Mitigating the One-Use Restriction in Attribute-Based Encryption

Lucas Kowalczyk * Jiahui Liu† Kailash Meiyappan‡ Tal Malkin§

Abstract

We present a key-policy attribute-based encryption scheme that is adaptively secure under a static assumption and is not directly affected by an attribute "one-use restriction." Our construction improves upon the only other such scheme (Takashima '17) by mitigating its downside of a ciphertext size that is dependent on the maximum size of any supported attribute set.

1 Introduction

Attribute-based encryption (ABE) is a type of public key encryption which allows for fine-grained access control to encrypted data. In Key-Policy ABE, ciphertexts are associated with attributes, and secret-keys are associated with Boolean access policies that take in a set of attributes and return True if the key is capable of decrypting ciphertexts associated with that set and return False otherwise. Security guarantees that (potentially colluding) users without an authorized key should not be able to learn anything about an encrypted message. (A dual variant called Ciphertext-Policy ABE swaps the roles of attributes and access policies to be associated with the secret keys and ciphertexts respectively).

One way to make security proofs for ABE more attainable is to consider restricted notions of security. For KP-ABE, the notion of *selective security* requires the adversary to commit to a target set of attributes for the challenge ciphertext that will be attacked at the start of the security game. The earliest constructions of ABE using bilinear groups were proven secure in this model [GPSW06, Wat11].The notion of *semi-adaptive security* [JW14] requires the adversary to commit to a target set of attributes, but allows the adversary to see the public parameters first. These notions are obviously not realistic attack scenarios, so a KP-ABE scheme would ideally satisfy the notion of *adaptive security* (full security), where the challenge attribute set can be chosen adaptively (in response to public parameters and any amount of secret keys received). The first construction of ABE achieving adaptive security appeared in [LOS+10], employing the dual system encryption methodology [Wat09] in its security reduction.

Another way to make proving security of ABE schemes easier is to reduce security to parameterized assumptions like *q-type assumptions*, where the size of the elements included in the assumption's challenge grows with some property of the adversary. q-type assumptions were used in the ABE constructions of [Wat11, LW12] to prove security. However, the security of dynamic assumptions like q-type assumptions is not well-understood, and the assumptions are often closely related to the scheme in which they are used. For example, the assumption may include a number of group

*Columbia University luke@cs.columbia.edu
†Columbia University jl4161@columbia.edu
‡Columbia University kkm2142@columbia.edu
§Columbia University tal@cs.columbia.edu

elements that scales with the number of queries made by the adversary in the security proof. Further, it is known that many q-type assumptions become stronger as q grows [Che06], so we would ideally like to reduce security of ABE constructions to better understood assumptions of a static size, like the Decisional Linear Assumption (DLIN) or the Symmetric External Diffie-Hellman Assumption (SXDH).

A natural class of access policies one would like to be able to support in an ABE construction is that of general Boolean formulas. Unfortunately, it has proven tremendously difficult to construct efficient ABE for general Boolean formulas with adaptive security under static assumptions. All constructions except for [Tak17] suffer from a "one-use restriction." That is, they only natively support read-once Boolean formulas, or formulas where attributes are used at most once in inputs. One way to extend such constructions to support formulas that use attributes more than once (say, k times) is to use k copies of new "meta-attributes" that stand for each use of the original attribute, and are handled as a group [LOS+10]. The downside of this approach is that it destroys the compactness of the construction – for KP-ABE, the size of the ciphertexts no longer depends on just the attribute set of the ciphertext, but also on the complexity of the formulas that the scheme supports (namely, the ciphertexts grow linearly with the maximum number of attribute uses in any formula supported). Ciphertexts associated with n' attributes in a scheme like [LOS+10] where policies can reuse attributes at most k times are of size $O(n' \cdot k)$.

Takashima presented the first KP-ABE scheme (proven adaptively secure from static assumptions) with ciphertexts that do not grow directly with the number of attribute uses [Tak17], but unfortunately, the construction still has a dependence on the set of allowed policies. Specifically, ciphertexts are of size $O(n + r)$, where n is the maximum size of any supported attribute set and r is the maximum number of columns in any policy matrix supported (this is the policy dependency). For (fan-in 2) Boolean formulas, standard techniques [LW11a] to translate the formula into a policy matrix result in r being equal to the number of AND gates in the formula. Additionally, this dependence on n, the maximum size of any supported attribute set rather than only the attribute set of the relevant ciphetext is undesirable, since one can imagine the size of each ciphertext's associated attribute set varying wildly from the worst-case maximum-sized set supported by the system. In fact, it is unclear whether $O(n + r)$-size ciphertexts are ever an asymptotic improvement over the $O(n' \cdot k)$-size ciphertexts of all other known ABE schemes proved adaptively secure under static assumptions.

1.1 Our Result

In this work, we describe a KP-ABE construction that mitigates one of the two undesirable dependencies of [Tak17], featuring ciphertexts of size $O(n' + r)$ instead of $O(n + r)$ (while remaining adaptively secure from a static assumption: the Symmetric Diffie-Hellman Assumption (SXDH) and allowing the reuse of attributes in its monotone span program policies). This significant improvement allows us to rigorously argue that there exist classes of access policies for which our construction enjoys an asymptotic improvement over the state of the art. We note that our construction is for the small-universe setting, where attributes come from a polynomial (in the security parameter) sized universe that is fixed upon setup, whereas the construction of [Tak17] supports an attribute universe that may be exponentially large. This allows us to focus on the techniques required to asymptotically improve the ciphertext size. Our scheme is likely translatable to accommodate a large attribute universe without sacrificing asymptotic efficiency in a straightforward way using the compiler technique of [CGKW18], but we leave this for future work.

Our construction avoids a dependence on k, the multiplicity of attribute-reuse in supported policies, but retains the dependence on r, the number of columns in supported policy matrices. We

reference	$\|sk\|$	$\|ct\|$	assumption
[LOS+10]	$O(\|f\|)$	$O(n' \cdot k)$	DLIN
[OT12]	$O(\|f\|)$	$O(n' \cdot k)$	DLIN
[CGW15]	$O(\|f\|)$	$O(n' \cdot k)$	k-LIN
[CGKW18]	$O(\|f\|)$	$O(n' \cdot k)$	SXDH
[Tak17]	$O(\|f\|)$	$O(n + r)$	DLIN
Ours	$O(\|f\|)$	$O(n' + r)$	SXDH

Figure 1: Summary of several KP-ABE schemes proven adaptively secure under static assumptions for monotone span programs. Here, n' is the number of attributes associated to the ciphertext, n is the maximum size of any supported attribute set, r is the maximum number of columns in any policy matrix, and k is the maximum number of attribute reuses in any policy (except in the name for the "k-LIN" assumption, which is unrelated and an unfortunate overloading).

view reducing this last dependence to achieve truly compact adaptively secure ABE from a static assumption as an interesting open problem.

1.2 Comparing Perfomance

Figure 1 contains a comparison of several KP-ABE schemes proven adaptively secure under static assumptions for monotone span programs.

An obvious question in comparing our construction to the state-of-the-art is: how does r compare to k? Is $n' + r$ ever better than $n' \cdot k$? It is easy to come up with individual formulas where this is the case, but it's not obvious that such a formula can't always be "compressed" to an equivalent formula that has less attribute-reuse. In general, circuit/formula minimization questions like this are difficult to answer.

Fortunately, we can make a simple counting argument to show that indeed there are classes of functions which cannot be expressed using Boolean formulas with much smaller maximum attribute reuse than the maximum number of AND gates within the class. To see this, consider some subset of x attributes in the attribute universe. There are 2^{2^x} Boolean functions on these attributes, and we can express each function as a DNF in the naive way as a formula that uses at most $O(2^x)$ AND gates. So, for this class of functions, $r = O(2^x)$.

However, counting the number of different Boolean formulas that could attempt to realize these functions using a maximum k reuses of any attribute shows that at least $k = \Omega(2^x)$ attribute-reuses are required to realize all of the functions in this class. In this case, we see our construction enjoys a multiplicative to additive improvement (from $n' \cdot \Omega(2^x)$ to $n' + O(2^x)$).

1.3 Technical Details

Our construction can be seen as combining the best of both worlds between the construction of [Tak17], which is the first to be independent of the number of attribute-reuses, and the lineage of [GPSW06, LOS+10, KL15, CGKW18], which enjoys ciphertexts that are independent of the size of the attribute universe (they depend only on the number of attributes actually associated with the ciphertext).

Specifically, all of these schemes are based on linear secret sharing and are built using bilinear groups. Given a matrix M representing a monotone span program, linear secret shares of α are constructed by choosing randomness r_i, then computing $M \cdot (\alpha, r_2, ..., r_m)$ to obtain a vector of shares $\vec{\lambda}$. [GPSW06, LOS+10, KL15, CGKW18] embed these shares into their constructions' secret keys, where they are hidden by attribute-randomness that can only be removed using corresponding

$$\{g^{\lambda_j + a_{\rho(j)} y_j}, \quad g^{y_j}\}_{j \in M}$$

Figure 2: Example secret key

elements from a ciphertext. See Figure 2 for an example. A crucial step of the dual-system proof [Wat09] of adaptive security occurs when secret shares in the dual "semi-functional" space of a secret key are changed from sharing 0 to sharing a random element α' (in [LW12], this is the change from "nominal semi-functional" to "temporary semi-functional"). This is the step of the proof that uses the fact that the keys requested by an adversary are not allowed to decrypt the challenge ciphertext, to argue that there exists different randomness r'_i where a sharing of 0 using the r_i randomness looks identically distributed to a sharing of random α' using the r'_i randomness, as long as the only shares seen are not allowed to reconstruct the secret. Crucially, the alternative randomness r'_i is not defined until the challenge ciphertext is requested (as the challenge ciphertext defines which shares in the key are allowed to be seen). The constructions in the [LOS+10] lineage therefore require that the change in the secret shares in their keys be information theoretic (so they can be implicitly changed upon challenge ciphertext creation). This turns out to be the root of the one-time attribute use restriction (reusing attributes prevents this information-theoretic argument from working).

[Tak17] employs a technique of delayed share construction to get around this problem. Specifically, the construction does not construct a secret sharing $\{\lambda_j\}$ and embed them in the secret key, but instead keeps the components that generate λ_j ($\vec{M}_j$ and $(\alpha, r_2, ..., r_m)$) separate *until decryption*. The $\vec{M}_j$ portion is embedded in the key and the randomness $(\alpha, r_2, ..., r_m)$ is stored *in the ciphertext*. Decryption computes the dot product of these two components to implicitly construct λ_j that function in the same way as before. The advantage of this approach is that the randomness used in the secret shares is not needed until the challenge ciphertext is requested, so computational assumptions can be used to side-step the one-time attribute use restriction that comes with information-theoretic changes.

Like [LOS+10], [Tak17] also masks secret key components, making them only available to ciphertexts associated with the appropriate attributes. However [Tak17] does this via a somewhat blunt tool: namely, its secret keys contain a vector $\vec{y}$ which can encode orthogonality relationships with any subset of the attributes associated with a ciphertext and whose length is as large as the maximum attribute set supported by the system.

In contrast, the "share encapsulation" in [LOS+10] demonstrated in Figure 2 can be thought of as using a vector of dimension 2 to perform the same job. $(a_{\rho(j)} y_j, y_j)$ is being used to hide the share λ_j and share retrieval will be allowed only give a ciphertext with an "orthogonal" vector: $(s, -s a_{\rho(j)})$. Our construction can be seen as essentially replacing [Tak17]'s vector $\vec{y}$ with constant-dimensioned vectors like this, resulting in a ciphertext dependent only on the number of attributes associated with it, just like all previous schemes. Doing so makes the dual-system hybrid more delicate, as it requires careful management of rerandomization across the now greatly reduced dimensions. More detail about how the proof structure handles this is provided in Section 5.2.

1.4 Related Work

Additional work on ABE in the bilinear setting includes various constructions of KP-ABE and CP-ABE schemes (e.g. [BSW07, OSW07, GJPS08, JW14]), schemes supporting multiple authorities (e.g. [Cha07, CC09, OT13, LW11a]), and schemes supporting large attribute universes (e.g. [LW11b, OT12, RW13, Att14, KL15, Att16, BV16, GKW6b, AC17, CGKW18]).

The construction of [GVW13] supports circuit access policies rather than monotone span

programs or Boolean formulas, which makes it more expressive than any known bilinear scheme. It was proven selectively secure under the standard LWE assumption. The construction of [BV16] later extended this to semi-adaptive security for circuit access policies from LWE. Proving full adaptive security for a ABE scheme supporting circuits from LWE or an assumption on bilinear maps is an interesting open problem.

Circuit policies are supported by the construction in [GGH+13] based on multilinear maps. This scheme is proven selectively secure, under a particular computational hardness assumption for multilinear groups. The multilinear scheme in [GGHZ14] achieves adaptive security, relying on computational hardness assumptions in multilinear groups.

2 Preliminaries

We will write $a \leftarrow \mathbb{Z}_p$ to denote choosing a uniformly at random from set $\mathbb{Z}_p$ and will abuse notation to use $j \in M$ as a subscript to denote each index j of the rows M_j of matrix M.

2.1 Prime Order Bilinear Groups

We construct our system in prime order asymmetric bilinear groups. We let $\mathcal{G}$ denote a group generator - an algorithm which takes a security parameter λ as input and outputs (p, G, H, G_T, e), where p is a prime, G, H and G_T are cyclic groups of order p, and $e : G \times H \rightarrow G_T$ is a map with the following properties:

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

We refer to G, H as the *source groups* and G_T as the *target group*. We assume that the group operations in G, H and G_T and the map e are computable in polynomial time with respect to λ, and the group descriptions of G, H and G_T include a generator of each group.

2.2 Dual Pairing Vector Spaces

We will employ the concept of dual pairing vector spaces from [OT08, OT09], where we'll denote choosing random dual orthogonal bases as: $(\mathbb{B}, \mathbb{B}^*) \leftarrow \mathit{Dual}(\mathbb{Z}_p^n)$. Such bases are collections of linearly independent vectors chosen at random up to orthogonality constraints ($\vec{b}_i \cdot \vec{b}_i^* = 1, \vec{b}_i \cdot \vec{b}_j^* = 0$ for $i \neq j$). For example, one can implement $\mathit{Dual}(\mathbb{Z}_p^n$ by choosing a random invertible matrix B, setting $\mathbb{B} := B$ which then defines $\mathbb{B}^*$ as $\mathbb{B}^* := (B^{-1})^T$. Note that the dual basis generation procedure satisfies the property that, if R is an invertible matrix, then $(\mathbb{B}, \mathbb{B}^*)$ and $(R \cdot \mathbb{B}, (R^{-1})^T \cdot \mathbb{B}^*)$ are distributed identically when $(\mathbb{B}, \mathbb{B}^*) \leftarrow \mathit{Dual}(\mathbb{Z}_p^n)$. We will use this fact in our security proof to introduce new randomness into free dimensions of the construction as well as to embed computational assumptions. Finally, we will write $g^{\vec{v}}$ to denote the vector of group elements $(g^{v_1}, ..., g^{v_n})$, and will use the notation: $(x_1, ..., x_n)_{\mathbb{B}}$ to denote $g^{x_1 \vec{b}_1} \cdot ... \cdot g^{x_n \vec{b}_n}$.

2.3 Complexity Assumptions

The security of our system will be reduced to the Symmetric External Diffie-Hellman assumption (SXDH). We use the notation $x \leftarrow S$ to express that element x is chosen uniformly at random from the finite set S.

Symmetric External Diffie-Hellman Assumption (SXDH) The SXDH problem in G is stated as follows: given an asymmetric bilinear group (G, H) of prime order p with respective generators g, h, and given g^a, g^b and $T = g^{ab+r^*} \in G$ where $a, b \leftarrow \mathbb{Z}_p$ and either $r^* = 0$ or $r \leftarrow \mathbb{Z}_p$, output "yes" if r is a random element of $\mathbb{Z}_p$ and "no" otherwise. The SXDH problem in H is stated symmetrically, swapping the role of G and H.

