denote the G_{p_2} part of τ_j, we have that $d_j = b_j a_j$ modulo p_2 (i.e. the G_{p_2} part of H_j is $h^{b_j a_j}$, the G_{p_2} part of L is h^{b_j}, and the G_{p_2} part of K_i is $h^{b_j z_{i,j}}$. Note that the value of $z_{i,j}$ modulo p_2 is uncorrelated from the value of s_i modulo p_1.

$\mathcal{A}$ sends $\mathcal{B}$ two messages x_0, x_1 and an access matrix (A^*, ρ). To make the semi-functional challenge ciphertext, $\mathcal{B}$ implicitly sets $g_j^{k_j} = X_{1,j}$ and $h^{c_j} = X_{2,j}$. It chooses random values $u_{2,j}, \ldots, u_{n,j} \in \mathbb{Z}_N$ and defines the vector u'_j

as $u'_j = (a_j, u_{2,j}, \ldots, u_{n,j})$. It also chooses random exponents $r'_{x,j} \in \mathbb{Z}_N$. The ciphertext is formed as:

$A_j = e(g,g)^{x_j} e(g_j, X_{1,J} X_{2,j}), B_j = X_{1,j} X_{2,j}$,

$C_{x,j} = X_{1,J} X_{2,J}^{A_x \cdot \boldsymbol{v_j}} X_{1,J} X_{2,J}^{-r_{x,j} s_{\rho(x)}}, D_x = X_1 X_2^{r_{x,j}} \forall x$ and $\forall j \in [n]$.

We note that this sets $v_j = s_j a_j^{-1} u'_j$ and $u_j = c_j u'_j$, so s_j is being shared in the G_{p_1} subgroup and $c_j a_j$ is being shared in the G_{p_2} subgroup. This also implicitly sets $r_{x,j} = r'_{x,j} s_j, \gamma_{x,j} = -c_j r'_{x,j}$. The values $z_{\rho(x),j} = s_{\rho(x),j}$ match those in the k^{th} key if it is semi-functional of type 1, as required.

The k^{th} key and ciphertext are almost properly distributed, except for the fact that the first coordinate of u_j (which equals $a_j c_j$) is correlated with the value of a modulo p_2 that also appears in key k if it is semi-functional. In fact, if the k^{th} key could decrypt the challenge ciphertext we would have $c_j d_j - b_j u_{1,j} = c_j b_j a_j - b_j c_j a_j = 0$ modulo p_2, so our key is either normal or nominally semi-functional. We must argue that this is hidden to the attacker $\mathcal{A}$, who cannot request any keys that can decrypt the challenge ciphertext.

To argue that the value being shared in G_{p_2} in the challenge ciphertext is information- theoretically hidden, we appeal to our restriction that attributes are only used once in labeling the rows of the matrix. Since the k^{th} key cannot decrypt the challenge ciphertext, the rowspace R formed by the rows of the matrix whose attributes are in the key does not include the vector $(1, 0, \ldots, 0)$. (We may assume this holds modulo p_2 - if not, a non-trivial factor of N can be found, which breaks our complexity assumptions.) This means there is some vector w_j which is orthogonal to R and not orthogonal to $(1, 0, ..., 0)$ (modulo p_2). We fix a basis including w, and we can then write $u_j = f_j w_j + u''_j$ modulo p_2, where $f_j \in \mathbb{Z}_{p_2}$ and u''_j is in the span of the basis elements not equal to w_j (and u''_j is distributed uniformly randomly in this space). We note that u''_j reveals no information about f_j , and that $u_{1,j} = u_j \cdot (1, 0, \ldots, 0)$ cannot be determined from u''_j alone - some information about f_j is needed, since w_j is not orthogonal to $(1, 0, \ldots, 0)$. However, the shares corresponding to rows whose attributes are in the key only reveal information about u''_j, since w_j is orthogonal to R.

The only places $f_j w_j$ appears are in equations of the form:

$A^*_x \cdot u_j + \gamma_{x,j} z_{\rho(x),j}$,

where the $\rho(x)$'s are each *unique* attributes not appearing the k^{th} key. As long as each $\gamma_{x,j}$ is not congruent to 0 modulo p_2, each of these equations introduces a new unknown $z_{\rho(x),j}$ that appears nowhere else, and so no information about f_j can be learned by the attacker. More precisely, for each potential value of $u_{1,j}$, there are an equal number of solutions to these equations, so each value is equally likely. Hence, the value being shared in the G_{p_2} subgroup in the semi-functional ciphertext is information-theoretically hidden, as long as each $\gamma_{x,j}$ is non-zero modulo p_2. The probability that any of the $\gamma_{x,j}$ values are congruent to 0 modulo p_2 is negligible. Thus, the ciphertext and key k_j are properly distributed in the attacker's view with probability negligibly close to 1.

