Semi-Adaptive Attribute-Based Encryption and Improved Delegation for Boolean Formula[⋆]

Jie Chen[1,⋆⋆] and Hoeteck Wee[2,⋆⋆⋆]

[1] Department of Computer Science and Technology, East China Normal University
[2] École Normale Supérieure, Paris

Abstract. We consider *semi-adaptive* security for attribute-based encryption, where the adversary specifies the challenge attribute vector after it sees the public parameters but before it makes any secret key queries. We present two constructions of semi-adaptive attribute-based encryption under static assumptions with *short* ciphertexts. Previous constructions with short ciphertexts either achieve the weaker notion of selective security, or require parameterized assumptions.

As an application, we obtain improved delegation schemes for Boolean formula with *semi-adaptive* soundness, where correctness of the computation is guaranteed even if the client's input is chosen adaptively depending on its public key. Previous delegation schemes for formula achieve one of adaptive soundness, constant communication complexity, or security under static assumptions; we show how to achieve semi-adaptive soundness and the last two simultaneously.

[⋆] The research leading to these results has received funding from the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013 Grant Agreement no. 339563 CryptoCloud).

[⋆⋆] Email: `s080001@e.ntu.edu.sg`. Supported by Science and Technology Commission of Shanghai Municipality under Grants 14YF1404200, 13JC1403500, and the National Natural Science Foundation of China under Grant 61172085. Part of this work was done at Nanyang Technological University, supported by the National Research Foundation of Singapore under Research Grant NRF-CRP2-2007-03.

[⋆⋆⋆] Email: `wee@di.ens.fr`. CNRS (UMR 8548) and INRIA. Supported in part by the French ANR-12-INSE-0014 SIMPATIC Project. Part of this work was done at Columbia University, supported by NSF Award CNS-1319021, and at Ruhr-Universität Bochum as a Research Fellow of the Alexander von Humboldt Foundation.

1 Introduction

Attribute-based encryption (ABE) [33, 19] is an emerging paradigm for public-key encryption which enables fine-grained control of access to encrypted data. In traditional public-key encryption, access to the encrypted data is all or nothing: given the secret key, one can decrypt and read the entire plaintext, but without it, nothing about the plaintext is revealed (other than its length). In ABE, a ciphertext is labeled with an attribute vector $\mathbf{x}$, and a secret key is associated with an access policy specified as a Boolean formula, and the secret key decrypts the ciphertext if and only if $\mathbf{x}$ satisfies the access policy.[1] It is easy to see that ABE is a generalization of identity-based encryption (IBE) [34, 5, 13]. The security requirement for ABE stipulates that it resists collusion attacks, namely any group of users collectively learns nothing about the plaintext if none of them is individually authorized to decrypt the ciphertext.

Delegation. A delegation scheme allows a computationally weak client to delegate expensive computations to the cloud, with the assurance that a malicious cloud cannot convince the client to accept an incorrect computation [18, 16, 4, 14]. Recent work of Parno, Raykova and Vaikuntanathan [32] showed that any ABE with encryption time at most linear in the length of the attribute vector immediately yields a delegation scheme for Boolean formula. There is an initial pre-processing phase which fixes the formula f the client wishes to compute and produces some public key. Afterwards, to delegate computation on an input x, the client only needs to send a single message. Moreover, the ensuing delegation scheme satisfies public delegatability, namely anyone can delegate computations to the cloud; as well as public verifiability, namely anyone can check the cloud's work (given a "verification" key published by the client).

State of the art. Since the introduction of ABE and motivated in part by the connection to delegation, there is now a large body of work providing constructions with incomparable trade-offs amongst efficiency, security guarantees and security assumptions [19, 2, 27, 31, 26]; a summary of this work is presented in Fig 1. A key measure of efficiency is the ciphertext size and the encryption time; ideally, we want this to depend at most linearly in the length of the attribute vector and independent of the size of the access structure. For security guarantees, the two primary notions are selective and adaptive security; in the more restrictive setting of selective security, the adversary must specify the challenge attribute vector prior to seeing the public parameters. Finally, the security of the schemes rely on the assumed hardness of some computational problem in bilinear groups; here, we prefer prime-order instantiations over composite-order ones, and static assumptions over parameterized ones.

1.1 Our Contributions

We introduce the notion of *semi-adaptive* security for ABE and delegation. In ABE, this means that the adversary specifies the challenge attribute vector after it sees the public parameters but before it makes any secret key queries. This is stronger than selective security but weaker than adaptive security. In delegation, this means that the client's input may depend on the public key but is independent of the worker's evaluation key. In addition, we provide new constructions of efficient semi-adaptively secure ABE and delegation schemes under static assumptions.