Definition 1. *SXDH Assumption in (G, H): no polynomial time algorithm can achieve non-negligible advantage in deciding the SXDH problem in G or the SXDH problem in H.*

2.4 Background for ABE

We now give required background material on Linear Secret Sharing Schemes, the formal definition of a KP-ABE scheme, and the security definition we will use.

2.4.1 Monotone Span Programs / Linear Secret Sharing Schemes

Our construction uses linear secret-sharing schemes (LSSS) to realize monotone span program access structures [MW93]. We use the following definition (adapted from [Bei96]). In the context of ABE, attributes will play the role of parties and will be represented as indexes $i \in [|\mathcal{U}|]$ for a fixed universe $\mathcal{U}$.

Definition 2. *(Linear Secret-Sharing Schemes (LSSS)) A secret sharing scheme Π over a set of attributes is called* linear *(over $\mathbb{Z}_p$) if*

1. *The shares belonging to all attributes form a vector over $\mathbb{Z}_p$.*
2. *There exists an $\ell \times n$ matrix Λ called the share-generating matrix for Π. The matrix Λ has ℓ rows and n columns. For all $j = 1, \ldots, \ell$, the j^{th} row of Λ is labeled by an attribute $i = \rho(j)$ (ρ is a mapping that maintains the relationship between matrix rows and attributes). 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 Λv is the vector of ℓ shares of the secret s according to Π. The share $(\Lambda v)_j = \lambda_j$ belongs to attribute $i = \rho(j)$.*

We note the *linear reconstruction* property: we suppose that Π is an LSSS. We let S denote an authorized set. Then there is a subset $S^* \subseteq S$ such that the vector $(1, 0, \ldots, 0)$ is in the span of rows of Λ indexed by S^*, and there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in S^*}$ such that, for any valid shares $\{\lambda_i\}$ of a secret s according to Π, we have: $\sum_{i \in S^*} \omega_i \lambda_i = s$. These constants $\{\omega_i\}$ can be found in time polynomial in the size of the share-generating matrix Λ [Bei96]. For unauthorized sets, no such S^*, $\{\omega_i\}$ exist.

For any set S of unauthorized shares, since the vector $(1, 0, ..., 0)$ is not in the span of rows indexed by S, then there is some vector $\vec{w}$ that is orthogonal to all of the rows of Λ indexed by S but is not orthogonal to $(1, 0, ..., 0)$. By scaling this vector, we can maintain these orthogonality relationships and force the first coordinate w_1 to be 1. Our proof of security will use the existence of this vector.

2.4.2 KP-ABE Definition

A key-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$(\text{PP}, m, S) \rightarrow \text{CT}$ The encryption algorithm takes in the public parameters PP, the message m, and a set of attributes S. It will output a ciphertext CT. We assume that S is implicitly included in CT.

KeyGen$(\text{MSK}, \text{PP}, \mathbb{A}) \rightarrow \text{SK}$ The key generation algorithm takes in the master secret key MSK, the public parameters PP, and an access structure $\mathbb{A}$ over the universe of attributes. It outputs a private key SK which can be used to decrypt ciphertexts encrypted under a set of attributes which satisfies $\mathbb{A}$. We assume that $\mathbb{A}$ 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 encrypted under a set of attributes S, and a private key SK for an access structure $\mathbb{A}$. If the set of attributes of the ciphertext satisfies the access structure of the private key, it outputs the message m.

2.4.3 Adaptive Security for KP-ABE Systems

We define adaptive security for KP-ABE Systems in terms of the following game:

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

Phase 1 The attacker queries the challenger for private keys corresponding to access structures.

Challenge The attacker declares two equal length messages M_0, M_1 and a set of attributes $A \subseteq \mathcal{U}$ where $\mathcal{U}$ is the attribute universe such that A does not satisfy the access structure of any of the keys requested in Phase 1. The challenger flips a random coin $\beta \in \{0, 1\}$, encrypts M_β under S to yield ciphertext CT_β and gives CT_β to the attacker.

Phase 2 The attacker queries the challenger for private keys corresponding to access structures that are not satisfied by S.

Guess The attacker outputs a guess β'.

Definition 3. *The advantage of an attacker $\mathcal{A}$ in this game is defined as* $Adv_{\mathcal{A}}^{KP-ABE}(\lambda) = \Pr[\beta = \beta'] - \frac{1}{2}$.

Definition 4. *A key-policy attribute based encryption scheme is adaptively secure if no polynomial time algorithm can achieve a non-negligible advantage in the above security game.*

3 Construction

Setup$(\lambda, \mathcal{U}) \rightarrow PP, MSK$ The setup algorithm chooses an asymmetric bilinear group $\mathcal{G}(\lambda) \rightarrow (p, G, H, G_T, e)$. It then chooses random generators $g \in G, h \in H$. For $i \in [k]$ where $k = |\mathcal{U}|$ it

chooses values $a_i \leftarrow \mathbb{Z}_p$. It then generates random dual orthonormal sets:

$$(\mathbb{D}, \mathbb{D}^*) \leftarrow Dual(\mathbb{Z}_p^6)$$
$$(\mathbb{B}, \mathbb{B}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)})$$
$$(\mathbb{A}_i, \mathbb{A}_i^*) \leftarrow Dual(\mathbb{Z}_p^3) \text{ for } i \in [k]$$

The public parameters PP are:

$$e(g,h)$$
$$(\vec{e}_1)_{\mathbb{D}^*}, (\vec{e}_2)_{\mathbb{D}^*}$$
$$\{(\vec{e}_i)_{\mathbb{B}^*}\}_{i\in[r+1]}$$
$$\{(a_i, 0, 0)_{\mathbb{A}_i^*}\}_{i\in[k]}$$

The MSK is:

$$(\vec{e}_1)_{\mathbb{D}}, (\vec{e}_2)_{\mathbb{D}}$$
$$\{(\vec{e}_i)_{\mathbb{B}}\}_{i\in r+1}$$
$$\{(1, 0, 0)_{\mathbb{A}_i}\}_{i\in[k]}$$

Such a construction is equipped to create keys for access policies which include attributes $i \in \mathcal{U}$.

Encrypt$(m, S, PP) \rightarrow CT$ The encryption algorithm draws $\alpha, \Delta, s, z_i \leftarrow \mathbb{Z}_p$ (for $i \in [r]$) and forms the ciphertext as:

$$CT_S = (C_0, C_1, C_2, \{C_{3,i}\}_{i\in S})$$

where

$$C_0 := m \cdot e(g,h)^{\alpha}$$
$$C_1 := (\alpha, -\Delta, \vec{0}^2, \vec{0}^2)_{\mathbb{D}^*}$$
$$C_2 := (\Delta, z_2, ..., z_r, s, \vec{0}^{r+1}, \vec{0}^{r+1})_{\mathbb{B}^*}$$
$$C_{3,i} := (sa_i, 0, 0)_{\mathbb{A}_i^*}$$

(This implicitly includes S)

KeyGen$(MSK, M, PP) \rightarrow SK$ The key generation algorithm takes in the public parameters, master secret key, and LSSS access matrix M. It chooses a random exponent $x \leftarrow \mathbb{Z}_p$. For each row j (associated with attribute $\rho(j)$) in the policy matrix M, it chooses exponent $y_j \leftarrow \mathbb{Z}_p$ and outputs the secret key:

$$SK_M = (K_1, \{K_{2,j}, K_{3,j}\}_{j\in M})$$

where:

$$K_1 := (1, x, \vec{0}^2, \vec{0}^2)_{\mathbb{D}}$$
$$K_{2,j} := (\text{——}x\vec{M}_j\text{——}, a_{\rho(j)}y_j, \vec{0}^{r+1}, \vec{0}^{r+1})_{\mathbb{B}}$$
$$K_{3,j} := (-y_j, 0, 0)_{\mathbb{A}_{\rho(j)}}$$

Decrypt$(CT_S, SK_M, PP) \to m$ Given ciphertext $CT_S = (C_0, C_1, C_2, \{C_{3,i}\}_{i \in S})$ and secret key $SK_M = (K_1, \{K_{2,j}, K_{3,j}\}_{j \in M})$, if S satisfies M, then there is a set S^* of policy row indices such that $j \in S^* \implies \rho(j) \in S$ and there exist efficiently computable constants ω_j such that $\sum_{j \in S^*} \omega_j M_j \cdot \vec{z} = \Delta$ (recall section 2.4.1). The decryption algorithm computes these ω_j and then computes:

$$B = \prod_{j \in S^*} e(C_2, K_{2,j})^{\omega_j} \cdot e(C_{3,\rho(j)}, K_{3,j})^{\omega_j}$$

$$D = e(C_1, K_1)$$

and finally, computes and outputs:

$$\frac{C_0}{B \cdot D} = m$$

4 Correctness

This scheme satisfies correctness since:

$$\begin{aligned}
B &= \prod_{j \in S^*} e(C_2, K_{2,j})^{\omega_j} \cdot e(C_{3,\rho(j)}, K_{3,j})^{\omega_j} \\
&= \prod_{j \in S^*} e\begin{pmatrix} (\Delta, z_2, ..., z_r, & s, & \vec{0}^{r+1}, \vec{0}^{r+1})_{\mathbb{B}^*}, \\ (\text{——} x\vec{M}_j \text{——}, & a_{\rho(j)} y_j, & \vec{0}^{r+1}, \vec{0}^{r+1})_{\mathbb{B}} \end{pmatrix}^{\omega_j} \cdot e\begin{pmatrix} (& sa_{\rho(j)}, 0, 0)_{\mathbb{A}^*_{\rho(j)}}, \\ & (-y_j, 0, 0)_{\mathbb{A}_{\rho(j)}} \end{pmatrix}^{\omega_j} \\
&= \prod_{j \in S^*} e(g,h)^{x\omega_j\lambda_j + s\omega_j a_{\rho(j)} y_j} \cdot e(g,h)^{-s\omega_j a_{\rho(j)} y_j} \\
&= e(g,h)^{x \sum_{j \in S^*} \omega_j \lambda_j} \\
&= e(g,h)^{x\Delta}
\end{aligned}$$

$$\begin{aligned}
D &= e(C_1, K_1) \\
&= e\begin{pmatrix} (\alpha, -\Delta, \vec{0}^2, \vec{0}^2)_{\mathbb{D}^*} \\ (1, \quad x, \ \vec{0}^2, \vec{0}^2)_{\mathbb{D}} \end{pmatrix} \\
&= e(g,h)^{\alpha - x\Delta}
\end{aligned}$$

and finally:

$$\begin{aligned}
\frac{C_0}{B \cdot D} &= \frac{m \cdot e(g,h)^{\alpha}}{e(g,h)^{x\Delta} \cdot e(g,h)^{\alpha - x\Delta}} \\
&= m
\end{aligned}$$

5 Proof of Security

In this section, we will prove our main theorem of security for our construction: Theorem 30. We first describe a litany of auxiliary ciphertext and secret key distributions that will be used the hybrid proof of this theorem.

5.1 Auxiliary Ciphertext and Secret Key Distributions

The security proof is a dual system hybrid over a sequence of games with different types of keys and ciphertexts. In these definitions, $\vec{w}$ is the vector described in Section 2.4.1, which is defined relative to the policy of the ith requested secret key in hybrid games superscripted by i (and is orthogonal to the rows of M in the ith key which are associated with attributes in the challenge ciphertext, while having a first coordinate of 1).

Note that we omit the message encapsulation component $m \cdot e(g,h)^{\alpha}$ from each semi-functional ciphertext description, since it is the same for all variants. Unless specifically mentioned, the distributions of all elements remain‘ unchanged from the last time they were defined.

Type$_0$ Type 0 keys and ciphertexts are simply the keys and ciphertexts of the regular construction.

$$CT_0 := \begin{bmatrix} \alpha & -\Delta \\ 0 & 0 \\ 0 & 0 \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} \Delta & z_2 & \dots & z_r & s \\ 0 & \dots & \dots & \dots & 0 \\ 0 & \dots & \dots & \dots & 0 \end{bmatrix}_{\mathbb{B}^*} \left\{ \begin{bmatrix} -sa_i \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i \in S}$$

$$SK_0 := \begin{bmatrix} 1 & x \\ 0 & 0 \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & \dots & xM_{r,j} & a_{\rho(j)} y_j \\ 0 & 0 & \dots & \dots & 0 \\ 0 & 0 & \dots & \dots & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M}$$

Type$_1$

$$CT_1 := \begin{bmatrix} \alpha & -\Delta \\ 0 & -\Delta' \\ 0 & -\Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} \Delta & z_2 & \dots & z_r & s \\ — & — & \Delta' \vec{w} & — & s' \\ — & — & \Delta' \vec{w} & — & s'' \end{bmatrix}_{\mathbb{B}^*} \left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i \in S}$$

where $s', s'' \Delta' \leftarrow \mathbb{Z}_p$ (and $\vec{w}$ is defined relative to the policy of the ith requested secret key in hybrids superscripted by i).

$$SK_1 := SK_0$$

Type$_2$

$$CT_2 := CT_1$$

$$SK_2 := \begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & \dots & xM_{r,j} & a_{\rho(j)} y_j \\ x'M_{1,j} & x'M_{2,j} & \dots & x'M_{r,j} & 0 \\ 0 & 0 & \dots & \dots & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M}$$

where $x' \leftarrow \mathbb{Z}_p$.

Type$_{3,k}$ Keys of Type 3 to Type 11 are also subindexed by k, which runs from 1 to r: which is the number of rows in the key's policy matrix. In all types, the K_1 component of the secret key remains the same as in Type 3. The $(K_{2,j}, K_{3,j})$ components for $j < k$ are the same as in $SK_{11,k-1}$. These components for $j > k$ (up to r) are the same as in SK_2. The kth $(K_{2,k}, K_{3,k})$ components of the key of Type$_{3,k}$ are described below:

$$CT_{3,k} := CT_2$$

$$SK_{3,k} := \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ x'M_{1,k} + t_1 & x'M_{2,k} + t_2 & ... & x'M_{r,k} + t_r & 0 \\ -t_1 & -t_2 & ... & -t_r & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(k)}}$$

Type$_{4,k}$

$$CT_{4,k} := CT_2$$

$$SK_{4,k} := \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ 0 & 0 & ... & 0 & 0 \\ x'M_{1,k} & x'M_{2,k} & ... & x'M_{r,k} & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(k)}}$$

Type$_{5,k}$

$$CT_{5,k} := CT_2$$

$$SK_{5,k} := \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ 0 & 0 & ... & 0 & 0 \\ x'M_{1,k} & x'M_{2,k} & ... & x'M_{r,k} & a_{\rho(k)}y'_k \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ y'_k \end{bmatrix}_{\mathbb{A}_{\rho(k)}}$$

Type$_{6,k}$

$$CT_{6,k} := \begin{bmatrix} \alpha & -\Delta \\ 0 & -\Delta' \\ 0 & -\Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} \Delta & z_2 & \dots & z_r & s \\ — & — & \Delta'\vec{w} & — & s' \\ — & — & \Delta'\vec{w} & — & s'' \end{bmatrix}_{\mathbb{B}^*}$$

$$\left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i \neq \rho(k) \in S} \bigcup \left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''\tilde{a}_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i = \rho(k) \in S}$$

Note that the only difference from CT_2 is in the vector for attribute $i = \rho(k)$, if it is in the set S.

$$SK_{6,k} := \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ 0 & 0 & ... & ... & 0 \\ x'M_{1,k} & x'M_{2,k} & ... & x'M_{r,k} & \tilde{a}_{\rho(k)}y'_k \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ y'_k \end{bmatrix}_{\mathbb{A}_{\rho(k)}}$$

(recall that the $(K_{2,j}, K_{3,j})$ components for $j \neq k$ are the same as in Type$_{5,k}$)

Type$_{7,k}$

$$CT_{7,k} := CT_{6,k}$$

$$SK_{7,k} := \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ 0 & 0 & ... & ... & 0 \\ x^*M_{1,k} & x^*M_{2,k} & ... & x^*M_{r,k} & \tilde{a}_{\rho(k)}y'_k \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ y'_k \end{bmatrix}_{\mathbb{A}_{\rho(k)}}$$

where $x^* \leftarrow \mathbb{Z}_p$.

