New Proof Methods for Attribute-Based Encryption: Achieving Full Security through Selective Techniques

Allison Lewko[1] * and Brent Waters[2] **

[1] The University of Texas at Austin
alewko@cs.utexas.edu
[2] University of Texas at Austin
bwaters@cs.utexas.edu

Abstract. We develop a new methodology for utilizing the prior techniques to prove selective security for functional encryption systems as a direct ingredient in devising proofs of full security. This deepens the relationship between the selective and full security models and provides a path for transferring the best qualities of selectively secure systems to fully secure systems. In particular, we present a Ciphertext-Policy Attribute-Based Encryption scheme that is proven fully secure while matching the efficiency of the state of the art selectively secure systems.

1 Introduction

Functional encryption presents a vision for public key cryptosystems that provide a strong combination of flexibility, efficiency, and security. In a functional encryption scheme, ciphertexts are associated with descriptive values x, secret keys are associated with descriptive values y, and a function $f(x, y)$ determines what a user with a key for value y should learn from a ciphertext with value x. One well-studied example of functional encryption is attribute-based encryption (ABE), first introduced in [30], in which ciphertexts and keys are associated with access policies over attributes and subsets of attributes. A key will decrypt a ciphertext if and only if the associated set of attributes satisfies the associated access policy. There are two types of ABE systems: Ciphertext-Policy ABE (CP-ABE), where ciphertexts are associated with access policies and keys are associated with sets of attributes, and Key-Policy ABE (KP-ABE), where keys

* Supported by a Microsoft Research PhD Fellowship.

** Supported by NSF CNS-0915361 and CNS-0952692, AFOSR Grant No: FA9550-08-1-0352, DARPA through the U.S. Office of Naval Research under Contract N00014-11-1-0382, DARPA N11AP20006, Google Faculty Research award, the Alfred P. Sloan Fellowship, and Microsoft Faculty Fellowship, and Packard Foundation Fellowship. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Department of Defense or the U.S. Government.

are associated with access policies and ciphertexts are associated with sets of attributes.

To achieve desired flexibility, one strives to construct ABE systems for suitably expressive types of access policies over many attributes. Current constructions allow boolean formulas or linear secret sharing schemes as access policies. This high level of flexibility means that keys and ciphertexts have rich structure, and there is a very large space of possible access policies and attribute sets. This presents a challenge to proving security, since a suitable notion of security in this setting must enforce collusion resistance, meaning that several users should not be able to decrypt a message that none of them are individually authorized to read. Hence a security proof must consider an attacker who can collect many different keys, just not a single one that is authorized to decrypt the ciphertext.

This requires security reductions to balance two competing goals: the simulator must be powerful enough to provide the attacker with the many keys that it adaptively requests, but it must also lack some critical knowledge that it can gain from the attacker's success. The first security proofs in the standard model for ABE systems (e.g. [30, 19, 34]) followed a very natural paradigm for balancing these two goals known as partitioning. This proof technique was previously used in the context of identity-based encryption [9, 11, 6, 7, 32]. In a partitioning proof, the simulator sets up the system so that the space of all possible secret keys is partitioned into two pieces: keys that the simulator can make and those that it cannot. To ensure that the keys the attacker requests all fall in the set of keys the simulator can produce and that any key capable of decrypting the challenge ciphertext falls in the opposite set, the prior works [30, 19, 34] had to rely on a weaker security model known as *selective security*. In the selective security model, the attacker must declare up front what the challenge ciphertext will be, *before* seeing the public parameters.

This notion of selective security is quite useful as an intermediary step, but is rather unsatisfying as an end goal. In the setting of identity-based encryption, the need for selectivity was overcome by arranging for the simulator to "guess" a partition and abort when the attacker violated its constraints [32]. However, the richer structure of attribute-based systems appears to doom this approach to incur exponential loss, since one must guess a partition that respects the partial ordering induced by the powers allocated to the individual keys.

Dual System Encryption With the goal of moving beyond the constraints of the partitioning paradigm, Waters introduced the dual system encryption methodology [33]. In a dual system security proof, the simulator is always prepared to make *any* key and *any* challenge ciphertext. The high level idea of the methodology is as follows. There are two types of keys and ciphertexts: normal and semi-functional. A key will decrypt a ciphertext properly unless *both* the key and the ciphertext are semi-functional, in which case decryption will fail with all but negligible probability. The normal keys and ciphertexts are used in the real system, while the semi-functional objects are gradually introduced in the hybrid security proof - first the ciphertext is changed from normal to semi-functional, and then the secret keys given to the attacker are changed from normal to semi-functional one by one. Ultimately, we arrive at a security game in which the simulator only has to produce semi-functional objects and security can be proved directly.

The most critical step of the hybrid proof is when a key turns semi-functional: at this point, we must leverage the fact that the key is not authorized to decrypt the (now semi-functional) challenge ciphertext in order to argue that the attacker cannot detect the change in the key. However, since we are not imposing a partition on the simulator, there is no constraint preventing the simulator itself from creating a key that is authorized to decrypt and testing the nature of the key for itself by attempting to decrypt the semi-functional ciphertext. In the first application of dual system encryption to ABE [22], this paradox was averted by ensuring that the simulator could only produce a key that would be *correlated* with the semi-functional ciphertext so that decryption would succeed in the simulator's view, regardless of the presence or absence of semi-functionality. This correlation between a semi-functional key and semi-functional ciphertext was called *nominal semi-functionality*. It was argued that this correlation was hidden information-theoretically from the attacker, who cannot request keys authorized to decrypt the challenge ciphertext. This provided the first proof of full security for an ABE scheme in the standard model.

The One-Use Restriction The information-theoretic argument in [22] required a one-use restriction on attributes in access formulas/LSSS matrices, meaning that a single attribute could only be used once in a policy. This can be extended to a system which allows reuse of attributes by setting a fixed bound M on the maximum number of times an attribute may be used and having separate parameters for each use. This scales the size of the public parameters by M, as well as the size of secret keys for CP-ABE systems[3]. This approach incurs a very significant loss in efficiency, and has been inherited by all fully secure schemes to date ([24, 28] employ the same technique). This loss in efficiency is costly enough to limit the potential applications of fully secure schemes. As an example, the recent work of [2] building verifiable computation schemes from KP-ABE systems only produces meaningful results when one starts with a KP-ABE scheme that can be proven secure *without* incurring the blowup of this encoding technique.

Our work eliminates this efficiency loss and allows unrestricted use of attributes while still proving full security in the standard model. Our main observation is motivated by the intuition that the information-theoretic step of the prior dual system proof is ceding too much ground to the attacker, since a computational argument would suffice. In fact, we are able to resurrect the earlier selective proof techniques inside the framework of dual system encryption in order to retake ground and obtain a wholly computational proof of full security.

Our Techniques Dual system encryption is typically implemented by designing a "semi-functional space" where semi-functional components of keys and ciphertexts will behave like a parallel copy of the normal components of the system,

[3] For KP-ABE systems, the ciphertext size instead will grow multiplicatively with M.

except divorced from the public parameters. This provides a mechanism allowing for *delayed parameters* in the semi-functional space, meaning that relevant variables can be defined later in the simulation instead of needing to be fixed in the setup phase. The hybrid structure of a dual system encryption argument is implemented by additionally providing a mechanism for *key isolation*, meaning that some or all of the semi-functional parameters will only be relevant to the distribution of a single semi-functional key at a time.