New ABE Schemes. Our first result is a semi-adaptively secure ABE whose efficiency matches the state-of-the-art selectively secure ABE [2]:

[1] This is typically referred to as key-policy ABE in the literature, which is the focus of this paper. A different line of works, e.g. [12, 20, 37, 27, 26], considers ciphertext-policy ABE, where the ciphertext is labeled with a formula and the secret key is associated with an attribute vector.

reference	security	Enc time	CT size	MPK size	SK size	group	assumption
GPSW06 [19]	selective	$O(n)^*$	$O(n)^*$	$O(n)$	$O(\ell)$	prime	static
ALP11 [2]	selective	$O(n)$	$O(1)$	$O(n)$	$O(n\ell)$	prime	parameterized in n
ALP11 + LW10	selective	$O(n)$	$O(1)$	$O(n)$	$O(n\ell)$	composite	static
T14 [35]	selective	$O(n)$	$O(1)$	$O(n)$	$O(n\ell)$	prime	static
LOSTW10 [27]	adaptive	$O(nM)^*$	$O(nM)^*$	$O(nM)$	$O(\ell)$	composite	static
OT10 [31]	adaptive	$O(nM)^*$	$O(nM)^*$	$O(nM)$	$O(\ell)$	prime	static
LW12 [26]	adaptive	$O(n)^*$	$O(n)^*$	$O(n)$	$O(\ell)$	prime	parameterized in ℓ
A14 [1]	adaptive	$O(n)$	$O(1)$	$O(n)$	$O(n\ell)$	composite	parameterized in ℓ
Construction 1	semi-adaptive	$O(n)$	$O(1)$	$O(n)$	$O(n\ell)$	composite	static
Construction 2	semi-adaptive	$O(n)^*$	$O(n)^*$	$O(n)$	$O(\ell)$	prime	static

Fig. 1. Summary of existing KP-ABE schemes. Here, n denotes the universe size, M is the maximum number of times an attribute may be used, and $\ell \leq nM$ is the number of rows in the matrix $\mathbf{M}$ of the access structure. Encryption time is given in terms of group operations, and CT, PP, SK sizes are given in terms of group elements. For CT, we omit the additive overhead of n *bits* in order to transmit the attribute vector. For the quantities marked with *, n may be replaced with number of non-zero entries in the attribute vector $\mathbf{x} \in \{0,1\}^n$, which could be much smaller than n. Note that ALP11, T14 and A14 achieve large universe, we restrict the attribute universe to $[n]$ for comparison.

> **(Informal Theorem)** There exists a semi-adaptively secure ABE with constant-size ciphertexts. Encryption time is linear in the length of the attribute vector and independent of the size of the access structure. The security of the scheme is based on static assumptions in composite-order groups.

We also achieve an analogous result in prime-order groups based on the SXDH Assumption; however, the ciphertext size is linear in the length of the attribute vector. Throughout this work, when we refer to ciphertext size, we measure the number of group elements, and we omit the additive overhead of n *bits* needed to transmit the attribute vector.

New Delegation Schemes. Starting from our semi-adaptively secure ABE, we obtain improved delegation schemes for Boolean formula with *semi-adaptive* soundness, where correctness of the computation is guaranteed even if the client's input is chosen adaptively depending on its public key. We note that achieving semi-adaptive soundness is important in practice, since we would like to reuse the same public key across multiple inputs, which could lead to correlation between the input and the public key. Previous delegation schemes for formula achieve one of adaptive soundness [26, 17], constant communication complexity[2] [2], or security under static assumptions [19]; we achieve semi-adaptive soundness and the last two simultaneously. We compare our schemes with prior works in Fig 2. We stress that in applications such as delegating computation from mobile devices on cellular networks where bandwidth is a premium, reducing the client's communication from $O(n\lambda)$ bits to $n + O(\lambda)$ bits represents substantial savings.

1.2 Our Techniques

Following our recent works [38, 9] and inspired in part by [26], we rely on Waters' dual system encryption methodology [36, 25] to reduce the problem of building a (public-key) semi-adaptively secure ABE to that of building a private-key selectively secure ABE. Recall that dual system encryption is typically implemented by designing a "semi-functional space" where semi-functional components of keys and ciphertexts will behave like a parallel copy of the normal components of the system, except divorced from the public parameters. In particular, we will embed the private-key selectively secure ABE into the semi-functional space.

