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

Allison Lewko *
University of Texas at Austin
alewko@cs.utexas.edu

Brent Waters †
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 [31], 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 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

*Supported by 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.

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. [31, 19, 36]) 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, 34]. 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 [31, 19, 36] 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 [34]. 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 [35]. 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 [23], 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 [23] 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[1]. This approach incurs a very significant loss in efficiency, and has been inherited by all fully secure schemes to date ([25, 29] 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, 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.

[1]For KP-ABE systems, it is the ciphertext size that will grow multiplicatively with M.

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 [36], 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 [35, 29]) and in composite order bilinear groups (e.g. in [24, 23]). 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. We then 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 [27, 28, 29], along with the relevant observations in [22]. The formal statements of our complexity assumptions for each setting can be found in Sections 2.2 and 5.1. 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.

1.1 Other Related Work

The roots of attribute-based encryption trace back to identity-based encryption (IBE), which was first conceived by Shamir [32] 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, 7, 8, 15, 16, 24, 1, 12, 26].

There have been several prior constructions of attribute-based encryption which have been shown to be selectively secure in the standard model [31, 19, 13, 30, 18, 36] 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).

1.2 Organization

In Section 2, we give the relevant background on CP-ABE systems and composite order bilinear groups, as well as formal statements of the complexity assumptions we rely on in the composite order setting. In Section 3, we present our CP-ABE system in composite order bilinear groups. In Section 4, we prove its full security. In Section 5, we give the relevant background for prime order bilinear groups, state our complexity assumptions in this context, present the prime order

variant of our CP-ABE construction, and prove its full security. In Appendix A, we provide a reduction between assumptions that we use in in the prime order setting. In Appendix B, we justify our q-based assumption in the generic group model.

2 Preliminaries

2.1 Composite Order Bilinear Groups

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 $N = p_1 p_2 p_3$ 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.

2.2 Complexity Assumptions

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 [24, 23], for example, and holds in the generic group model:

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

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

$$g_1 \xleftarrow{R} G_{p_1},\ g_2, X_2, Y_2 \xleftarrow{R} G_{p_2},\ g_3 \xleftarrow{R} G_{p_3}$$

$$\alpha, s \xleftarrow{R} \mathbb{Z}_N,$$

$$D = (\mathbb{G}, g_1, g_2, g_3, g_1^{\alpha} X_2, g_1^{s} Y_2),$$

$$T_0 = e(g_1, g_1)^{\alpha s},\ T_1 \xleftarrow{R} G_T.$$

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

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

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:

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

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 [36], 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 Appendix B, 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 $\mathcal{A}$.

2.3 Background for ABE

We now give required background material on access structures, the formal definition of a CP-ABE scheme, and the security definition we will use.

2.3.1 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., $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}} \setminus \{\}$. 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. As in the definition above over $\mathbb{Z}_p$, we say a set of attributes S is authorized if the rows of the access matrix A labeled by elements of S have the vector $(1, 0, \ldots, 0)$ in their span modulo N. However, in our security proof for our composite order system, we will further assume that for an unauthorized set, the corresponding rows of A do not include the vector $(1, 0, \ldots, 0)$ in their span modulo p_2. We may assume this because if an adversary can produce an access matrix A over $\mathbb{Z}_N$ and an unauthorized set over $\mathbb{Z}_N$ that is authorized over $\mathbb{Z}_{p_2}$, then this can be used to produce a non-trivial factor of the group order N, which would violate our general subgroup decision assumption.

2.3.2 CP-ABE Definition

A ciphertext-policy attribute-based encryption system consists of four algorithms: Setup, Encrypt, KeyGen, and Decrypt.

Setup$(\lambda, \mathcal{U}) \rightarrow (\text{PP}, \text{MSK})$ The setup algorithm takes in the security parameter λ and the attribute universe description $\mathcal{U}$. It outputs the public parameters PP and a master secret key MSK.

Encrypt$(\text{PP}, M, \mathbb{A}) \rightarrow \text{CT}$ The encryption algorithm takes in the public parameters PP, the message M, and an access structure $\mathbb{A}$ over the universe of attributes. It will output a ciphertext CT such that only users whose private keys satisfy the access structure $\mathbb{A}$ should be able to extract M. We assume that $\mathbb{A}$ is implicitly included in CT.

KeyGen$(\text{MSK}, \text{PP}, S) \rightarrow \text{SK}$ The key generation algorithm takes in the master secret key MSK, the public parameters PP, and a set of attributes S. It outputs a private key SK. We assume that S is implicitly included in SK.

Decrypt$(\text{PP}, \text{CT}, \text{SK}) \rightarrow M$ The decryption algorithm takes in the public parameters PP, a ciphertext CT, and a private key SK. If the set of attributes of the private key satisfies the access structure of the ciphertext, it outputs the message M.

2.3.3 Security Model for CP-ABE

We now give the full security definition for CP-ABE systems. This is described by a security game between a challenger and an attacker. The game proceeds as follows:

Setup The challenger runs the Setup algorithm and sends the public parameters PP to the attacker.

Phase 1 The attacker adaptively queries the challenger for private keys corresponding to sets of attributes $S_1, \ldots, S_{Q_1}$. Each time, the challenger responds with a secret key obtained by running $\text{KeyGen}(\text{MSK}, \text{PP}, S_k)$.

Challenge The attacker declares two equal length messages M_0 and M_1 and an access structure $\mathbb{A}$. This access structure cannot be satisfied by any of the queried attribute sets $S_1, \ldots, S_{Q_1}$. The challenger flips a random coin $b \in \{0, 1\}$, and encrypts M_b under $\mathbb{A}$, producing CT. It sends CT to the attacker.

Phase 2 The attacker adaptively queries the challenger for private keys corresponding to sets of attributes $S_{Q_1+1}, \ldots, S_Q$, with the added restriction that none of these satisfy $\mathbb{A}^*$. Each time, the challenger responds with a secret key obtained by running $\text{KeyGen}(\text{MSK}, \text{PP}, S_k)$.

Guess The attacker outputs a guess b' for b.

The advantage of an attacker is this game is defined to be $Pr[b = b'] - \frac{1}{2}$.

Definition 3. *A ciphertext-policy attribute-based encryption system is fully secure if all polynomial time attackers have at most a negligible advantage in this security game.*

Selective security is defined by adding an initialization phase where the attacker must declare $\mathbb{A}$ before seeing PP.

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 [36], 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.

Setup$(\lambda, \mathcal{U}) \to \text{PP}, \text{MSK}$ 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($\text{MSK}, S, \text{PP}) \to \text{SK}$ 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($(A, \rho), \text{PP}, M) \to \text{CT}$ 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):

$$C_0 = M e(g,g)^{\alpha s}, \;\; C = g^s, \;\; C' = (g^\kappa)^s,$$

$$C_j = (g^a)^{A_j \cdot v} H_{\rho(j)}^{-r_j}, \;\; D_j = g^{r_j} \;\; \forall j \in [\ell].$$

(The notation $[\ell]$ denotes the set $\{1, \ldots, \ell\}$.)

Decrypt($\text{CT}, \text{PP}, \text{SK}) \to M$ 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:

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

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,

$$e(C_j, K'') e(D_j, K_{\rho(j)}) = e(g,g)^{at A_j \cdot v},$$

so we have:

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

4 Security Proof

We now prove the following theorem:

Theorem 4. *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 2.2, our CP-ABE scheme defined in Section 3 is fully secure (in the sense of Definition 3).*

Our security proof is obtained via a hybrid argument over a sequence of games. We let Game_{real} denote the real security game as defined in Section 2.3.3. 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:

$$C_0,\ C g_2^{s'},\ C' g_2^{s' \kappa'},\ \{C_j g_2^{a' A_j \cdot w} g_2^{-\eta_{\rho(j)} \gamma_j},\ D_j g_2^{\gamma_j}\}.$$

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:

$$K g_2^{a't' + \kappa' u'},\ K' g_2^{u'},\ K'' g_2^{t'},\ K_i g_2^{t' \eta_i}\ \forall i \in S.$$

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 [36].

The original versions of these arguments in [19, 36] relied on assumptions very close to ours, with the main difference being that the assumptions in [19, 36] 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.

Lemma 5. *Under the general subgroup decision assumption, no polynomial time attacker can achieve a non-negligible difference in advantage between Game_{real} and Game_0.*

Proof. Given a PPT attacker $\mathcal{A}$ achieving a non-negligible difference in advantage between Game_{real} and Game_0, we will create a PPT algorithm $\mathcal{B}$ to break the general subgroup decision assumption with sets $Z_0 := \{1\}, Z_1 := \{1, 2\}, Z_2 := \{1\}, Z_3 := \{3\}$. Our $\mathcal{B}$ is given g_1, g_3, T, where g_1 is a generator of G_{p_1}, g_3 is a generator of G_{p_3}, and T is either a random element of G_{p_1} or a random element of $G_{p_1 p_2}$. $\mathcal{B}$ will simulate either Game_{real} or Game_0 with $\mathcal{A}$, depending on the nature of T.

$\mathcal{B}$ chooses random exponents $\alpha, a, \kappa, \{h_i\} \in \mathbb{Z}_N$ and sets the public parameters as:

$$\text{PP} = \{N,\ g = g_1,\ g^a = g_1^a,\ g^\kappa = g_1^\kappa,\ e(g,g)^\alpha = e(g_1,g_1)^\alpha,\ H_i = g_1^{h_i}\ \forall i\}.$$

It gives these to $\mathcal{A}$. We note that $\mathcal{B}$ knows the MSK. When $\mathcal{A}$ requests a secret key, $\mathcal{B}$ can call the normal key generation algorithm to create one.

At some point, $\mathcal{A}$ requests a challenge ciphertext for an access matrix (A, ρ) and messages M_0, M_1. We let $\ell \times n$ denote the dimensions of A. $\mathcal{B}$ chooses a random bit b and encrypts M_b as follows. It implicitly sets g^s equal to the G_{p_1} part of T. It chooses a random vector $\tilde{v} \in \mathbb{Z}_N^n$ with first entry equal to 1. It implicitly sets $v = s\tilde{v}$. It also chooses random exponents $\tilde{r}_j \in \mathbb{Z}_N$ for each j from 1 to ℓ. It implicitly sets $r_j = s\tilde{r}_j$. We note that the values of $s, v, r_j\ \forall j$ are properly distributed modulo p_1. The ciphertext is formed as:

$$C_0 = M_b e(g_1, T)^\alpha,\ C = T,\ C' = T^\kappa,$$

$$C_j = T^{aA_j \cdot \tilde{v}} T^{-\tilde{r}_j h_{\rho(j)}},\ D_j = T^{\tilde{r}_j}\ \forall j.$$

If $T \in G_{p_1}$, this is a properly distributed normal ciphertext, and $\mathcal{B}$ has properly simulated Game_{real} with $\mathcal{A}$. If $T \in G_{p_1 p_2}$, then this is a semi-functional ciphertext, with components in G_{p_2} set as: $g_2^{s'}$ is the G_{p_2} part of T, κ' is equal to the value of κ modulo p_2, a' is equal to the value of a modulo p_2, w is equal to $s'\tilde{v}$ modulo p_2, and for each j, $\eta_{\rho(j)}$ is equal to the value of $h_{\rho(j)}$ modulo p_2 and γ_j is equal to the value of $s'\tilde{r}_j$ modulo p_2. We note that these values are properly distributed, since the modulo p_1 and p_2 values of an element chosen uniformly at random modulo N are distributed as *independent* uniform random values by the Chinese Remainder Theorem. We also note that the public parameters only information-theoretically reveal the values of a, κ, h_i modulo p_1. Hence when $T \in G_{p_1 p_2}$, $\mathcal{B}$ has properly simulated Game_0 with $\mathcal{A}$. Thus, $\mathcal{B}$ can leverage the non-negligible difference in $\mathcal{A}$'s advantage between these games to achieve a non-negligible advantage against the general subgroup decision assumption. □

Lemma 6. *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*.*

Proof. Given a PPT attacker $\mathcal{A}$ achieving a non-negligible difference in advantage between Game_{k-1} and Game_k for some k between 1 and Q, we will create a PPT algorithm $\mathcal{B}$ to break the general subgroup decision assumption with sets $Z_0 := \{1,3\}, Z_1 := \{1,2,3\}, Z_2 := \{1\}, Z_3 := \{3\}, Z_4 := \{1,2\}, Z_5 := \{2,3\}$. Our $\mathcal{B}$ is given $g_1, g_3, X_1X_2, Y_2Y_3, T$, where g_1, X_1 are generators of G_{p_1}, g_3, Y_3 are generators of G_{p_3}, X_2 is a generator of G_{p_2}, and T is either a random element of $G_{p_1p_3}$ or a random element of $G_{p_1p_2p_3}$. $\mathcal{B}$ will simulate either Game_{k-1} or Game_k^N with $\mathcal{A}$, depending on the nature of T.

$\mathcal{B}$ 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 so can use the regular key generation algorithm to produce normal keys in response to $\mathcal{A}$'s later key queries.

To respond to $\mathcal{A}$'s first $k-1$ key queries, $\mathcal{B}$ produces semi-functional keys as follows. First it uses the regular key generation algorithm to produce a normal key $K, K', K'', \{K_i\}_{i\in S}$. Then it chooses a random exponent $\tau \in \mathbb{Z}_N$ and forms the semi-functional key as:

$$K(Y_2Y_3)^\tau,\ K',\ K'',\ \{K_i\}_{i\in S}.$$

We note that these semi-functional keys are properly distributed, as Y_2^τ is distributed as a uniformly random element of G_{p_2}.

To form the semi-functional challenge ciphertext for an $\ell \times n$ access matrix (A, ρ) and message M_b, $\mathcal{B}$ chooses random exponents $\tilde{r}_j \in \mathbb{Z}_N$ for all j from 1 to ℓ. It also chooses a random vector $\tilde{v} \in \mathbb{Z}_N$ with first entry equal to 1. It will implicitly set $g^s = X_1$, $v = s\tilde{v}$, and $g^{r_j} = X_1^{\tilde{r}_j}$. It computes the ciphertext as:

$$C_0 = M_b(g_1, X_1X_2)^\alpha,\ C = X_1X_2,\ C' = (X_1X_2)^\kappa,$$

$$C_j = (X_1X_2)^{aA_j\cdot\tilde{v}}(X_1X_2)^{-h_{\rho(j)}\tilde{r}_j},\ D_j = (X_1X_2)^{\tilde{r}_j}\ \forall j.$$

We note that this implicitly sets $g_2^{s'} = X_2$, $\kappa' = \kappa$ modulo p_2, $a' = a$ modulo p_2, $\eta_{\rho(j)} = h_{\rho(j)}$ modulo p_2 for all j, and $g_2^{\gamma_j} = X_2^{\tilde{r}_j}$. To see that this is a properly distributed semi-functional ciphertext, note that the values of a, κ, and $\{h_i\}$ modulo p_2 are *not revealed* by the public parameters.

To create the k^{th} requested key for an attribute set S, $\mathcal{B}$ chooses a random exponent $\tilde{u} \in \mathbb{Z}_N$ and random elements $R, R', R'', \{R_i\} \in G_{p_3}$. It sets:

$$K = g_1^\alpha T^a T^{\tilde{u}\kappa} R,\ K' = T^{\tilde{u}}R',\ K'' = TR'',\ K_i = T^{h_i}R_i\ \forall i \in S.$$

This implicitly sets g^t to be the G_{p_1} part of T, and g^u to be the G_{p_1} part of $T^{\tilde{u}}$. We note that these are properly distributed as (independently) random elements of G_{p_1}. Now, if $T \in G_{p_1p_3}$, this is a properly distributed normal key. If $T \in G_{p_1p_2p_3}$, this is a properly distributed nominal semi-functional key, with the values $a' = a$ modulo p_2, $\kappa' = \kappa$ modulo p_2, and each η_i equal to the value of h_i modulo p_2 as in the semi-functional ciphertext.