In combination, these two mechanisms mean that the semi-functional space has its own fresh parameters that can be decided on the fly by the simulator when they become relevant, and they are only relevant for the semi-functional ciphertext and a single semi-functional key. Previous dual system encryption arguments have used the isolated use of these delayed semi-functional parameters as a source of entropy in the attacker's view to make an information-theoretic argument. We observe that these mechanisms can also be used to implement prior techniques for selective security proofs, without needing to impose the selective restriction on the attacker.

To explain this more precisely, we consider the critical step in the hybrid security proof when a particular key becomes semi-functional. We conceptualize the unpublished semi-functional parameters as being defined belatedly when the simulator first issues either the key in question or the semi-functional ciphertext. For concreteness, we consider a CP-ABE system. If the ciphertext is issued first, then the simulator learns the challenge policy *before* defining the delayed semi-functional parameters - this is closely analogous to the setting of selective security for a CP-ABE system. If the key is issued first, then the simulator learns the relevant set of attributes before defining the delayed semi-functional parameters, and this is closely analogous to the setting of selective security for a KP-ABE system. This provides us an opportunity to combine the techniques used to prove selective security for both CP-ABE and KP-ABE systems with the dual system encryption methodology in order to obtain a new proof of full security that maintains the efficiency of selectively secure systems.

Our Results Since our approach utilizes selective techniques for both CP-ABE and KP-ABE in order to prove full security for either kind of system, we inherit the kinds of complexity assumptions needed to prove selective security in both settings. The KP-ABE scheme of [19] is proven selectively secure under the decisional bilinear Diffie-Hellman assumption, and so we are able to rely on the relatively simple 3-party Diffie-Hellman assumption for this part of our proof. The most efficient selectively secure CP-ABE scheme that is known is provided in [34], and it is proven secure under a q-based assumption (meaning that the number of terms in the assumption is parameterized by a value q that depends on the behavior of the attacker). Hence we inherit the need to rely on a q-based assumption in our security proof as well.

The dual system encryption methodology has previously been implemented both in prime order bilinear groups (e.g. in [33, 28]) and in composite order bilinear groups (e.g. in [23, 22]). The two settings provide different but roughly interchangeable mechanisms for executing delayed parameters and key isolation, and our techniques are compatible with either setting. We first present a CP-ABE construction and security proof in composite order groups, relying on a few instances of the general subgroup decision assumption to execute the delayed parameters and key isolation mechanisms. In the full version, we also present an analogous CP-ABE construction and security proof in prime order groups, relying on the decisional linear assumption for these functions. To translate our construction from the composite order setting to the prime order setting, we employ the dual pairing vector space framework developed in [26–28], along with the relevant observations in [21]. The formal statements of our complexity assumptions for the composite order setting can be found in Section 2.1, while those for the prime order setting can be found in the full version. Though we present only CP-ABE schemes in this work, we expect that our techniques are equally applicable to the KP-ABE setting.

We view our work as providing a new view of the relationship between the selective and full security models, as we illustrate a methodology for using techniques in the selective context as direct building blocks for a full security proof. We suspect that any new improvements in selectively secure systems may now translate to improvements in the full security setting. In particular, a new proof of selective security for an efficient CP-ABE system relying on a static (non q-based) assumption could likely be combined with our techniques to yield a fully secure scheme of comparable efficiency under similar assumptions. This remains an important open problem.

Other Related Work The roots of attribute-based encryption trace back to identity-based encryption (IBE), which was first conceived by Shamir [31] and then constructed by Boneh and Franklin [9] and Cocks [14]. This concept was extended to the notion of hierarchical identity-based encryption (HIBE) by Horwitz and Lynn [20], and this was first constructed by Gentry and Silverberg [17]. Subsequent constructions of IBE and HIBE can be found in [11, 6–8, 15, 16, 23, 1, 12, 25]. There have been several prior constructions of attribute-based encryption which have been shown to be selectively secure in the standard model [30, 19, 13, 29, 18, 34] or proven secure in the generic group model [5] (this is a heuristic model intended to capture an attacker who can only access group operations in a black-box fashion).

2 Preliminaries

Here we present the relevant background on composite order bilinear groups and state the complexity assumptions we use in this context. We also give background on LSSS access structures. Further background on CP-ABE systems and their formal security definition can be found in the full version.

2.1 Composite Order Bilinear Groups and Complexity Assumptions

We will first construct our system in composite order bilinear groups, which were introduced in [10]. We let $\mathcal{G}$ denote a group generator - an algorithm which takes a security parameter λ as input and outputs a description of a bilinear group G. We define $\mathcal{G}$'s output as (N, G, G_T, e), where $N = p_1p_2p_3$ is a product of three distinct primes, G and G_T are cyclic groups of order N, and $e : G^2 \to G_T$ is a map such that:

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

We refer to G as the *source group* and G_T as the *target group*. We assume that the group operations in G and G_T and the map e are computable in polynomial time with respect to λ, and the group descriptions of G and G_T include a generator of each group. We 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. We note that these subgroups are "orthogonal" to each other under the bilinear map e: if $f_i \in G_{p_i}$ and $f_j \in G_{p_j}$ for $i \neq j$, then $e(f_i, f_j)$ is the identity element in G_T. If g_1 generates G_{p_1}, g_2 generates G_{p_2}, and g_3 generates G_{p_3}, then every element f of G can be expressed as $g_1^{c_1} g_2^{c_2} g_3^{c_3}$ for some values $c_1, c_2, c_3 \in \mathbb{Z}_N$. We will refer to $g_1^{c_1}$ as the "G_{p_1} part of f", for example.

We now present the complexity assumptions we will use in composite order bilinear groups. We use the notation $X \xleftarrow{R} S$ to express that X is chosen uniformly randomly from the finite set S. We will consider groups G whose orders are products of three distinct primes. For any non-empty set $Z \subseteq \{1, 2, 3\}$, there is a corresponding subgroup of G of order $\prod_{i \in Z} p_i$. We denote this subgroup by G_Z. Our first assumption has been previously used in [23, 22], for example, and holds in the generic group model:

Assumption 1 Given a group generator $\mathcal{G}$, we define the following distribution:

$$\mathbb{G} = (N = p_1p_2p_3, G, G_T, e) \xleftarrow{R} \mathcal{G},$$

$$g_1 \xleftarrow{R} G_{p_1},\ g_2, X_2, Y_2 \xleftarrow{R} G_{p_2},\ g_3 \xleftarrow{R} G_{p_3},\ \alpha, s \xleftarrow{R} \mathbb{Z}_N,$$

$$D = (\mathbb{G}, g_1, g_2, g_3, g_1^{\alpha} X_2, g_1^{s} Y_2),\ T_0 = e(g_1, g_1)^{\alpha s},\ T_1 \xleftarrow{R} G_T.$$

We define the advantage of an algorithm $\mathcal{A}$ in breaking this assumption to be:

$$Adv^1_{\mathcal{G},\mathcal{A}}(\lambda) := \big|Pr[\mathcal{A}(D, T_0) = 1] - Pr[\mathcal{A}(D, T_1) = 1]\big|.$$

We say that $\mathcal{G}$ satisfies Assumption 1 if $Adv^1_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any PPT algorithm $\mathcal{A}$.