We proceed to outline the constructions of private-key ABE with short ciphertexts:

[2] Here, we refer to the client's communication overhead beyond sending the n-bit input, as measured in group elements.

reference	security	$\lvert\mathsf{EK_F}\rvert$	client's communication in bits	worker's complexity	groups	assumptions
GPSW06 [19]	selective	$O(\ell)$	$O(n\lambda)$	$O(\ell)$	prime	static
ALP11 [2]	selective	$O(n\ell)$	$n + O(\lambda)$	$O(n\ell)$	prime	parameterized in n
T14 [35]	selective	$O(n\ell)$	$n + O(\lambda)$	$O(n\ell)$	prime	static
GGPR13 [17]	adaptive	$O(\ell)$	$n + O(\lambda)$	$O(\ell)$	prime	parameterized in ℓ
LW12 [26]	adaptive	$O(\ell)$	$O(n\lambda)$	$O(\ell)$	prime	parameterized in ℓ
A14 [1]	adaptive	$O(n\ell)$	$n + O(\lambda)$	$O(n\ell)$	composite	parameterized in ℓ
Construction 1	semi-adaptive	$O(n\ell)$	$n + O(\lambda)$	$O(n\ell)$	composite	static
Construction 2	semi-adaptive	$O(\ell)$	$O(n\lambda)$	$O(\ell)$	prime	static

Fig. 2. Summary of existing publicly verifiable computation schemes. GGPR13 supports NC. The remaining schemes only support NC^1 and are obtained using the transformation of [32]. Here, $\lvert\mathsf{EK_F}\rvert$ is the worker's evaluation key, n is the bit length of the input and ℓ is the size of the formula. In all the schemes, the public key is $O(n)$ group elements, delegation and verification complexity of client is $O(n)$ group operations, computation complexity of worker is also given in terms of group operations.

- For our composite-order scheme with constant-size ciphertext, we use a private-key variant of the selectively secure ABE scheme of Attrapadung, Libert and Panafieu (ALP) in [2]. Our main insight is that in the private-key setting with a single challenge ciphertext, we can replace the use of parameterized assumptions in the ALP scheme with the basic DDH assumption. Roughly speaking, fix an attribute i that does not appear in the challenge attribute. We can then rely on the DDH assumption to mask all the LSSS shares of the master secret key corresponding to attribute i (c.f. Section 3 overview and Lemma 2).[3] The formal security proof is more involved since we need to instantiate this argument within the dual system framework.
- For our prime-order scheme with $O(n)$-size ciphertext, the private-key selectively secure ABE we use is essentially that of Goyal et al. [19], which is in fact a public-key scheme and yields ciphertexts of length $O(n)$. To combine this scheme with the dual system framework, we rely on dual pairing vector spaces [29, 30, 15, 23, 11]. Here, we will also use the SXDH assumption to boost *statistical* entropy in the semi-functional key space into arbitrarily large amounts of *computational* entropy in the same space (c.f. Lemma 6) as we will need to mask an arbitrarily large number of shares corresponding to a single attribute.

For both schemes, we are able to exploit random self-reducibility to obtain security loss that do not depend on the number of secret key queries or the size of the boolean formula (but may depend on the input size n). In contrast, all known adaptively secure ABE schemes incur a loss that is at least linear in both the number of secret key queries and the size of the boolean formula (sometimes implicitly, by either making a "one-use" restriction or using a parameterized assumption).

Discussion. We believe that semi-adaptive security also constitutes a useful and meaningful intermediate security goal in our pursuit of new techniques for achieving full, adaptive security. We leave as an open problem obtaining a semi-adaptively secure ABE with constant-size ciphertext under the SXDH and DLIN assumptions. In particular, the existing tools in [23, 11, 10] are not directly applicable to our scheme.

Additional related work. In an independent work, Takashima [35] proposed a selectively secure KP-ABE scheme with constant-size ciphertexts under the DLIN assumption, which results in a delegation scheme

[3] In an earlier submission, an anonymous reviewer asked if it is possible to obtain the composite-order scheme by combining the Lewko-Waters ABE [26] with the ALP scheme. We clarify here that this approach (should it pan out) would inherit the parameterized assumption from [2]. In particular, none of the prior works either implicitly or explicitly build a private-key ABE with constant-size ciphertexts from static assumptions.

with constant communication complexity and security under static assumptions but only achieving selective soundness. Gennaro, Gentry, Parno and Raykova [17] constructed a delegation scheme achieving adaptive soundness and supporting NC but its security relies on parameterized assumptions.