Thus, when $T \in G_{p_1p_3}$, $\mathcal{B}$ has properly simulated Game_{k-1}, and when $T \in G_{p_1p_2p_3}$, $\mathcal{B}$ has properly simulated Game_k^N. Hence $\mathcal{B}$ can leverage $\mathcal{A}$'s non-negligible difference in advantage to achieve a non-negligible advantage against the general subgroup decision assumption.

□

Lemma 7. *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:

$$K g_2^{\kappa' u'} T,\ K' g_2^{u'},\ K'' g_2^z,\ K_i = (g_2^z)^{\eta_i}\ \forall i \in S.$$

We observe that if $T = g_2^{xyz}$, this will be a properly distributed nominal semi-functional key, and when T is random in G_{p_2}, this will be 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 $\gamma_j = y\tilde{\eta}_{\rho(j)}^{-1} A_j \cdot \tilde{w} + \tilde{\gamma}_j$. We note that all of these values are properly distributed modulo p_2.

It forms the semi-functional ciphertext as:

$$C_0,\ C g_2^{s'},\ C' g_2^{s'\kappa'},$$

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

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

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

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

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 8. *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: $g_1, g_3, g_2, g_2^f, g_2^{df}, g_2^{c^i}\ \forall i \in [2q]\backslash\{q+1\}$, $g_2^{c^i/b_j}\ \forall i \in [2q]\backslash\{q+1\}, j \in [q]$, $g_2^{dfb_j}\ \forall j \in [q]$, $g_2^{dfc^i b_{j'}/b_j}\ \forall i \in [q], j, j' \in [q]$ 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

$$\text{PP} = \{N,\ g = g_1,\ g^a = g_1^a,\ g^\kappa = g_1^\kappa,\ e(g_1, g_1)^\alpha,\ H_i = g_1^{h_i}\ \forall i\}.$$

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 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

$$\begin{aligned} a' A_j \cdot w - \eta_{\rho(j)} \gamma_j &= dfc(A_{j,1} + \ldots + c^n A_{j,n}) + y_2 A_{j,2} + \ldots + y_n A_{j,n} && (1) \\ &\quad - dfb_j \left(\sum_{j' \in J_{\rho(j)}} cA_{j',1}/b_{j'} + \ldots + c^n A_{j',n}/b_{j'} \right) && (2) \\ &\quad - dfb_j \tilde{\eta}_{\rho(j)} - \tilde{\gamma}_j \tilde{\eta}_{\rho(j)} - \tilde{\gamma}_j \left(\sum_{j' \in J_{\rho(j)}} cA_{j',1}/b_{j'} + \ldots + c^n A_{j',n}/b_{j'} \right) && (3) \end{aligned}$$

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

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

$$t' = \theta_1 c^q + \theta_2 c^{q-1} + \ldots + \theta_n c^{q-n+1}.$$

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

$$a't' + k'u' = \theta_1 d c^{q+1} - \tilde{\kappa}\left(\theta_2 c^q + \ldots + \theta_n c^{q-n+2}\right) + df\tilde{u} + f\tilde{\kappa}\tilde{u}.$$

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 9. *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.*

Proof. This is nearly identical to the proof of Lemma 6, except that $\mathcal{B}$ uses Y_2Y_3 to place a random G_{p_2} component on the K part of the k^{th} key to make it a semi-functional key in the case that T has no G_{p_2} component.

□

Lemma 10. *Under Assumption 1, no polynomial attacker can achieve a non-negligible difference in advantage between Game_Q and Game_{final}.*

Proof. Given a PPT attacker $\mathcal{A}$ achieving a non-negligible difference in advantage between Game_Q and Game_{final}, we will create a PPT algorithm $\mathcal{B}$ to break Assumption 1. $\mathcal{B}$ is given $g_1, g_2, g_3, g_1^{\alpha} X_2, g_1^{s} Y_2, T$, where T is either $e(g_1, g_1)^{\alpha s}$ or a random element of G_T. $\mathcal{B}$ will simulate either Game_Q or Game_{final} with $\mathcal{A}$ depending on the nature of T.

$\mathcal{B}$ chooses random exponents $a, \kappa, \{h_i\} \in \mathbb{Z}_N$ and sets the public parameters as:

$$\text{PP} = \{N,\ g = g_1,\ g^a = g_1^a,\ g^{\kappa} = g_1^{\kappa},\ e(g,g)^{\alpha} = e(g_1^{\alpha} X_2, g_1),\ H_i = g_1^{h_i}\ \forall i\}.$$

It gives these to $\mathcal{A}$.

When $\mathcal{A}$ requests a key for an attribute set S, $\mathcal{B}$ creates a semi-functional key as follows. It chooses random exponents $t, u, \gamma \in \mathbb{Z}_N$ and samples random elements $R, R', R'', \{R_i\} \in G_{p_3}$ (this can be done by raising g_3 to random exponents modulo N). The key is formed as:

$$K = (g_1^{\alpha} X_2) g_1^{at} g_1^{\kappa u} R g_2^{\gamma},\ K' = g_1^{u} R',\ K'' = g_1^{t} R'',\ K_i = g_1^{h_i t}\ \forall i \in S.$$

We note that this is a properly distributed semi-functional key.

To produce the semi-functional ciphertext for some $\ell \times n$ access matrix (A, ρ), $\mathcal{B}$ chooses random exponents $\tilde{r}_j \in \mathbb{Z}_N$ for each j from 1 to ℓ and a random vector $\tilde{v}$ with first entry equal

to 1. It implicitly sets $v = s\tilde{v}$ and $r_j = s\tilde{r}_j$. We note that these values are properly distributed. It forms the ciphertext as:

$$C_0 = M_b T,\ C = g_1^s Y_2,\ C' = (g_1^s Y_2)^\kappa,$$

$$C_j = (g_1^s Y_2)^{aA_j \cdot \tilde{v} - h_{\rho(j)} \tilde{r}_j},\ D_j = (g_1^s Y_2)^{\tilde{r}_j}.$$

We note that this is a semi-functional ciphertext with $g_2^{s'}$ equal to Y_2, κ' equal to the value of κ modulo p_2, w equal to $s'\tilde{v}$ modulo p_2, $g_2^{\gamma_j} = Y_2^{\tilde{r}_j}$, and $\eta_{\rho(j)}$ equal to the value of $h_{\rho(j)}$ modulo p_2 for each j. We note that these values are all properly distributed because Y_2 is a random element of G_{p_2}, and the values of $\kappa, h_i, \tilde{r}_j, \tilde{v}$ modulo p_2 are distributed independently of their values modulo p_1. Therefore, if $T = e(g_1, g_1)^{\alpha s}$, this is a properly distributed semi-functional encryption of M_b, and $\mathcal{B}$ has properly simulated Game_q. If T is a random element of G_T, then this is a properly distributed semi-functional encryption of a random message, and $\mathcal{B}$ has properly simulated Game_{final}. Hence $\mathcal{B}$ can leverage $\mathcal{A}$'s non-negligible difference in advantage to achieve a non-negligible advantage against Assumption 1. □

This completes our proof of Theorem 4. We note that it is not necessary to include explicitly in the statement of the theorem that we are assuming it is hard to find a non-trivial factor of N, since this is implied by the general subgroup decision assumption.

5 Our System in Prime Order Groups

In this section, we will present a prime order analog of our composite order result. This is obtained by combining our composite order construction and proof with the translation techniques developed in [22]. We first present the necessary background and state our complexity assumptions in the prime order setting.

5.1 Background and Complexity Assumptions

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 (p, G, G_T, e), where p is a prime, G and G_T are cyclic groups of order p, and $e : G^2 \to G_T$ is a map such that:

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

We will 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.

Our complexity assumptions in the prime order setting will be nearly identical to our assumptions in the composite order setting, except that the general subgroup decision assumption and the assumption used for the final game will be replaced by the decisional linear assumption.

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

$$\mathbb{G} := (p, G, G_T, e) \xleftarrow{R} \mathcal{G},$$

$$g, f, v \xleftarrow{R} G,\ c_1, c_2 \xleftarrow{R} \mathbb{Z}_p,$$

$$D := (\mathbb{G}, g, f, v, f^{c_1}, v^{c_2}),$$

$$T_0 = g^{c_1+c_2},\ T_1 \xleftarrow{R} G.$$

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

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

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

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

$$\mathbb{G} = (p, G, G_T, e) \xleftarrow{R} \mathcal{G},$$
$$g \xleftarrow{R} G,\ x, y, z \xleftarrow{R} \mathbb{Z}_p,$$
$$D = (\mathbb{G}, g, g^x, g^y, g^z),$$
$$T_0 = g^{xyz},\ T_1 \xleftarrow{R} G.$$

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

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

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

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

$$\mathbb{G} = (p, G, G_T, e) \xleftarrow{R} \mathcal{G},$$
$$g \xleftarrow{R} G,\ c, d, f, b_1, \ldots, b_q \xleftarrow{R} \mathbb{Z}_p,$$

The adversary will be given:

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

We additionally define

$$T_0 = g^{dc^{q+1}},\ T_1 \xleftarrow{R} G.$$

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 $\mathcal{A}$.

Dual Pairing Vector Spaces In order to simulate an analog of the orthogonal subgroups present in composite order groups, our construction will use dual pairing vector spaces, a tool introduced by Okamoto and Takashima [27, 28, 29]. We will replace single group elements with tuples of group elements, denoted by $g^{\vec{v}}$, where $\vec{v} = (v_1, \ldots, v_n)$ is a vector over $\mathbb{Z}_p$. This notation should be interpreted as:

$$g^{\vec{v}} := (g^{v_1}, \ldots, g^{v_n}).$$

When we write something like "raise $g^{\vec{v}}$ to the power c" for a scalar $c \in \mathbb{Z}_p$, we mean raising each component to the power c, i.e.:

$$\left(g^{\vec{v}}\right)^c := g^{c\vec{v}} = (g^{cv_1}, \ldots, g^{cv_n}).$$

We define a bilinear map e_n on n-tuples of G by pairing componentwise and multiplying the results in G_T:

$$e_n(g^{\vec{v}}, g^{\vec{w}}) := \prod_{i=1}^{n} e(g^{v_i}, g^{w_i}) = e(g, g)^{\vec{v} \cdot \vec{w}},$$

where the dot product is computed modulo p.

For a fixed (constant) dimension n, we say two bases $\mathbb{B} := (\vec{b}_1, \ldots, \vec{b}_n)$ and $\mathbb{B}^* := (\vec{b}^*_1, \ldots, \vec{b}^*_n)$ of $\mathbb{Z}_p^n$ are "dual orthonormal" when:

$$\vec{b}_i \cdot \vec{b}^*_j = 0 \pmod{p},$$

whenever $i \neq j$, and

$$\vec{b}_i \cdot \vec{b}^*_i = \psi$$

for all i, where ψ is a nonzero element of $\mathbb{Z}_p$. (This is a slight abuse of the terminology "orthonormal", since ψ is not constrained to be 1.) For a generator $g \in G$, we note that

$$e_n(g^{\vec{b}_i}, g^{\vec{b}^*_j}) = 1$$

whenever $i \neq j$, where 1 here denotes the identity element in G_T.

We let $Dual(\mathbb{Z}_p^n, \psi)$ denote the set of pairs of dual orthonormal bases of dimension n with dot products $\vec{b}_i \cdot \vec{b}^*_i = \psi$. We let $(\mathbb{B}, \mathbb{B}^*) \xleftarrow{R} Dual(\mathbb{Z}_p^n, \psi)$ denote choosing a random pair of bases from this set.

Dual pairing vector spaces provide a workable analog to the prime order subgroups present in composite order groups, since they come equipped with orthogonal subspaces under the pairing e_n. The notion of a subgroup can now be replaced by a subspace in the exponent, particularly a span of a subset of the basis vectors in a pair of dual orthonormal bases.

We will use a lemma noted in [22] which roughly states that if one starts by sampling a random pair of dual orthonormal bases and then applies a linear change of basis to a subset of the basis vectors (maintaining the orthonormal properties), the resulting bases are also distributed as a random pair, independent of the change of basis that was applied. More formally, we let $(\mathbb{B}, \mathbb{B}^*)$ denote a pair of dual orthonormal bases over $\mathbb{Z}_p^n$, and we let $A \in \mathbb{Z}_p^{m \times m}$ be an invertible matrix for some $m \leq n$. We let $S_m \subseteq [n]$ be a subset of size m. We then define new dual orthonormal bases $\mathbb{B}_A, \mathbb{B}^*_A$ as follows. We let B_m denote the $n \times m$ matrix over $\mathbb{Z}_p$ whose columns are the vectors $\vec{b}_i \in \mathbb{B}$ such that $i \in S_m$. Then $B_m A$ is also an $n \times m$ matrix. We form $\mathbb{B}_A$ by retaining all of the vectors $\vec{b}_i \in \mathbb{B}$ for $i \notin S_m$ and exchanging the $\vec{b}_i$ for $i \in S_m$ with the columns of $B_m A$. To define $\mathbb{B}^*_A$, we similarly let B^*_m denote the $n \times m$ matrix over $\mathbb{Z}_p$ whose columns are the vectors $\vec{b}^*_i \in \mathbb{B}^*$ such that $i \in S_m$. Then $B^*_m (A^{-1})^t$ is also an $n \times m$ matrix, where $(A^{-1})^t$ denotes the transpose of A^{-1}. We form $\mathbb{B}^*_A$ by retaining all of the vectors $\vec{b}^*_i \in \mathbb{B}^*$ for $i \notin S_m$ and exchanging the $\vec{b}_i$ for $i \in S_m$ with the columns of $B^*_m (A^{-1})^t$. We have:

Lemma 11. *For any fixed positive integers $m \leq n$, any fixed invertible $A \in \mathbb{Z}_p^{m \times m}$ and set $S_m \subseteq [n]$ of size m, if $(\mathbb{B}, \mathbb{B}^*) \xleftarrow{R} Dual(\mathbb{Z}_p^n, \psi)$, then $(\mathbb{B}_A, \mathbb{B}_A^*)$ is also distributed as a random sample from $Dual(\mathbb{Z}_p^n, \psi)$. In particular, the distribution of $(\mathbb{B}_A, \mathbb{B}_A^*)$ is independent of A.*

This follows simply from noting that one can recover $\mathbb{B}, \mathbb{B}^*$ uniquely from $\mathbb{B}_A, \mathbb{B}_A^*$ (with A fixed).

In [22], the "Subspace Assumption" is introduced as a prime order substitute of the general subgroup decision assumption in composite order groups. It is based on the observation that if one a given $g^{\vec{v}}$ say, then one cannot tell if $\vec{v}$ is in the span of $\vec{b}_1^*, \vec{b}_2^*$ or the larger span of $\vec{b}_1^*, \vec{b}_2^*, \vec{b}_3^*$ when one is *not* given $g^{\vec{b}_3}$ (though one can be given $g^{\vec{w}}$ for $\vec{w}$ in the span of $\vec{b}_1, \vec{b}_2, \vec{b}_3$, for example). The subspace assumption is implied by the decisional linear assumption and helps clarify how DLIN allows one to expand/contract spaces in the exponent in the dual pairing vector space framework, similar to the effect of the general subgroup decision assumption in the composite order setting.

The statement of the subspace assumption in [22] involves one dual orthonormal basis pair of dimension n, and is also parameterized by a positive integer $k \leq \frac{n}{3}$ which controls how many 2-dimensional subspaces of the total n-dimensional space are expanding to 3-dimensional subspaces. It is remarked that one could additionally generalize the assumption to involve multiple bases - we will use such a generalization here, and we let the parameter m denote the number of bases. Each basis pair has its own dimension n_i and its own parameter k_i. For completeness, we include a proof that our statement of the subspace assumption here is implied by DLIN in Appendix A, though the proof is essentially the same as given in [22]. We note that this reduction holds for *any* valid choices of the parameters m, n_i, k_i. For the simpler statement of the subspace assumption without the clutter of multiple bases pairs, see [22].