Type$_{8,k}$

$$CT_{8,k} := CT_2$$

$$SK_{8,k} := \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ 0 & 0 & ... & ... & 0 \\ x^*M_{1,k} & x^*M_{2,k} & ... & x^*M_{r,k} & a_{\rho(k)}y'_k \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ y'_k \end{bmatrix}_{\mathbb{A}_{\rho(k)}}$$

Type$_{9,k}$

$$CT_{9,k} := CT_2$$

$$SK_{9,k} := \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ 0 & 0 & ... & ... & 0 \\ x^*M_{1,k} & x^*M_{2,k} & ... & x^*M_{r,k} & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(k)}}$$

Type$_{10,k}$

$$CT_{10,k} := CT_2$$

$$SK_{10,k} := \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ -t_1 & -t_2 & ... & -t_r & 0 \\ x^*M_{1,k} + t_1 & x^*M_{2,k} + t_2 & ... & x^*M_{r,k} + t_r & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(k)}}$$

Type$_{11,k}$

$$CT_{11,k} := CT_2$$

$$SK_{11,k} := \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ x^*M_{1,k} & x^*M_{2,k} & ... & x^*M_{r,k} & 0 \\ 0 & 0 & ... & ... & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(k)}}$$

Type$_{12}$

$$CT_{12} := CT_2$$

$$SK_{12} := \begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ 0 & 0 & ... & 0 & 0 \\ 0 & 0 & ... & ... & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M}$$

Type$_{13}$

$$CT_{13} := \begin{bmatrix} \alpha & -\Delta \\ 0 & -\Delta' \\ 0 & -\Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} \Delta & z_2 & \dots & z_r & s \\ — & — & \Delta'\vec{e}_1 & — & s' \\ — & — & \Delta'\vec{e}_1 & — & s'' \end{bmatrix}_{\mathbb{B}^*} \left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i \in S}$$

$$SK_{13} := SK_{12}$$

Type$_{14}$

$$CT_{14} := CT_{13}$$

$$SK_{14} := \begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x'\lambda_j & 0 & ... & ... & 0 \\ 0 & 0 & ... & 0 & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M}$$

where $\lambda_j := \vec{M}_j \cdot (1, z'_2, ..., z'_r)$ for $z'_i \leftarrow \mathbb{Z}_p$.

Type$_{15_k}$ Type 15 keys and ciphertexts are subindexed by k which runs from attribute 1 to $|\mathcal{U}|$.

$$CT_{15_k} := CT_{14}$$

$$SK_{15_k} := \begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x'\lambda_j & 0 & ... & ... & \tilde{a}_{\rho(j)}y'_j \\ 0 & 0 & ... & 0 & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ y'_j \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M \text{ where } \rho(j) \leq k, \rho(j) \notin S}$$

$$\bigcup \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x'\lambda_j & 0 & ... & ... & 0 \\ 0 & 0 & ... & 0 & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M \text{ where } \rho(j) > k \text{ or } \rho(j) \in S}$$

Type$_{15/16_H}$ Type $15/16_H$ keys are a halfway point between Type $15_{|\mathcal{U}|}$ and Type $16_{|\mathcal{U}|}$ keys, where the $\tilde{a}_{\rho(j)}y_j$ are changed into freshly random $\tilde{a}'_j$ for each j where $\rho(j) \notin S$ by embedding SXDH challenge.

$$CT_{15_H} = CT_{16_H} := CT_{14}$$

$$SK_{15_H} := \begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x'\lambda_j & 0 & ... & ... & \tilde{a}'_j y'_j \\ 0 & 0 & ... & 0 & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ y'_j \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M \text{ where } \rho(j) \notin S}$$

$$\bigcup \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x'\lambda_j & 0 & ... & ... & 0 \\ 0 & 0 & ... & 0 & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M \text{ where } \rho(j) \in S}$$

$$SK_{16_H} := \begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x^*\lambda_j & 0 & ... & ... & \tilde{a}'_j y'_j \\ 0 & 0 & ... & 0 & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ y'_j \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M \text{ where } \rho(j) \notin S}$$

$$\bigcup \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x^*\lambda'_j & 0 & ... & ... & 0 \\ 0 & 0 & ... & 0 & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M \text{ where } \rho(j) \in S}$$

where $x^* \leftarrow \mathbb{Z}_p,\ \lambda'_j := \vec{M}_j \cdot (z'_1, z'_2, ..., z'_r)$ for $z'_i \leftarrow \mathbb{Z}_p$.

Type$_{16_k}$ Type 16 keys and ciphertexts are also subindexed by k which runs from attribute 1 to $|\mathcal{U}|$.

$$CT_{16_k} := CT_{15_k}$$

$$SK_{16_k} := \begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x^*\lambda'_j & 0 & ... & ... & \tilde{a}_{\rho(j)}y'_j \\ 0 & 0 & ... & 0 & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ y'_j \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M \text{ where } \rho(j) \leq k,\ \rho(j) \notin S}$$

$$\bigcup \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x^*\lambda_j & 0 & ... & ... & 0 \\ 0 & 0 & ... & 0 & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M \text{ where } \rho(j) > k \text{ or } \rho(j) \in S}$$

Type$_{17}$

$$CT_{17} := CT_{16_1}$$

$$SK_{17} := \begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ 0 & 0 & ... & ... & 0 \\ 0 & 0 & ... & 0 & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M}$$

Type$_{18}$

$$CT_{18} := \begin{bmatrix} \alpha^* & -\Delta \\ 0 & -\Delta' \\ 0 & -\Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} \Delta & z_2 & \dots & z_r & s \\ — & — & \Delta'\vec{e}_1 & — & s' \\ — & — & \Delta'\vec{e}_1 & — & s'' \end{bmatrix}_{\mathbb{B}^*} \left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}^*_i} \right\}_{i \in S}$$

where $\alpha^* \leftarrow \mathbb{Z}_p$.

$$SK_{18} := SK_{17}$$

5.2 Hybrid Structure

Our proof of security will consist of a hybrid sequence of games where the keys and challenge ciphertext are constructed according to various types. At a high level, the proof follows a typical dual-system hybrid structure, where the challenge ciphertext is first made "semi-functional," then the hybrid continues over the secret keys requested, transforming each key into a "semifunctional" variant which is useless to the attacker relative to the challenge (semifunctional) ciphertext.

There are two parts to the key hybrid: one that makes semifunctional keys which were requested before the challenge ciphertext and another that makes semifunctional keys which were requested after the challenge ciphertext. The high level reason for this difference is that for keys requested after the challenge ciphertext, the challenge attribute set is already known. This makes it easy to follow a standard selective security argument to make each key semifunctional (this sequence captioned "SF-Key After" in Figure 3). The harder part of the hybrid deals with making keys requested before the challenge ciphertext (and the challenge attribute set) is known. This is where we use the delayed randomness contained within our ciphertext as well as the fact that we allow the semifunctional ciphertext distributions to depend on the current key of the hybrid (this sequence

captioned "SF-Key Before" in Figure 3). This bifurcated approach to handling secret keys in a dual-system proof was first employed in [LW12] and later refined by [Att14, Att16]

A key step in our proof (and of [Tak17]) is Lemma 13, where each policy matrix row is isolated in turn against the ciphertext's $\vec{w}$ alternative randomness component and their dot product's distribution is used to argue that the row can be multiplied by an uncorrelated x^*. In [Tak17], this argument takes advantage of the inefficient $\vec{y}$ vector, but for us, we need to delicately thread just enough randomness through the single attribute element a_i hiding each row to accomplish the same feat. We accomplish this in Lemma 12.

Our hybrid starts with $Game_{real}$, the real security game. This has all keys and cipher texts of type Type_0. We then proceed through a sequence of games Game^i_j, which are indexed by each requested key i and subhybrid index ℓ. Let Q_1 be the number of key queries issued by the adversary before the challenge ciphertext, and let Q_2 be the number of key queries issued by the adversary after the challenge ciphertext.

Definition 5. *In* Game^ℓ_i *for* $i \leq 12$*, the first* $\ell - 1$ *keys are of type* Type_{12}*, the* ℓ*th key is of type* Type_i*, all remaining keys are of type* Type_0*, and the ciphertext is of type* Type_i*.*

Definition 6. *In* Game^ℓ_i *for* $i > 12$*, the first* Q_1 *keys are of type* Type_{12}*, the next* $\ell - 1$ *keys are of type* Type_{17}*, the next* $(Q_1 + \ell th)$ *key is of type* Type_i*, all remaining keys are of type* Type_0*, and the ciphertext is of type* Type_i*.*

Figure 3 shows the order of our hybrid games.

Game_{real} $\Rightarrow \text{Game}^1_1$

$\Rightarrow \text{Game}^1_2$
$\Rightarrow \text{Game}^1_{3,1} \Rightarrow ... \Rightarrow \text{Game}^1_{11,1}$
$\Rightarrow \text{Game}^1_{3,2} \Rightarrow ... \Rightarrow \text{Game}^1_{11,2}$
.....
$\Rightarrow \text{Game}^1_{3,r} \Rightarrow ... \Rightarrow \text{Game}^1_{11,r} \Rightarrow \text{Game}^1_{12}$
$\Rightarrow \text{Game}^2_1 \Rightarrow ... \Rightarrow \text{Game}^2_{12}$
.....
$\Rightarrow \text{Game}^i_1 \Rightarrow ... \Rightarrow \text{Game}^i_{12}$
.....
$\Rightarrow \text{Game}^{Q_1}_1 \Rightarrow ... \Rightarrow \text{Game}^{Q_1}_{12}$

$\Rightarrow \text{Game}^1_{13}$
$\Rightarrow \text{Game}^1_{14} \Rightarrow \text{Game}^1_{15,1} ... \Rightarrow \text{Game}^1_{15,|\mathcal{U}|} \Rightarrow \text{Game}^1_{15,H}$
$\Rightarrow \text{Game}^1_{16,H} \Rightarrow \text{Game}^1_{16,|\mathcal{U}|} ... \Rightarrow \text{Game}^1_{16,1} \Rightarrow \text{Game}^1_{17}$
...
$\Rightarrow \text{Game}^i_{14} \Rightarrow \text{Game}^i_{15,1} ... \Rightarrow \text{Game}^i_{15,|\mathcal{U}|} \Rightarrow \text{Game}^i_{15,H}$
$\Rightarrow \text{Game}^i_{16,H} \Rightarrow \text{Game}^i_{16,|\mathcal{U}|} ... \Rightarrow \text{Game}^i_{16,1} \Rightarrow \text{Game}^i_{17}$
...
$\Rightarrow \text{Game}^{Q_2}_{14} \Rightarrow \text{Game}^{Q_2}_{15,1} ... \Rightarrow \text{Game}^{Q_2}_{15,|\mathcal{U}|} \Rightarrow \text{Game}^{Q_2}_{15,H}$
$\Rightarrow \text{Game}^{Q_2}_{16,H} \Rightarrow \text{Game}^{Q_2}_{16,|\mathcal{U}|} ... \Rightarrow \text{Game}^{Q_2}_{16,1} \Rightarrow \text{Game}^{Q_2}_{17}$

$\Rightarrow \text{Game}^{Q_2}_{18}$

SF-CT, SF-Keys-Before, SF-Keys-After

Figure 3: Hybrid Game Sequence

5.3 Hybrid Indistinguishability Lemmas

The following is a sequence of hybrid indistinguishability lemmas that will be combined at the end to form our adaptive security theorem.

Lemma 7. *For any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_1$ such that*

$$|\mathrm{Adv}_{real}(\mathcal{A}, \lambda) - \mathrm{Adv}_1^1(\mathcal{A}, \lambda)| \leq \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_1, \lambda)$$

Proof. Given g, h, g^a, g^b and $T = g^{ab+r^*} \in G$ where either $r^* = 0$ or is a uniform random element of $\mathbb{Z}_p$, consider the following simulator $\mathcal{B}_1$ in the security game:

First, $\mathcal{B}_1$ generates orthonormal bases:

$$(\mathbb{J}, \mathbb{J}^*) \leftarrow Dual(\mathbb{Z}_p^6)$$
$$(\mathbb{F}, \mathbb{F}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)})$$
$$(\mathbb{K}_i, \mathbb{K}_i^*) \leftarrow Dual(\mathbb{Z}_p^3) \text{ for } i \in \mathcal{U}$$

The bases $(\mathbb{B}, \mathbb{B}^*)$ are implicitly defined as:

$$\vec{b}_i^* = \vec{f}_i^* + a\vec{f}_{r+1+i}^* + a\vec{f}_{2(r+1)+i}^* \text{ for } i \in [1, r]$$
$$\vec{b}_{r+1}^* = \vec{f}_{r+1}^* + a\tilde{s}'\vec{f}_{2(r+1)}^* + a\tilde{s}''\vec{f}_{3(r+1)}^*$$
$$\vec{b}_i^* = \vec{f}_i^* \text{ for all other } i$$

$$\vec{b}_{(r+1)+i} = \vec{f}_{(r+1)+i} - a\vec{f}_i \text{ for } i \in [1, r]$$
$$\vec{b}_{2(r+1)+i} = \vec{f}_{2(r+1)+i} - a\vec{f}_i \text{ for } i \in [1, r]$$
$$\vec{b}_{2(r+1)} = \vec{f}_{2(r+1)} - a\tilde{s}'\vec{f}_{(r+1)}$$
$$\vec{b}_{3(r+1)} = \vec{f}_{3(r+1)} - a\tilde{s}''\vec{f}_{(r+1)}$$
$$\vec{b}_i = \vec{f}_i \text{ for all other } i$$

where $\tilde{s}', \tilde{s}'' \leftarrow \mathbb{Z}_p$ are drawn by $\mathcal{B}_2$ (and later used in the challenge ciphertext construction). $(\mathbb{D}, \mathbb{D}^*)$ are implicitly defined as:

$$\vec{d}_2^* = \vec{j}_2^* + a\vec{j}_4^* + a\vec{j}_6^*$$
$$\vec{d}_i^* = \vec{j}_i^* \text{ for all other } i$$

$$\vec{d}_4 = \vec{j}_4 - a\vec{j}_2$$
$$\vec{d}_6 = \vec{j}_6 - a\vec{j}_2$$
$$\vec{d}_i = \vec{j}_i \text{ for all other } i$$

and $(\mathbb{A}_i, \mathbb{A}_i^*)$ are implicitly defined as:

$$\begin{aligned}
\vec{a}^*_{i,1} &= \vec{k}^*_{i,1} + a\tilde{s}'\vec{k}^*_{i,2} + a\tilde{s}''\vec{k}^*_{i,3} \\
\vec{a}^*_{i,2} &= \vec{k}^*_{i,2} \\
\vec{a}^*_{i,3} &= \vec{k}^*_{i,3}
\end{aligned}$$

$$\begin{aligned}
\vec{a}_{i,1} &= \vec{k}_{i,1} \\
\vec{a}_{i,2} &= \vec{k}_{i,2} - a\tilde{s}'\vec{k}_{i,1} \\
\vec{a}_{i,3} &= \vec{k}_{i,3} - a\tilde{s}''\vec{k}_{i,1}
\end{aligned}$$

The public parameters are then generated by drawing $a_i \leftarrow \mathbb{Z}_p$ for $i \in \mathcal{U}$ and forming:

$$\begin{aligned}
&e(g,h) \\
&(\vec{e}_1)_{\mathbb{D}^*}, (\vec{e}_2)_{\mathbb{D}^*} \\
&\{(\vec{e}_i)_{\mathbb{B}^*}\}_{i\in[r+1]} \\
&\{(a_i, 0, 0)_{\mathbb{A}_i^*}\}_{i\in[k]}
\end{aligned}$$

Note that the terms in the public parameter can be generated since all vectors in the $\mathbb{D}^*, \mathbb{B}^*, \mathbb{A}_i^*$ bases are known directly except for $\vec{d}^*_2$, $\vec{b}^*_i$ for $i \in [r+1]$, and all $\vec{a}_{i,1}$ and $\mathcal{B}_1$ can generate $\mathbf{d}^*_2$ by computing $g^{\vec{j}^*_2} \cdot (g^a)^{\vec{j}^*_4} = \mathbf{d}^*_2$, $\mathbf{b}^*_i$ for $i \in [r]$ by computing $g^{\vec{f}^*_i} \cdot (g^a)^{\vec{f}^*_{(r+1)+i}} \cdot (g^a)^{\vec{f}^*_{2(r+1)+i}} = \mathbf{b}^*_i$, and $\mathbf{a}^*_{i,1}$ by computing $g^{\vec{k}^*_{i,1}} \cdot (g^a)^{\tilde{s}'\vec{k}^*_{i,2}} \cdot (g^a)^{\tilde{s}''\vec{k}^*_{i,2}} = \mathbf{a}^*_{i,1}$.