We next define the General Subgroup Decision Assumption for composite order bilinear groups with three prime subgroups. This was first defined in [4] more generally for groups with an arbitrary number of prime order subgroups, but three will be sufficient for our purposes. We will only use a few specific instances of this assumption, but we prefer to state its full generality here for conciseness. We note that for our prime order construction, Assumption 1 and all instances of the General Subgroup Decision Assumption will be replaced by the Decisional Linear Assumption.

The General Subgroup Decision Assumption We let $\mathcal{G}$ denote a group generator and $Z_0, Z_1, Z_2, \ldots, Z_k$ denote a collection of non-empty subsets of $\{1,2,3\}$ where each Z_i for $i \geq 2$ satisfies either $Z_0 \cap Z_i \neq \emptyset \neq Z_1 \cap Z_i$ or $Z_0 \cap Z_i = \emptyset = Z_1 \cap Z_i$. We define the following distribution:

$$\mathbb{G} = (N = p_1 p_2 p_3, G, G_T, e) \xleftarrow{R} \mathcal{G},$$

$$g_{Z_2} \xleftarrow{R} G_{Z_2}, \ \ldots, \ g_{Z_k} \xleftarrow{R} G_{Z_k}$$

$$D = (\mathbb{G}, g_{Z_2}, \ldots, g_{Z_k}), \ T_0 \xleftarrow{R} G_{Z_0}, \ T_1 \xleftarrow{R} G_{Z_1}.$$

Fixing the collection of sets $Z_0, \ldots, Z_k$, we define the advantage of an algorithm $\mathcal{A}$ in breaking this assumption to be:

$$Adv^{SD}_{\mathcal{G},\mathcal{A}}(\lambda) := \big|Pr[\mathcal{A}(D, T_0) = 1] - Pr[\mathcal{A}(D, T_1) = 1]\big|.$$

We say that $\mathcal{G}$ satisfies the General Subgroup Decision Assumption if $Adv^{SD}_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any PPT algorithm $\mathcal{A}$ and any suitable collection of subsets $Z_0, \ldots, Z_k$. This can be thought of as a family of assumptions, parameterized by the choice of the sets $Z_0, \ldots, Z_k$. All of these individual assumptions hold in the generic group model, assuming it is hard to find a non-trivial factor of N. We will assume that $\frac{1}{p_i}$ is negligible in the security parameter for each prime factor p_i of N. In particular, this means we may assume (ignoring only negligible probability events) that when an element is randomly chosen from a subgroup of G, it is in fact a generator of that subgroup.

We next introduce an assumption that we call The Three Party Diffie-Hellman Assumption in a Subgroup. This is a close relative of the standard Decisional Bilinear Diffie-Hellman Assumption, but it has a challenge term remaining in the source group and takes place in a prime order subgroup of a composite order bilinear group. These adjustments from the usual DBDH assumption allow us to use our assumption in the semi-functional space for a particular key - without affecting the normal space or the other keys.

The Three Party Diffie-Hellman Assumption in a Subgroup Given a group generator $\mathcal{G}$, we define the following distribution:

$$\mathbb{G} = (N = p_1 p_2 p_3, G, G_T, e) \xleftarrow{R} \mathcal{G},$$

$$g_1 \xleftarrow{R} G_{p_1}, \ g_2 \xleftarrow{R} G_{p_2}, \ g_3 \xleftarrow{R} G_{p_3}, \ x, y, z \xleftarrow{R} \mathbb{Z}_N,$$

$$D = (\mathbb{G}, g_1, g_2, g_3, g_2^x, g_2^y, g_2^z), \ T_0 = g_2^{xyz}, \ T_1 \xleftarrow{R} G_{p_2}.$$

We define the advantage of an algorithm $\mathcal{A}$ in breaking this assumption to be:

$$Adv^{3DH}_{\mathcal{G},\mathcal{A}}(\lambda) := \big|Pr[\mathcal{A}(D, T_0) = 1] - Pr[\mathcal{A}(D, T_1) = 1]\big|.$$

We say that $\mathcal{G}$ satisfies The Three Party Diffie-Hellman Assumption if $Adv^{3DH}_{\mathcal{G},\mathcal{A}}(\lambda)$ is a negligible function of λ for any PPT algorithm $\mathcal{A}$.

We next introduce a q-based assumption that we call The Source Group q-Parallel BDHE Assumption in a Subgroup. This is a close relative of The Decisional q-parallel Bilinear Diffie-Hellman Exponent Assumption introduced in [34], except that its challenge term remains in the source group and it takes place in a prime order subgroup of a composite order bilinear group. In the full version, we prove that the prime order variant of this assumption holds in the generic group model (the proof for this version follows analogously). Below, we use the notation $[q]$, for example, to denote the set $\{1, 2, \ldots, q\}$.

The Source Group q-Parallel BDHE Assumption in a Subgroup Given a group generator $\mathcal{G}$ and a positive integer q, we define the following distribution:

$$\mathbb{G} = (N = p_1 p_2 p_3, G, G_T, e) \xleftarrow{R} \mathcal{G},$$

$$g_1 \xleftarrow{R} G_{p_1},\ g_2 \xleftarrow{R} G_{p_2},\ g_3 \xleftarrow{R} G_{p_3},\ c, d, f, b_1, \ldots, b_q \xleftarrow{R} \mathbb{Z}_N,$$

The adversary will be given:

$$\begin{aligned} D = (&\mathbb{G}, g_1, g_3, g_2, g_2^f, g_2^{df}, g_2^c, g_2^{c^2}, \ldots, g_2^{c^q}, g_2^{c^{q+2}}, \ldots, g_2^{c^{2q}}, \\ &g_2^{c^i/b_j}\ \forall i \in [2q] \setminus \{q+1\}, j \in [q], \\ &g_2^{dfb_j}\ \forall j \in [q],\ g_2^{dfc^i b_{j'}/b_j}\ \forall i \in [q], j, j' \in [q] \text{ s.t. } j \neq j'). \end{aligned}$$

We additionally define

$$T_0 = g_2^{dc^{q+1}},\ T_1 \xleftarrow{R} G_{p_2}.$$

We define the advantage of an algorithm $\mathcal{A}$ in breaking this assumption to be:

$$Adv^q_{\mathcal{G},\mathcal{A}}(\lambda) := \left| Pr[\mathcal{A}(D, T_0) = 1] - Pr[\mathcal{A}(D, T_1) = 1] \right|.$$

We say that $\mathcal{G}$ satisfies The Source Group q-Parallel BDHE Assumption in a Subgroup if $Adv^q_{\mathcal{G},\mathcal{A}}$ is a negligible function of λ for any PPT algorithm $\mathcal{A}$.

2.2 Access Structures

Definition 1. *(Access Structure [3]) Let $\{P_1, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. An* access structure *(respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, \ldots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}} \backslash \{\}$. The sets in $\mathbb{A}$ are called the* authorized sets*, and the sets not in $\mathbb{A}$ are called the* unauthorized sets*.*

In our setting, attributes will play the role of parties and we will consider only monotone access structures. One can (inefficiently) realize general access structures with our techniques by having the negation of an attribute be a separate attribute (so the total number of attributes doubles).