Thus, if $\tau \in G_{p_1 p_3}$, then $\mathcal{B}$ has properly simulated $Game_{k-1,2}$, and if $\tau \in G$ and all the $\gamma_{x,j}$ values are non-zero modulo p2, then $\mathcal{B}$ has properly simulated $Game_{k,1}$. $\mathcal{B}$ can therefore use the output of $\mathcal{A}$ to gain advantage negligibly close

to ϵ in breaking Assumption 2. $\square$

Proof of Lemma 3. This proof is very similar to the proof of the previous lemma, but the information- theoretic argument is no longer required. $\mathcal{B}$ is given $g, X_{1,j}X_{2,j}, X_{3,j}$, $Y_{2,j}Y_{3,j}, \tau_j$. It will simulate $Game_{k,1}$ or $Game_{k,2}$ with $\mathcal{A}$. It chooses random exponents $a_j, \alpha_j \in \mathbb{Z}_N$ and a random exponent $s_i \in \mathbb{Z}_N$ for each attribute i in the system. It then sends $\mathcal{A}$ the public parameters:

$PK = N, g_j, g_j^{a_j}, e(g,g)^{\alpha_j}, T_{i,j} = g_j^{s_i} \forall i, j.$

The $k-1$ semi-functional keys of type 2, the normal keys > k, and the challenge ciphertext are constructed exactly as in the previous lemma. This means the ciphertext is sharing the value $a_j c_j$ in the G_{p_2} subgroup. This time, this will not be correlated with key k in any way, so this value is random modulo p_2 (note that a modulo p_1 and a modulo p_2 are uncorrelated).

To make key k, we proceed as we did before, but we additionally choose a random exponent $\bar{g} \in \mathbb{Z}_N$ and set:

$S, H_j = g_j^{\alpha_j y_j} \tau_j^{a_j} R_{0,j} (Y_{2,j}Y_{3,j})^{\bar{g}},$

$L_j = \tau_j R'_{0,j}$, $K_{i,j} = \tau_j^{s_i} R_{i,j} \forall i \in S$, and $j \in [n]$.

The only change we have made here is adding the $(Y_{2,j}Y_{3,j}) \ldots \bar{g}$ term. This randomizes the G_{p_2} part of K, so the key is no longer nominally semi-functional. If we tried to decrypt the semi- functional ciphertext with it, decryption would fail (we no longer have the cancellation $c_j d_j - b_j u_{1,j} \equiv 0 (mod p_2)$).

If $T \in G_{p_1 p_3}$, this is a properly distributed semi-functional key of type 2. If $T \in G$, this is a properly distributed semi-functional key of type 1. Hence, $\mathcal{B}$ can use the output of $\mathcal{A}$ to gain advantage ϵ in breaking Assumption 2. $\square$

Proof of Lemma 4. Again, this proof is similar to the proofs of the previous lemmas. $\mathcal{B}$ is given $g_j, g_j^{\alpha_j} X_{2,j}, X_{3,j}, g_j^{k_j} Y_{2,j}, Z_{2,j}, \tau_j$. It will simulate $Game_{q,2}$ or $Game_{Final}$ with $\mathcal{A}$. It chooses a random exponent $a_j \in \mathbb{Z}_N$ and a random exponent $s_i \in \mathbb{Z}_N$ for each attribute i in the system. It takes α_j from the assumption term $g_j^{\alpha_j} X_{2,j}$. It then sends $\mathcal{A}$ the public parameters:

$PK = N, g_j, g_j^{a_j}, e(g_j, g_j)^{\alpha_j} = e(g_j, g_j^{\alpha_j} X_{2,j}), T_{i,j} = g_j^{s_i} \forall i.$

To make the semi-functional keys of type 2, $\mathcal{B}$ responds to each key request by choosing a random $t_j \in \mathbb{Z}_N$, random elements $R_{0,j}, R'_{0,j}, R_{i,j}$ of G_{p_3} , and setting:

$H_j = g_j^{\alpha_j y_j} g_j^{a_j t_j} Z_{2,j}^{t_j} R_{0,j}$, $L_j = g_j^{t_j} R'_{0,j}$, $K_{i,j} = T_{i,j}^{t_j} R_{i,j} \forall i \in S$, and $j \in [n]$.

as in the previous lemmas.

$\mathcal{A}$ sends $\mathcal{B}$ two messages x_0, x_1 and an access matrix (A^*, ρ). To make the semi-functional challenge ciphertext, $\mathcal{B}$ will take k_j from the assumption term $g_j^{k_j} Y_{2,j}$. It chooses random values $u_{2,j}, \ldots, u_{n,j} \in \mathbb{Z}_N$ and defines the vector u'_j as $u'_j = (a_j, u_{2,j}, \ldots, u_{n,j})$. It also chooses a random exponent $r'_{x,j} \in \mathbb{Z}_N$. The ciphertext is formed as:

$A_j = e(g,g)^{x_{\beta_j}} \tau_j, B_j = g_j^{k_j} Y_{2,j},$

$C_{x,j} = (g_j^{k_j} Y_{2,j})^{A^*_x \cdot \boldsymbol{u_j}} (g_j^{k_j} Y_{2,j})^{-r'_{x,j} s_{\rho(x)}}, D_x = (g_j^{k_j} Y_{2,j})^{r'_{x,j}} \forall x$ and $\forall j \in [n]$.

We note that this sets $v_j = k_j a_j^{-1} u'_j$ and $u_j = c_j u'_j$, so k_j is being shared in the G_{p_1} subgroup and $c_j a_j$ is being shared in the G_{p_2} subgroup. This also implicitly sets $r_{x,j} = r'_{x,j} k_j, \gamma_{x,j} = -c_j r'_{x,j}$.

If $\tau_j = e(g_j, g_j)^{k_j}$, this is a properly distributed semi-functional encryption of x_b. Otherwise, it is a properly distributed semi-functional encryption of a random message in G_T . Thus, $\mathcal{B}$ can use the output $\mathcal{A}$ to gain advantage ϵ in breaking Assumption 3. $\square$