Our m dual orthonormal bases pairs will be denoted by $(\mathbb{B}_1, \mathbb{B}_1^*), \ldots, (\mathbb{B}_m, \mathbb{B}_m^*)$. For each i from 1 to m, the basis vectors comprising $(\mathbb{B}_i, \mathbb{B}_i^*)$ will be denoted by $\vec{b}_{1,i}, \ldots, \vec{b}_{n_i,i}$ and $\vec{b}_{1,i}^*, \ldots, \vec{b}_{n_i,i}^*$ respectively.

Definition 12. *(The Subspace Assumption) Given a group generator $\mathcal{G}$, we define the following distribution:*

$$\mathbb{G} := (p, G, G_T, e) \xleftarrow{R} \mathcal{G},\ g \xleftarrow{R} G,\ \psi, \eta, \beta, \tau_1, \tau_2, \tau_3, \mu_1, \mu_2, \mu_3 \xleftarrow{R} \mathbb{Z}_p,$$

$$(\mathbb{B}_1, \mathbb{B}_1^*) \xleftarrow{R} Dual(\mathbb{Z}_p^{n_1}, \psi), \ldots, (\mathbb{B}_m, \mathbb{B}_m^*) \xleftarrow{R} Dual(\mathbb{Z}_p^{n_m}, \psi),$$

$$U_{1,i} := g^{\mu_1 \vec{b}_{1,i} + \mu_2 \vec{b}_{k_i+1,i} + \mu_3 \vec{b}_{2k_i+1,i}},\ U_{2,i} := g^{\mu_1 \vec{b}_{2,i} + \mu_2 \vec{b}_{k_i+2,i} + \mu_3 \vec{b}_{2k_i+2,i}},$$

$$\ldots,\ U_{k_i,i} := g^{\mu_1 \vec{b}_{k_i,i} + \mu_2 \vec{b}_{2k_i,i} + \mu_3 \vec{b}_{3k_i,i}}\ \forall i \in [m],$$

$$V_{1,i} := g^{\tau_1 \eta \vec{b}_{1,i}^* + \tau_2 \beta \vec{b}_{k_i+1,i}^*},\ V_{2,i} := g^{\tau_1 \eta \vec{b}_{2,i}^* + \tau_2 \beta \vec{b}_{k_i+2,i}^*},\ \ldots,\ V_{k_i,i} := g^{\tau_1 \eta \vec{b}_{k_i,i}^* + \tau_2 \beta \vec{b}_{2k_i,i}^*}\ \forall i \in [m],$$

$$W_{1,i} := g^{\tau_1 \eta \vec{b}_{1,i}^* + \tau_2 \beta \vec{b}_{k_i+1,i}^* + \tau_3 \vec{b}_{2k_i+1,i}^*},\ W_{2,i} := g^{\tau_1 \eta \vec{b}_{2,i}^* + \tau_2 \beta \vec{b}_{k_i+2,i}^* + \tau_3 \vec{b}_{2k_i+2,i}^*},$$

$$\ldots,\ W_{k_i,i} := g^{\tau_1 \eta \vec{b}_{k_i,i}^* + \tau_2 \beta \vec{b}_{2k_i,i}^* + \tau_3 \vec{b}_{3k_i,i}^*}\ \forall i \in [m],$$

$$D := \big(\mathbb{G}, g, \{g^{\vec{b}_{1,i}}, g^{\vec{b}_{2,i}}, \ldots, g^{\vec{b}_{2k_i,i}}, g^{\vec{b}_{3k_i+1,i}}, \ldots, g^{\vec{b}_{n_i,i}}, g^{\eta \vec{b}_{1,i}^*}, \ldots, g^{\eta \vec{b}_{k_i,i}^*}, g^{\beta \vec{b}_{k_i+1,i}^*},$$
$$\ldots, g^{\beta \vec{b}_{2k_i,i}^*}, g^{\vec{b}_{2k_i+1,i}^*}, \ldots, g^{\vec{b}_{n_i,i}^*}, U_{1,i}, U_{2,i}, \ldots, U_{k_i,i}\}_{i=1}^m,\ \mu_3\big).$$

We assume that for any PPT algorithm $\mathcal{A}$ (with output in $\{0,1\}$),

$$Adv_{\mathcal{G},\mathcal{A}} := \left| \mathbb{P}\left[\mathcal{A}\left(D, \{V_{1,i}, \ldots, V_{k_i,i}\}_{i=1}^m\right) = 1\right] - \mathbb{P}\left[\mathcal{A}\left(D, \{W_{1,i}, \ldots, W_{k_i,i}\}_{i=1}^m\right) = 1\right] \right|$$

is negligible in the security parameter λ.

To help the reader identity the important features of this assumption, we include heuristic illustrations of the $m = 1$, $n = 3$, $k = 1$ and $m = 1$, $n = 6$, $k = 2$ cases below, reproduced from [22] (with permission).

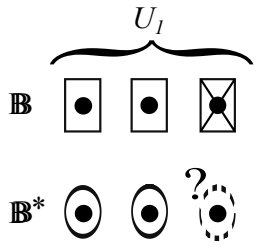

Figure 1: Subspace Assumption with $m = 1, n = 3, k = 1$

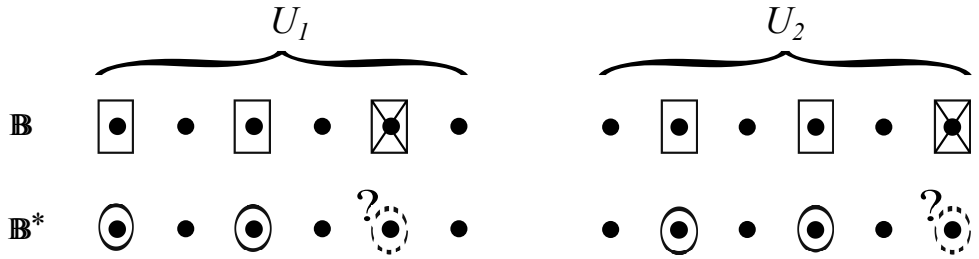

Figure 2: Subspace Assumption with $m = 1, n = 6, k = 2$

In these diagrams, the top rows illustrate the U terms, while the bottom rows illustrate the V, W terms. The solid ovals and rectangles indicate the presence of basis vectors. The crossed rectangles indicate basis elements of $\mathbb{B}$ that are present in U_1, U_2 but are not given out in isolation. The dotted ovals with question marks indicate the basis vectors whose presence depends on whether we consider the V's or the W's.

5.2 Construction

Our prime order construction very closely resembles our composite order construction, replacing the subgroups in the composite order setting with dual pairing vector spaces. The basis vectors which do not appear in the exponents of the construction are reserved as the semi-functional space. We note that there is no need to simulate the additional randomness provided by the G_{p_3} subgroup in the composite order setting. This is due to the asymmetry of the dual pairing vector spaces, since there are different bases for the ciphertexts and keys. In our security proof, knowledge of the full basis on the key side will allow the simulator to create other properly semi-functional keys while a single key is being isolated to change from normal to semi-functional. In most other respects, our security proof will be very closely analogous to the proof for our composite order system, with the subspace assumption playing the role of the general subgroup decision assumption.

Since the subspace assumption allows us to expand 2-dimensional spaces to 3-dimensional, we duplicate random exponents in our composite order construction in order to start with 2-dimensional random objects in the normal space. Our construction is influenced by taste and convenience - there are alternate choices one could make in translating our composite order scheme into the dual pairing vector space setting, but we have chosen to work with a 1-dimensional semi-functional space on some terms and a 2-dimensional semi-functional space on others. For the terms with a 2-dimensional semi-functional space, we have expanded the normal

space to be 4-dimensional. This allows us to expand simultaneously into both dimensions of the semi-functional space by a single application of the subspace assumption (with n_i's equal to 6 and k_i's equal to 2).

We assume that messages to be encrypted are elements of the target group G_T. We also conflate notation and consider the attribute universe to be $[\mathcal{U}] = \{1, 2, \ldots, \mathcal{U}\}$, so $\mathcal{U}$ serves both as a description of the attribute universe and as a count of the total number of attributes.

Setup$(\lambda, \mathcal{U}) \to \text{PP}, \text{MSK}$ The setup algorithm chooses a bilinear group G of prime order p and a generator g. It randomly chooses two pairs of dual orthonormal bases $(\mathbb{B}, \mathbb{B}^*), (\mathbb{B}_0, \mathbb{B}_0^*)$ of dimension 3 and $\mathcal{U}$ pairs of dual orthonormal bases $(\mathbb{B}_1, \mathbb{B}_1^*), \ldots, (\mathbb{B}_\mathcal{U}, \mathbb{B}_\mathcal{U}^*)$ of dimension 6, subject to the constraint that all of these share the same value of ψ. We let $\vec{b}_i, \vec{b}_i^*$ denote the basis vectors belonging to $(\mathbb{B}, \mathbb{B}^*)$, and $\vec{b}_{i,j}, \vec{b}_{i,j}^*$ denote the basis vectors belong to $(\mathbb{B}_j, \mathbb{B}_j^*)$ for each j from 0 to $\mathcal{U}$. The setup algorithm also chooses two random exponents $\alpha_1, \alpha_2 \in \mathbb{Z}_p$. The public parameters consist of:

$$\text{PP} := \{G, p, g^{\vec{b}_1}, g^{\vec{b}_2}, g^{\vec{b}_{1,0}}, g^{\vec{b}_{2,0}}, g^{\vec{b}_{1,i}}, \ldots, g^{\vec{b}_{4,i}} \ \forall i \in [\mathcal{U}],\ e(g,g)^{\alpha_1 \psi}, e(g,g)^{\alpha_2 \psi}\}.$$

The master secret key additionally contains:

$$\text{MSK} := \{g^{\alpha_1 \vec{b}_1^*}, g^{\alpha_2 \vec{b}_2^*}, g^{\vec{b}_1^*}, g^{\vec{b}_2^*}, g^{\vec{b}_{1,0}^*}, g^{\vec{b}_{2,0}^*}, g^{\vec{b}_{1,i}^*}, \ldots, g^{\vec{b}_{4,i}^*} \ \forall i \in [\mathcal{U}]\ \}.$$

KeyGen$(\text{MSK}, S, \text{PP}) \to \text{SK}$ The key generation algorithm chooses random exponents t_1, t_2, u_1, $u_2 \in \mathbb{Z}_p$ and computes:

$$K := g^{(\alpha_1 + t_1 + u_1)\vec{b}_1^* + (\alpha_2 + t_2 + u_2)\vec{b}_2^*},$$

$$K_0 := g^{u_1 \vec{b}_{1,0}^* + u_2 \vec{b}_{2,0}^*},$$

$$K_i := g^{t_1 \vec{b}_{1,i}^* + t_1 \vec{b}_{2,i}^* + t_2 \vec{b}_{3,i}^* + t_2 \vec{b}_{4,i}^*} \ \forall i \in S.$$

The secret key is

$$\text{SK} := \{S, K,\ K_0,\ \{K_i\}_{i \in S}\}.$$

Encrypt$((A, \rho), \text{PP}, M) \to \text{CT}$ We assume $M \in G_T$. For an $\ell \times n$ access matrix A, the encryption algorithm chooses random exponents $s_1, s_2, \{r_j^1, r_j^2\}_{j=1}^\ell \in \mathbb{Z}_p$. It also chooses random vectors $v_1, v_2 \in \mathbb{Z}_p^n$ with first entries equal to s_1 and s_2 respectively. The ciphertext is formed as (it additionally includes (A, ρ)):

$$M' := M e(g,g)^{\alpha_1 s_1 \psi} e(g,g)^{\alpha_2 s_2 \psi},\ C := g^{s_1 \vec{b}_1 + s_2 \vec{b}_2},$$

$$C_0 := g^{s_1 \vec{b}_{1,0} + s_2 \vec{b}_{2,0}},$$

$$C_j := g^{(A_j \cdot v_1 + r_j^1)\vec{b}_{1,\rho(j)} - r_j^1 \vec{b}_{2,\rho(j)} + (A_j \cdot v_2 + r_j^2)\vec{b}_{3,\rho(j)} - r_j^2 \vec{b}_{4,\rho(j)}} \ \forall j = 1, \ldots, \ell.$$

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

$$X := \prod_{\rho(j) \in S} e_6(C_j, K_{\rho(j)})^{\omega_j}$$

It then computes:

$$Y := e_3(K, C) / e_6(K_0, C_0).$$

The message is recovered as:

$$M = M' X / Y.$$

Correctness We observe that for each j,

$$e_6(C_j, K_{\rho(j)}) = e(g,g)^{(t_1 A_j \cdot v_1 + t_2 A_j \cdot v_2)\psi}.$$

Thus,

$$X := \prod_{\rho(j) \in S} e_6(C_j, K_{\rho(j)})^{\omega_j} = e(g,g)^{\psi(t_1 \sum_{\rho(j)\in S} \omega_j A_j \cdot v_1 + t_2 \sum_{\rho(j)\in S} \omega_j A_j \cdot v_2)} = e(g,g)^{\psi(t_1 s_1 + t_2 s_2)}.$$

We note that

$$Y = e(g,g)^{\psi(s_1(\alpha_1+t_1+u_1)+s_2(\alpha_2+t_2+u_2))} / e(g,g)^{\psi(s_1 u_1 + s_2 u_2)} = e(g,g)^{\psi(s_1(\alpha_1+t_1)+s_2(\alpha_2+t_2))},$$

and therefore:

$$M'X/Y = Me(g,g)^{\psi(s_1\alpha_1+s_2\alpha_2)} e(g,g)^{\psi(t_1 s_1 + t_2 s_2)} / e(g,g)^{\psi(s_1(\alpha_1+t_1)+s_2(\alpha_2+t_2))} = M.$$

5.3 Security Proof

We now prove:

Theorem 13. *Under the decisional linear assumption, the three party Diffie-Hellman assumption, and the source group q-parallel BDHE assumption defined in Section 5.1, our CP-ABE scheme defined in Section 5.2 is fully secure (in the sense of Definition 3).*

Our security proof here is quite similar to the proof for our composite order scheme. We begin by defining our various types of semi-functional keys and ciphertexts. The semi-functional space in the exponent will correspond to the span of $\vec{b}_3, \vec{b}^*_3$, the span of $\vec{b}_{3,0}, \vec{b}^*_{3,0}$, and the span of each $\vec{b}_{5,j}, \vec{b}_{6,j}, \vec{b}^*_{5,j}, \vec{b}^*_{6,j}$.

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 $K, K_0, \{K_i\}_{i\in S}$. One then chooses a random value $\gamma \in \mathbb{Z}_p$ and multiplies K by $g^{\gamma \vec{b}^*_3}$. The other components of the key remain unchanged.

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 $M', C, C_0, \{C_j\}$. One then chooses random values $s_3, \{r^3_j\} \in \mathbb{Z}_p$ and a random vector $v_3 \in \mathbb{Z}^n_p$ with first entry equal to s_3. The semi-functional ciphertext is:

$$M',\ Cg^{s_3 \vec{b}_3},\ C_0 g^{s_3 \vec{b}_{3,0}},\ C_j g^{(A_j \cdot v_3 + r^3_j)\vec{b}_{5,\rho(j)} - r^3_j \vec{b}_{6,\rho(j)}} \quad \forall j = 1, \ldots, \ell.$$

Nominal Semi-functional Keys 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 $K, K_0, \{K_i\}_{i\in S}$. One then chooses random values $t_3, u_3 \in \mathbb{Z}_p$. The nominal semi-functional key is:

$$Kg^{(t_3+u_3)\vec{b}^*_3},\ K_0 g^{u_3 \vec{b}^*_{3,0}},\ K_i g^{t_3 \vec{b}^*_{5,i} + t_3 \vec{b}^*_{6,i}} \quad \forall i \in S.$$

We note that a nominal semi-functional key still correctly decrypts a semi-functional ciphertext.