Linear Secret-Sharing Schemes Our construction will employ linear secret-sharing schemes (LSSS). We use the following definition adapted from [3].

Definition 2. *(Linear Secret-Sharing Schemes (LSSS)) A secret sharing scheme Π 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 A called the share-generating matrix for Π. The matrix A has ℓ rows and n columns. For all $j = 1, \ldots, \ell$, the j^{th} row of A is labeled by a party $\rho(j)$ (ρ is a function from $\{1, \ldots, \ell\}$ to $\mathcal{P}$). 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 Av is the vector of ℓ shares of the secret s according to Π. The share $(Av)_j$ belongs to party $\rho(j)$.*

We note the *linear reconstruction* property: we suppose that Π is an LSSS for access structure $\mathbb{A}$. We let S denote an authorized set, and define $I \subseteq \{1, \ldots, \ell\}$ as $I = \{j | \rho(j) \in S\}$. Then the vector $(1, 0, \ldots, 0)$ is in the span of rows of A indexed by I, and there exist constants $\{\omega_j \in \mathbb{Z}_p\}_{j \in I}$ such that, for any valid shares $\{\lambda_j\}$ of a secret s according to Π, we have: $\sum_{j \in I} \omega_j \lambda_j = s$. These constants $\{\omega_j\}$ can be found in time polynomial in the size of the share-generating matrix A [3]. For unauthorized sets, no such constants $\{\omega_j\}$ exist. For our composite order group construction, we will employ LSSS matrices over $\mathbb{Z}_N$, where N is a product of three distinct primes.

3 CP-ABE Construction

We now present our CP-ABE scheme in composite order groups. This closely resembles the selectively secure CP-ABE scheme in [34], but with a one extra group element for each key and ciphertext. This extra group element is helpful in performing a cancelation during our security proof (when we are dealing with Phase II queries). We note that the freshly random exponent t for each key serves to prevent collusion, since it "ties" together the user's attributes. Our main system resides in the G_{p_1} subgroup, while the G_{p_2} subgroup is reserved as the semi-functional space, and the G_{p_3} subgroup provides additional randomness on keys that helps to isolate keys in the hybrid argument. We assume that messages to be encrypted as elements of the target group G_T. The notation $[\ell]$ denotes the set $\{1, \ldots, \ell\}$.

Setup$(\lambda, \mathcal{U}) \to \mathrm{PP}, \mathrm{MSK}$ The setup algorithm chooses a bilinear group G of order $N = p_1 p_2 p_3$ (3 distinct primes). We let G_{p_i} denote the subgroup of order p_i in G. It then chooses random exponents $\alpha, a, \kappa \in \mathbb{Z}_N$, and a random group element $g \in G_{p_1}$. For each attribute $i \in \mathcal{U}$, it chooses a random value $h_i \in \mathbb{Z}_N$. The public parameters PP are $N, g, g^a, g^\kappa, e(g,g)^\alpha, H_i = g^{h_i} \forall i$. The master secret key MSK additionally contains g^α and a generator g_3 of G_{p_3}.

KeyGen($\mathrm{MSK}, S, \mathrm{PP}) \rightarrow \mathrm{SK}$ The key generation algorithm chooses random exponents $t, u \in \mathbb{Z}_N$, and random elements $R, R', R'', \{R_i\}_{i\in S} \in G_{p_3}$ (this can be done by raising a generator of G_{p_3} to random exponents modulo N). The secret key is: S, $K = g^{\alpha} g^{at} g^{\kappa u} R$, $K' = g^u R'$, $K'' = g^t R''$, $K_i = H_i^t R_i \; \forall i \in S$.

Encrypt($(A, \rho), \mathrm{PP}, M) \rightarrow \mathrm{CT}$ For A an $\ell \times n$ matrix and ρ a map from each row A_j of A to an attribute $\rho(j)$, the encryption algorithm chooses a random vector $v \in \mathbb{Z}_N^n$, denoted $v = (s, v_2, \ldots, v_n)$. For each row A_j of A, it chooses a random $r_j \in \mathbb{Z}_N$. The ciphertext is (we also include (A, ρ) in the ciphertext, though we do not write it below):

$$C_0 = Me(g,g)^{\alpha s},\ \ C = g^s,\ \ C' = (g^{\kappa})^s,\ C_j = (g^a)^{A_j \cdot v} H_{\rho(j)}^{-r_j},\ \ D_j = g^{r_j}\ \ \forall j \in [\ell].$$

Decrypt($\mathrm{CT}, \mathrm{PP}, \mathrm{SK}) \rightarrow M$ For a secret key corresponding to an authorized set S, the decryption algorithm computes constants $\omega_j \in \mathbb{Z}_N$ such that $\sum_{\rho(j)\in S} \omega_j A_j = (1, 0, \ldots, 0)$. It then computes:

$$e(C,K)e(C',K')^{-1} / \prod_{\rho(j)\in S} \left(e(C_j, K'')e(D_j, K_{\rho(j)})\right)^{\omega_j} = e(g,g)^{\alpha s}.$$

Then M can be recovered as $C_0/e(g,g)^{\alpha s}$.

Correctness We observe that $e(C,K)e(C',K')^{-1} = e(g,g)^{\alpha s} e(g,g)^{sat}$. For each j, $e(C_j, K'')e(D_j, K_{\rho(j)}) = e(g,g)^{atA_j \cdot v}$, so we have:

$$\prod_{\rho(j)\in S} \left(e(C_j, K'')e(D_j, K_{\rho(j)})\right)^{\omega_j} = e(g,g)^{at \sum_{\rho(j)\in S} \omega_j A_j \cdot v} = e(g,g)^{sat}.$$

4 Security Proof

We now prove the following theorem:

Theorem 1. *Under Assumption 1, the general subgroup decision assumption, the three party Diffie-Hellman assumption in a subgroup, and the source group q-parallel BDHE assumption in a subgroup defined in Section 2.1, our CP-ABE scheme defined in Section 3 is fully secure.*

Our security proof is obtained via a hybrid argument over a sequence of games. We let Game_{real} denote the real security game in the standard definition of full security for CP-ABE schemes (see the full version for a complete description). To describe the rest of the games, we must first define semi-functional keys and ciphertexts. We let g_2 denote a fixed generator of the subgroup G_{p_2}.

Semi-functional Keys To produce a semi-functional key for an attribute set S, one first calls the normal key generation algorithm to produce a normal key consisting of $K, K', K'', \{K_i\}_{i\in S}$. One then chooses a random element $W \in G_{p_2}$ and forms the semi-functional key as: $KW, K', K'', \{K_i\}_{i\in S}$. In other words, all of the elements remain unchanged except for K, which is multiplied by a random element of G_{p_2}.