Organization. We present our composite-order and prime-order constructions in Section 3 and Section A respectively, and the delegation schemes and associated definitions in Section B.

2 Preliminaries

Notation. We denote by $s \leftarrow_{\mathrm{R}} S$ the fact that s is picked uniformly at random from a finite set S and by $x, y, z \leftarrow_{\mathrm{R}} S$ that all x, y, z are picked independently and uniformly at random from S. By PPT, we denote a probabilistic polynomial-time algorithm. Throughout, we use 1^λ as the security parameter. We use $\cdot$ to denote multiplication (or group operation) as well as component-wise multiplication. We use lower case boldface to denote (column) vectors over scalars and upper case boldface to denote vectors of group elements as well as matrices. Given two vectors $\mathbf{x} = (x_1, x_2, \ldots), \mathbf{y} = (y_1, y_2, \ldots)$ over scalars, we use $\langle \mathbf{x}, \mathbf{y} \rangle$ to denote the standard dot product $\mathbf{x}^\top \mathbf{y}$. Given a group element g, we write $g^{\mathbf{x}}$ to denote $(g^{x_1}, g^{x_2}, \ldots)$; we define $g^{\mathbf{A}}$ where $\mathbf{A}$ is a matrix in an analogous way.

2.1 Access Structures

We define (monotone) access structures using the language of (monotone) span programs [21].

Definition 1 (access structure [3, 21]). *A* (monotone) access structure $\mathbb{A}$ *for attribute universe* $[n]$ *is a pair* $(\mathbf{M}, \rho)$ *where* $\mathbf{M}$ *is a* $\ell \times \ell'$ *matrix over* $\mathbb{Z}_N$ *and* $\rho : [\ell] \to [n]$*. Given* $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$*, we say that*

$$\mathbf{x} \textit{ satisfies } \mathbb{A} \textit{ iff } \mathbf{1} \in \mathsf{span}\langle \mathbf{M}_{\mathbf{x}} \rangle.$$

Here, $\mathbf{1} := (1, 0, \ldots, 0) \in \mathbb{Z}_N^{\ell'}$ *is a row vector;* $\mathbf{M}_{\mathbf{x}}$ *denotes the collection of vectors* $\{\mathbf{M}_j : x_{\rho(j)} = 1\}$ *where* $\mathbf{M}_j$ *denotes the j'th row of* $\mathbf{M}$*; and* span *refers to linear span of collection of (row) vectors over* $\mathbb{Z}_N$*.*

That is, $\mathbf{x}$ satisfies $\mathbb{A}$ iff there exists constants $\omega_1, \ldots, \omega_\ell \in \mathbb{Z}_N$ such that

$$\sum_{j : x_{\rho(j)} = 1} \omega_j \mathbf{M}_j = \mathbf{1}.$$

Observe that the constants $\{\omega_j\}$ can be computed in time polynomial in the size of the matrix $\mathbf{M}$ via Gaussian elimination.

2.2 Key-Policy Attribute-Based Encryption

A KP-ABE scheme consists of four algorithms $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{KeyGen}, \mathsf{Dec})$:

$\mathsf{Setup}(1^\lambda, [n]) \to (\mathrm{MPK}, \mathrm{MSK})$. The setup algorithm takes in a security parameter 1^λ, and an attribute universe $[n]$. It outputs public parameters MPK and a master secret key MSK.

$\mathsf{Enc}(\mathrm{MPK}, \mathbf{x}, m) \to \mathrm{CT}_{\mathbf{x}}$. The encryption algorithm takes in the public parameters MPK, an attribute vector $\mathbf{x}$, and a message m. It outputs a ciphertext $\mathrm{CT}_{\mathbf{x}}$.

$\mathsf{KeyGen}(\mathrm{MPK}, \mathrm{MSK}, \mathbb{A}) \to \mathrm{SK}_{\mathbb{A}}$. The key generation algorithm takes in the public parameters MPK, the master secret key MSK, and an access structure $\mathbb{A} := (\mathbf{M}, \rho)$. It outputs a secret key $\mathrm{SK}_{\mathbb{A}}$.

$\mathsf{Dec}(\textsc{mpk}, \textsc{sk}_{\mathbb{A}}, \textsc{ct}_{\mathbf{x}}) \to m$. The decryption algorithm takes in the public parameters MPK, a secret key $\textsc{sk}_{\mathbb{A}}$ for an access structure $\mathbb{A}$, and a ciphertext $\textsc{ct}_{\mathbf{x}}$ encrypted under an attribute vector $\mathbf{x}$. It outputs a message m if $\mathbf{x}$ satisfies $\mathbb{A}$.