Temporary Semi-functional Keys A temporary semi-functional key is similar to a nominal key, except that the semi-functional component attached to K will now be randomized (this will prevent correct decryption of a semi-functional ciphertext). 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 $K, K_0, \{K_i\}_{i \in S}$. One then chooses random values $t_3, u_3, \gamma \in \mathbb{Z}_p$. The temporary semi-functional key is formed as:

$$K g^{\gamma \vec{b}^*_3},\ K_0 g^{u_3 \vec{b}^*_{3,0}},\ K_i g^{t_3 \vec{b}^*_{5,i} + t_3 \vec{b}^*_{6,i}}\ \ \forall i \in S.$$

We define Game_{real}, Game_k, Game_k^N, Game_k^T, and Game_{final} as before. The hybrid security proof is accomplished in the following lemmas.

Lemma 14. *Under the subspace assumption, no polynomial time attacker can achieve a non-negligible difference in advantage between Game_{real} and Game_0.*

Proof. Given a PPT attacker $\mathcal{A}$ achieving a non-negligible difference in advantage between Game_{real} and Game_0, we will create a PPT algorithm $\mathcal{B}$ to break the subspace assumption. We will employ the subspace assumption with parameters $m = \mathcal{U} + 2$, $n_i = 3, k_i = 1$ for two values of i, and $n_i = 6, k_i = 2$ for the rest of the values of i. In order to reconcile the notation of the assumption with the notation of our construction as conveniently as possible, we will denote the bases involved in the assumption by $(\mathbb{D}, \mathbb{D}^*), (\mathbb{D}_0, \mathbb{D}_0^*) \in Dual(\mathbb{Z}_p^3, \psi)$ and $(\mathbb{D}_1, \mathbb{D}_1^*), \ldots, (\mathbb{D}_{\mathcal{U}}, \mathbb{D}_{\mathcal{U}}^*) \in Dual(\mathbb{Z}_p^6, \psi)$. $\mathcal{B}$ is given (we will ignore the U terms and μ_3 because they will not be needed):

$$G, p, g, g^{\vec{d}_1}, g^{\vec{d}_2}, g^{\vec{d}_{1,0}}, g^{\vec{d}_{2,0}}, \{g^{\vec{d}_{1,i}}, \ldots, g^{\vec{d}_{4,i}}\}_{i \in [\mathcal{U}]},$$

$$g^{\eta \vec{d}^*_1}, g^{\beta \vec{d}^*_2}, g^{\vec{d}^*_3}, g^{\eta \vec{d}^*_{1,0}}, g^{\beta \vec{d}^*_{2,0}}, g^{\vec{d}^*_{3,0}}, \{g^{\eta \vec{d}^*_{1,i}}, g^{\eta \vec{d}^*_{2,i}}, g^{\beta \vec{d}^*_{3,i}}, g^{\beta \vec{d}^*_{4,i}}, g^{\vec{d}^*_{5,i}}, g^{\vec{d}^*_{6,i}}\}_{i \in [\mathcal{U}]},$$

$$T_1, T_{1,0}, \{T_{1,i}, T_{2,i}\}_{i \in [\mathcal{U}]}.$$

The exponents of the unknown terms $T_1, T_{1,0}$ are distributed either as $\tau_1 \eta \vec{d}^*_1 + \tau_2 \beta \vec{d}^*_2$ and $\tau_1 \eta \vec{d}^*_{1,0} + \tau_2 \beta \vec{d}^*_{2,0}$ respectively, or as $\tau_1 \eta \vec{d}^*_1 + \tau_2 \beta \vec{d}^*_2 + \tau_3 \vec{d}^*_3$ and $\tau_1 \eta \vec{d}^*_{1,0} + \tau_2 \beta \vec{d}^*_{2,0} + \tau_3 \vec{d}^*_{3,0}$ respectively. Similarly, the exponents of the unknown terms $T_{1,i}, T_{2,i}$ are distributed either as $\tau_1 \eta \vec{d}^*_{1,i} + \tau_2 \beta \vec{d}^*_{3,i}$ and $\tau_1 \eta \vec{d}^*_{2,i} + \tau_2 \beta \vec{d}^*_{4,i}$ respectively, or as $\tau_1 \eta \vec{d}^*_{1,i} + \tau_2 \beta \vec{d}^*_{3,i} + \tau_3 \vec{d}^*_{5,i}$ and $\tau_1 \eta \vec{d}^*_{2,i} + \tau_2 \beta \vec{d}^*_{4,i} + \tau_3 \vec{d}^*_{6,i}$ respectively. It is $\mathcal{B}$'s task to determine if these τ_3 contributions are present or not.

$\mathcal{B}$ implicitly sets the bases for the construction as:

$$\vec{b}_1 = \eta \vec{d}^*_1,\ \vec{b}_2 = \beta \vec{d}^*_2,\ \vec{b}_3 = \vec{d}^*_3,\ \vec{b}^*_1 = \eta^{-1} \vec{d}_1,\ \vec{b}^*_2 = \beta^{-1} \vec{d}_2,\ \vec{b}^*_3 = \vec{d}_3,$$

$$\vec{b}_{1,0} = \eta \vec{d}^*_{1,0},\ \vec{b}_{2,0} = \beta \vec{d}^*_{2,0},\ \vec{b}_{3,0} = \vec{d}^*_{3,0},\ \vec{b}^*_{1,0} = \eta^{-1} \vec{d}_{1,0},\ \vec{b}^*_{2,0} = \beta^{-1} \vec{d}_{2,0},\ \vec{b}^*_{3,0} = \vec{d}_{3,0},$$

$$\vec{b}_{1,i} = \eta \vec{d}^*_{1,i},\ \vec{b}_{2,i} = \eta \vec{d}^*_{2,i},\ \vec{b}_{3,i} = \beta \vec{d}^*_{3,i},\ \vec{b}_{4,i} = \beta \vec{d}^*_{4,i},\ \vec{b}_{5,i} = \vec{d}^*_{5,i},\ \vec{b}_{6,i} = \vec{d}^*_{6,i}\ \forall i,$$

$$\vec{b}^*_{1,i} = \eta^{-1} \vec{d}_{1,i},\ \vec{b}^*_{2,i} = \eta^{-1} \vec{d}_{2,i},\ \vec{b}^*_{3,i} = \beta^{-1} \vec{d}_{3,i},\ \vec{b}^*_{4,i} = \beta^{-1} \vec{d}_{4,i},\ \vec{b}^*_{5,i} = \vec{d}_{5,i},\ \vec{b}^*_{6,i} = \vec{d}_{6,i}\ \forall i.$$

We note that these are properly distributed because $(\mathbb{D}, \mathbb{D}^*)$, $(\mathbb{D}_0, \mathbb{D}_0^*)$, etc. are randomly chosen (up to sharing the same ψ value).

$\mathcal{B}$ can use the terms given in the assumption to produce $g^{\vec{b}_1}$, $g^{\vec{b}_2}$, $g^{\vec{b}_{1,0}}$, $g^{\vec{b}_{2,0}}$, $\{g^{\vec{b}_{1,i}}, \ldots, g^{\vec{b}_{4,i}}\}$ for the public parameters. $\mathcal{B}$ chooses random values $\tilde{\alpha}_1, \tilde{\alpha_2} \in \mathbb{Z}_p$. It implicitly sets $\alpha_1 = \eta \tilde{\alpha}_1$ and $\alpha_2 = \beta \tilde{\alpha_2}$. This allows it to produce

$$e(g,g)^{\alpha_1 \psi} = \left(e_n(g^{\vec{d}_1}, g^{\eta \vec{d}^*_1}) \right)^{\tilde{\alpha}_1},\ e(g,g)^{\alpha_2 \psi} = \left(e_n(g^{\vec{d}_2}, g^{\beta \vec{d}^*_2}) \right)^{\tilde{\alpha}_2}.$$

$\mathcal{B}$ gives the public parameters to $\mathcal{A}$.

To produce a normal key for an attribute set S, $\mathcal{B}$ proceeds as follows. It chooses random values $\tilde{t}_1, \tilde{t}_2, \tilde{u}_1, \tilde{u}_2 \in \mathbb{Z}_p$. It implicitly sets $t_1 = \eta\tilde{t}_1$, $t_2 = \beta\tilde{t}_2$, $u_1 = \eta\tilde{u}_1$, $u_2 = \beta\tilde{u}_2$. It forms the key as:

$$K = g^{(\alpha_1+t_1+u_1)\vec{b}_1^*+(\alpha_2+t_2+u_2)\vec{b}_2^*} = \left(g^{\vec{d}_1}\right)^{\tilde{\alpha}_1+\tilde{t}_1+\tilde{u}_1} \left(g^{\vec{d}_2}\right)^{\tilde{\alpha}_2+\tilde{t}_2+\tilde{u}_2},$$

$$K_0 = g^{u_1\vec{b}_{1,0}^*+u_2\vec{b}_{2,0}^*} = \left(g^{\vec{d}_{1,0}}\right)^{\tilde{u}_1} \left(g^{\vec{d}_{2,0}}\right)^{\tilde{u}_2},$$

$$K_i = g^{t_1\vec{b}_{1,i}^*+t_1\vec{b}_{2,i}^*+t_2\vec{b}_{3,i}^*+t_2\vec{b}_{4,i}^*} = \left(g^{\vec{d}_{1,i}}\right)^{\tilde{t}_1} \left(g^{\vec{d}_{2,i}}\right)^{\tilde{t}_1} \left(g^{\vec{d}_{3,i}}\right)^{\tilde{t}_2} \left(g^{\vec{d}_{4,i}}\right)^{\tilde{t}_2} \ \forall i \in S.$$

To produce the challenge ciphertext for an access matrix (A, ρ) of size $\ell \times n$, $\mathcal{B}$ implicitly sets $s_1 = \tau_1$ and $s_2 = \tau_2$. It chooses a random vector $v \in \mathbb{Z}_p^n$ with first entry equal to 1. It also chooses random vectors $\tilde{v}_1, \tilde{v}_2 \in \mathbb{Z}_p^n$ with first entries equal to 0. It will implicitly set $v_1 = s_1 v + \tilde{v}_1$ and $v_2 = s_2 v + \tilde{v}_2$. We note that these are properly distributed as independent, random vectors with first entries equal to s_1 and s_2 respectively. For each j from 1 to ℓ, $\mathcal{B}$ also chooses random values $\tilde{r}_j^1, \tilde{r}_j^2, \tilde{r}_j^3 \in \mathbb{Z}_p$. It implicitly sets $r_j^1 = \tilde{r}_j^3\tau_1 + \tilde{r}_j^1$, $r_j^2 = \tilde{r}_j^3\tau_2 + \tilde{r}_j^2$. We note that these values are properly distributed because $\tilde{r}_j^1, \tilde{r}_j^2$ are random. The ciphertext is formed as:

$$M' = M_b \left(e_3(g^{\vec{d}_1}, T_1)\right)^{\tilde{\alpha}_1} \left(e_3(g^{\vec{d}_2}, T_1)\right)^{\tilde{\alpha}_2}, \ C = T_1, \ C_0 = T_{1,0},$$

$$C_j = \left(T_{1,\rho(j)}\right)^{A_j\cdot v+\tilde{r}_j^3} \left(T_{2,\rho(j)}\right)^{-\tilde{r}_j^3} \left(g^{\eta\vec{d}_{1,\rho(j)}^*}\right)^{A_j\cdot\tilde{v}_1+\tilde{r}_j^1} \left(g^{\eta\vec{d}_{2,\rho(j)}^*}\right)^{-\tilde{r}_j^1} \left(g^{\beta\vec{d}_{3,\rho(j)}^*}\right)^{A_j\cdot\tilde{v}_2+\tilde{r}_j^2} \left(g^{\beta\vec{d}_{4,\rho(j)}^*}\right)^{-\tilde{r}_j^2}$$

for all j from 1 to ℓ.

If the exponents of the T terms *do not* include the τ_3 terms, then the exponent vector of C is $s_1\vec{b}_1 + s_2\vec{b}_2$, the exponent vector of C_0 is $s_1\vec{b}_{1,0} + s_2\vec{b}_{2,0}$, and the exponent vector of C_j is:

$$\begin{aligned} &= (A_j\cdot\tau_1 v + A_j\cdot\tilde{v}_1 + \tau_1\tilde{r}_j^3 + \tilde{r}_j^1)\eta\vec{d}_{1,\rho(j)}^* + (-\tau_1\tilde{r}_j^3 - \tilde{r}_j^1)\eta\vec{d}_{2,\rho(j)}^* \\ &\quad +(A_j\cdot\tau_2 v + A_j\cdot\tilde{v}_2 + \tau_2\tilde{r}_j^3 + \tilde{r}_j^2)\beta\vec{d}_{3,\rho(j)}^* + (-\tau_2\tilde{r}_j^3 - \tilde{r}_j^2)\beta\vec{d}_{4,\rho(j)}^* \\ &= (A_j\cdot v_1 + r_j^1)\vec{b}_{1,\rho(j)} - r_j^1\vec{b}_{2,\rho(j)} + (A_j\cdot v_2 + r_j^2)\vec{b}_{3,\rho(j)} - r_j^2\vec{b}_{4,\rho(j)}. \end{aligned}$$

Thus we have a properly distributed normal ciphertext in this case.

If the exponents of the T terms *do* include the τ_3 terms, then the exponent vector of C is $s_1\vec{b}_1 + s_2\vec{b}_2 + s_3\vec{b}_3$, where $s_3 := \tau_3$, the exponent vector of C_0 is $s_1\vec{b}_{1,0} + s_2\vec{b}_{2,0} + s_3\vec{b}_{3,0}$, and the exponent vector of each C_j is:

$$(A_j\cdot v_1 + r_j^1)\vec{b}_{1,\rho(j)} - r_j^1\vec{b}_{2,\rho(j)} + (A_j\cdot v_2 + r_j^2)\vec{b}_{3,\rho(j)} - r_j^2\vec{b}_{4,\rho(j)}$$
$$+(A_j\cdot v + \tilde{r}_j^3)\tau_3\vec{b}_{5,\rho(j)} - \tilde{r}_j^3\tau_3\vec{b}_{6,\rho(j)}.$$

This is a properly distributed semi-functional ciphertext with $v_3 = \tau_3 v$ and $r_j^3 = \tau_3\tilde{r}_3^j$. (Note that these values are distributed randomly and independently from v_1, v_2, r_j^1, r_j^2.)

Thus, when the τ_3 terms are absent, $\mathcal{B}$ properly simulates Game_{real}, and when the τ_3 terms are present, $\mathcal{B}$ properly simulates Game_0. As a result, $\mathcal{B}$ can leverage $\mathcal{A}$'s non-negligible difference in advantage between these games to gain a non-negligible advantage against the subspace assumption.