Semi-functional Ciphertexts To produce a semi-functional ciphertext for an LSSS matrix (A, ρ), one first calls the normal encryption algorithm to produce a normal ciphertext consisting of $C_0, C, C', \{C_j, D_j\}$. One then chooses random exponents $a', \kappa', s' \in \mathbb{Z}_N$, a random vector $w \in \mathbb{Z}_N^n$ with s' as its first entry, a random exponent $\eta_i \in \mathbb{Z}_N$ for each attribute i, and a random exponent $\gamma_j \in \mathbb{Z}_N$ for each $j \in [\ell]$. The semi-functional ciphertext is formed as: $C_0,\ Cg_2^{s'},\ C'g_2^{s'\kappa'},\ \{C_j g_2^{a'A_j \cdot w} g_2^{-\eta_{\rho(j)}\gamma_j},\ D_j g_2^{\gamma_j}\}$.

We observe that the structure of the elements in G_{p_2} here is similar to the structure in G_{p_1}, but is unrelated to the public parameters. More specifically, s' plays the role of s, w plays the role of v, a' plays the role of a, κ' plays the role of κ, $\eta_{\rho(j)}$ plays the role of $h_{\rho(j)}$, and γ_j plays the role of r_j. While the values of a, κ, and the values $h_{\rho(j)}$ are determined modulo p_1 by the public parameters, the values of $a', \kappa', \eta_{\rho(j)}$ are freshly random modulo p_2. These values $a', \kappa', \{\eta_i\}$ are chosen randomly once and then fixed - these same values will also be involved in additional types of semi-functional keys which we will define below.

We let Q denote the total number of key queries that the attacker makes. For each k from 0 to Q, we define Game$_k$ as follows.

Game$_k$ In this game, the ciphertext given to the attacker is semi-functional, as are the first k keys. The remaining keys are normal.

The outer structure of our hybrid argument will progress as follows. First, we transition from Game$_{real}$ to Game$_0$, then to Game$_1$, next to Game$_2$, and so on. We ultimately arrive at Game$_Q$, where the ciphertext and *all* of the keys given to the attacker are semi-functional. We then transition to Game$_{final}$, which is defined to be like Game$_Q$, except that the ciphertext given to the attacker is a semi-functional encryption of a *random message*. This will complete our security proof, since any attacker has a zero advantage in this final game.

The transitions from Game$_{real}$ to Game$_0$ and from Game$_Q$ to Game$_{final}$ are relatively easy, and can be accomplished directly via computational assumptions. The transitions from Game$_{k-1}$ to Game$_k$ require more intricate arguments. For these steps, we will need to treat Phase I key requests (before the challenge ciphertext) and Phase II key requests (after the challenge ciphertext) differently. We will also need to define two additional types of semi-functional keys:

Nominal Semi-functional Keys These keys will share the values a', κ', η_i modulo p_2 with the semi-functional ciphertext. To produce a nominal semi-functional key for an attribute set S, one first calls the normal key generation algorithm to produce a normal key consisting of $K, K', K'', \{K_i\}_{i \in S}$. One then chooses random exponents $t', u' \in \mathbb{Z}_N$ and forms the nominal semi-functional key as: $Kg_2^{a't'+\kappa'u'},\ K'g_2^{u'},\ K''g_2^{t'},\ K_i g_2^{t'\eta_i}\ \forall i \in S$. We note that a nominal semi-functional key still correctly decrypts a semi-functional ciphertext, since the terms in the G_{p_2} will cancel out upon completion of the decryption algorithm.

Temporary Semi-functional Keys These keys will still share the values η_i modulo p_2 with the semi-functional ciphertext, but the G_{p_2} component attached to K will now be randomized. More formally, to produce a temporary semi-functional key for an attribute set S, one first calls the normal key generation algorithm to produce a normal key consisting of $K, K', K'', \{K_i\}_{i \in S}$. One then chooses a random $W \in G_{p_2}$ and random exponents $t', u' \in \mathbb{Z}_N$. The temporary semi-functional key is formed as: $KW,\ K' g_2^{u'},\ K'' g_2^{t'},\ K_i g_2^{t'\eta_i}\ \forall i \in S$.

For each k from 1 to Q, we define the following additional games:

Game$_k^N$ This is like Game$_k$, except that the k^{th} key given to the attacker is a nominal semi-functional key. The first $k-1$ keys are still semi-functional in the original sense, while the remaining keys are normal.

Game$_k^T$ This is like Game$_k$, except that the k^{th} key given to the attacker is a temporary semi-functional key. The first $k-1$ keys are still semi-functional in the original sense, while the remaining keys are normal.

The fact that the values a', κ', η_i are shared among semi-functional ciphertexts, nominal semi-functional keys, and temporary semi-functional keys means that these values are fixed whenever they first appear in a security game. This could be when the semi-functional ciphertext is generated, when a nominal semi-functional key is generated, or in the case of the η_i values, when a temporary semi-functional key is generated. The structure of temporary semi-functional keys is designed to fit the outcome of applying selective proof techniques to a single key and ciphertext pair within our hybrid game sequence.

In order to get from Game$_{k-1}$ to Game$_k$ in our hybrid argument, we will transition first from Game$_{k-1}$ to Game$_k^N$, then to Game$_k^T$, and finally to Game$_k$. The transition from Game$_k^N$ to Game$_k^T$ will require different computational assumptions for Phase I and Phase II key queries. We let Q_1 denote the number of Phase I queries, and we will address this transition separately for $k \leq Q_1$ and $k > Q_1$. Our handling of Phase I queries will closely resemble the selective security proof strategy for KP-ABE in [19], while our handling of Phase II queries will closely resemble the selective security proof strategy for CP-ABE in [34].

The original versions of these arguments in [19, 34] relied on assumptions very close to ours, with the main difference being that the assumptions in [19, 34] had challenge terms in the target group G_T instead of G. This is because the selective security arguments could afford to deal with all keys at once, and hence could use an assumption with a challenge in the target group to change the ciphertext to an encryption of a random message. This kind of change simultaneously affects the interaction of the ciphertext with *all* keys. In our hybrid framework, we need to handle keys individually, and hence we use an assumption with a challenge in the source group to change the nature of individual keys one at a time, saving our progress incrementally until we arrive at the final step and can afford to change to an encryption of a random message.

Our hybrid argument is accomplished in the following lemmas. Due to space constraints, some of the more standard lemma proofs are omitted here, but can be found in the full version.

Lemma 1. *Under the general subgroup decision assumption, no polynomial time attacker can achieve a non-negligible difference in advantage between Game$_{real}$ and Game$_0$.*

Lemma 2. *Under the general subgroup decision assumption, no polynomial time attacker can achieve a non-negligible difference in advantage between* Game_{k-1} *and* Game_k^N *for any* k *from 1 to* Q.

The proofs of these first two lemmas can be found in the full version.

Lemma 3. *Under the three party Diffie-Hellman assumption in a subgroup (and assuming it is hard to find a non-trivial factor of* N*), no polynomial time attacker can achieve a non-negligible difference in advantage between* Game_k^N *and* Game_k^T *for an* k *from 1 to* Q_1 *(recall these are all the Phase I queries).*

Proof. Given a PPT attacker $\mathcal{A}$ achieving a non-negligible difference in advantage between Game_k^N and Game_k^T for some k between 1 and Q_1, we will create a PPT algorithm $\mathcal{B}$ to break the three party Diffie-Hellman assumption in a subgroup. $\mathcal{B}$ is given $g_1, g_2, g_3, g_2^x, g_2^y, g_2^z, T$, where T is either g_2^{xyz} or a random element of G_{p_2}. $\mathcal{B}$ will simulate either Game_k^N or Game_k^T with $\mathcal{A}$ depending on the nature of T.