Similarly, note that all vectors in $\mathbb{D}, \mathbb{B}, \mathbb{A}_i$ are known explicitly except for $\vec{d}_4, \vec{d}_6, \vec{b}_{(r+1)+i}, \vec{b}_{2(r+1)+i}$ for $i \in [r+1]$, and $\vec{a}_{i,2}, \vec{a}_{i,3}$ for $i \in \mathcal{U}$, but for normal secret keys of $Type_0$, those entries are all 0. So, $\mathcal{B}_1$ is able to generate appropriately distributed secret keys of $Type_0$ for all secret key requests.

For the challenge ciphertext request, let $\vec{w}$ be defined relative to the first requested secret key (this will be known upon ciphertext generation since this secret key is defined to be the first request before the challenge ciphertext request). Recall that the first coordinate of $\vec{w}$ is always 1. $\mathcal{B}_1$ draws $\alpha, \tilde{s}, \tilde{z}_i \leftarrow \mathbb{Z}_p$ for $i \in [r]$ and constructs:

$$m \cdot e(g,h)^\alpha, \begin{bmatrix} \alpha & -(b) \\ 0 & -(ab+r) \\ 0 & -(ab+r) \end{bmatrix}_{\mathbb{J}^*}$$

$$\begin{bmatrix} (b) & (b)+\tilde{z}_2 & \dots & (b)+\tilde{z}_r & (b)+\tilde{s} \\ (ab+r) & (ab+r)w_2+(a)\tilde{z}_2 & \dots & (ab+r)w_r+(a)\tilde{z}_r & (ab+r)\tilde{s}'+(a)\tilde{s}'\tilde{s} \\ (ab+r) & (ab+r)w_2+(a)\tilde{z}_2 & \dots & (ab+r)w_r+(a)\tilde{z}_r & (ab+r)\tilde{s}''+(a)\tilde{s}''\tilde{s} \end{bmatrix}_{\mathbb{F}^*}$$

$$\left\{ \begin{bmatrix} -(b)a_i - \tilde{s}a_i \\ -(ab+r)\tilde{s}'a_i - (a)\tilde{s}'\tilde{s}a_i \\ -(ab+r)\tilde{s}''a_i - (a)\tilde{s}'\tilde{s}a_i \end{bmatrix}_{\mathbb{K}_i^*} \right\}_{i\in S}$$

which in the right bases, is:

$$m \cdot e(g,h)^\alpha, \begin{bmatrix} \alpha & -b \\ 0 & -r \\ 0 & -r \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} b & b+\tilde{z}_2 & \dots & b+\tilde{z}_r & b+\tilde{s} \\ — & — & & r\vec{w} & — & r\tilde{s}' \\ — & — & & r\vec{w} & — & r\tilde{s}'' \end{bmatrix}_{\mathbb{B}^*} \left\{ \begin{bmatrix} -(b+\tilde{s})a_i \\ -r\tilde{s}'a_i \\ -r\tilde{s}''a_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i\in S}$$

which is a properly distributed ciphertext of $Type_0$ when $r = 0$ where $\Delta = b, s = b + \tilde{s}$ and $z_i = b + \tilde{z}_i$, and is a properly distributed ciphertext of $Type_1$ when $r \leftarrow \mathbb{Z}_p$ where $\Delta = b, \Delta' = r$, $s = b + \tilde{s}, s' = r\tilde{s}', s'' = r\tilde{s}''$, and $z_i = b + \tilde{z}_i$.

So, $\mathcal{B}_1$ is able to exactly simulate Game_{real} and Game_1^1 when its SXDH challenge has $r = 0$ and $r \leftarrow \mathbb{Z}_p$ respectively. It then follows that for any difference in $\mathcal{A}$'s advantage between the two games, $\mathcal{B}_1$ can use $\mathcal{A}$'s answers to achieve that same advantage in the SXDH game. □

Lemma 8. *For $1 \leq \ell \leq Q_1$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{2,\ell}$ such that*

$$|\text{Adv}_1^\ell(\mathcal{A}, \lambda) - \text{Adv}_2^\ell(\mathcal{A}, \lambda)| \leq \text{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{2,\ell}, \lambda)$$

Proof. Given g, h, h^a, h^b and $T = h^{ab+r^*} \in G$ where either $r^* = 0$ or is a uniform random element of $\mathbb{Z}_p$, consider the following simulator $\mathcal{B}_{2,\ell}$ in the security game:

First, $\mathcal{B}_{2,\ell}$ generates orthonormal bases:

$$(\mathbb{J}, \mathbb{J}^*) \leftarrow Dual(\mathbb{Z}_p^6)$$
$$(\mathbb{F}, \mathbb{F}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)})$$
$$(\mathbb{A}_i, \mathbb{A}_i^*) \leftarrow Dual(\mathbb{Z}_p^3) \text{ for } i \in \mathcal{U}$$

The bases $(\mathbb{B}, \mathbb{B}^*)$ are implicitly defined as:

$$\vec{b}_i = \vec{f}_i + a\vec{f}_{(r+1)+i} \text{ for } i \in [1, r]$$
$$\vec{b}_i = \vec{f}_i \text{ for all other } i$$

$$\vec{b}^*_{(r+1)+i} = \vec{f}^*_{(r+1)+i} - a\vec{f}^*_i \text{ for } i \in [1, r]$$
$$\vec{b}^*_i = \vec{f}^*_i \text{ for all other } i$$

$(\mathbb{D}, \mathbb{D}^*)$ are implicitly defined as:

$$\vec{d}_2 = \vec{j}_2 + a\vec{j}_4$$
$$\vec{d}_i = \vec{j}_i \text{ for all other } i$$

$$\vec{d}^*_4 = \vec{j}^*_4 - a\vec{j}^*_2$$
$$\vec{d}^*_i = \vec{j}^*_i \text{ for all other } i$$

The public parameters are then generated by drawing $a_i \leftarrow \mathbb{Z}_p$ for $i \in \mathcal{U}$ and forming:

$$e(g, h)$$
$$(\vec{e}_1)_{\mathbb{D}^*}, (\vec{e}_2)_{\mathbb{D}^*}$$
$$\{(\vec{e}_i)_{\mathbb{B}^*}\}_{i \in [r+1]}$$
$$\{(a_i, 0, 0)_{\mathbb{A}_i^*}\}_{i \in [k]}$$

Note that the terms in the public parameter can be generated since all vectors in the $\mathbb{D}^*, \mathbb{B}^*, \mathbb{A}_i^*$ with nonzero components in the public parameters are known directly.

Similarly, note that all vectors in $\mathbb{D}, \mathbb{B}, \mathbb{A}_i$ are known explicitly except for $\vec{d}_2$ and $\vec{b}_i$ for $i \in [r]$ and these can be constructed by taking $h^{\vec{j}_2} \cdot (h^a)^{\vec{j}_4} = \mathbf{d}_2$, and $h^{\vec{f}_i} \cdot (h^a)^{\vec{f}_{(r+1)+i}} = \mathbf{b}_i$ for $i \in [r]$. So,

$\mathcal{B}_{2,\ell}$ is able to generate appropriately distributed secret keys of $Type_{12}$ for all secret key requests before the ℓth key as well as keys of $Type_0$ for requests after the ℓth request.

For the challenge ciphertext request, let $\vec{w}$ be defined relative to the ℓth requested secret key (this will be known upon ciphertext generation since this secret key is defined to be one of the Q_1 requested before the challenge ciphertext). $\mathcal{B}_{2,\ell}$ draws $\alpha, \Delta', s, s', s'', \tilde{z}_i \leftarrow \mathbb{Z}_p$ for $i \in [r]$ and constructs:

$$m \cdot e(g,h)^{\alpha}, \begin{bmatrix} \alpha & 0 \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{J}^*}$$

$$\begin{bmatrix} 0 & \tilde{z}_2 & \dots & \tilde{z}_r & s \\ \Delta' & \Delta' w_2 & \dots & \Delta' w_r & s' \\ \Delta' & \Delta' w_2 & \dots & \Delta' w_r & s'' \end{bmatrix}_{\mathbb{F}^*}$$

$$\left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i \in S}$$

which in the right bases, is:

$$m \cdot e(g,h)^{\alpha}, \begin{bmatrix} \alpha & a\Delta' \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} a\Delta' & a\Delta' w_2 + \tilde{z}_2 & \dots & a\Delta' w_r + \tilde{z}_r & s \\ — & — & \Delta' \vec{w} & — & s' \\ — & — & \Delta' \vec{w} & — & s'' \end{bmatrix}_{\mathbb{B}^*} \left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i \in S}$$

which is a properly distributed ciphertext of $Type_1 = Type_2$ where $\Delta = a\Delta'$, and $z_i = a\Delta' + \tilde{z}_i$.

For the ℓth key request, $\mathcal{B}_{2,\ell}$ draws $y_j \leftarrow \mathbb{Z}_p$ for $j \in M$ and constructs:

$$\begin{bmatrix} 1 & (b) \\ 0 & (ab+r) \\ 0 & 0 \end{bmatrix}_{\mathbb{J}}$$

$$\left\{ \begin{bmatrix} (b)M_{1,j} & (b)M_{2,j} & ... & (b)M_{r,j} & a_{\rho(j)} y_j \\ (ab+r)M_{1,j} & (ab+r)M_{2,j} & ... & (ab+r)M_{r,j} & 0 \\ 0 & 0 & ... & ... & 0 \end{bmatrix}_{\mathbb{F}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M}$$

which in the right bases, is:

$$\begin{bmatrix} 1 & b \\ 0 & r \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} bM_{1,j} & bM_{2,j} & ... & bM_{r,j} & a_{\rho(j)} y_j \\ rM_{1,j} & rM_{2,j} & ... & rM_{r,j} & 0 \\ 0 & 0 & ... & ... & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M}$$

which is a properly distributed secret key of $Type_1$ when $r = 0$ where $x = b$ and is a properly distributed secret key of $Type_2$ when $r \leftarrow \mathbb{Z}_p$ where $x = b$ and $x' = r$.

So, $\mathcal{B}_{2,\ell}$ is able to exactly simulate Game_1^{ℓ} and Game_2^{ℓ} when its SXDH challenge has $r = 0$ and $r \leftarrow \mathbb{Z}_p$ respectively. It then follows that for any difference in $\mathcal{A}$'s advantage between the two games, $\mathcal{B}_{2,\ell}$ can use $\mathcal{A}$'s answers to achieve that same advantage in the SXDH game. $\square$

For the connection to the following lemma, note that Game_2^ℓ is equivalent to $\text{Game}_{11,0}^\ell$.

Lemma 9. *For $1 \leq \ell \leq Q_1$, for $1 \leq k \leq r$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{3,k,\ell}$ such that*

$$|\text{Adv}_{11,k-1}^\ell(\mathcal{A}, \lambda) - \text{Adv}_{3,k}^\ell(\mathcal{A}, \lambda)| \leq \text{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{3,k,\ell}, \lambda)$$

Proof. Given g, h, h^a, h^b and $T = h^{ab+r^*} \in G$ where either $r^* = 0$ or is a uniform random element of $\mathbb{Z}_p$, consider the following simulator $\mathcal{B}_{3,k,\ell}$ in the security game:

First, $\mathcal{B}_{3,k,\ell}$ generates orthonormal bases:

$$\begin{aligned}
&(\mathbb{D}, \mathbb{D}^*) \leftarrow Dual(\mathbb{Z}_p^6) \\
&(\mathbb{F}, \mathbb{F}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)}) \\
&(\mathbb{A}_i, \mathbb{A}_i^*) \leftarrow Dual(\mathbb{Z}_p^3) \text{ for } i \in \mathcal{U}
\end{aligned}$$

The bases $(\mathbb{B}, \mathbb{B}^*)$ are implicitly defined as:

$$\begin{aligned}
&\vec{b}_{r+1} = \vec{f}_{r+1} + \sum_{i=1}^{r} a\tilde{t}_i \vec{f}_{(r+1)+i} \\
&\vec{b}_{(r+1)+i} = \vec{f}_{(r+1)+i} + \vec{f}_{2(r+1)+i} \text{ for } i \in [1, r] \\
&\vec{b}_i = \vec{f}_i \text{ for all other } i
\end{aligned}$$

$$\begin{aligned}
&\vec{b}_{(r+1)+i}^* = \vec{f}_{(r+1)+i}^* - a\tilde{t}_i \vec{f}_{r+1}^* \text{ for } i \in [1, r] \\
&\vec{b}_{2(r+1)+i}^* = \vec{f}_{2(r+1)+i}^* - \vec{f}_{(r+1)+i}^* + a\tilde{t}_i \vec{f}_{(r+1)}^* \text{ for } i \in [1, r] \\
&\vec{b}_i^* = \vec{f}_i^* \text{ for all other } i
\end{aligned}$$

where $\mathcal{B}_{3,k,\ell}$ draws $\tilde{t}_i \leftarrow \mathbb{Z}_p$.

The public parameters are then generated by drawing $a_i \leftarrow \mathbb{Z}_p$ for $i \in \mathcal{U}$ and forming:

$$\begin{aligned}
&e(g, h) \\
&(\vec{e}_1)_{\mathbb{D}^*}, (\vec{e}_2)_{\mathbb{D}^*} \\
&\{(\vec{e}_i)_{\mathbb{B}^*}\}_{i \in [r+1]} \\
&\{(a_i, 0, 0)_{\mathbb{A}_i^*}\}_{i \in [k]}
\end{aligned}$$

Note that the terms in the public parameter can be generated since all vectors in the $\mathbb{D}^*, \mathbb{B}^*, \mathbb{A}_i^*$ with nonzero components in the public parameters are known directly.

Similarly, note that all vectors with nonzero components in $Type_0$ and $Type_{12}$ secret keys are known explicitly except for $\vec{b}_{r+1}$ and this can be constructed by taking $h^{\vec{f}_{r+1}} \cdot \prod_{i=1}^{r} (h^a)^{\tilde{t}_i \vec{f}_{(r+1)+i}} = \mathbf{b}_i$ for $i \in [r]$. So, $\mathcal{B}_{3,k,\ell}$ is able to generate appropriately distributed secret keys of $Type_{12}$ for all secret key requests before the ℓth key as well as keys of $Type_0$ for requests after the ℓth request.

For the challenge ciphertext request, let $\vec{w}$ be defined relative to the ℓth requested secret key (this will be known upon ciphertext generation since this secret key is defined to be one of the Q_1

requested before the challenge ciphertext). $\mathcal{B}_{3,k,\ell}$ draws $\alpha, \Delta, \Delta', s, s', s'', z_i \leftarrow \mathbb{Z}_p$ for $i \in [r]$ and constructs:

$$m \cdot e(g,h)^{\alpha}, \begin{bmatrix} \alpha & \Delta \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{D}^*}$$

$$\begin{bmatrix} \Delta & z_2 & \dots & z_r & s \\ 0 & 0 & \dots & 0 & s' \\ — & — & \Delta'\vec{w} & — & s'' \end{bmatrix}_{\mathbb{F}^*}$$

$$\left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i \in S}$$

which in the right bases, is:

$$m \cdot e(g,h)^{\alpha}, \begin{bmatrix} \alpha & \Delta \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} \Delta & z_2 & \dots & z_r & s \\ — & — & \Delta'\vec{w} & — & s' \\ — & — & \Delta'\vec{w} & — & s'' \end{bmatrix}_{\mathbb{B}^*} \left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i \in S}$$

which is a properly distributed ciphertext of $Type_2 = Type_{3,k} = Type_{11,k-1}$.