□

Lemma 15. *Under the subspace 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.*

Proof. Given a PPT attacker $\mathcal{A}$ achieving a non-negligible difference in advantage between Game_{k-1} and Game_k^N for some k, we will create a PPT algorithm $\mathcal{B}$ to break the subspace assumption. We will employ the subspace assumption with parameters $m = \mathcal{U}+2$, $n_i = 3, k_i = 1$ for two values of i, and $n_i = 6, k_i = 2$ for the rest of the values of i. In order to reconcile the notation of the assumption with the notation of our construction as conveniently as possible, we will denote the bases involved in the assumption by $(\mathbb{B}, \mathbb{B}^*), (\mathbb{B}_0, \mathbb{B}_0^*) \in Dual(\mathbb{Z}_p^3, \psi)$ and $(\mathbb{B}_1, \mathbb{B}_1^*), \ldots, (\mathbb{B}_{\mathcal{U}}, \mathbb{B}_{\mathcal{U}}^*) \in Dual(\mathbb{Z}_p^6, \psi)$. $\mathcal{B}$ is given (we will ignore μ_3 because it will not be needed):

$$G, p, g, g^{\vec{b}_1}, g^{\vec{b}_2}, g^{\vec{b}_{1,0}}, g^{\vec{b}_{2,0}}, \{g^{\vec{b}_{1,i}}, \ldots, g^{\vec{b}_{4,i}}\}_{i\in[\mathcal{U}]},$$

$$g^{\eta\vec{b}_1^*}, g^{\beta\vec{b}_2^*}, g^{\vec{b}_3^*}, g^{\eta\vec{b}_{1,0}^*}, g^{\beta\vec{b}_{2,0}^*}, g^{\vec{b}_{3,0}^*}, \{g^{\eta\vec{b}_{1,i}^*}, g^{\eta\vec{b}_{2,i}^*}, g^{\beta\vec{b}_{3,i}^*}, g^{\beta\vec{b}_{4,i}^*}, g^{\vec{b}_{5,i}^*}, g^{\vec{b}_{6,i}^*}\}_{i\in[\mathcal{U}]},$$

$$U_1 = g^{\mu_1\vec{b}_1+\mu_2\vec{b}_2+\mu_3\vec{b}_3}, U_{1,0} = g^{\mu_1\vec{b}_{1,0}+\mu_2\vec{b}_{2,0}+\mu_3\vec{b}_{3,0}},$$

$$\{U_{1,i} = g^{\mu_1\vec{b}_{1,i}+\mu_2\vec{b}_{3,i}+\mu_3\vec{b}_{5,i}}, U_{2,i} = g^{\mu_1\vec{b}_{2,i}+\mu_2\vec{b}_{4,i}+\mu_3\vec{b}_{6,i}}\}_{i\in[\mathcal{U}]},$$

$$T_1, T_{1,0}, \{T_{1,i}, T_{2,i}\}_{i\in[\mathcal{U}]}.$$

The exponents of the unknown terms $T_1, T_{1,0}$ are distributed either as $\tau_1\eta\vec{b}_1^*+\tau_2\beta\vec{b}_2^*$ and $\tau_1\eta\vec{b}_{1,0}^*+\tau_2\beta\vec{b}_{2,0}^*$ respectively, or as $\tau_1\eta\vec{b}_1^*+\tau_2\beta\vec{b}_2^*+\tau_3\vec{b}_3^*$ and $\tau_1\eta\vec{b}_{1,0}^*+\tau_2\beta\vec{b}_{2,0}^*+\tau_3\vec{b}_{3,0}^*$ respectively. Similarly, the exponents of the unknown terms $T_{1,i}, T_{2,i}$ are distributed either as $\tau_1\eta\vec{b}_{1,i}^* + \tau_2\beta\vec{b}_{3,i}^*$ and $\tau_1\eta\vec{b}_{2,i}^*+\tau_2\beta\vec{b}_{4,i}^*$ respectively, or as $\tau_1\eta\vec{b}_{1,i}^*+\tau_2\beta\vec{b}_{3,i}^*+\tau_3\vec{b}_{5,i}^*$ and $\tau_1\eta\vec{b}_{2,i}^*+\tau_2\beta\vec{b}_{4,i}^*+\tau_3\vec{b}_{6,i}^*$ respectively. It is $\mathcal{B}$'s task to determine if these τ_3 contributions are present or not.

$\mathcal{B}$ implicitly sets $(\mathbb{B}, \mathbb{B}^*)$, $(\mathbb{B}_0, \mathbb{B}_0^*)$, $\{(\mathbb{B}_i, \mathbb{B}_i^*)\}$ as the bases for the construction. It chooses random values $\tilde{\alpha}_1, \tilde{\alpha}_2 \in \mathbb{Z}_p$ and implicitly sets $\alpha_1 = \eta\tilde{\alpha}_1$ and $\alpha_2 = \beta\tilde{\alpha}_2$. This allows it to compute $e(g,g)^{\alpha_1\psi}$ as $e_3(g^{\vec{b}_1}, g^{\eta\vec{b}_1^*})^{\tilde{\alpha}_1}$, and $e(g,g)^{\alpha_2\psi}$ as $e_3(g^{\vec{b}_2}, g^{\beta\vec{b}_2^*})^{\tilde{\alpha}_2}$. $\mathcal{B}$ can thus produce the public parameters, and it gives these to $\mathcal{A}$.

To respond to $\mathcal{A}$'s first $k-1$ key queries, $\mathcal{B}$ acts as follows. To produce a semi-functional key for an attribute set S, it chooses random values $\tilde{t}_1, \tilde{t}_2, \tilde{u}_1, \tilde{u}_2, \gamma \in \mathbb{Z}_p$. It implicitly sets $t_1 = \eta\tilde{t}_1$, $t_2 = \beta\tilde{t}_2$, $u_1 = \eta\tilde{u}_1$, $u_2 = \beta\tilde{u}_2$. It forms the key as:

$$K = \left(g^{\eta\vec{b}_1^*}\right)^{\tilde{\alpha}_1+\tilde{t}_1+\tilde{u}_1} \left(g^{\beta\vec{b}_2^*}\right)^{\tilde{\alpha}_2+\tilde{t}_2+\tilde{u}_2} g^{\gamma\vec{b}_3^*},$$

$$K_0 = \left(g^{\eta\vec{b}_{1,0}^*}\right)^{\tilde{u}_1} \left(g^{\beta\vec{b}_{2,0}^*}\right)^{\tilde{u}_2},$$

$$K_i = \left(g^{\eta\vec{b}_{1,i}^*}\right)^{\tilde{t}_1} \left(g^{\eta\vec{b}_{2,i}^*}\right)^{\tilde{t}_1} \left(g^{\beta\vec{b}_{3,i}^*}\right)^{\tilde{t}_2} \left(g^{\beta\vec{b}_{4,i}^*}\right)^{\tilde{t}_2} \ \forall i \in S.$$

In this way, $\mathcal{B}$ produces properly distributed semi-functional keys in response to the first $k-1$ key requests. We note that $\mathcal{B}$ can similarly produce normal keys in response to key requests $k+1$ and onward using the same procedure except leaving off $g^{\gamma\vec{b}_3^*}$ from K.

To create the k^{th} key for some attribute set S, $\mathcal{B}$ proceeds as follows. It chooses random values $\tilde{u}_1, \tilde{u}_2, \tilde{u}_3 \in \mathbb{Z}_p$. It implicitly sets $t_1 = \eta\tau_1$, $t_2 = \beta\tau_2$, $u_1 = \eta(\tau_1\tilde{u}_3 + \tilde{u}_1)$, and $u_2 = \beta(\tau_2\tilde{u}_3 + \tilde{u}_2)$. We note that these values are independently random because $\tau_1, \tau_2, \tilde{u}_1, \tilde{u}_2$ are independently random. The key is formed as:

$$K = \left(g^{\eta\vec{b}_1^*}\right)^{\tilde{\alpha_1}+\tilde{u}_1} \left(g^{\beta\vec{b}_2^*}\right)^{\tilde{\alpha}_2+\tilde{u}_2} T_1(T_1)^{\tilde{u}_3},$$

$$K_0 = \left(g^{\eta\vec{b}_{1,0}^*}\right)^{\tilde{u}_1} \left(g^{\beta\vec{b}_{2,0}^*}\right)^{\tilde{u}_2} T_{1,0}^{\tilde{u}_3},$$

$$K_i = T_{1,i} T_{2,i} \; \forall i \in S.$$

If the exponents of the T terms here *do not* include the τ_3 terms, then this is a properly distributed normal key. If they *do* include the τ_3 terms, then this is a properly distributed nominal semi-functional key with $t_3 = \tau_3$ and $u_3 = \tau_3 \tilde{u}_3$. (Note that these values are random and independent of t_1, t_2, u_1, u_2.)

To create the semi-functional ciphertext for some $n \times \ell$ access matrix (A, ρ), $\mathcal{B}$ chooses random values $\tilde{r}_j^1, \tilde{r}_j^2, \tilde{r}_j^3 \in \mathbb{Z}_p$ for each j from 1 to ℓ. It also chooses a random vector $v \in \mathbb{Z}_p^n$ with first entry equal to 1, and random vectors $\tilde{v}_1, \tilde{v}_2 \in \mathbb{Z}_p^n$ with first entries equal to 0. It implicitly sets $s_1 = \mu_1$, $s_2 = \mu_2$, $s_3 = \mu_3$, $v_1 = s_1 v + \tilde{v}_1$, $v_2 = s_2 v + \tilde{v}_2$, $v_3 = s_3 v$, $r_j^1 = \mu_1 \tilde{r}_j^3 + \tilde{r}_j^1$, $r_j^2 = \mu_2 \tilde{r}_j^3 + \tilde{r}_j^2$, and $r_j^3 = \mu_3 \tilde{r}_j^3$. We note that these values are properly distributed. The ciphertext is formed as:

$$M' = M_b e_3(U_1, g^{\eta \vec{b}_1^*})^{\tilde{\alpha}_1} e_3(U_1, g^{\beta \vec{b}_2^*})^{\tilde{\alpha}_2}, \; C = U_1, \; C_0 = U_{1,0},$$

$$C_j = \left(g^{\vec{b}_{1,\rho(j)}}\right)^{A_j \cdot \tilde{v}_1 + \tilde{r}_j^1} \left(g^{\vec{b}_{2,\rho(j)}}\right)^{-\tilde{r}_j^1} \left(g^{\vec{b}_{3,\rho(j)}}\right)^{A_j \cdot \tilde{v}_2 + \tilde{r}_j^2} \left(g^{\vec{b}_{4,\rho(j)}}\right)^{-\tilde{r}_j^2} U_{1,\rho(j)}^{A_j \cdot v + \tilde{r}_j^3} U_{2,\rho(j)}^{-\tilde{r}_j^3},$$

for all j from 1 to ℓ.

Thus, when the τ_3 terms are absent, $\mathcal{B}$ properly simulates Game_{k-1}, and when the τ_3 terms are present, $\mathcal{B}$ properly simulates Game_k^N. As a result, $\mathcal{B}$ can leverage $\mathcal{A}$'s non-negligible difference in advantage between these games to gain a non-negligible advantage against the subspace assumption. □

Lemma 16. *Under the three party Diffie-Hellman assumption, no polynomial time attacker can achieve a non-negligible difference in advantage between Game_k^N and Game_k^T for any 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. $\mathcal{B}$ is given g, g^x, g^y, g^z, T, where T is either g^{xyz} or a random element of G. $\mathcal{B}$ will simulate either Game_k^N or Game_k^T with $\mathcal{A}$ depending on the nature of T.

$\mathcal{B}$ chooses random dual orthonormal bases $(\mathbb{D}, \mathbb{D}^*), (\mathbb{D}_0, \mathbb{D}_0^*)$ of dimension 3 and $(\mathbb{D}_1, \mathbb{D}_1^*)$, …, $(\mathbb{D}_{\mathcal{U}}, \mathbb{D}_{\mathcal{U}}^*)$ of dimension 6, all with the same value of ψ. It then implicitly sets $(\mathbb{B}, \mathbb{B}^*)$ and $(\mathbb{B}_0, \mathbb{B}_0^*)$ as follows:

$$\vec{b}_1 = \vec{d}_1, \; \vec{b}_2 = \vec{d}_2, \; \vec{b}_3 = (xy)^{-1} \vec{d}_3, \; \vec{b}_1^* = \vec{d}_1^*, \; \vec{b}_2^* = \vec{d}_2^*, \; \vec{b}_3^* = xy \vec{d}_3^*,$$

$$\vec{b}_{1,0} = \vec{d}_{1,0}, \; \vec{b}_{2,0} = \vec{d}_{2,0}, \; \vec{b}_{3,0} = (xy)^{-1} \vec{d}_{3,0}, \; \vec{b}_{1,0}^* = \vec{d}_{1,0}^*, \; \vec{b}_{2,0}^* = \vec{d}_{2,0}^*, \; \vec{b}_{3,0}^* = xy \vec{d}_{3,0}^*.$$

We note $(\mathbb{B}, \mathbb{B}^*)$ and $(\mathbb{B}_0, \mathbb{B}_0^*)$ are properly distributed.

$\mathcal{B}$ sets the *normal* portions of $(\mathbb{B}_1, \mathbb{B}_1^*)$, …, $(\mathbb{B}_{\mathcal{U}}, \mathbb{B}_{\mathcal{U}}^*)$ as follows:

$$\vec{b}_{1,i} = \vec{d}_{1,i}, \; \vec{b}_{2,i} = \vec{d}_{2,i}, \; \vec{b}_{3,i} = \vec{d}_{3,i}, \; \vec{b}_{4,i} = \vec{d}_{4,i} \; \forall i = 1, \ldots, \mathcal{U},$$

$$\vec{b}_{1,i}^* = \vec{d}_{1,i}^*, \; \vec{b}_{2,i}^* = \vec{d}_{2,i}^*, \; \vec{b}_{3,i}^* = \vec{d}_{3,i}^*, \; \vec{b}_{4,i}^* = \vec{d}_{4,i}^* \; \forall i = 1, \ldots, \mathcal{U}.$$

The semi-functional portions of these bases will be set later (at which point we may verify that all of $(\mathbb{B}_1, \mathbb{B}_1^*)$, …, $(\mathbb{B}_{\mathcal{U}}, \mathbb{B}_{\mathcal{U}}^*)$ are properly distributed).

$\mathcal{B}$ chooses $\alpha_1, \alpha_2 \in \mathbb{Z}_p$ randomly. We observe that $\mathcal{B}$ can now produce the public parameters, and also knows the master secret key (enabling it to create normal keys). It gives the public parameters to $\mathcal{A}$. To create the first $k-1$ semi-functional keys in response to $\mathcal{A}$'s key requests, $\mathcal{B}$

first creates a normal key, then raises $g^{\vec{d}^*_3}$ to a random exponent in $\mathbb{Z}_p$ and multiplies this by K. We are using here that $\mathcal{B}$ does not need to know $g^{\vec{b}^*_3}$ precisely in order to create well-distributed semi-functional keys - it suffices for $\mathcal{B}$ to know $g^{c\vec{b}^*_3}$ for some (nonzero) $c \in \mathbb{Z}_p$.

$\mathcal{A}$ requests the k^{th} key for some attribute set $S \subset [\mathcal{U}]$. At this point, $\mathcal{B}$ implicitly defines the semi-functional parts of the bases $(\mathbb{B}_1, \mathbb{B}^*_1)$, ..., $(\mathbb{B}_\mathcal{U}, \mathbb{B}^*_\mathcal{U})$ as follows (note that these have not been involved in the game before this):

$$\vec{b}_{5,i} = x^{-1}\vec{d}_{5,i},\ \vec{b}_{6,i} = \vec{d}_{6,i}\ \vec{b}^*_{5,i} = x\vec{d}^*_{5,i},\ \vec{b}^*_{6,i} = \vec{d}^*_{6,i}\ \forall i \notin S,$$

$$\vec{b}_{5,i} = \vec{d}_{5,i},\ \vec{b}_{6,i} = \vec{d}_{6,i},\ \vec{b}^*_{5,i} = \vec{d}^*_{5,i},\ \vec{b}^*_{6,i} = \vec{d}^*_{6,i}\ \forall i \in S.$$

We observe that all of $(\mathbb{B}, \mathbb{B}^*)$, $(\mathbb{B}_0, \mathbb{B}^*_0)$, $(\mathbb{B}_1, \mathbb{B}^*_1)$, ..., $(\mathbb{B}_\mathcal{U}, \mathbb{B}^*_\mathcal{U})$ are properly distributed, and their distribution is independent of x, y, and S (the involvement of x, y, and S is only present in $\mathcal{B}$'s view and is information-theoretically hidden from $\mathcal{A}$, see Lemma 11).