$\mathcal{B}$ first chooses random exponents $\alpha, a, \kappa, \{h_i\} \in \mathbb{Z}_N$ and sets the public parameters as: $\text{PP} = \{N,\ g = g_1,\ g^a = g_1^a,\ g^\kappa = g_1^\kappa,\ e(g_1, g_1)^\alpha,\ H_i = g_1^{h_i}\ \forall i\}$. It gives these to $\mathcal{A}$. We note that $\mathcal{B}$ knows the MSK, and hence can use the normal key generation algorithm to make normal keys in response to $\mathcal{A}$'s key requests from the $k+1$ request and onward. To respond to $\mathcal{A}$'s first $k-1$ key requests, $\mathcal{B}$ creates semi-functional keys by first creating a normal key and then multiplying K by a random element of G_{p_2} (this can be obtained by raising the generator g_2 to a random exponent modulo N).

We let S denote the attribute set requested in the k^{th} key query by $\mathcal{A}$. Since we are assuming the k^{th} key query occurs in Phase I, S is declared *before* $\mathcal{B}$ must produce the challenge ciphertext. This allows $\mathcal{B}$ to define the values η_i modulo p_2 to be shared by the k^{th} key and the semi-functional ciphertext *after* learning the set S. To set these values, $\mathcal{B}$ chooses random exponents $\eta_i \in \mathbb{Z}_N$ for each $i \in S$. For $i \notin S$, it implicitly sets η_i modulo p_2 to be equal to $x\tilde{\eta}_i$ modulo p_2, where random exponents $\tilde{\eta}_i \in \mathbb{Z}_N$ are chosen for each $i \notin S$. It also implicitly sets a' equal to xy modulo p_2.

To form the k^{th} key, $\mathcal{B}$ first calls the normal key generation algorithm to produce a normal key, $K, K', K'', \{K_i\}_{i \in S}$. It then chooses random exponents $\kappa', u' \in \mathbb{Z}_N$ and implicitly sets t' modulo p_2 equal to z modulo p_2. It sets the key as: $Kg_2^{\kappa' u'}T,\ K'g_2^{u'},\ K''g_2^z,\ K_i = (g_2^z)^{\eta_i}\ \forall i \in S$. We observe that if $T = g_2^{xyz}$, this is a properly distributed nominal semi-functional key, and when T is random in G_{p_2}, this is a properly distributed temporary semi-functional key.

To create the semi-functional challenge ciphertext for an $\ell \times n$ access matrix (A, ρ) and message M_b, $\mathcal{B}$ first runs the normal encryption algorithm to produce a normal ciphertext, $C_0, C, C', \{C_j, D_j\}_{j \in [\ell]}$. We note the attribute set S cannot satisfy the access policy of (A, ρ). As a result, $\mathcal{B}$ can efficiently find a vector $\tilde{w} \in \mathbb{Z}_N^n$ such that $\tilde{w} \cdot A_j = 0$ modulo N for all j such that $\rho(j) \in S$ and the first entry of $\tilde{w}$ is nonzero modulo each prime dividing N. Such a vector will exist as long as $(1, 0, \ldots, 0)$ is not in the span of $\{A_j\}_{\rho(j) \in S}$ modulo each of p_1, p_2, p_3. We may assume this holds with all but negligible probability, since we are assuming it is hard to find a non-trivial factor of N. This vector $\tilde{w}$ can be efficiently found by performing row reduction modulo N (we note that if one encounters a nonzero, non-invertible element of N during this process, then one has found a nontrivial factor of N). Once $\tilde{w}$ is found, its first entry can be randomized by multiplying the vector by a random value modulo N. Thus, we may assume the first entry of $\tilde{w}$ is random modulo p_2. We call this first entry s'.

$\mathcal{B}$ also chooses a random vector $w' \in \mathbb{Z}_N^n$ with first entry equal to 0. It will implicitly set the sharing vector w modulo p_2 so that $a'w = xy\tilde{w} + w'$ (i.e. $w = \tilde{w} + (xy)^{-1}w'$). We note that w is randomly distributed since the first entry of $\tilde{w}$ is random and the remaining entries of w' are random. $\mathcal{B}$ also chooses random values $\gamma_j \in \mathbb{Z}_N$ for each j such that $\rho(j) \in S$, and random values $\tilde{\gamma}_j \in \mathbb{Z}_N$ for each j such that $\rho(j) \notin S$. For these j's such that $\rho(j) \notin S$, it will implicitly set $\gamma_j = y\tilde{\eta}^{-1}_{\rho(j)} A_j \cdot \tilde{w} + \tilde{\gamma}_j$. We note that all of these values are properly distributed modulo p_2.

It forms the semi-functional ciphertext as:

$$C_0,\ Cg_2^{s'},\ C'g_2^{s'\kappa'},\ \ C_j g_2^{A_j \cdot w'} g_2^{-\eta_{\rho(j)}\gamma_j},\ D_j g_2^{\gamma_j}\ \forall j \text{ s.t. } \rho(j) \in S,$$

$$C_j g_2^{A_j \cdot w'} (g_2^x)^{-\tilde{\eta}_{\rho(j)}\tilde{\gamma}_j},\ D_j (g_2^y)^{\tilde{\eta}^{-1}_{\rho(j)} A_j \cdot \tilde{w}} g_2^{\tilde{\gamma}_j}\ \forall j \text{ s.t. } \rho(j) \notin S.$$

To see that this is a properly formed semi-functional ciphertext, note that for j such that $\rho(j) \notin S$:

$$a' A_j \cdot w - \eta_{\rho(j)}\gamma_j = A_j \cdot (xy\tilde{w} + w') - x\tilde{\eta}_{\rho(j)}(y\tilde{\eta}^{-1}_{\rho(j)} A_j \cdot \tilde{w} + \tilde{\gamma}_j) = A_j \cdot w' - x\tilde{\eta}_{\rho(j)}\tilde{\gamma}_j.$$

Here, $\mathcal{B}$ has embedded a y into the γ_j term and used the x embedded in the $\eta_{\rho(j)}$ term to cancel out the xy term in $a'A_j \cdot w$ that it cannot produce.

When $T = g_2^{xyz}$, $\mathcal{B}$ has properly simulated Game$_k^N$, and when T is random in G_{p_2}, $\mathcal{B}$ has properly simulated Game$_k^T$. Hence $\mathcal{B}$ can leverage $\mathcal{A}$'s non-negligible difference in advantage between these games to achieve a non-negligible advantage against the three party Diffie-Hellman assumption in a subgroup.