For the ℓth key request, $\mathcal{B}_{3,k,\ell}$ draws $x, x', x^*, y_j \leftarrow \mathbb{Z}_p$ for $j \in M$ and constructs:

$$\begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{J}}$$

$$\left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x^*M_{1,j} + a_{\rho(j)}y_j\tilde{t}_1(a) & x^*M_{2,j} + a_{\rho(j)}y_j\tilde{t}_2(a) & ... & x^*M_{r,j} + a_{\rho(j)}y_j\tilde{t}_r(a) & 0 \\ x^*M_{1,j} & x^*M_{2,j} & ... & x^*M_{r,j} & 0 \end{bmatrix}_{\mathbb{F}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j<k \in M}$$

$$\cup$$

$$\left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x'M_{1,j} + a_{\rho(j)}y_j\tilde{t}_1(a) & x'M_{2,j} + a_{\rho(j)}y_j\tilde{t}_2(a) & ... & x'M_{r,j} + a_{\rho(j)}y_j\tilde{t}_r(a) & 0 \\ x'M_{1,j} & x'M_{2,j} & ... & x'M_{r,j} & 0 \end{bmatrix}_{\mathbb{F}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j>k \in M}$$

$$\cup$$

$$\left\{ \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}(b) \\ x'M_{1,k} + a_{\rho(k)}\tilde{t}_1(ab+r) & x'M_{2,k} + a_{\rho(k)}\tilde{t}_2(ab+r) & ... & x'M_{r,k} + a_{\rho(k)}\tilde{t}_r(ab+r) & 0 \\ x'M_{1,k} & x'M_{2,k} & ... & x'M_{r,k} & 0 \end{bmatrix}_{\mathbb{F}} \begin{bmatrix} (b) \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(k)}} \right\}$$

which in the right bases, is:

$$\begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}}$$

$$\left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x^*M_{1,j} & x^*M_{2,j} & ... & x^*M_{r,j} & 0 \\ 0 & 0 & ... & ... & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j<k \in M}$$

$$\cup$$

$$\left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & ... & xM_{r,j} & a_{\rho(j)}y_j \\ x'M_{1,j} & x'M_{2,j} & ... & x'M_{r,j} & 0 \\ 0 & 0 & ... & ... & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j>k \in M}$$

$$\cup$$

$$\left\{ \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}b \\ x'M_{1,k} + \tilde{t}_1 r a_{\rho(k)} & x'M_{2,k} + \tilde{t}_2 r a_{\rho(k)} & ... & x'M_{r,k} + \tilde{t}_r r a_{\rho(k)} & 0 \\ -\tilde{t}_1 r a_{\rho(k)} & \tilde{t}_2 r a_{\rho(k)} & ... & -\tilde{t}_r r a_{\rho(k)} & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} b \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(k)}} \right\}$$

which is a properly distributed secret key of $Type_{11,k-1}$ when $r = 0$ where $y_k = b$ and is a properly distributed secret key of $Type_{3,k}$ when $r \leftarrow \mathbb{Z}_p$ where $y_k = b$ and $t_i = \tilde{t}_i r a_{\rho(k)}$.

So, $\mathcal{B}_{3,k,\ell}$ is able to exactly simulate $\text{Game}^{\ell}_{11,k-1}$ and $\text{Game}^{\ell}_{3,k}$ when its SXDH challenge has $r = 0$ and $r \leftarrow \mathbb{Z}_p$ respectively. It then follows that for any difference in $\mathcal{A}$'s advantage between the two games, $\mathcal{B}_{3,k,\ell}$ can use $\mathcal{A}$'s answers to achieve that same advantage in the SXDH game. □

Lemma 10. *For $1 \leq \ell \leq Q_1$, for $1 \leq k \leq r$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{4,k,\ell}$ such that:*

$$|\text{Adv}^{\ell}_{3,k}(\mathcal{A}, \lambda) - \text{Adv}^{\ell}_{4,k}(\mathcal{A}, \lambda)| \leq \text{Adv}^{SXDH}_{\mathcal{G}}(\mathcal{B}_{4,k,\ell}, \lambda)$$

Proof. Looking at the $K_{2,k}, K_{3,k}$ components of the $\text{Type}_{3,k}$ ℓth key (the ones containing the t_i and the only things that differ in definition between $\text{Game}^{\ell}_{3,k}$ and $\text{Game}^{\ell}_{4,k}$):

$$\begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ x'M_{1,k} + t_1 & x'M_{2,k} + t_2 & ... & x'M_{r,k} + t_r & 0 \\ -t_1 & -t_2 & ... & -t_r & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(k)}}$$

we see that this is identically distributed to:

$$\begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ -\tilde{t}_1 & -\tilde{t}_2 & ... & -\tilde{t}_r & 0 \\ x'M_{1,k} + \tilde{t}_1 & x'M_{2,k} + \tilde{t}_2 & ... & x'M_{r,k} + \tilde{t}_r & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(k)}}$$

where implicitly $\tilde{t}_i = -xM_{1,k} - t_i$.

So, the transition from $\text{Type}_{3,k}$ key to $\text{Type}_{4,k}$ is symmetric to the one from $\text{Type}_{11,k-1}$ to $\text{Type}_{3,k}$, and by applying the simulation from Lemma 9 accordingly (flipping the embeddings of the bottom two rows of $\mathbb{B}, \mathbb{B}^*$), $\mathcal{B}_{4,k,\ell}$ could use $\mathcal{A}$'s answers to achieve the same advantage in the SXDH game.

□

Lemma 11. *For $1 \leq \ell \leq Q_1$, for $1 \leq k \leq r$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{5,k,\ell}$ such that:*

$$|\text{Adv}^{\ell}_{4,k}(\mathcal{A}, \lambda) - \text{Adv}^{\ell}_{5,k}(\mathcal{A}, \lambda)| \leq \text{Adv}^{SXDH}_{\mathcal{G}}(\mathcal{B}_{5,k,\ell}, \lambda)$$

Proof. Given g, h, h^a, h^b and $T = h^{ab+r^*} \in G$ where either $r^* = 0$ or is a uniform random element of $\mathbb{Z}_p$, consider the following simulator $\mathcal{B}_{5,k,\ell}$ in the security game:

First, $\mathcal{B}_{5,k,\ell}$ generates orthonormal bases:

$$(\mathbb{D}, \mathbb{D}^*) \leftarrow Dual(\mathbb{Z}_p^6)$$
$$(\mathbb{F}, \mathbb{F}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)})$$
$$(\mathbb{K}_i, \mathbb{K}_i^*) \leftarrow Dual(\mathbb{Z}_p^3) \text{ for } i \in \mathcal{U}$$

The bases $(\mathbb{B}, \mathbb{B}^*)$ are implicitly defined as:

$$\vec{b}_{(r+1)} = \vec{f}_{(r+1)} + a\vec{f}_{3(r+1)}$$
$$\vec{b}_i = \vec{f}_i \text{ for all other } i$$

$$\vec{b}^*_{3(r+1)} = \vec{f}^*_{3(r+1)} - a\vec{f}^*_{(r+1)}$$
$$\vec{b}^*_i = \vec{f}^*_i \text{ for all other } i$$

$(\mathbb{A}_i, \mathbb{A}_i^*)$ are implicitly defined as:

$$\vec{a}_{1,i} = \vec{k}_{1,i} + a\vec{k}_{3,i}$$
$$\vec{a}_{2,i} = \vec{k}_{2,i}$$
$$\vec{a}_{3,i} = \vec{k}_{3,i}$$

$$\vec{a}^*_{1,i} = \vec{k}^*_{1,i}$$
$$\vec{a}^*_{2,i} = \vec{k}^*_{2,i}$$
$$\vec{a}^*_{3,i} = \vec{k}^*_{3,i} - a\vec{k}^*_{1,i}$$

The public parameters are then generated by drawing $a_i \leftarrow \mathbb{Z}_p$ for $i \in \mathcal{U}$ and forming:

$$e(g,h)$$
$$(\vec{e}_1)_{\mathbb{D}^*}, (\vec{e}_2)_{\mathbb{D}^*}$$
$$\{(\vec{e}_i)_{\mathbb{B}^*}\}_{i\in[r+1]}$$
$$\{(a_i, 0, 0)_{\mathbb{A}_i^*}\}_{i\in[k]}$$

Note that the terms in the public parameter can be generated since all vectors in the $\mathbb{D}^*, \mathbb{B}^*, \mathbb{A}_i^*$ with nonzero components in the public parameters are known directly.

Similarly, note that all vectors with nonzero components in $Type_0$ and $Type_{12}$ secret keys are known explicitly except for $\vec{b}_{r+1}$ and this can be constructed by taking $h^{\vec{f}_{r+1}} \cdot (h^a)^{\vec{f}_{3(r+1)}} = \mathbf{b}_{(r+1)}$. So, $\mathcal{B}_{5,k,\ell}$ is able to generate appropriately distributed secret keys of $Type_{12}$ for all secret key requests before the ℓth key as well as keys of $Type_0$ for requests after the ℓth request.

For the challenge ciphertext request, let $\vec{w}$ be defined relative to the ℓth requested secret key (this will be known upon ciphertext generation since this secret key is defined to be one of the Q_1 requested before the challenge ciphertext). $\mathcal{B}_{5,k,\ell}$ draws $\alpha, \Delta, \Delta', s', s'', z_i \leftarrow \mathbb{Z}_p$ for $i \in [r]$ and constructs:

$$m \cdot e(g,h)^\alpha, \begin{bmatrix} \alpha & \Delta \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} \Delta & z_2 & \dots & z_r & 0 \\ — & — & \Delta'\vec{w} & — & s' \\ — & — & \Delta'\vec{w} & — & s'' \end{bmatrix}_{\mathbb{F}^*} \left\{ \begin{bmatrix} 0 \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{K}_i^*} \right\}_{i\in S}$$

which in the right bases, is:

$$m \cdot e(g,h)^{\alpha}, \begin{bmatrix} \alpha & \Delta \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} \Delta & z_2 & \dots & z_r & as'' \\ — & — & \Delta'\vec{w} & — & s' \\ — & — & \Delta'\vec{w} & — & s'' \end{bmatrix}_{\mathbb{B}^*} \left\{ \begin{bmatrix} -as''a_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i \in S}$$

which is a properly distributed ciphertext of $Type_2 = Type_{4,k} = Type_{5,k}$ where $s = as''$.

For the ℓth key request, $\mathcal{B}_{5,k,\ell}$ draws $x, x', x^*, y_j \leftarrow \mathbb{Z}_p$ for $j \in M$ and constructs:

$$\begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}}$$

$$\left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & \dots & xM_{r,j} & a_{\rho(j)}y_j \\ x^*M_{1,j} & x^*M_{2,j} & \dots & x^*M_{r,j} & 0 \\ 0 & 0 & \dots & \dots & a_{\rho(j)}y_j(a) \end{bmatrix}_{\mathbb{F}} \begin{bmatrix} y_j \\ 0 \\ y_j(a) \end{bmatrix}_{\mathbb{K}_{\rho(j)}} \right\}_{j<k \in M}$$

$$\cup$$

$$\left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & \dots & xM_{r,j} & a_{\rho(j)}y_j \\ x'M_{1,j} & x'M_{2,j} & \dots & x'M_{r,j} & 0 \\ 0 & 0 & \dots & \dots & a_{\rho(j)}y_j(a) \end{bmatrix}_{\mathbb{F}} \begin{bmatrix} y_j \\ 0 \\ y_j(a) \end{bmatrix}_{\mathbb{K}_{\rho(j)}} \right\}_{j>k \in M}$$

$$\cup$$

$$\left\{ \begin{bmatrix} xM_{1,k} & xM_{2,k} & \dots & xM_{r,k} & a_{\rho(k)}(b) \\ 0 & 0 & \dots & 0 & 0 \\ x'M_{1,k} & x'M_{2,k} & \dots & x'M_{r,k} & a_{\rho(k)}(ab+r) \end{bmatrix}_{\mathbb{F}} \begin{bmatrix} (b) \\ 0 \\ (ab+r) \end{bmatrix}_{\mathbb{K}_{\rho(k)}} \right\}$$

which in the right bases, is:

$$\begin{bmatrix} 1 & x \\ 0 & x' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}}$$

$$\left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & \dots & xM_{r,j} & a_{\rho(j)}y_j \\ x^*M_{1,j} & x^*M_{2,j} & \dots & x^*M_{r,j} & 0 \\ 0 & 0 & \dots & \dots & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j<k \in M}$$

$$\cup$$

$$\left\{ \begin{bmatrix} xM_{1,j} & xM_{2,j} & \dots & xM_{r,j} & a_{\rho(j)}y_j \\ x'M_{1,j} & x'M_{2,j} & \dots & x'M_{r,j} & 0 \\ 0 & 0 & \dots & \dots & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j>k \in M}$$

$$\cup$$

$$\left\{ \begin{bmatrix} xM_{1,k} & xM_{2,k} & \dots & xM_{r,k} & a_{\rho(k)}b \\ 0 & 0 & \dots & 0 & 0 \\ x'M_{1,k} & x'M_{2,k} & \dots & x'M_{r,k} & a_{\rho(k)}r \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} b \\ 0 \\ r \end{bmatrix}_{\mathbb{A}_{\rho(k)}} \right\}$$

which is a properly distributed secret key of $Type_{4,k}$ when $r = 0$ where $y_k = b$ and is a properly distributed secret key of $Type_{5,k}$ when $r \leftarrow \mathbb{Z}_p$ where $y_k = b$ and $y'_k = r$.

So, $\mathcal{B}_{5,k,\ell}$ is able to exactly simulate $\text{Game}^{\ell}_{4,k}$ and $\text{Game}^{\ell}_{5,k}$ when its SXDH challenge has $r = 0$ and $r \leftarrow \mathbb{Z}_p$ respectively. It then follows that for any difference in $\mathcal{A}$'s advantage between the two games, $\mathcal{B}_{5,k,\ell}$ can use $\mathcal{A}$'s answers to achieve that same advantage in the SXDH game.

□

Lemma 12. *For $1 \leq \ell \leq Q_1$, for $1 \leq k \leq r$, for any adversary $\mathcal{A}$,*

$$|\text{Adv}^{\ell}_{5,k}(\mathcal{A}, \lambda) - \text{Adv}^{\ell}_{6,k}(\mathcal{A}, \lambda)| = 0$$

Proof. Given the bases:

$$(\mathbb{D}, \mathbb{D}^*) \leftarrow Dual(\mathbb{Z}_p^6)$$
$$(\mathbb{F}, \mathbb{F}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)})$$
$$(\mathbb{K}_i, \mathbb{K}_i^*) \leftarrow Dual(\mathbb{Z}_p^3) \text{ for } i \in \mathcal{U}$$

Define the bases $(\mathbb{B}, \mathbb{B}^*)$ as:

$$\vec{b}_{3(r+1)} = \tilde{a}\vec{f}_{3(r+1)}$$
$$\vec{b}_i = \vec{f}_i \text{ for all other } i$$

$$\vec{b}^*_{3(r+1)} = \tilde{a}^{-1}\vec{f}^*_{3(r+1)}$$
$$\vec{b}^*_i = \vec{f}^*_i \text{ for all other } i$$

for a random $\tilde{a} \leftarrow \mathbb{Z}_p$, define $(\mathbb{A}_i, \mathbb{A}_i^*)$ as equal to $(\mathbb{K}_i, \mathbb{K}_i^*)$ for $i = \rho(k)$ and define $(\mathbb{A}_i, \mathbb{A}_i^*)$ as:

$$\vec{a}_{1,i} = \vec{k}_{1,i}$$
$$\vec{a}_{2,i} = \vec{k}_{2,i}$$
$$\vec{a}_{3,i} = \tilde{a}\vec{k}_{3,i}$$

$$\vec{a}^*_{1,i} = \vec{k}^*_{1,i}$$
$$\vec{a}^*_{2,i} = \vec{k}^*_{2,i}$$
$$\vec{a}^*_{3,i} = \tilde{a}^{-1}\vec{k}^*_{3,i}$$

for $i \neq \rho(k)$.