To create the k^{th} key, $\mathcal{B}$ first creates a normal key with components $K, K_0, \{K_i\}_{i\in S}$. To create the semi-functional components, it chooses a random value $\tilde{u}_3$ and implicitly sets $t_3 = z$, $u_3 = (xy)^{-1}\tilde{u}_3$. It then forms the semi-functional component for K_0 as

$$g^{u_3\vec{b}^*_{3,0}} = g^{\tilde{u}_3\vec{d}^*_{3,0}}$$

and the semi-functional component for each K_i as

$$g^{t_3\vec{b}^*_{5,i} + t_3\vec{b}^*_{6,i}} = (g^z)^{\vec{d}^*_{5,i} + \vec{d}^*_{6,i}}\ \forall i \in S.$$

It forms the semi-functional component for K as:

$$T^{\vec{d}^*_3} g^{\tilde{u}_3\vec{d}^*_3}.$$

If $T = g^{xyz}$, then the exponent vector here is $xyz\vec{d}^*_3 + \tilde{u}_3\vec{d}^*_3 = (z+u_3)\vec{b}^*_3$, as required for a nominal semi-functional key. Otherwise, this exponent vector is distributed as a random multiple of $\vec{b}^*_3$, as required for a temporary semi-functional key. $\mathcal{B}$ multiplies these semi-functional components with the normal $K, K_0, \{K_i\}_{i\in S}$ to produce the key it gives to $\mathcal{A}$.

At some *later* point, $\mathcal{A}$ requests the challenge ciphertext for some $\ell \times n$ access matrix (A, ρ) that is *not* satisfied by the attribute set S. $\mathcal{B}$ first creates a normal ciphertext with components $M', C, C_0, \{C_j\}_{j=1}^\ell$. To create the semi-functional components, $\mathcal{B}$ first computes a vector $\nu \in \mathbb{Z}_p^n$ that has first entry equal to 1 and is orthogonal to all of the rows A_j of A such that $\rho(j) \in S$ (such a vector must exist since S fails to satisfy A, and it is efficiently computable). $\mathcal{B}$ also chooses a random vector $\tilde{v}_3 \in \mathbb{Z}_p^n$ subject to the constraint that the first entry is zero. It implicitly sets $s_3 = xy$ and sets $v_3 = xy\nu + x\tilde{v}_3$. We note that s_3 is random because all of the dual orthonormal bases are distributed independently of x, y, and v_3 is distributed as a random vector with first entry equal to s_3. $\mathcal{B}$ also chooses random values $r^3_j \in \mathbb{Z}_p$ for all j such that $\rho(j) \in S$ and random values $\tilde{r}^3_j \in \mathbb{Z}_p$ for all j such that $\rho(j) \notin S$. For values of j such that $\rho(j) \notin S$, it implicitly sets $r^3_j = x\tilde{r}^3_j$. $\mathcal{B}$ can then produce the semi-functional components of the ciphertext as:

$$g^{s_3\vec{b}_3} = g^{\vec{d}_3},\ g^{s_3\vec{b}_{3,0}} = g^{\vec{d}_{3,0}},$$

$$g^{(A_j\cdot v_3 + r^3_j)\vec{b}_{5,\rho(j)} - r^3_j\vec{b}_{6,\rho(j)}} = (g^y)^{A_j\cdot\nu\vec{d}_{5,\rho(j)}}\, g^{(A_j\cdot\tilde{v}_3 + \tilde{r}^3_j)\vec{d}_{5,\rho(j)}}\, (g^x)^{-\tilde{r}^3_j\vec{d}_{6,\rho(j)}}\ \forall j \text{ s.t. } \rho(j) \notin S,$$

$$g^{(A_j\cdot v_3 + r^3_j)\vec{b}_{5,\rho(j)} - r^3_j\vec{b}_{6,\rho(j)}} = (g^x)^{A_j\cdot\tilde{v}_3\vec{d}_{5,\rho(j)}}\, g^{r^3_j\vec{d}_{5,\rho(j)} - r^3_j\vec{d}_{6,\rho(j)}}\ \forall j \text{ s.t. } \rho(j) \in S.$$

Here we have used the fact that $A_j \cdot \nu \equiv 0$ modulo p to avoid needing to produce a multiple of $g^{xy\vec{d}_{5,\rho(j)}}$ for j such that $\rho(j) \in S$.

$\mathcal{B}$ multiplies these semi-functional components by the normal components to form the semi-functional ciphertext, which is gives to $\mathcal{A}$. It can respond to the rest of $\mathcal{A}$'s key queries by calling the normal key generation algorithm. If $T = g^{xyz}$, then $\mathcal{B}$ has properly simulated Game_K^N, and if T is a random group element, then $\mathcal{B}$ has properly simulated Game_k^T. Thus, $\mathcal{B}$ can leverage $\mathcal{A}$'s non-negligible difference in advantage between these games to gain a non-negligible advantage against the three party Diffie-Hellman assumption. □

Lemma 17. *Under the source group q-parallel BDHE assumption, 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. Our $\mathcal{B}$ is given: $g, g^f, g^{df}, g^{c^i}\ \forall i \in [2q] \setminus \{q+1\}$, $g^{c^i/b_j}\ \forall i \in [2q] \setminus \{q+1\}, j \in [q]$, $g^{dfb_j}\ \forall j \in [q]$, $g^{dfc^i b_{j'}/b_j}\ \forall i \in [q], j, j' \in [q]$ such that $j \neq j'$, and T, where T is either equal to $g^{dc^{q+1}}$ 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 dual orthonormal bases $(\mathbb{D}, \mathbb{D}^*), (\mathbb{D}_0, \mathbb{D}_0^*)$ of dimension 3 and $(\mathbb{D}_1, \mathbb{D}_1^*)$, ..., $(\mathbb{D}_\mathcal{U}, \mathbb{D}_\mathcal{U}^*)$ of dimension 6, all with the same value of ψ. It then implicitly sets $(\mathbb{B}, \mathbb{B}^*)$ and $(\mathbb{B}_0, \mathbb{B}_0^*)$ as follows:

$$\vec{b}_1 = \vec{d}_1,\ \vec{b}_2 = \vec{d}_2,\ \vec{b}_3 = (cd)^{-1}\vec{d}_3,\ \vec{b}_1^* = \vec{d}_1^*,\ \vec{b}_2^* = \vec{d}_2^*,\ \vec{b}_3^* = (cd)\vec{d}_3^*,$$

$$\vec{b}_{1,0} = \vec{d}_{1,0},\ \vec{b}_{2,0} = \vec{d}_{2,0},\ \vec{b}_{3,0} = (c)^{-1}\vec{d}_{3,0},\ \vec{b}_{1,0}^* = \vec{d}_{1,0}^*,\ \vec{b}_{2,0}^* = \vec{d}_{2,0}^*,\ \vec{b}_{3,0}^* = (c)\vec{d}_{3,0}^*.$$

We note that $(\mathbb{B}, \mathbb{B}^*)$ and $(\mathbb{B}_0, \mathbb{B}_0^*)$ are properly distributed.

$\mathcal{B}$ sets the *normal* portions of $(\mathbb{B}_1, \mathbb{B}_1^*)$, ..., $(\mathbb{B}_\mathcal{U}, \mathbb{B}_\mathcal{U}^*)$ as follows:

$$\vec{b}_{1,i} = \vec{d}_{1,i},\ \vec{b}_{2,i} = \vec{d}_{2,i},\ \vec{b}_{3,i} = \vec{d}_{3,i},\ \vec{b}_{4,i} = \vec{d}_{4,i}\ \forall i = 1, \ldots, \mathcal{U},$$

$$\vec{b}_{1,i}^* = \vec{d}_{1,i}^*,\ \vec{b}_{2,i}^* = \vec{d}_{2,i}^*,\ \vec{b}_{3,i}^* = \vec{d}_{3,i}^*,\ \vec{b}_{4,i}^* = \vec{d}_{4,i}^*\ \forall i = 1, \ldots, \mathcal{U}.$$

The semi-functional portions of these bases will be set later (at which point we may verify that all of $(\mathbb{B}_1, \mathbb{B}_1^*)$, ..., $(\mathbb{B}_\mathcal{U}, \mathbb{B}_\mathcal{U}^*)$ are properly distributed).

$\mathcal{B}$ chooses $\alpha_1, \alpha_2 \in \mathbb{Z}_p$ randomly. We observe that $\mathcal{B}$ can now produce the public parameters, and also knows the master secret key (enabling it to create normal keys). It gives the public parameters to $\mathcal{A}$. To create the first $k-1$ semi-functional keys in response to $\mathcal{A}$'s key requests, $\mathcal{B}$ first creates a normal key, then raises $g^{\vec{d}_3^*}$ to a random exponent in $\mathbb{Z}_p$ and multiplies this by K. As in the proof of the previous lemma, we note here that $\mathcal{B}$ does not need to know $g^{\vec{b}_3^*}$ precisely in order to create well-distributed semi-functional keys.

Before requesting the k^{th} key, $\mathcal{A}$ will request the challenge ciphertext for some access matrix (A, ρ) of size $\ell \times n$, where both $\ell, n \leq q$. For each attribute i, we let J_i denote the set of indices $j \in [\ell]$ such that $\rho(j) = i$. For each i, $\mathcal{B}$ chooses a random value $\tilde{\eta}_i$ and defines a value $\eta_i \in \mathbb{Z}_p$ by

$$\eta_i = \tilde{\eta}_i + \sum_{j \in J_i} cA_{j,1}/b_j + \ldots + c^n A_{j,n}/b_j.$$

At this point, $\mathcal{B}$ implicitly sets the semi-functional portions of the bases $(\mathbb{B}_1, \mathbb{B}_1^*)$, ..., $(\mathbb{B}_\mathcal{U}, \mathbb{B}_\mathcal{U}^*)$ as follows (note that these have played no role in the game before this point):

$$\vec{b}_{5,i} = \vec{d}_{5,i},\ \vec{b}_{6,i} = \eta_i^{-1}\vec{d}_{6,i}\ \vec{b}_{5,i}^* = \vec{d}_{5,i}^*,\ \vec{b}_{6,i}^* = \eta_i\vec{d}_{6,i}^*\ \forall i.$$

We observe that $(\mathbb{B}_1, \mathbb{B}_1^*)$, ..., $(\mathbb{B}_\mathcal{U}, \mathbb{B}_\mathcal{U}^*)$ are properly distributed.

To create the challenge ciphertext, $\mathcal{B}$ first creates a normal ciphertext using the normal encryption algorithm. To create the semi-functional components, it implicitly sets $s_3 = cdf$. It also chooses random values $y_2, \ldots, y_n \in \mathbb{Z}_p$ and random values $\tilde{r}_j^3 \in \mathbb{Z}_p$ for each $j \in [\ell]$. It implicitly sets $v_3 = (cdf, dfc^2 + y_2, \ldots, dfc^n + y_n)$. This is distributed as a random vector with first entry equal to s_3. For each $j \in [\ell]$, $\mathcal{B}$ implicitly sets $r_3^j = -dfb_j\eta_{\rho(j)} + \tilde{r}_3^j\eta_{\rho(j)}$. These are distributed as uniformly random elements because each $\tilde{r}_j^3$ is random and $\eta_{\rho(j)} \neq 0$ (with all but negligible probability). We observe:

$$\begin{aligned} A_j \cdot v_3 + r_j^3 &= df(cA_{j,1} + c^2A_{j,2} + \ldots + c^nA_{j,n}) + A_{j,2}y_2 + \ldots + A_{j,n}y_n \\ &\quad -dfb_j\left(\tilde{\eta}_{\rho(j)} + \sum_{j' \in J_{\rho(j)}} cA_{j',1}/b_{j'} + \ldots + c^nA_{j',n}/b_{j'}\right) + \tilde{r}_j^3\eta_{\rho(j)} \end{aligned}$$

By definition, $j \in J_{\rho(j)}$, so we have some cancelation here:

$$A_j \cdot v_3 + r_j^3 = A_{j,2}y_2 + \ldots + A_{j,n}y_n - dfb_j\left(\tilde{\eta}_{\rho(j)} + \sum_{j' \in J_{\rho(j)} \setminus \{j\}} cA_{j',1}/b_{j'} + \ldots + c^nA_{j'n}/b_{j'}\right) + \tilde{r}_j^3\eta_{\rho(j)}.$$

We now see that $\mathcal{B}$ can compute $g^{A_j \cdot v_3 + r_j^3}$ using the terms it is given in the assumption, enabling it to produce $g^{(A_j \cdot v_3 + r_j^3)\vec{d}_{5,\rho(j)}} = g^{(A_j \cdot v_3 + r_j^3)\vec{b}_{5,i}}$. We also see that

$$-r_j^3\vec{b}_{6,\rho(j)} = -r_3^j\eta_{\rho(j)}^{-1}\vec{d}_{6,i} = (dfb_j - \tilde{r}_3^j)\vec{d}_{6,i},$$

so $\mathcal{B}$ can also produce $g^{-r_j^3\vec{b}_{6,\rho(j)}}$. In this way, $\mathcal{B}$ produces the semi-functional component of C_j for each j with the proper distribution.

$\mathcal{B}$ also produces the semi-functional components of C and C_0 as:

$$g^{s_3\vec{b}_3} = \left(g^f\right)^{\vec{d}_3}, \; g^{s_3\vec{b}_{3,0}} = \left(g^{df}\right)^{\vec{d}_{3,0}}.$$

It gives the resulting properly distributed semi-functional ciphertext to $\mathcal{A}$.

At some *later* point in the game, $\mathcal{A}$ requests the k^{th} key for some attribute set S. $\mathcal{B}$ can create the normal parts of the key using the normal key generation algorithm. To create the semi-functional parts, $\mathcal{B}$ proceeds as follows. Since S does not satisfy (A, ρ), $\mathcal{B}$ can (efficiently) compute a vector $w \in \mathbb{Z}_p^n$ such that its first entry is non-zero and w is orthogonal (modulo p) to all rows A_j of A such that $\rho(j) \in S$. We may assume the first entry of w is randomized. $\mathcal{B}$ implicitly sets $t_3 = w_1c^q + \ldots + w_nc^{q-n+1}$, which is properly distributed because w_1 is random (and c is nonzero with all but negligible probability). $\mathcal{B}$ also chooses a random value $\tilde{u}_3$ and implicitly sets $u_3 = -w_2c^{q-1} - \ldots - w_nc^{q-n+1} + fc^{-1}\tilde{u}_3$. This is properly distributed because $\tilde{u}_3$ is random (and fc^{-1} is nonzero with all but negligible probability).

We observe that

$$(t_3 + u_3)\vec{b}_3^* = (w_1dc^{q+1} + df\tilde{u}_3)\vec{d}_3^*.$$

$\mathcal{B}$ forms the semi-functional part of K as: $T^{w_1\vec{d}_3^*}\left(g^{df}\right)^{\tilde{u}_3\vec{d}_3^*}$. If $T = g^{dc^{q+1}}$, this is equal to $g^{(t_3+u_3)\vec{b}_3^*}$, as required for a nominal semi-functional key. Otherwise, this exponent is distributed as a random multiple of $\vec{b}_3^*$, as required for a temporary semi-functional key. We also have

$$u_3\vec{b}_{3,0}^* = (-w_2c^q - \ldots - w_nc^{q-n+2} + f\tilde{u}_3)\vec{d}_{3,0}^*,$$

enabling $\mathcal{B}$ to produce $g^{u_3\vec{b}_{3,0}^*}$ using the terms given in the assumption.

Now, $\mathcal{B}$ can also produce g^{t_3}, and hence can compute $g^{t_3 \vec{b}^*_{5,i}} = g^{t_3 \vec{d}^*_{5,i}}$ for each $i \in S$. We observe

$$t_3 \vec{b}^*_{6,i} = t_3 \eta_i \vec{d}^*_{6,i},$$

and

$$t_3 \eta_i = \left(w_1 c^q + \ldots + w_n c^{q-n+1}\right) \left(\tilde{\eta}_i + \sum_{j \in J_i} c A_{j,1}/b_j + \ldots + c^n A_{j,n}/b_j \right).$$

For each $j \in J_i$, we have $\rho(j) = i$. So for $i \in S$, we have $A_j \cdot w = 0$ modulo p for every $j \in J_i$. Thus, all of the terms involving c^{q+1} cancel, and we are left with terms that can be created in the exponent from the group elements given in the assumption (note that $n \leq q$, so $2q$ is an upper bound on the powers of c involved here). This shows that $\mathcal{B}$ can create $g^{t_3 \vec{b}_{6,i}}$ for all $i \in S$, and hence can produce properly distributed semi-functional components for each K_i of the k^{th} key.