Correctness. For all $(\textsc{mpk}, \textsc{msk}) \leftarrow \mathsf{Setup}(1^\lambda, [n])$, all access structures $\mathbb{A}$, all decryption keys $\textsc{sk}_{\mathbb{A}}$, all messages m, all $\mathbf{x}$ satisfying $\mathbb{A}$, we have $\Pr[\mathsf{Dec}(\textsc{mpk}, \textsc{sk}_{\mathbb{A}}, \mathsf{Enc}(\textsc{mpk}, \mathbf{x}, m)) = m] = 1$.

2.3 Semi-Adaptive Security Model

We now formalize the notation of *semi-adaptive* security for KP-ABE. Briefly, the adversary specifies the challenge attribute vector after it sees the public parameters and before it makes any secret key queries. The security game is defined by the following experiment, played by a challenger and an adversary $\mathcal{A}$.

Setup. The challenger runs the setup algorithm to generate $(\textsc{mpk}, \textsc{msk})$. It gives MPK to $\mathcal{A}$.

Challenge Attribute. $\mathcal{A}$ gives the challenger a challenge $\mathbf{x}^*$.

Phase 1. $\mathcal{A}$ adaptively requests keys for access structures $\mathbb{A}$ with the constraint $\mathbf{x}^*$ does not satisfy $\mathbb{A}$. The challenger responds with the corresponding secret key $\mathsf{SK}_{\mathbb{A}}$, which it generates by running the key generation algorithm.

Challenge Ciphertext. $\mathcal{A}$ submits two equal-length messages m_0 and m_1. The challenger picks $\beta \leftarrow_{\text{R}} \{0,1\}$, and encrypts m_β under $\mathbf{x}^*$ by running the encryption algorithm. It sends the ciphertext to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ continues to issue key queries as in **Phase 1**.

Guess. $\mathcal{A}$ must output a guess β' for β.

The advantage $\mathsf{Adv}_{\mathcal{A}}^{\text{KP-ABE}}(\lambda)$ of an adversary $\mathcal{A}$ is defined to be $\Pr[\beta' = \beta] - 1/2$.

Definition 2. *A KP-ABE scheme is* semi-adaptively secure *if all PPT adversaries achieve at most a negligible advantage in the above security game.*

2.4 Composite Order Bilinear Groups

Composite order bilinear groups were first introduced in [7] and used in [22, 25, 27]. A generator $\mathcal{G}$ takes as input a security parameter 1^λ and outputs a description $\mathbb{G} := (N, G_N, G_T, e)$, where N is product of distinct primes of $\Theta(\lambda)$ bits, G_N and G_T are cyclic groups of order N, and $e : G_N \times G_N \to G_T$ is a map with the following properties:

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

We require that the group operations in G_N and G_T as well the bilinear map e are computable in deterministic polynomial time with respect to λ. Furthermore, the group descriptions of G_N and G_T include generators of the respective cyclic groups. We use G_n to denote the subgroup of G_N of order n, where n divides N.

Computational Assumptions. We now state the three static assumptions that are required in our security proof. The first two assumptions are introduced in [25] and also used in [27]. The third assumption which basically asserts that the DDH problem is hard in the G_{p_2}-subgroup. This assumption is essentially implied by the composite 3-party Diffie-Hellman (3PDH) assumption in [6]. We provide more discussion and justification of this assumption in Section C. All three assumptions hold in the generic group model under the assumption finding a non-trivial factor of N is hard.

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

$$\begin{aligned}
&\mathbb{G} := (N = p_1 p_2 p_3, G_N, G_T, e) \leftarrow_{\mathrm{R}} \mathcal{G},\\
&g_1, U_1 \leftarrow_{\mathrm{R}} G_{p_1},\ U_2 \leftarrow_{\mathrm{R}} G_{p_2},\ g_3 \leftarrow_{\mathrm{R}} G_{p_3},\\
&T_0 \leftarrow_{\mathrm{R}} G_{p_1},\ T_1 \leftarrow_{\mathrm{R}} G_{p_1 p_2},\\
&D := (\mathbb{G}; g_1, U_1 U_2, g_3).
\end{aligned}$$

We assume that for any PPT algorithm $\mathcal{A}$,

$$\mathsf{Adv}^{\mathsf{AS1}}_{\mathcal{A}}(\lambda) := \big|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\big|$$

is negligible in the security parameter λ.