Consider simulating $\text{Game}^{\ell}_{5,k}$ with the $(\mathbb{D}, \mathbb{D}^*), (\mathbb{B}, \mathbb{B}^*), (\mathbb{A}_i, \mathbb{A}_i^*)$ bases. We will see that doing so is equivalent to simulating $\text{Game}^{\ell}_{6,k}$ with the $(\mathbb{D}, \mathbb{D}^*), (\mathbb{F}, \mathbb{F}^*), (\mathbb{K}_i, \mathbb{K}_i^*)$ bases. Since both sets of bases come from the same distribution, then the two games are identical.

The only key-side elements that have a non-zero component in a vector that has been transformed between the two bases are the $K_{2,k}, K_{3,k}$ components of the (simulated $\text{Type}_{5,k}$) ℓth key:

$$\left\{ \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ 0 & 0 & ... & 0 & 0 \\ x'M_{1,k} & x'M_{2,k} & ... & x'M_{r,k} & a_{\rho(k)}y'_k \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_k \\ 0 \\ y'_k \end{bmatrix}_{\mathbb{A}_{\rho(k)}} \right\}$$

which in the $(\mathbb{D}, \mathbb{D}^*), (\mathbb{F}, \mathbb{F}^*), (\mathbb{K}_i, \mathbb{K}_i^*)$ looks like:

$$\left\{ \begin{bmatrix} xM_{1,k} & xM_{2,k} & ... & xM_{r,k} & a_{\rho(k)}y_k \\ 0 & 0 & ... & 0 & 0 \\ x'M_{1,k} & x'M_{2,k} & ... & x'M_{r,k} & a_{\rho(k)}\tilde{a}y'_k \end{bmatrix}_{\mathbb{F}} \begin{bmatrix} y_k \\ 0 \\ y'_k \end{bmatrix}_{\mathbb{K}_{\rho(k)}} \right\}$$

a properly distributed $\text{Type}_{6,k}$ key where $\tilde{a}_{\rho(k)} = \tilde{a}a_{\rho(k)}$.

The only ciphertext-side elements that have a non-zero component in a vector that has been transformed between the two bases are the $C_{2,k}, K_{3,k}$ components of the (simulated $\text{Type}_{5,k}$) challenge ciphertext:

$$m \cdot e(g,h)^{\alpha}, \begin{bmatrix} \alpha & \Delta \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} \Delta & z_2 & \dots & z_r & s \\ — & — & \Delta'\vec{w} & — & s' \\ — & — & \Delta'\vec{w} & — & \tilde{s}'' \end{bmatrix}_{\mathbb{B}^*} \left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -\tilde{s}''a_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i \in S}$$

which in the $(\mathbb{D}, \mathbb{D}^*), (\mathbb{F}, \mathbb{F}^*), (\mathbb{K}_i, \mathbb{K}_i^*)$ looks like:

$$m \cdot e(g,h)^{\alpha}, \begin{bmatrix} \alpha & \Delta \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} \Delta & z_2 & \dots & z_r & s \\ — & — & \Delta'\vec{w} & — & s' \\ — & — & \Delta'\vec{w} & — & \tilde{s}''\tilde{a}^{-1} \end{bmatrix}_{\mathbb{F}^*}$$

$$\left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -\tilde{s}''\tilde{a}^{-1}a_i \end{bmatrix}_{\mathbb{K}_i^*} \right\}_{i \neq \rho(k) \in S} \bigcup \left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -(\tilde{s}''\tilde{a}^{-1})(\tilde{a}a_i) \end{bmatrix}_{\mathbb{K}_i^*} \right\}_{i = \rho(k) \in S}$$

which is a properly distributed $\text{Type}_{6,k}$ challenge ciphertext where $s'' = \tilde{s}''\tilde{a}^{-1}$, $\tilde{a}_{\rho(k)} = a_{\rho(k)}\tilde{a}$.

These are the only elements that interact with the transformed basis vectors during the simulation, the transformation results in the same distribution of basis vectors, and we showed the using one set simulates $\text{Game}_{5,k}^{\ell}$ and using the other simulates $\text{Game}_{6,k}^{\ell}$, therefore the two games are identical and no adversary can achieve a difference in advantage between the two.

□

Lemma 13. *For $1 \leq \ell \leq Q_1$, for $1 \leq k \leq r$, for any adversary $\mathcal{A}$,*

$$|\text{Adv}_{6,k}^{\ell}(\mathcal{A}, \lambda) - \text{Adv}_{7,k}^{\ell}(\mathcal{A}, \lambda)| = 0$$

Proof. This follows from a direct application of Lemma 8 from [Tak17] (which relies on Lemma 3 from [OT10]). Essentially, the lemma says that since the only nonzero elements in the third row of the $(\mathbb{B}, \mathbb{B}^*)$ basis are in the single challenge ciphertext and the ℓth key's kth components $K_{2,k}$, then their distribution is dependent only on their dot product. We reproduce the lemma in Appendix A's Lemma 31.

If $a_{\rho(k)} \in S$ for the challenge ciphertext, then we have that $\vec{w} \cdot \vec{M}_k = \vec{w} \cdot x'\vec{M}_k = 0$. So, by Lemma 31 we can switch $x'\vec{M}_k$ to $x^*\vec{M}_k$ in the ℓth key's kth component without changing the distribution, since $\vec{w} \cdot x^*\vec{M}_k = 0$.

If $a_{\rho(k)} \notin S$ for the challenge ciphertext, then we know that $\tilde{a}_{\rho_k}$ appears only in the ℓth key's kth component. So, we can use the same lemma to argue that its randomness blinds the dot product. Specifically, we look at the dot product of the previous vectors plus the additional $3(r+1)$th component, which is equal to: $\vec{w} \cdot x'\vec{M}_k + s''\tilde{a}_{\rho(k)}$, which is uniformly random because of the presence of $\tilde{a}_{\rho(k)}$. So, again by Lemma 31, we can switch $x'\vec{M}_k$ to $x^*\vec{M}_k$ in the ℓth key's kth component without changing the distribution, since the new dot product $\vec{w} \cdot x^*\vec{M}_k + s''\tilde{a}_{\rho(k)}$ is also uniformly randomly distributed. □

Lemma 14. *For $1 \leq \ell \leq Q_1$, for $1 \leq k \leq r$, for any adversary $\mathcal{A}$,*

$$|\mathrm{Adv}^{\ell}_{7,k}(\mathcal{A}, \lambda) - \mathrm{Adv}^{\ell}_{8,k}(\mathcal{A}, \lambda)| = 0$$

Lemma 15. *For $1 \leq \ell \leq Q_1$, for $1 \leq k \leq r$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{9,k,\ell}$ such that:*

$$|\mathrm{Adv}^{\ell}_{8,k}(\mathcal{A}, \lambda) - \mathrm{Adv}^{\ell}_{9,k}(\mathcal{A}, \lambda)| \leq \mathrm{Adv}^{SXDH}_{\mathcal{G}}(\mathcal{B}_{9,k,\ell}, \lambda)$$

Lemma 16. *For $1 \leq \ell \leq Q_1$, for $1 \leq k \leq r$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{10,k,\ell}$ such that:*

$$|\mathrm{Adv}^{\ell}_{9,k}(\mathcal{A}, \lambda) - \mathrm{Adv}^{\ell}_{10,k}(\mathcal{A}, \lambda)| \leq \mathrm{Adv}^{SXDH}_{\mathcal{G}}(\mathcal{B}_{10,k,\ell}, \lambda)$$

Lemma 17. *For $1 \leq \ell \leq Q_1$, for $1 \leq k \leq r$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{11,k,\ell}$ such that:*

$$|\mathrm{Adv}^{\ell}_{10,k}(\mathcal{A}, \lambda) - \mathrm{Adv}^{\ell}_{11,k}(\mathcal{A}, \lambda)| \leq \mathrm{Adv}^{SXDH}_{\mathcal{G}}(\mathcal{B}_{11,k,\ell}, \lambda)$$

Proof. These proofs follow the same arguments as Lemma 12, Lemma 11, Lemma 10, and Lemma 9 respectively, in reverse, and using x^* instead of x' in the ℓth key's kth component. □

Lemma 18. *For $1 \leq \ell \leq Q_1$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{12,\ell}$ such that:*

$$|\mathrm{Adv}^{\ell}_{11,r}(\mathcal{A}, \lambda) - \mathrm{Adv}^{\ell}_{12}(\mathcal{A}, \lambda)| \leq \mathrm{Adv}^{SXDH}_{\mathcal{G}}(\mathcal{B}_{12,\ell}, \lambda)$$

Proof. This lemma is almost essentially a reverse of Lemma 8 using x^* instead of x', except the x' in the $\mathbb{D}$ basis of the ℓth key is *not* removed. So, we will show that we can fix it so it does not disappear when the rest of the x^* components go to 0:

Given g, h, h^a, h^b and $T = h^{ab+r^*} \in G$ where either $r^* = 0$ or is a uniform random element of $\mathbb{Z}_p$, consider the following simulator $\mathcal{B}_{12,\ell}$ in the security game:

First, $\mathcal{B}_{12,\ell}$ generates orthonormal bases:

$$(\mathbb{J}, \mathbb{J}^*) \leftarrow Dual(\mathbb{Z}_p^6)$$
$$(\mathbb{F}, \mathbb{F}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)})$$
$$(\mathbb{A}_i, \mathbb{A}_i^*) \leftarrow Dual(\mathbb{Z}_p^3) \text{ for } i \in \mathcal{U}$$

The bases $(\mathbb{B}, \mathbb{B}^*)$ are implicitly defined as:

$$\vec{b}_i = \vec{f}_i + a\vec{f}_{(r+1)+i} \text{ for } i \in [1, r]$$
$$\vec{b}_i = \vec{f}_i \text{ for all other } i$$

$$\vec{b}^*_{(r+1)+i} = \vec{f}^*_{(r+1)+i} - a\vec{f}^*_i \text{ for } i \in [1, r]$$
$$\vec{b}^*_i = \vec{f}^*_i \text{ for all other } i$$

$(\mathbb{D}, \mathbb{D}^*)$ are implicitly defined as:

$$\vec{d}_2 = \vec{j}_2 + a\vec{j}_4$$
$$\vec{d}_i = \vec{j}_i \text{ for all other } i$$

$$\vec{d}^*_4 = \vec{j}^*_4 - a\vec{j}^*_2$$
$$\vec{d}^*_i = \vec{j}^*_i \text{ for all other } i$$

The public parameters are then generated by drawing $a_i \leftarrow \mathbb{Z}_p$ for $i \in \mathcal{U}$ and forming:

$$e(g,h)$$
$$(\vec{e}_1)_{\mathbb{D}^*}, (\vec{e}_2)_{\mathbb{D}^*}$$
$$\{(\vec{e}_i)_{\mathbb{B}^*}\}_{i\in[r+1]}$$
$$\{(a_i, 0, 0)_{\mathbb{A}^*_i}\}_{i\in[k]}$$

Note that the terms in the public parameter can be generated since all vectors in the $\mathbb{D}^*, \mathbb{B}^*, \mathbb{A}^*_i$ with nonzero components in the public parameters are known directly.

Similarly, note that all vectors in $\mathbb{D}, \mathbb{B}, \mathbb{A}_i$ are known explicitly except for $\vec{d}_2$ and $\vec{b}_i$ for $i \in [r]$ and these can be constructed by taking $h^{\vec{j}_2} \cdot (h^a)^{\vec{j}_4} = \mathbf{d}_2$, and $h^{\vec{f}_i} \cdot (h^a)^{\vec{f}_{(r+1)+i}} = \mathbf{b}_i$ for $i \in [r]$. So, $\mathcal{B}_{12,\ell}$ is able to generate appropriately distributed secret keys of $Type_0$ for all secret key requests before the ℓth key as well as keys of $Type_{12}$ for requests after the ℓth request.

For the challenge ciphertext request, let $\vec{w}$ be defined relative to the ℓth requested secret key (this will be known upon ciphertext generation since this secret key is defined to be one of the Q_1 requested before the challenge ciphertext). $\mathcal{B}_{12,\ell}$ draws $\alpha, \Delta', s, s', s'', \tilde{z}_i \leftarrow \mathbb{Z}_p$ for $i \in [r]$ and constructs:

$$m \cdot e(g,h)^\alpha, \begin{bmatrix} \alpha & 0 \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{J}^*}$$

$$\begin{bmatrix} 0 & \tilde{z}_2 & \dots & \tilde{z}_r & s \\ \Delta' & \Delta' w_2 & \dots & \Delta' w_r & s' \\ \Delta' & \Delta' w_2 & \dots & \Delta' w_r & s'' \end{bmatrix}_{\mathbb{F}^*}$$

$$\left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}^*_i} \right\}_{i\in S}$$

which in the right bases, is:

$$m \cdot e(g,h)^\alpha, \begin{bmatrix} \alpha & a\Delta' \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} a\Delta' & a\Delta' w_2 + \tilde{z}_2 & \dots & a\Delta' w_r + \tilde{z}_r & s \\ — & — & \Delta'\vec{w} & — & s' \\ — & — & \Delta'\vec{w} & — & s'' \end{bmatrix}_{\mathbb{B}^*} \left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}^*_i} \right\}_{i\in S}$$

which is a properly distributed ciphertext of $Type_1 = Type_2$ where $\Delta = a\Delta'$, and $z_i = a\Delta' + \tilde{z}_i$.

For the ℓth key request, $\mathcal{B}_{12,\ell}$ draws $\tilde{x}', y_j \leftarrow \mathbb{Z}_p$ for $j \in M$ and constructs:

$$\begin{bmatrix} 1 & (b) \\ 0 & (ab + r) + \tilde{x}' \\ 0 & 0 \end{bmatrix}_{\mathbb{J}}$$

$$\left\{ \begin{bmatrix} (b)M_{1,j} & (b)M_{2,j} & ... & (b)M_{r,j} & a_{\rho(j)}y_j \\ (ab+r)M_{1,j} & (ab+r)M_{2,j} & ... & (ab+r)M_{r,j} & 0 \\ 0 & 0 & ... & ... & 0 \end{bmatrix}_{\mathbb{F}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M}$$

which in the right bases, is:

$$\begin{bmatrix} 1 & b \\ 0 & r+\tilde{x}' \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} bM_{1,j} & bM_{2,j} & ... & bM_{r,j} & a_{\rho(j)}y_j \\ rM_{1,j} & rM_{2,j} & ... & rM_{r,j} & 0 \\ 0 & 0 & ... & ... & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M}$$

which is a properly distributed secret key of $Type_{11,r}$ when $r \leftarrow \mathbb{Z}_p$ where $x = b, x' = r + \tilde{x}'$, and $x^* = r$ and is a properly distributed secret key of $Type_{12}$ when $r = 0$ where $x = b, x' = r + \tilde{x}' = \tilde{x}'$.

So, $\mathcal{B}_{12,\ell}$ is able to exactly simulate $\text{Game}_{11,r}^{\ell}$ and Game_{12}^{ℓ} when its SXDH challenge has $r \leftarrow \mathbb{Z}_p$ and $r = 0$ respectively. It then follows that for any difference in $\mathcal{A}$'s advantage between the two games, $\mathcal{B}_{12,\ell}$ can use $\mathcal{A}$'s answers to achieve that same advantage in the SXDH game.