$\mathcal{B}$ can respond to the rest of $\mathcal{A}$'s key requests by producing normal keys via the normal key generation algorithm. If $T = g^{dc^{q+1}}$, then $\mathcal{B}$ has properly simulated Game^N_k. If T is distributed randomly, then $\mathcal{B}$ has properly simulated Game^T_k. Thus, $\mathcal{B}$ can leverage $\mathcal{A}$'s non-negligible difference in advantage between these games to achieve a non-negligible advantage against the source group q-parallel BDHE assumption. □

Lemma 18. *Under the subspace assumption, no polynomial time attacker can achieve a non-negligible difference in advantage between Game^T_k and Game_k for any k from 1 to Q.*

Proof. This proof is almost identical to the proof of Lemma 15, except that $\mathcal{B}$ adds an additional term of $g^{\gamma \vec{b}^*_3}$ to K for the k^{th} key (where it chooses $\gamma \in \mathbb{Z}_p$ randomly). This ensures that when the τ_3 terms are not present, the k^{th} key will be a properly distributed semi-functional key. □

Lemma 19. *Under the subspace assumption, no polynomial time attacker can achieve a non-negligible difference in advantage between Game_Q and Game_{final}.*

Proof. Given a PPT attacker $\mathcal{A}$ achieving a non-negligible difference in advantage between Game_Q and Game_{final}, we will create a PPT algorithm $\mathcal{B}$ to break the subspace assumption. We will employ the subspace assumption with parameters $m = \mathcal{U} + 2$, $n_i = 3, k_i = 1$ for two values of i, and $n_i = 6, k_i = 2$ for the rest of the values of i. To coincide with our notation for the construction, we will denote the bases involved in the assumption by $(\mathbb{B}, \mathbb{B}^*), (\mathbb{B}_0, \mathbb{B}^*_0) \in \mathit{Dual}(\mathbb{Z}^3_p, \psi)$ and $(\mathbb{B}_1, \mathbb{B}^*_1), \ldots, (\mathbb{B}_\mathcal{U}, \mathbb{B}^*_\mathcal{U}) \in \mathit{Dual}(\mathbb{Z}^6_p, \psi)$. $\mathcal{B}$ is given (we will ignore μ_3 and $T_{1,0}, \{T_{1,i}, T_{2,i}\}_{i \in [\mathcal{U}]}$ because they will not be needed):

$$G, p, g, g^{\vec{b}_1}, g^{\vec{b}_2}, g^{\vec{b}_{1,0}}, g^{\vec{b}_{2,0}}, \{g^{\vec{b}_{1,i}}, \ldots, g^{\vec{b}_{4,i}}\}_{i \in [\mathcal{U}]},$$

$$g^{\eta \vec{b}^*_1}, g^{\beta \vec{b}^*_2}, g^{\vec{b}^*_3}, g^{\eta \vec{b}^*_{1,0}}, g^{\beta \vec{b}^*_{2,0}}, g^{\vec{b}^*_{3,0}}, \{g^{\eta \vec{b}^*_{1,i}}, g^{\eta \vec{b}^*_{2,i}}, g^{\beta \vec{b}^*_{3,i}}, g^{\beta \vec{b}^*_{4,i}}, g^{\vec{b}^*_{5,i}}, g^{\vec{b}^*_{6,i}}\}_{i \in [\mathcal{U}]},$$

$$U_1 = g^{\mu_1 \vec{b}_1 + \mu_2 \vec{b}_2 + \mu_3 \vec{b}_3}, U_{1,0} = g^{\mu_1 \vec{b}_{1,0} + \mu_2 \vec{b}_{2,0} + \mu_3 \vec{b}_{3,0}},$$

$$\{U_{1,i} = g^{\mu_1 \vec{b}_{1,i} + \mu_2 \vec{b}_{3,i} + \mu_3 \vec{b}_{5,i}}, U_{2,i} = g^{\mu_1 \vec{b}_{2,i} + \mu_2 \vec{b}_{4,i} + \mu_3 \vec{b}_{6,i}}\}_{i \in [\mathcal{U}]},\ T_1.$$

The exponent of the unknown term T_1 is distributed either as $\tau_1 \eta \vec{b}^*_1 + \tau_2 \beta \vec{b}^*_2$, or as $\tau_1 \eta \vec{b}^*_1 + \tau_2 \beta \vec{b}^*_2 + \tau_3 \vec{b}^*_3$. It is $\mathcal{B}$'s task to determine if this τ_3 contribution is present or not.

$\mathcal{B}$ sets $(\mathbb{B}, \mathbb{B}^*)$, $(\mathbb{B}_0, \mathbb{B}^*_0)$, $\{(\mathbb{B}_i, \mathbb{B}^*_i)\}$ as the bases for the construction. It will implicitly set $\alpha_1 = \eta \tau_1$ and $\alpha_2 = \beta \tau_2$. It forms $e(g,g)^{\alpha_1 \psi}$ as $e_3(T_1, g^{\vec{b}_1})$ and $e(g,g)^{\alpha_2 \psi}$ as $e_3(T_1, g^{\vec{b}_2})$. It gives the public parameters to $\mathcal{A}$.

To create a semi-functional key for an attribute set S, $\mathcal{B}$ proceeds as follows. It chooses random values $\tilde{t}_1, \tilde{t}_2, \tilde{u}_1, \tilde{u}_2, \tilde{\gamma} \in \mathbb{Z}_p$. It implicitly sets $t_1 = \eta\tilde{t}_1$, $t_2 = \beta\tilde{t}_2$, $u_1 = \eta\tilde{u}_1$, and $u_2 = \beta\tilde{u}_2$. It creates the key as:

$$K = T_1 \left(g^{\eta\vec{b}^*_1}\right)^{\tilde{t}_1+\tilde{u}_1} \left(g^{\beta\vec{b}^*_2}\right)^{\tilde{t}_2+\tilde{u}_2} \left(g^{\vec{b}^*_3}\right)^{\tilde{\gamma}},$$

$$K_0 = \left(g^{\eta\vec{b}^*_{1,0}}\right)^{\tilde{u}_1} \left(g^{\beta\vec{b}^*_{2,0}}\right)^{\tilde{u}_2},$$

$$K_i = \left(g^{\eta\vec{b}^*_{1,i}}\right)^{\tilde{t}_1} \left(g^{\eta\vec{b}^*_{2,i}}\right)^{\tilde{t}_1} \left(g^{\beta\vec{b}^*_{3,i}}\right)^{\tilde{t}_2} \left(g^{\beta\vec{b}^*_{4,i}}\right)^{\tilde{t}_2} \ \forall i \in S.$$

We note that the multiple of $\vec{b}^*_3$ appearing in the exponent of K is either equal to $\tilde{\gamma}$ or $\tilde{\gamma} + \tau_3$, depending on the nature of T_1. Either way, this is a properly distributed semi-functional key (whose distribution is independent of τ_3 even if it is present).

To create the semi-functional ciphertext for some $n \times \ell$ access matrix (A, ρ), $\mathcal{B}$ can use the same procedure employed in the proof of Lemma 15 to use the U terms to provide the semi-functional components. We repeat the description of this procedure below for the reader's convenience. The only difference here comes in computing the blinding factor for M'.

$\mathcal{B}$ chooses random values $\tilde{r}^1_j, \tilde{r}^2_j, \tilde{r}^3_j \in \mathbb{Z}_p$ for each j from 1 to ℓ. It also chooses a random vector $v \in \mathbb{Z}^n_p$ with first entry equal to 1, and random vectors $\tilde{v}_1, \tilde{v}_2 \in \mathbb{Z}^n_p$ with first entries equal to 0. It implicitly sets $s_1 = \mu_1$, $s_2 = \mu_2$, $s_3 = \mu_3$, $v_1 = s_1 v + \tilde{v}_1$, $v_2 = s_2 v + \tilde{v}_2$, $v_3 = s_3 v$, $r^1_j = \mu_1\tilde{r}^3_j + \tilde{r}^1_j$, $r^2_j = \mu_2\tilde{r}^3_j + \tilde{r}^2_j$, and $r^3_j = \mu_3\tilde{r}^3_j$. We note that these values are properly distributed. The ciphertext is formed as:

$$M' = M_b e_3(U_1, T_1),\ C = U_1,\ C_0 = U_{1,0},$$

$$C_j = \left(g^{\vec{b}_{1,\rho(j)}}\right)^{A_j\cdot\tilde{v}_1+\tilde{r}^1_j} \left(g^{\vec{b}_{2,\rho(j)}}\right)^{-\tilde{r}^1_j} \left(g^{\vec{b}_{3,\rho(j)}}\right)^{A_j\cdot\tilde{v}_2+\tilde{r}^2_j} \left(g^{\vec{b}_{4,\rho(j)}}\right)^{-\tilde{r}^2_j} U^{A_j\cdot v+\tilde{r}^3_j}_{1,\rho(j)} U^{-\tilde{r}^3_j}_{2,\rho(j)},$$

for all j from 1 to ℓ.

If the exponent of T_1 is equal to $\tau_1\eta\vec{b}^*_1 + \tau_2\beta\vec{b}^*_2$, then we have

$$e_3(U_1, T_1) = e(g, g)^{(\alpha_1 s_1 + \alpha_2 s_2)\psi},$$

and hence we have a properly distributed semi-functional encryption of M_b, as required in Game_Q. If instead the exponent of T_1 is equal to $\tau_1\eta\vec{b}^*_1 + \tau_2\beta\vec{b}^*_2 + \tau_3\vec{b}^*_3$, then we have

$$e_3(U_1, T_1) = e(g, g)^{(\alpha_1 s_1 + \alpha_2 s_2 + \mu_3\tau_3)\psi}.$$

Since τ_3 is random (and independent of the semi-functional keys and the rest of the ciphertext), this blinding factor is distributed as a freshly random group element of G_T. Therefore the ciphertext is distributed as a semi-functional encryption of a random message, as required in Game_{final}. Thus, $\mathcal{B}$ can leverage $\mathcal{A}$'s non-negligible difference in advantage between these games to achieve a non-negligible advantage against the subspace assumption. □

Combining the above lemmas with Lemma 20 in Appendix A, we have completed the proof of Theorem 13.

6 Conclusion

We have presented CP-ABE schemes in composite order and prime order bilinear groups that are fully secure and allow arbitrary reuse of attributes in access policies. Along the way, we have developed a methodology for combining proof techniques in the selective setting with dual system encryption in order to obtain full security proofs. As a consequence of the limitations of current selective techniques for CP-ABE systems, we inherit a reliance on a q-based complexity assumption. Obtaining a proof of full security in the standard model from static assumptions *without* imposing any efficiency-losing restrictions remains an important open problem.

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] J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT*, pages 146–162, 2008.

[22] A. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In *EUROCRYPT*, pages 318–335, 2012.

[23] 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.

[24] A. Lewko and B. Waters. New techniques for dual system encryption and fully secure hibe with short ciphertexts. In *TCC*, pages 455–479, 2010.

[25] A. Lewko and B. Waters. Decentralizing attribute-based encryption. In *EUROCRYPT*, pages 568–588, 2011.

[26] A. Lewko and B. Waters. Unbounded hibe and attribute-based encryption. In *EUROCRYPT*, pages 547–567, 2011.

[27] T. Okamoto and K. Takashima. Homomorphic encryption and signatures from vector decomposition. In *Pairing*, pages 57–74, 2008.

[28] T. Okamoto and K. Takashima. Hierarchical predicate encryption for inner-products. In *ASIACRYPT*, pages 214–231, 2009.

[29] T. Okamoto and K. Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *CRYPTO*, pages 191–208, 2010.

[30] 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.

[31] A. Sahai and B. Waters. Fuzzy identity based encryption. In *EUROCRYPT*, pages 457–473, 2005.

[32] A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.

[33] V. Shoup. Lower bounds for discrete logarithms and related problems. In *EUROCRYPT*, page 256266, 1997.

[34] B. Waters. Efficient identity-based ecnryption without random oracles. In *EUROCRYPT*, pages 114–127, 2005.

[35] B. Waters. Dual system encryption: realizing fully secure ibe and hibe under simple assumptions. In *CRYPTO*, pages 619–636, 2009.

[36] B. Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *PKC*, pages 53–70, 2011.

A Proof that DLIN Implies the Subspace Assumption

Lemma 20. *If the decisional linear assumption holds for a group generator $\mathcal{G}$, then the subspace assumption stated in Definition 12 also holds for $\mathcal{G}$ (for any fixed values of $m, n_1 \geq 3k_1, \ldots, n_m \geq 3k_m$).*

Proof. We assume there exists a PPT algorithm $\mathcal{A}$ breaking the subspace assumption with non-negligible advantage (for some fixed positive integers $m, n_1, \ldots, n_m, k_1, \ldots, k_m$ satisfying $n_1 \geq 3k_1, \ldots, n_m \geq 3k_m$). We create a PPT algorithm $\mathcal{B}$ which breaks the decisional linear assumption with non-negligible advantage. $\mathcal{B}$ is given $g, f, v, f^{c_1}, v^{c_2}, T$, where T is either $g^{c_1+c_2}$ or T is a uniformly random element of G. We let ℓ_f denote the discrete logarithm base g of f and ℓ_v denote the discrete logarithm base g of v, i.e. $f = g^{\ell_f}$ and $v = g^{\ell_v}$.

$\mathcal{B}$ simulates the subspace assumption for $\mathcal{A}$ as follows. $\mathcal{B}$ first chooses ψ at random from Z_p, then (independently) samples random dual orthonormal bases $(\mathbb{D}_1, \mathbb{D}_1^*) \in Dual(Z_p^{n_1}, \psi)$, …, $(\mathbb{D}_m, \mathbb{D}_m^*) \in Dual(\mathbb{Z}_p^{n_m}, \psi)$. $\mathcal{B}$ then implicitly sets:

$$\eta\vec{b}^*_{1,i} = \vec{d}^*_{2k_i+1,i} + \ell_f\vec{d}^*_{1,i},\ \eta\vec{b}^*_{2,i} = \vec{d}^*_{2k_i+2,i} + \ell_f\vec{d}^*_{2,i},\ \ldots,\ \eta\vec{b}^*_{k_i,i} = \vec{d}^*_{3k_i,i} + \ell_f\vec{d}^*_{k_i,i},$$

$$\beta\vec{b}^*_{k_i+1,i} = \vec{d}^*_{2k_i+1,i} + \ell_v\vec{d}^*_{k_i+1,i},\ \beta\vec{b}^*_{k_i+2,i} = \vec{d}^*_{2k_i+2,i} + \ell_v\vec{d}^*_{k_i+2,i},\ \ldots,\ \beta\vec{b}^*_{2k_i,i} = \vec{d}^*_{3k_i,i} + \ell_v\vec{d}^*_{2k_i,i},$$

$$\vec{b}^*_{2k_i+1,i} = \vec{d}^*_{2k_i+1,i},\ \ldots,\vec{b}^*_{n_i,i} = \vec{d}^*_{n_i,i}$$

for each i from 1 to m. In other words, $\mathcal{B}$ has set $\eta = \ell_f$ and $\beta = \ell_v$, with $\vec{b}^*_{1,i} = \eta^{-1}\vec{d}^*_{2k_i+1,i} + \vec{d}^*_{1,i}$ for example.