Lemma 4. *Under the source group q-parallel BDHE assumption in a subgroup (and assuming it is hard to find a non-trivial factor of N), no polynomial time attacker can achieve a non-negligible difference in advantage between Game$_k^N$ and Game$_k^T$ for a $k > Q_1$ using an access matrix (A, ρ) of size $\ell \times n$ where $\ell, n \leq q$.*

Proof. Given a PPT attacker $\mathcal{A}$ achieving a non-negligible difference in advantage between Game$_k^N$ and Game$_k^T$ for some k such that $Q_1 < k \leq Q$ using an access matrix with dimensions $\leq q$, we will create a PPT algorithm $\mathcal{B}$ to break the source group q-parallel BDHE assumption in a subgroup. Our $\mathcal{B}$ is given: $g_1, g_3, g_2, g_2^f, g_2^{df}, g_2^{c^i}\ \forall i \in [2q] \setminus \{q+1\}$, $g_2^{c^i/b_j}\ \forall i \in [2q] \setminus \{q+1\}, j \in [q]$, $g_2^{dfb_j}\ \forall j \in [q]$, $g_2^{dfc^i b_{j'}/b_j}\ \forall i \in [q], j, j' \in [q]$ such that $j \neq j'$, and T, where T is either equal to $g_2^{dc^{q+1}}$ or is a random element of G_{p_2}. $\mathcal{B}$ will simulate either Game$_k^N$ or Game$_k^T$ with $\mathcal{A}$, depending on T.

$\mathcal{B}$ chooses random exponents $\alpha, a, \kappa, \{h_i\} \in \mathbb{Z}_N$, and sets the public parameters as PP $= \{N,\ g = g_1,\ g^a = g_1^a,\ g^\kappa = g_1^\kappa,\ e(g_1, g_1)^\alpha,\ H_i = g_1^{h_i}\ \forall i\}$. It gives these to $\mathcal{A}$. We note that $\mathcal{B}$ knows the MSK, and hence can use the normal key generation algorithm to make normal keys in response to $\mathcal{A}$'s key requests from the $k+1$ request and onward. To make the first $k-1$ semi-functional keys, $\mathcal{B}$ can first make a normal key and then multiply the K by a random element of G_{p_2} (this can be obtained by raising g_2 to a random exponent modulo N).

Since we are assuming the k^{th} key query is a Phase II key query, $\mathcal{A}$ will request the challenge ciphertext for some $\ell \times n$ access matrix (A, ρ) *before* requesting the k^{th} key. This allows $\mathcal{B}$ to define the exponents $a', \kappa', \{\eta_i\}$ *after* seeing (A, ρ). $\mathcal{B}$ chooses random values $\tilde{\kappa}, \{\tilde{\eta}_i\} \in \mathbb{Z}_N$. It will implicitly set $a' = cd$ modulo p_2 and $\kappa' = d + \tilde{\kappa}$ modulo p_2. For each attribute i, we let J_i denote the set of indices j such that $\rho(j) = i$. $\mathcal{B}$ define $g_2^{\eta_i}$ as:

$$g_2^{\eta_i} = g_2^{\tilde{\eta}_i} \prod_{j \in J_i} \left(g_2^{c/b_j}\right)^{A_{j,1}} \cdot \left(g_2^{c^2/b_j}\right)^{A_{j,2}} \cdots \left(g_2^{c^n/b_j}\right)^{A_{j,n}}.$$

We note that all of these terms $g_2^{c/b_j}, \ldots, g_2^{c^n/b_j}$ are available to $\mathcal{B}$, since we are assuming $n, \ell \leq q$. We note that a' is uniformly random because d is random, κ' is randomized by $\tilde{\kappa}$, and each η_i is randomized by $\tilde{\eta}_i$.

To form the challenge ciphertext, $\mathcal{B}$ chooses random exponents $\{\tilde{\gamma}_j\} \in \mathbb{Z}_N$. It creates the normal components of the ciphertext as in the encryption algorithm. To create the semi-functional components (the parts in G_{p_2}), it implicitly sets $s' = f$ modulo p_2 and $\gamma_j = dfb_j + \tilde{\gamma}_j$ for each j from 1 to ℓ. We note that these values are properly distributed because $f, \tilde{\gamma}_j$ are random. It also chooses random values $y_2, \ldots, y_n \in \mathbb{Z}_N$ and implicitly sets the sharing vector w as:

$$w := (f, fc + y_2(a')^{-1}, \ldots, fc^{n-1} + y_n(a')^{-1}).$$

This is properly distributed as a random vector up to the constraint that the first entry is $s' = f$ (note that a' is nonzero with all but negligible probability).

For each j from 1 to ℓ, we observe that

$$a'A_j \cdot w - \eta_{\rho(j)}\gamma_j = df(cA_{j,1} + \ldots + c^n A_{j,n}) + y_2 A_{j,2} + \ldots + y_n A_{j,n} \quad (1)$$

$$-dfb_j \left(\sum_{j' \in J_{\rho(j)}} cA_{j',1}/b_{j'} + \ldots + c^n A_{j',n}/b_{j'} \right) \quad (2)$$

$$-dfb_j \tilde{\eta}_{\rho(j)} - \tilde{\gamma}_j \tilde{\eta}_{\rho(j)} - \tilde{\gamma}_j \left(\sum_{j' \in J_{\rho(j)}} cA_{j',1}/b_{j'} + \ldots + c^n A_{j',n}/b_{j'} \right) \quad (3)$$

Since $j \in J_{\rho(j)}$, the first quantity in (1) will be canceled by (2). What is left of (2) will be terms of the form $dfc^i b_j/b_{j'}$, where $i \leq n \leq q$ and $j \neq j'$. We note that $\mathcal{B}$ is given all of these in the exponent of g_2 in the assumption. $\mathcal{B}$ also has $g_2^{dfb_j}$ for all j from 1 to $q \geq \ell$ and $g_2^{c^i/b_{j'}}$ for all $j' \in J_{\rho(j)}, i \leq n \leq q$. Thus, $\mathcal{B}$ can form $g_2^{a'A_j\cdot w-\eta_{\rho(j)}\gamma_j}$ for each j. It can also compute $g_2^{s'}=g_2^f$, $g_2^{s'\kappa'}=g_2^{df}\left(g_2^f\right)^{\tilde{\kappa}}$, and $g_2^{\gamma_j}=g_2^{dfb_j}g_2^{\tilde{\gamma}_j}$. $\mathcal{B}$ multiplies these G_{p_2} components by the normal ciphertext to produce the semi-functional ciphertext, which it gives to $\mathcal{A}$.

Now, when $\mathcal{A}$ later requests the k^{th} key for some attribute set S not satisfying (A,ρ), $\mathcal{B}$ responds as follows. It first creates a normal key by calling the usual key generation algorithm. To create the semi-functional components, it first chooses a vector $\theta=(\theta_1,\ldots,\theta_n)\in\mathbb{Z}_N^n$ such that $\theta\cdot A_j=0$ modulo N for all j such that $\rho(j)\in S$ and the first entry of θ is nonzero modulo each prime dividing N. Such a vector will exist as long as $(1,0,\ldots,0)$ is not in the span of $\{A_j\}_{\rho(j)\in S}$ modulo each of p_1,p_2,p_3. As in the proof of the previous lemma, we may assume this holds with all but negligible probability and we note that such a θ can be efficiently computed.