□

Lemma 19. *For $1 \leq \ell < Q_1$, for any adversary$\mathcal{A}$,*

$$|\text{Adv}_{12}^{\ell}(\mathcal{A}, \lambda) - \text{Adv}_{1}^{\ell+1}(\mathcal{A}, \lambda)| = 0$$

Proof. The only difference between these two games is that the $\vec{w}$ used in the challenge ciphertext is switched from being defined relative to the ℓth requested secret key to the $\ell + 1$th requested key. We will call the new vector $\vec{v}$. All secret keys contain zeroes on the opposite side, so this can be done with a simple change of basis:

Specifically, consider the following sets of bases:

$$(\mathbb{D}, \mathbb{D}^*) \leftarrow Dual(\mathbb{Z}_p^6)$$
$$(\mathbb{F}, \mathbb{F}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)})$$
$$(\mathbb{A}_i, \mathbb{A}_i^*) \leftarrow Dual(\mathbb{Z}_p^3) \text{ for } i \in \mathcal{U}$$

and define the bases $(\mathbb{B}, \mathbb{B}^*)$ as:

$$\vec{b}_{(r+1)+1}^* = \vec{f}_{(r+1)+1}^* + \sum_{i=2}^{r} (v_i - w_i)\vec{f}_{(r+1)+i}^*$$
$$\vec{b}_i^* = \vec{f}_i^* \text{ for all other } i$$
$$\vec{b}_{(r+1)+i} = \vec{f}_{(r+1)+i} - (v_i - w_i)\vec{f}_{(r+1)+1} \text{ for } i \in [2, r]$$
$$\vec{b}_i = \vec{f}_i \text{ for all other } i$$

The transformation results in the same distribution of basis vectors between $((\mathbb{D}, \mathbb{D}^*), (\mathbb{F}, \mathbb{F}^*), (\mathbb{A}_i, \mathbb{A}_i^*))$ and $((\mathbb{D}, \mathbb{D}^*), (\mathbb{B}, \mathbb{B}^*), (\mathbb{A}_i, \mathbb{A}_i^*))$, and yet using the second set to simulate $\text{Game}_1^{\ell+1}$ implicitly simulates Game_{12}^{ℓ} in the first basis. Therefore, the two games are identical and no adversary can achieve a difference in advantage between the two.

□

Now we begin the second half of the hybrid: handling keys requested *after* the challenge ciphertext. This is a more conventional hybrid reminiscent of the selective security proofs for static-assumption-based ABE schemes.

Lemma 20. *For any adversary $\mathcal{A}$,*

$$|\mathrm{Adv}_{12}^{Q_1}(\mathcal{A},\lambda) - \mathrm{Adv}_{13}^{1}(\mathcal{A},\lambda)| = 0$$

Proof. The only difference between these two games is that the $\vec{w}$ used in the challenge ciphertext is switched from being defined relative to the Q_1th requested secret key (the last before the challenge ciphertext) to a static standard basis vector: $\vec{e}_1$. All secret keys contain zeroes on the opposite side, so this can be done with a simple change of basis:

Specifically, consider the following sets of bases:

$$(\mathbb{D},\mathbb{D}^*) \leftarrow Dual(\mathbb{Z}_p^6)$$
$$(\mathbb{F},\mathbb{F}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)})$$
$$(\mathbb{A}_i,\mathbb{A}_i^*) \leftarrow Dual(\mathbb{Z}_p^3) \text{ for } i \in \mathcal{U}$$

and define the bases $(\mathbb{B},\mathbb{B}^*)$ as:

$$\vec{b}^*_{(r+1)+1} = \vec{f}^*_{(r+1)+1} + \sum_{i=2}^{r} -w_i \vec{f}^*_{(r+1)+i}$$
$$\vec{b}^*_i = \vec{f}^*_i \text{ for all other } i$$
$$\vec{b}_{(r+1)+i} = \vec{f}_{(r+1)+i} + w_i \vec{f}_{(r+1)+1} \text{ for } i \in [2,r]$$
$$\vec{b}_i = \vec{f}_i \text{ for all other } i$$

The transformation results in the same distribution of basis vectors between $((\mathbb{D},\mathbb{D}^*),(\mathbb{F},\mathbb{F}^*),(\mathbb{A}_i,\mathbb{A}_i^*))$ and $((\mathbb{D},\mathbb{D}^*),(\mathbb{B},\mathbb{B}^*),(\mathbb{A}_i,\mathbb{A}_i^*))$, and yet using the second set to simulate Game_{13}^1 implicitly simulates $\mathrm{Game}_{12}^{Q_1}$ in the first basis. Therefore, the two games are identical and no adversary can achieve a difference in advantage between the two.

□

For the connection to the previous lemma, note that $\mathrm{Game}_{13}^{\ell}$ is equivalent to $\mathrm{Game}_{17}^{\ell-1}$.

Lemma 21. *For $1 \le \ell \le Q_2$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{14,\ell}$ such that:*

$$|\mathrm{Adv}_{17}^{\ell-1}(\mathcal{A},\lambda) - \mathrm{Adv}_{14}^{\ell}(\mathcal{A},\lambda)| \le \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{14,\ell},\lambda)$$

Proof. Given g, h, h^a, h^b and $T = h^{ab+r^*} \in G$ where either $r^* = 0$ or is a uniform random element of $\mathbb{Z}_p$, consider the following simulator $\mathcal{B}_{14,\ell}$ in the security game:

First, $\mathcal{B}_{14,\ell}$ generates orthonormal bases:

$$(\mathbb{J},\mathbb{J}^*) \leftarrow Dual(\mathbb{Z}_p^6)$$
$$(\mathbb{F},\mathbb{F}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)})$$
$$(\mathbb{A}_i,\mathbb{A}_i^*) \leftarrow Dual(\mathbb{Z}_p^3) \text{ for } i \in \mathcal{U}$$

The bases $(\mathbb{B}, \mathbb{B}^*)$ are implicitly defined as:

$$\vec{b}_i = \vec{f}_i + az'_i \vec{f}_{(r+1)+1} \text{ for } i \in [1, r]$$
$$\vec{b}_i = \vec{f}_i \text{ for all other } i$$

$$\vec{b}^*_{(r+1)+1} = \vec{f}^*_{(r+1)+1} - \sum_{i=1}^{r} az'_i \vec{f}^*_i$$
$$\vec{b}^*_i = \vec{f}^*_i \text{ for all other } i$$

for $z'_1 = 1$ and randomly drawn $z'_i \leftarrow \mathbb{Z}_p$ for $i > 1$.

$(\mathbb{D}, \mathbb{D}^*)$ are implicitly defined as:

$$\vec{d}_2 = \vec{j}_2 + a\vec{j}_4$$
$$\vec{d}_i = \vec{j}_i \text{ for all other } i$$

$$\vec{d}^*_4 = \vec{j}^*_4 - a\vec{j}^*_2$$
$$\vec{d}^*_i = \vec{j}^*_i \text{ for all other } i$$

The public parameters are then generated by drawing $a_i \leftarrow \mathbb{Z}_p$ for $i \in \mathcal{U}$ and forming:

$$e(g, h)$$
$$(\vec{e}_1)_{\mathbb{D}^*}, (\vec{e}_2)_{\mathbb{D}^*}$$
$$\{(\vec{e}_i)_{\mathbb{B}^*}\}_{i \in [r+1]}$$
$$\{(a_i, 0, 0)_{\mathbb{A}^*_i}\}_{i \in [k]}$$

Note that the terms in the public parameter can be generated since all vectors in the $\mathbb{D}^*, \mathbb{B}^*, \mathbb{A}^*_i$ with nonzero components in the public parameters are known directly.

Similarly, note that all vectors in $\mathbb{D}, \mathbb{B}, \mathbb{A}_i$ are known explicitly except for $\vec{d}_2$ and $\vec{b}_i$ for $i \in [r]$ and these can be constructed by taking $h^{\vec{j}_2} \cdot (h^a)^{\vec{j}_4} = \mathbf{d}_2$, and $h^{\vec{f}_i} \cdot (h^a)^{\tilde{z}'_i \vec{f}_{(r+1)+i}} = \mathbf{b}_i$ for $i \in [r]$. So, $\mathcal{B}_{14,\ell}$ is able to generate appropriately distributed secret keys of $Type_0$ for all secret key requests after the $Q_1 + \ell$th key as well as keys of $Type_{12} = Type_{17}$ for requests before the $Q_1 + \ell$th request.

For the challenge ciphertext request, $\mathcal{B}_{14,\ell}$ draws $\alpha, \Delta', s, s', s'', \tilde{z}_i \leftarrow \mathbb{Z}_p$ for $i \in [r]$ and constructs:

$$m \cdot e(g, h)^{\alpha}, \begin{bmatrix} \alpha & 0 \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{J}^*}$$

$$\begin{bmatrix} 0 & \tilde{z}_2 & \dots & \tilde{z}_r & s \\ \Delta' & 0 & \dots & 0 & s' \\ \Delta' & 0 & \dots & 0 & s'' \end{bmatrix}_{\mathbb{F}^*}$$

$$\left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}^*_i} \right\}_{i \in S}$$

which in the right bases, is:

$$m \cdot e(g,h)^{\alpha}, \begin{bmatrix} \alpha & a\Delta' \\ 0 & \Delta' \\ 0 & \Delta' \end{bmatrix}_{\mathbb{D}^*} \begin{bmatrix} a\Delta' & a\Delta' z_2' + \tilde{z}_2 & \dots & a\Delta' z_r' + \tilde{z}_r & s \\ — & — & \Delta'\vec{e}_1 & — & s' \\ — & — & \Delta'\vec{e}_1 & — & s'' \end{bmatrix}_{\mathbb{B}^*} \left\{ \begin{bmatrix} -sa_i \\ -s'a_i \\ -s''a_i \end{bmatrix}_{\mathbb{A}_i^*} \right\}_{i \in S}$$

which is a properly distributed ciphertext of $Type_{17} = Type_{13} = Type_{14}$ where $\Delta = a\Delta'$, and $z_i = a\Delta' z_i' + \tilde{z}_i$.

For the ℓth key request, $\mathcal{B}_{14,\ell}$ draws $y_j \leftarrow \mathbb{Z}_p$ for $j \in M$ and constructs:

$$\begin{bmatrix} 1 & (b) \\ 0 & (ab+r) \\ 0 & 0 \end{bmatrix}_{\mathbb{J}}$$

$$\left\{ \begin{bmatrix} (b)M_{1,j} & (b)M_{2,j} & ... & (b)M_{r,j} & a_{\rho(j)}y_j \\ \sum_{i=1}^{r}(ab+r)z_i' M_{i,j} & 0 & ... & 0 & 0 \\ 0 & 0 & ... & ... & 0 \end{bmatrix}_{\mathbb{F}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M}$$

which in the right bases, is:

$$\begin{bmatrix} 1 & b \\ 0 & r \\ 0 & 0 \end{bmatrix}_{\mathbb{D}} \left\{ \begin{bmatrix} bM_{1,j} & bM_{2,j} & ... & bM_{r,j} & a_{\rho(j)}y_j \\ r\sum_{i=1}^{r} z_i' M_{i,j} & 0 & ... & 0 & 0 \\ 0 & 0 & ... & ... & 0 \end{bmatrix}_{\mathbb{B}} \begin{bmatrix} y_j \\ 0 \\ 0 \end{bmatrix}_{\mathbb{A}_{\rho(j)}} \right\}_{j \in M}$$

which is a properly distributed secret key of $Type_0$ when $r = 0$ where $x = b$ and is a properly distributed secret key of $Type_{14}$ when $r \leftarrow \mathbb{Z}_p$ where $x = b$, $x' = r$, and $\lambda_j = \sum_{i=1}^{r} \tilde{z}_i' M_{i,j} = \vec{M}_j \cdot (1, z_2', ..., z_r')$.

So, $\mathcal{B}_{14,\ell}$ is able to exactly simulate $\text{Game}_{17}^{\ell-1}$ and Game_{14}^{ℓ} when its SXDH challenge has $r = 0$ and $r \leftarrow \mathbb{Z}_p$ respectively. It then follows that for any difference in $\mathcal{A}$'s advantage between the two games, $\mathcal{B}_{14,\ell}$ can use $\mathcal{A}$'s answers to achieve that same advantage in the SXDH game.

□

For the connection to the previous lemma, note that Game_{14}^{ℓ} is equivalent to $\text{Game}_{15,0}^{\ell}$.

Lemma 22. *For $1 \leq \ell \leq Q_2$, for $1 \leq k \leq |\mathcal{U}|$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{15,k,\ell}$ such that:*

$$|\text{Adv}_{15,k-1}^{\ell}(\mathcal{A}, \lambda) - \text{Adv}_{15,k}^{\ell}(\mathcal{A}, \lambda)| \leq \text{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{15,k,\ell}, \lambda)$$

Proof. First, note that if $k \in S$ for the challenge ciphertext's attribute set, then $\text{Game}_{15,k-1}^{\ell}$ is identical to $\text{Game}_{15,k}^{\ell}$, in which case the adversary's difference in advantage is 0. So, we only need to account for the worse case when $k \notin S$.

In this case, we can use an embedding that is symmetric to that of Lemma 11 to bring in $\tilde{a}_j y_j'$ for j where $\rho(j) = k$. The $\mathcal{B}_{15,k,\ell}$ that performs this simulation can use $\mathcal{A}$'s answers to achieve that same advantage in the SXDH game. □

Lemma 23. *For $1 \leq \ell \leq Q_2$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{15,H,\ell}$ such that:*

$$|\mathrm{Adv}^{\ell}_{15,\mathcal{U}}(\mathcal{A},\lambda) - \mathrm{Adv}^{\ell}_{15,H}(\mathcal{A},\lambda)| \leq \mathrm{Adv}^{SXDH}_{\mathcal{G}}(\mathcal{B}_{15,H,\ell},\lambda)$$

Proof. This proof follows a standard selective security argument to embed the SXDH challenge in exactly the rows of the $Q_1 + \ell$th key for which $\rho(j) \notin S$, changing the $y'_j \tilde{a}_{\rho(j)}$ to $y'_j \tilde{a}'_j$ (that is, creating independent randomness $\tilde{a}'_j$ for each j). This can be done since the $Q_1 + \ell$th key is requested *after* the challenge ciphertext, and therefore it is known upon key generation which rows contain $\tilde{a}_{\rho(j)}$ that don't need to be replicated on the other side in the challenge ciphertext (and can therefore serve as an embedding for the SXDH challenge). □

Lemma 24. *For $1 \leq \ell \leq Q_2$, for any adversary $\mathcal{A}$,*

$$|\mathrm{Adv}^{\ell}_{15,H}(\mathcal{A},\lambda) - \mathrm{Adv}^{\ell}_{16,H}(\mathcal{A},\lambda)| = 0$$

Proof. This follows from the standard information-theoretic security of the secret sharing scheme for λ'. Since the key is unauthorized, we can switch the λ' from sharing 1 to sharing a random $z' \leftarrow \mathbb{Z}_p$ without changing the distribution of shares that aren't masked by $\tilde{a}'_j$ (and the $\tilde{a}'_j$ information-theoretically hide the λ_j that do change – we can "mix" the dimensions that contain the λ_j and the $\tilde{a}'_j y'_j$ via a change of basis to see the hiding property). The resulting shares $\lambda'_j := \vec{M}_j \cdot (z'_1, z'_2, ..., z'_r)$ for $z'_i \leftarrow \mathbb{Z}_p$ have the property that $\{x'\lambda'_j\}_j$ is distributed the same as $\{x^*\lambda'_j\}_j$. □

Lemma 25. *For $1 \leq \ell \leq Q_2$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{16,U,\ell}$ such that:*

$$|\mathrm{Adv}^{\ell}_{16,H}(\mathcal{A},\lambda) - \mathrm{Adv}^{\ell}_{16,\mathcal{U}}(\mathcal{A},\lambda)| \leq \mathrm{Adv}^{SXDH}_{\mathcal{G}}(\mathcal{B}_{16,H,\ell},\lambda)$$