$\mathcal{B}$ sets the dual basis as:

$$\vec{b}_{1,i} = \vec{d}_{1,i},\ \vec{b}_{2,i} = \vec{d}_{2,i},\ \ldots,\vec{b}_{2k_i,i} = \vec{d}_{2k_i,i},$$

$$\vec{b}_{2k_i+1,i} = \vec{d}_{2k_i+1,i} - \ell_f^{-1}\vec{d}_{1,i} - \ell_v^{-1}\vec{d}_{k_i+1,i},\ \ldots,\ \vec{b}_{3k_i,i} = \vec{d}_{3k_i,i} - \ell_f^{-1}\vec{d}_{k_i,i} - \ell_v^{-1}\vec{d}_{2k_i,i},$$

$$\vec{b}_{3k_i+1,i} = \vec{d}_{3k_i+1,i},\ \ldots,\ \vec{b}_{n_i,i} = \vec{d}_{n_i,i}.$$

We note that each pair $(\mathbb{B}_1, \mathbb{B}_1^*), \ldots, (\mathbb{B}_m, \mathbb{B}_m^*)$ is a indeed a pair of dual orthonormal bases. We also note that $\mathcal{B}$ can produce all of $g^{\eta\vec{b}^*_{1,i}}, \ldots, g^{\eta\vec{b}^*_{k_i,i}}$, $g^{\beta\vec{b}^*_{k_i+1,i}}, \ldots, g^{\beta\vec{b}^*_{2k_i,i}}$, $g^{\vec{b}^*_{2k_i+1,i}}, \ldots, g^{\vec{b}^*_{n_i,i}}$, $g^{\vec{b}_{1,i}}, \ldots, g^{\vec{b}_{2k_i,i}}$, and $g^{\vec{b}_{3k_i+1,i}}, \ldots, g^{\vec{b}_{n_i,i}}$ for each i from 1 to m, but *cannot produce* $g^{\vec{b}_{2k_i+1,i}}, \ldots, g^{\vec{b}_{3k_i,i}}$.

We now observe that $\eta = \ell_f$, $\beta = \ell_v$, $\vec{b}_1, \ldots, \vec{b}_n$ and $(\mathbb{B}_1, \mathbb{B}_1^*), \ldots, (\mathbb{B}_m, \mathbb{B}_m^*)$ are properly distributed. Indeed, this follows from the fact that ℓ_f, ℓ_v are randomly distributed and from Lemma 11, since we have applied a change of basis to each $(\mathbb{D}_i, \mathbb{D}_i^*)$.

Now $\mathcal{B}$ creates $U_{1,i}, \ldots, U_{k_i,i}$ for each i as follows. It chooses random values $\mu'_1, \mu'_2, \mu'_3 \in \mathbb{Z}_p$. It sets:

$$U_{1,i} = g^{\mu'_1\vec{b}_{1,i} + \mu'_2\vec{b}_{k_i+1,i} + \mu'_3\vec{d}_{2k_i+1,i}}.$$

We note that

$$\mu_1'\vec{b}_{1,i} + \mu_2'\vec{b}_{k_i+1,i} + \mu_3'\vec{d}_{2k_i+1,i} = (\mu_1' + \ell_f^{-1}\mu_3')\vec{b}_{1,i} + (\mu_2' + \ell_v^{-1}\mu_3')\vec{b}_{k_i+1,i} + \mu_3'\vec{b}_{2k_i+1,i}.$$

In other words, $\mathcal{B}$ has implicitly set $\mu_1 = \mu_1' + \ell_f^{-1}\mu_3'$, $\mu_2 = \mu_2' + \ell_v^{-1}\mu_3'$, and $\mu_3 = \mu_3'$. We note that these values are uniformly random, and μ_3 is known to $\mathcal{B}$. $\mathcal{B}$ can then similarly form $U_{2,i}, \ldots, U_{k_i,i}$ as:

$$U_{2,i} = g^{\mu_1'\vec{b}_{2,i} + \mu_2'\vec{b}_{k_i+2,i} + \mu_3'\vec{d}_{2k_i+2,i}}, \ \ldots, \ U_{k_i,i} = g^{\mu_1'\vec{b}_{k_i,i} + \mu_2'\vec{b}_{2k_i,i} + \mu_3'\vec{d}_{3k_i,i}}.$$

$\mathcal{B}$ then implicitly sets $\tau_1 = c_1$ and $\tau_2 = c_2$. We note that for each i from 1 to m:

$$\begin{aligned}
\tau_1\eta\vec{b}^*_{1,i} + \tau_2\beta\vec{b}^*_{k_i+1,i} &= (c_1+c_2)\vec{d}^*_{2k_i+1,i} + c_1\ell_f\vec{d}^*_{1,i} + c_2\ell_v\vec{d}^*_{k_i+1,i}, \\
&\vdots \\
\tau_1\eta\vec{b}^*_{k_i,i} + \tau_2\beta\vec{b}^*_{2k_i,i} &= (c_1+c_2)\vec{d}^*_{3k_i,i} + c_1\ell_f\vec{d}^*_{k_i,i} + c_2\ell_v\vec{d}^*_{2k_i,i}.
\end{aligned}$$

The terms which are multiples of $c_1\ell_f$ and $c_2\ell_v$ are not difficult for $\mathcal{B}$ to produce as exponents of g, since $\mathcal{B}$ has $f^{c_1} = g^{c_1\ell_f}$ and $v^{c_2} = g^{c_2\ell_v}$. For the multiples of $c_1 + c_2$, $\mathcal{B}$ needs to use T.

$\mathcal{B}$ computes for each i:

$$T_{1,i} = T^{\vec{d}^*_{2k_i+1,i}} \left(f^{c_1}\right)^{\vec{d}^*_{1,i}} \left(v^{c_2}\right)^{\vec{d}^*_{k_i+1,i}}, \ \ldots, \ T_{k_i,i} = T^{\vec{d}^*_{3k_i,i}} \left(f^{c_1}\right)^{\vec{d}^*_{k_i,i}} \left(v^{c_2}\right)^{\vec{d}^*_{2k_i,i}}.$$

If $T = g^{c_1+c_2}$, then these are distributed as $V_{1,i}, \ldots, V_{k_i,i}$. If $T = g^{c_1+c_2+w}$ (for random w), then these are distributed as $W_{1,i}, \ldots, W_{k_i,i}$, with τ_3 implicitly set to w.

$\mathcal{B}$ gives

$$D := \big(\{g^{\vec{b}_{1,i}}, g^{\vec{b}_{2,i}}, \ldots, g^{\vec{b}_{2k_i,i}}, g^{\vec{b}_{3k_i+1,i}}, \ldots, g^{\vec{b}_{n_i,i}}, g^{\eta\vec{b}^*_{1,i}}, \ldots, g^{\eta\vec{b}^*_{k_i,i}}, g^{\beta\vec{b}^*_{k_i+1,i}}, \ldots, g^{\beta\vec{b}^*_{2k_i,i}},$$

$$g^{\vec{b}^*_{2k_i+1,i}}, \ldots, g^{\vec{b}^*_{n_i,i}}, U_{1,i}, U_{2,i}, \ldots, U_{k_i,i}\}_{i=1}^m, \mu_3\big)$$

to $\mathcal{A}$, along with $\{T_{1,i} \ldots, T_{k_i,i}\}_{i=1}^m$. $\mathcal{B}$ can then leverage $\mathcal{A}$'s non-negligible advantage in distinguishing between the distributions $\{V_{1,i}, \ldots, V_{k_i,i}\}$ and $\{W_{1,i}, \ldots, W_{k_i,i}\}$ to achieve a non-negligible advantage in distinguishing $T = g^{c_1+c_2}$ from $T = g^{c_1+c_2+w}$, violating the decisional linear assumption. □

B Proof of Our q-Based Assumption in the Generic Group Model

We prove a lower bound for the complexity of our source group parallel BDHE assumption in the generic group model. We do this for the prime order version of the assumption, though the proof for the composite-order version in a subgroup is analogous.

In the generic group model [33], an adversary is not given direct access to the group, but rather only receives "handles" representing elements. It must interact with an oracle to perform the group operation (multiplication and division are both enabled) and obtain handles for new elements. It is assumed that it can only use handles which it has previously received from the oracle. We consider an experiment where an adversary is given handles for the group elements given out in the assumption as well as a handle for the challenge term T_β (here, β is a uniformly random bit). The adversary may then interact with the oracle to perform group operations and pairings. It is then given the handles for the group elements resulting from these operations.

Finally, the adversary must guess the bit β. The difference between the adversary's success probability and one half is defined to be its advantage. For other examples of uses of the generic group model to justify assumptions in bilinear groups, see [8, 21].

We now develop some convenient notation. We consider $c, d, f, b_1, \ldots, b_q$ as variables over $\mathbb{Z}_p$, and we define $\mathcal{M}$ to be the following set of rational functions over these variables:

$$\mathcal{M} := \{1, f, df, c^1, \ldots, c^q, c^{q+2}, \ldots, c^{2q},\ c^i/b_j\ \forall i \in [2q] \setminus \{q+1\},\ \forall j \in [q],$$

$$dfb_j\ \forall j \in [q],\ dfc^i b_{j'}/b_j\ \forall i \in [q],\ \forall j, j' \in [q] \text{ s.t. } j \neq j'\}.$$

These are the exponents of the group elements given out in our source group q-parallel BDHE assumption. We let $E(\mathcal{M})$ denote the set of all pairwise products of functions in $\mathcal{M}$. This set of rational functions represents the exponents of elements in G_T that can be obtained by pairing elements with exponents in $\mathcal{M}$.

We say a function T is *dependent* on a set of functions $\mathcal{S} = \{S_1, \ldots, S_k\}$ if there exist constants $r_1, \ldots, r_k \in \mathbb{Z}_p$ such that $T = r_1 S_1 + \cdots + r_k S_k$. This is an equality of functions over $\mathbb{Z}_p$, and hence must hold for *all* settings of the variables. If no such constants exist, we say that T is *independent* of $\mathcal{S}$. We begin by establishing the following lemma.

Lemma 21. *For each function $M \in \mathcal{M} \cup \{dc^{q+1}\}$, the product $M \cdot dc^{q+1}$ is independent of $E(\mathcal{M}) \cup dc^{q+1}(\mathcal{M} \setminus M)$. (Here, $dc^{q+1}(\mathcal{M} \setminus M)$ denotes the set formed by removing M from $\mathcal{M}$ and then multiplying all remaining elements by dc^{q+1}.)*

Proof. We note that every element of $\mathcal{M} \cup \{dc^{q+1}\}$ and every element of $E(\mathcal{M}) \cup dc^{q+1}(\mathcal{M} \setminus M)$ is a ratio of monomials. Hence, the only way $M(dc^{q+1})$ can be dependent on $E(\mathcal{M}) \cup dc^{q+1}(\mathcal{M} \setminus M)$ is if it is in fact *contained* in the set $E(\mathcal{M}) \cup dc^{q+1}(\mathcal{M} \setminus M)$. First, we note that d^2c^{2q+2} is not contained in $E(\mathcal{M}) \cup dc^{q+1}\mathcal{M}$. For any $M \in \mathcal{M}$, it is clear that $dc^{q+1}M \notin dc^{q+1}(\mathcal{M} \setminus M)$. Thus it suffices to prove that for each M, $dc^{q+1}M \notin E(\mathcal{M})$. In other words, we must show that $E(\mathcal{M})$ does not intersect with the set $dc^{q+1}\mathcal{M}$ (the set formed by multiplying each element of $\mathcal{M}$ by dc^{q+1}. To see this, we examine the set $dc^{q+1}\mathcal{M}$.

By definition, we have that

$$dc^{q+1}\mathcal{M} = \{dc^{q+1}, dfc^{q+1}, d^2fc^{q+1}, dc^{q+2}, \ldots, dc^{2q+1}, dc^{2q+3}, \ldots, dc^{3q+1},$$

$$dc^i/b_j\ \forall j \in [q],\ \forall i \in \{q+2, \ldots, 3q+1\} \setminus \{2q+2\},\ d^2fb_jc^{q+1}\ \forall j \in [q],$$

$$d^2fc^ib_{j'}/b_j\ \forall i \in \{q+2, \ldots, 2q+1\},\ \forall j, j' \in [q] \text{ s.t. } j \neq j'\}.$$

We now must check if any of these are in $E(\mathcal{M})$, which is the set of pairwise products of things in $\mathcal{M}$. We observe that in $\mathcal{M}$, every occurrence of d is accompanied by f and f^{-1} never appears: so it is impossible for $E(\mathcal{M})$ to contain any elements which have higher powers of d than f. This rules out all of the elements in $dc^{q+1}\mathcal{M}$ above except for dfc^{q+1}.

To also rule out dfc^{q+1}, we consider all the possible ways it might be formed as a product of two elements of $\mathcal{M}$. Since this term contains f, one of the two factors in $\mathcal{M}$ must be a term containing f. We note that neither f or df can be one of the factors, since $dc^{q+1}, c^{q+1} \notin \mathcal{M}$. We note that an element of the form dfb_j cannot be one of the two factors, since $c^{q+1}/b_j \notin \mathcal{M}$. Similarly, an element of the form $dfc^ib_{j'}/b_j$ cannot be one of the two factors, since $c^{q+1-i}b_j/b_{j'} \notin \mathcal{M}$. We have thus dismissed all ways the f could be obtained, and we conclude that $dfc^{q+1} \notin E(\mathcal{M})$. □

We now proceed similarly to the proof strategy in [8, 21] to establish the following theorem:

Theorem 22. *For any adversary $\mathcal{A}$ that makes Q queries to the oracles computing the group operations in G, G_T and the bilinear map $e : G \times G \to G_T$, the advantage of $\mathcal{A}$ against the source group q-parallel BDHE assumption in the generic group model is at most $O\left(\frac{Q^2 q}{p}\right)$.*

Proof. In the real experiment, the variables $c, d, f, b_1, \ldots, b_q$ are first set randomly, and then the adversary is given handles for the group elements corresponding to the terms given out in the assumption and to T_β. $\mathcal{A}$ can then issue oracle queries for handles corresponding to products, divisions, or pairings of elements that it already has handles for. We define a new experiment in which the variables are never concretely instantiated, but instead the handles correspond to formal functions. Two elements are now given the same handle if and only if they are equal as formal functions over the variables.

This differs only from the real experiment when it happens that two formal functions are unequal, but happen to coincide for the particular choice of the variable settings. All of the functions created in the course of the experiment are linear combinations of rational functions whose numerators are polynomials of degree $\leq 4q$ and whose denominators are always among $\{b_1, \ldots, b_q\}$. Multiplying such a rational function by the product $b_1 \cdots b_q$ will thus yield a polynomial of degree $\leq 5q$. The probability that two formal polynomials of degree $\leq 5q$ are unequal but happen to be equal for a random setting of the variables modulo p is upper bounded by $\frac{5q}{p}$ by the Schwartz-Zippel Lemma. Thus, the probability of the particular setting of the variables causing a difference between the two experiments is $O\left(\frac{Q^2 q}{p}\right)$.

Now, in this new experiment where formal variables are maintained, the only way for the adversary to have any advantage in guessing β is for it to generate a two formal functions during the course of the experiment that are the same only when β takes a particular value. In our case, this must mean that the attacker generates two functions that are equal only when the challenge term is $g^{dc^{q+1}}$. We can rearrange terms and express this equality as $dc^{q+1}h_1 = h_2$ for functions h_1 and h_2 satisfying the following constraints: h_1 must be non-zero and generated in G (so without using pairings). Thus, h_1 must be a linear combination of elements of $\mathcal{M} \cup \{dc^{q+1}\}$. Also, h_2 must be a linear combination of elements in $E(\mathcal{M})$ (note that this set includes $\mathcal{M}$ since $1 \in \mathcal{M}$). But this means that for some $M \in \mathcal{M} \cup \{dc^{q+1}\}$, $dc^{q+1}M$ is dependent on $E(\mathcal{M}) \cup dc^{q+1}(\mathcal{M} \setminus M)$, contradicting Lemma 21. □