$\mathcal{B}$ chooses a random value $\tilde{u}\in\mathbb{Z}_N$ and implicitly sets

$$u'=-\theta_2c^q-\theta_3c^{q-1}-\ldots-\theta_nc^{q-n+2}+f\tilde{u},$$

$$t'=\theta_1c^q+\theta_2c^{q-1}+\ldots+\theta_nc^{q-n+1}.$$

We note that these are random modulo p_2 because $\tilde{u}$ and θ_1 are random (and c,f are nonzero with all but negligible probability). We observe that $\mathcal{B}$ can now form $g_2^{u'}$ and $g_2^{t'}$ as follows:

$$g_2^{u'}=\left(g_2^{c^q}\right)^{-\theta_2}\left(g_2^{c^{q-1}}\right)^{-\theta_3}\ldots\left(g_2^{c^{q-n+2}}\right)^{-\theta_n}\left(g_2^f\right)^{\tilde{u}},$$

$$g_2^{t'}=\left(g_2^{c^q}\right)^{\theta_1}\left(g_2^{c^{q-1}}\right)^{\theta_2}\ldots\left(g_2^{c^{q-n+1}}\right)^{\theta_n}.$$

For each attribute $i\in S$, we recall that the vector θ is orthogonal to A_j for all rows j such that $\rho(j)=i$ (i.e. all $j\in J_i$). Thus, we observe:

$$t'\eta_i=t'\tilde{\eta}_i+\sum_{j\in J_i}\sum_{\substack{m_1,m_2=1\\m_1\neq m_2}}^{n}\theta_{m_1}A_{j,m_2}b_j^{-1}c^{q+1+m_2-m_1}.$$

Since $q+1+m_2-m_1$ is always in the set $[2q]\setminus\{q+1\}$, $\mathcal{B}$ can compute $g_2^{t'\eta_i}$ from the terms it is given in the assumption. We also have that

$$a't'+k'u'=\theta_1dc^{q+1}-\tilde{\kappa}\left(\theta_2c^q+\ldots+\theta_nc^{q-n+2}\right)+df\tilde{u}+f\tilde{\kappa}\tilde{u}.$$

Therefore, $\mathcal{B}$ creates the semi-functional term for key component K as:

$$T^{\theta_1}\left(g_2^{c^q}\right)^{-\tilde{\kappa}\theta_2}\ldots\left(g_2^{c^{q-n+2}}\right)^{-\tilde{\kappa}\theta_n}\left(g_2^{df}\right)^{\tilde{u}}\left(g_2^f\right)^{\tilde{\kappa}\tilde{u}}.$$

If $T=g_2^{dc^{q+1}}$, then this is a properly distributed nominal semi-functional key. If T is a random element of G_{p_2}, this is a properly distributed temporary semi-functional key. Hence, $\mathcal{B}$ has properly simulated either Game_k^N or Game_k^T, depending on T, and can therefore leverage $\mathcal{A}$'s non-negligible difference in advantage to break the source group q-parallel BDHE assumption in a subgroup.

Lemma 5. *Under the general subgroup decision assumption, no polynomial time attacker can achieve a non-negligible difference in advantage between* $Game_k^T$ *and* $Game_k$ *for any k from 1 to Q.*

Lemma 6. *Under Assumption 1, no polynomial attacker can achieve a non-negligible difference in advantage between* $Game_Q$ *and* $Game_{final}$.

The proofs of these last two lemmas can be found in the full version. This completes the proof of Theorem 1.

References

1. S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (h)ibe in the standard model. In *EUROCRYPT*, pages 553–572, 2010.
2. M. Raykova B. Parno and V. Vaikuntanathan. How to delegate and verify in public: Verifiable computation from attribute-based encryption. In *TCC*, pages 422–439, 2012.
3. A. Beimel. Secure schemes for secret sharing and key distribution. PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.
4. M. Bellare, B. Waters, and S. Yilek. Identity-based encryption secure against selective opening attack. In *TCC*, pages 235–252, 2011.
5. John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 321–334.
6. D. Boneh and X. Boyen. Efficient selective-id secure identity based encryption without random oracles. In *EUROCRYPT*, pages 223 – 238, 2004.
7. D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *CRYPTO*, pages 443–459, 2004.
8. D. Boneh, X. Boyen, and E. Goh. Hierarchical identity based encryption with constant size ciphertext. In *EUROCRYPT*, pages 440–456, 2005.
9. D. Boneh and M. Franklin. Identity based encryption from the weil pairing. In *CRYPTO*, pages 213–229, 2001.
10. D. Boneh, E. Goh, and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In *TCC*, pages 325–342, 2005.
11. R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. In *EUROCRYPT*, pages 255–271, 2003.
12. D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, pages 523–552, 2010.
13. L. Cheung and C. Newport. Provably secure ciphertext policy abe. In *CCS*, pages 456–465, 2007.
14. C. Cocks. An identity based encryption scheme based on quadratic residues. In *Proceedings of the 8th IMA International Conference on Cryptography and Coding*, pages 26–28, 2001.
15. C. Gentry. Practical identity-based encryption without random oracles. In *EUROCRYPT*, pages 445–464, 2006.
16. C. Gentry and S. Halevi. Hierarchical identity based encryption with polynomially many levels. In *TCC*, pages 437–456, 2009.
17. C. Gentry and A. Silverberg. Hierarchical id-based cryptography. In *ASIACRYPT*, pages 548–566, 2002.

18. V. Goyal, A. Jain, O. Pandey, and A. Sahai. Bounded ciphertext policy attribute-based encryption. In *ICALP*, 2008.
19. V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute based encryption for fine-grained access control of encrypted data. In *ACM conference on Computer and Communications Security*, pages 89–98, 2006.
20. J. Horwitz and B. Lynn. Toward hierarchical identity-based encryption. In *EUROCRYPT*, pages 466–481, 2002.
21. A. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In *EUROCRYPT*, pages 318–335, 2012.
22. A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *EUROCRYPT*, pages 62–91, 2010.
23. A. Lewko and B. Waters. New techniques for dual system encryption and fully secure hibe with short ciphertexts. In *TCC*, pages 455–479, 2010.
24. A. Lewko and B. Waters. Decentralizing attribute-based encryption. In *EUROCRYPT*, pages 568–588, 2011.
25. A. Lewko and B. Waters. Unbounded hibe and attribute-based encryption. In *EUROCRYPT*, pages 547–567, 2011.
26. T. Okamoto and K. Takashima. Homomorphic encryption and signatures from vector decomposition. In *Pairing*, pages 57–74, 2008.
27. T. Okamoto and K. Takashima. Hierarchical predicate encryption for inner-products. In *ASIACRYPT*, pages 214–231, 2009.
28. T. Okamoto and K. Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *CRYPTO*, pages 191–208, 2010.
29. R. Ostrovksy, A. Sahai, and B. Waters. Attribute based encryption with non-monotonic access structures. In *ACM conference on Computer and Communications Security*, pages 195–203, 2007.
30. A. Sahai and B. Waters. Fuzzy identity based encryption. In *EUROCRYPT*, pages 457–473, 2005.
31. A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.
32. B. Waters. Efficient identity-based ecnryption without random oracles. In *EUROCRYPT*, pages 114–127, 2005.
33. B. Waters. Dual system encryption: realizing fully secure ibe and hibe under simple assumptions. In *CRYPTO*, pages 619–636, 2009.
34. B. Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *PKC*, pages 53–70, 2011.