Proof. This proof is symmetric to Lemma 23, where we embed SXDH in the j rows that are not authorized to make the reverse transformation (while using $x^*\lambda^*$ instead of $x'\lambda'$ in the ℓth key). □

Lemma 26. *For $1 \leq \ell \leq Q_2$, for $|\mathcal{U}| \geq k \geq 1$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{16,k-1,\ell}$ such that:*

$$|\mathrm{Adv}^{\ell}_{16,k}(\mathcal{A},\lambda) - \mathrm{Adv}^{\ell}_{16,k-1}(\mathcal{A},\lambda)| \leq \mathrm{Adv}^{SXDH}_{\mathcal{G}}(\mathcal{B}_{16,k-1,\ell},\lambda)$$

Proof. This proof is symmetric to Lemma 22, but in reverse (and using $x^*\lambda^*$ instead of $x'\lambda'$ in the ℓth key). □

Lemma 27. *For $1 \leq \ell \leq Q_2$, for any polynomial time adversary $\mathcal{A}$, there exists a polynomial time $\mathcal{B}_{17,\ell}$ such that:*

$$|\mathrm{Adv}^{\ell}_{16,0}(\mathcal{A},\lambda) - \mathrm{Adv}^{\ell}_{17}(\mathcal{A},\lambda)| \leq \mathrm{Adv}^{SXDH}_{\mathcal{G}}(\mathcal{B}_{17,\ell},\lambda)$$

Proof. This proof is nearly identical to Lemma 18, except instead of removing the whole row of $x^*\vec{M}_j$, we are just removing a single component of $x^*\lambda^*$ (while retaining the x' in the $\mathbb{D}$ basis). □

Lemma 28. *For any adversary $\mathcal{A}$,*

$$|\mathrm{Adv}^{Q_2}_{17}(\mathcal{A},\lambda) - \mathrm{Adv}^{Q_2}_{18}(\mathcal{A},\lambda)| = 0$$

Proof. In both the these games, all keys are semi-functional of $Type_{12} = Type_{17}$. The only difference between these two games the challenge ciphertext is switched to one that completely hides the message m (the α blinding m is decoupled from the α^* in the $\mathbb{D}^*$ component). Since all secret keys are also semi-functional, we can accomplish this with a simple change of basis that takes advantage of the decoupled x' in the secret keys:

Specifically, consider the following sets of bases:

$$(\mathbb{J}, \mathbb{J}^*) \leftarrow Dual(\mathbb{Z}_p^6)$$
$$(\mathbb{B}, \mathbb{B}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)})$$
$$(\mathbb{A}_i, \mathbb{A}_i^*) \leftarrow Dual(\mathbb{Z}_p^3) \text{ for } i \in \mathcal{U}$$

and define the bases $(\mathbb{D}, \mathbb{D}^*)$ as:

$$\vec{d}_1 = \vec{j}_1 - \tilde{\alpha}^* \vec{j}_4$$
$$\vec{d}_i = \vec{j}_i \text{ for all other } i$$

$$\vec{d}_4^* = \vec{j}_4^* - \tilde{\alpha}^* \vec{j}_1^*$$
$$\vec{d}_i^* = \vec{j}_i^* \text{ for all other } i$$

The transformation results in the same distribution of basis vectors between $((\mathbb{J}, \mathbb{J}^*), (\mathbb{B}, \mathbb{B}^*), (\mathbb{A}_i, \mathbb{A}_i^*))$ and $((\mathbb{D}, \mathbb{D}^*), (\mathbb{B}, \mathbb{B}^*), (\mathbb{A}_i, \mathbb{A}_i^*))$, and yet using the second set to simulate $\text{Game}_{18}^{Q_2}$ implicitly simulates $\text{Game}_{17}^{Q_2}$ in the first basis (where the x' in all secret keys absorbs the $\tilde{\alpha}^*$ introduced when using vector $\vec{j}_1$). Therefore, the two games are identical and no adversary can achieve a difference in advantage between the two. □

Lemma 29. *For any adversary $\mathcal{A}$,*

$$\text{Adv}_{18}^{Q_2}(\mathcal{A}, \lambda) = 0$$

Proof. In this game, the m in the challenge ciphertext is masked by $e(g,h)^\alpha$ for a completely independent α, so its distribution is independent of the game's challenge β. Therefore, no adversary can achieve a nonzero advantage in this game. □

Theorem 30. *Under the SXDH assumption, our KP-ABE construction is adaptively secure against any polynomial time adversary $\mathcal{A}$.*

Proof. Summing up Lemmas 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26,

27, 28, and 29, and using the triangle inequality, we get that for any polynomial time adversary $\mathcal{A}$:

$$\begin{aligned}
\mathrm{Adv}_{real}(\mathcal{A},\lambda) &= |\mathrm{Adv}_{real}(\mathcal{A},\lambda) - \mathrm{Adv}_{18}^{Q_2}(\mathcal{A}\lambda)| \\
&\leq \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_1,\lambda) + \sum_{\ell=1}^{Q_1} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{2,\ell},\lambda) \\
&\quad + \sum_{\ell=1}^{Q_1}\sum_{k=1}^{r} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{3,k,\ell},\lambda) + \sum_{\ell=1}^{Q_1}\sum_{k=1}^{r} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{4,k,\ell},\lambda) \\
&\quad + \sum_{\ell=1}^{Q_1}\sum_{k=1}^{r} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{5,k,\ell},\lambda) + \sum_{\ell=1}^{Q_1}\sum_{k=1}^{r} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{9,k,\ell},\lambda) \\
&\quad + \sum_{\ell=1}^{Q_1}\sum_{k=1}^{r} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{10,k,\ell},\lambda) + \sum_{\ell=1}^{Q_1}\sum_{k=1}^{r} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{11,k,\ell},\lambda) \\
&\quad + \sum_{\ell=1}^{Q_1} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{12,\ell},\lambda) + \sum_{\ell=1}^{Q_2} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{14,\ell},\lambda) \\
&\quad + \sum_{\ell=1}^{Q_2}\sum_{k=1}^{\mathcal{U}} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{15,k,\ell},\lambda) + \sum_{\ell=1}^{Q_2} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{15,H,\ell},\lambda) \\
&\quad + \sum_{\ell=1}^{Q_2} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{16,H,\ell},\lambda) + \sum_{\ell=1}^{Q_2}\sum_{k=1}^{\mathcal{U}} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{16,k-1,\ell},\lambda) \\
&\quad + \sum_{\ell=1}^{Q_2} \mathrm{Adv}_{\mathcal{G}}^{SXDH}(\mathcal{B}_{17,\ell},\lambda)
\end{aligned}$$

Under the SXDH assumption, each of the Adv quantities is a negligible function of λ, and since Q_1, Q_2, r are polynomial functions of lambda, then the sum (bounding $\mathrm{Adv}_{real}(\mathcal{A},\lambda)$) is also a negligble function of λ, and therefore our construction is adaptively secure. □

References

[AC17] S. Agrawal and M. Chase. Fame: Fast attribute-based message encryption. In *CCS*, 2017.

[Att14] N. Attrapadung. Dual system encryption via doubly selective security: Frame- work, fully secure functional encryption for regular languages, and more. In *EUROCRYPT*, page 57577, 2014.

[Att16] N. Attrapadung. Dual system encryption framework in prime-order groups via computational pair encodings. In *ASIACRYPT*, page 91623, 2016.

[Bei96] A. Beimel. Secure schemes for secret sharing and key distribution. PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.

[BSW07] 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, 2007.

[BV16] Z. Brakerski and V. Vaikuntanathan. Circuit-abe from lwe: Unbounded attributes and semi-adaptive security. In *CRYPTO*, pages 363–384, 2016.

[CC09] M. Chase and S. S. M. Chow. Improving privacy and security in multi-authority attribute-based encryption. In *Proceedings of the 2009 ACM Conference on Computer and Communications Security*, pages 121–130, 2009.

[CGKW18] J. Chen, J. Gong, L. Kowalczyk, and H. Wee. Unbounded abe via bilinear entropy expansion, revisited. In *EUROCRYPT*, pages 503–534, 2018.

[CGW15] J. Chen, R. Gay, and H. Wee. Improved dual system abe in prime-order groups via predicate encodings. In *EUROCRYPT*, pages 595–624, 2015.

[Cha07] M. Chase. Multi-authority attribute based encryption. In *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, pages 515–534, 2007.

[Che06] J. H. Cheon. Security analysis of the strong diffie-hellman problem. In *EUROCRYPT*, pages 1–11, 2006.

[GGH+13] S. Garg, C. Gentry, S. Halevi, A. Sahai, and B. Waters. Attribute-based encryption for circuits from multilinear maps. In *CRYPTO*, pages 479–499, 2013.

[GGHZ14] S. Garg, C. Gentry, S. Halevi, and M. Zhandry. Fully secure attribute based encryption from multilinear maps. *IACR Cryptology ePrint Archive*, 2014:622, 2014.

[GJPS08] V. Goyal, A. Jain, O. Pandey, and A. Sahai. Bounded ciphertext policy attribute-based encryption. In *ICALP*, 2008.

[GKW6b] R. Goyal, V. Koppula, and B. Waters. Semi-adaptive security and bundling functionalities made generic and easy. In *TCC*, pages 361–388, 2016b.

[GPSW06] 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.

[GVW13] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Attribute-based encryption for circuits. In *STOC*, pages 545–554, 2013.

[IW14] Y. Ishai and H. Wee. Partial garbling schemes and their applications. In *ICALP*, pages 650–662, 2014.

[JW14] C. Jie and H. Wee. Semi-adaptive attribute-based encryption and improved delegation for boolean formula. In *SCN*, pages 277–297, 2014.

[KL15] L. Kowalczyk and A. B. Lewko. Bilinear entropy expansion from the decisional linear assumption. In *CRYPTO*, pages 524–541, 2015.

[LOS+10] 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.

[LW11a] A. Lewko and B. Waters. Decentralizing attribute-based encryption. In *EUROCRYPT*, pages 568–588, 2011.

[LW11b] A. Lewko and B. Waters. Unbounded hibe and attribute-based encryption. In *EUROCRYPT*, pages 547–567, 2011.

[LW12] A. B. Lewko and B. Waters. New proof methods for attribute-based encryption: Achieving full security through selective techniques. In *CRYPTO*, pages 180–198, 2012.

[MW93] M.Karchmer and A. Wigderson. On span programs. In *CCC*, pages 102–111, 1993.

[OSW07] 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.

[OT08] T. Okamoto and K. Takashima. Homomorphic encryption and signatures from vector decomposition. In *Pairing*, pages 57–74, 2008.

[OT09] T. Okamoto and K. Takashima. Hierarchical predicate encryption for inner-products. In *ASIACRYPT*, pages 214–231, 2009.

[OT10] T. Okamoto and K. Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *CRYPTO*, pages 191–208, 2010.

[OT12] T. Okamoto and K. Takashima. Fully secure unbounded inner-product and attribute-based encryption. In *ASIACRYPT*, pages 349–366, 2012.

[OT13] T. Okamoto and K. Takashima. Decentralized attribute-based signatures. In *PKC*, pages 125–142, 2013.

[RW13] Y. Rouselakis and B. Waters. Practical constructions and new proof methods for large universe attribute-based encryption. In *2013 ACM Conference on Computer and Communications Security*, pages 463–474, 2013.

[Tak17] K. Takashima. New proof techniques for dlin-based adaptively secure attribute-based encryption. In *ACISP*, pages 85–105, 2017.

[Wat09] B. Waters. Dual system encryption: realizing fully secure ibe and hibe under simple assumptions. In *CRYPTO*, pages 619–636, 2009.

[Wat11] B. Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *PKC*, pages 53–70, 2011.

A Adaptation of Inner Product Argument

We will now reproduce an adaptation of Lemma 8 from [Tak17]:

Lemma 31. *Given nonzero $\vec{x}_1, \vec{y}_1, \vec{x}_2, \vec{y}_2 \in \mathbb{Z}_p^{r+1}$ such that $\langle \vec{x}_1, \vec{y}_1 \rangle = \langle \vec{x}_1, \vec{y}_1 \rangle$, after drawing $(\mathbb{B}, \mathbb{B}^*) \leftarrow \mathit{Dual}(\mathbb{Z}_p^{3(r+1)})$, the following two distributions are identical:*

$$\left((\vec{0}, \vec{0}, \vec{x}_1)_{\mathbb{B}}, (\vec{0}, \vec{0}, \vec{y}_1)_{\mathbb{B}^*}, \{\mathbf{b}_i, \mathbf{b}_i^*\}_{i \in [1,2(r+1)]}\right) \sim \left((\vec{0}, \vec{0}, \vec{x}_2)_{\mathbb{B}}, (\vec{0}, \vec{0}, \vec{y}_2)_{\mathbb{B}^*}, \{\mathbf{b}_i, \mathbf{b}_i^*\}_{i \in [1,2(r+1)]}\right)$$

Proof. First, we proceed in two cases: when the dot product k is zero, and when it is nonzero.

If $\langle \vec{x}_1, \vec{y}_1 \rangle = \langle \vec{x}_1, \vec{y}_1 \rangle = k$ is nonzero, then consider the invertible matrix R_1 formed as follows: the first row of R_1 is $\vec{y}_1$, and the remaining rows of R_1 are (any) $n-1$ linearly independent vectors in the orthogonal space of $\vec{x}_1$ (note that the first row is independent of these vectors since $\vec{y}_1$ has nonzero projection on to $\vec{x}_1$ and so R_1 is full rank and invertible).

Note that this matrix satisfies the following properties:

$$R_1 \vec{x}_1 = \begin{bmatrix} k \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad (R_1^{-1})^T \vec{y}_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Therefore there is also an invertible R_2 such that:

$$R_2 \vec{x}_2 = \begin{bmatrix} k \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad (R_2^{-1})^T \vec{y}_2 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

In the second case, say $\langle \vec{x}_1, \vec{y}_1 \rangle = \langle \vec{x}_1, \vec{y}_1 \rangle = 0$. Then then consider the invertible matrix R_1 formed as follows: the first row of R_1 is $\frac{1}{\langle \vec{x}_1, \vec{x}_1 \rangle} \vec{x}_1$, the second row of R_1 is $\vec{y}_1$, and the remaining rows of R_1 are (any) $n-2$ linearly independent vectors in the orthogonal space of $\vec{x}_1, \vec{y}_1$.

Note that this matrix is full rank and satisfies the following properties:

$$R_1 \vec{x}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad (R_1^{-1})^T \vec{y}_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Therefore there is also an invertible R_2 such that:

$$R_2 \vec{x}_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad (R_2^{-1})^T \vec{y}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

In both cases, the matrix $R_2^{-1} R_1$ is invertible and satisfies:

$$R_2^{-1} R_1 \vec{x}_1 = \vec{x}_2, \quad ((R_2^{-1} R_1)^{-1})^T \vec{y}_1 = \vec{y}_2$$

.

Now, recall from Section 2.2 that the distributions of $(\mathbb{B}, \mathbb{B}^*)$ and $(R \cdot \mathbb{B}, (R^{-1})^T \cdot \mathbb{B}^*)$ where R is an invertible matrix and $(\mathbb{B}, \mathbb{B}^*) \leftarrow Dual(\mathbb{Z}_p^{3(r+1)})$ are identical.

Finally, let R be the matrix:

$$R := \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & R_2^{-1} R_1 \end{bmatrix}$$

(where each block is of dimension $(r+1)$).

Then using $(\mathbb{B}, \mathbb{B}^*)$ to construct $\left((\vec{0}, \vec{0}, \vec{x}_1)_{\mathbb{B}}, (\vec{0}, \vec{0}, \vec{y}_1)_{\mathbb{B}^*}, \{\mathbf{b}_i, \mathbf{b}_i^*\}_{i \in [1,2(r+1)]}\right)$ yields the left side in the equation of the lemma, yet using the identically distributed $(R \cdot \mathbb{B}, (R^{-1})^T \cdot \mathbb{B}^*)$ basis results in the right side. Therefore, the two distributions are identical. □