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

$$\begin{aligned}
&\mathbb{G} := (N = p_1 p_2 p_3, G_N, G_T, e) \leftarrow_{\mathrm{R}} \mathcal{G},\\
&\alpha, s \leftarrow_{\mathrm{R}} \mathbb{Z}_N,\\
&g_1 \leftarrow_{\mathrm{R}} G_{p_1},\ g_2, X_2, Y_2 \leftarrow_{\mathrm{R}} G_{p_2},\ g_3 \leftarrow_{\mathrm{R}} G_{p_3},\\
&T_0 := e(g_1, g_1)^{\alpha s},\ T_1 \leftarrow_{\mathrm{R}} G_T,\\
&D := (\mathbb{G}; g_1, g_1^{\alpha} X_2, g_1^{s} Y_2, g_2, g_3).
\end{aligned}$$

We assume that for any PPT algorithm $\mathcal{A}$,

$$\mathsf{Adv}^{\mathsf{AS2}}_{\mathcal{A}}(\lambda) := \big|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\big|$$

is negligible in the security parameter λ.

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

$$\begin{aligned}
&\mathbb{G} := (N = p_1 p_2 p_3, G_N, G_T, e) \leftarrow_{\mathrm{R}} \mathcal{G},\\
&x, y, z \leftarrow_{\mathrm{R}} \mathbb{Z}_N,\\
&g_1, U_1 \leftarrow_{\mathrm{R}} G_{p_1},\ g_2, U_2 \leftarrow_{\mathrm{R}} G_{p_2},\ g_3, X_3, Y_3, U_3, W_3 \leftarrow_{\mathrm{R}} G_{p_3},\\
&T_0 := g_2^{xy} W_3,\ T_1 := g_2^{xy+z} W_3,\\
&D := (\mathbb{G}; g_1, U_1 U_2, g_2^{x} X_3, g_2^{y} Y_3, g_2 U_3, g_3).
\end{aligned}$$

We assume that for any PPT algorithm $\mathcal{A}$,

$$\mathsf{Adv}^{\mathsf{AS3}}_{\mathcal{A}}(\lambda) := \big|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\big|$$

is negligible in the security parameter λ.

3 Semi-Adaptive ABE with Constant-Size Ciphertext

Overview. The starting point of our construction is the following variant of the ALP KP-ABE in [2]:

$$\begin{aligned} \mathrm{MPK} &:= (g, g^{\mathbf{w}}, e(g,g)^{\alpha}) \\ \mathrm{CT}_{\mathbf{x}} &:= (g^{s}, g^{s\langle \mathbf{w},\mathbf{x}\rangle}, e(g,g)^{\alpha s} \cdot m) \\ \mathrm{SK}_{\mathbb{A}} &:= (g^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}}, g^{r_j} \;:\; j \in [\ell]) \end{aligned}$$

where $\alpha_1, \ldots, \alpha_\ell$ are LSSS shares of α for the access structure $\mathbb{A}$. Our construction proceeds by embedding this scheme into composite-order groups. As noted in the introduction, our main insight is to analyze this scheme in the private-key, selective setting. Fix a selective challenge $\mathbf{x}^* \in \{0,1\}^n$ and an index $k \in [n]$ and an access structure $\mathbb{A}$ not satisfied by $\mathbf{x}^*$. We proceed via a case analysis to argue that $\mathrm{SK}_{\mathbb{A}}$ hides α computationally:

- if $x^*_k = 0$, then the shares $\{\alpha_j : \rho(j) = k\}$ reveal no information about α via the secret sharing property.
- if $x^*_k = 1$, then the ciphertext reveals no information about w_k (and since we are in the private-key setting, there is no MPK). Then, by the DDH assumption, $\{g^{\alpha_j + r_j w_k}, g^{r_j} : \rho(j) = k\}$ computationally hides α_j.

The formal security proof is more involved since we need to instantiate this argument within the dual system framework.

3.1 Construction

- $\mathsf{Setup}(1^\lambda, [n])$: On input an attribute universe $[n]$, generate $\mathbb{G} := (N = p_1 p_2 p_3, G_N, G_T, e) \leftarrow_{\mathrm{R}} \mathcal{G}$, pick $\alpha \leftarrow_{\mathrm{R}} \mathbb{Z}_N, \mathbf{w} \leftarrow_{\mathrm{R}} \mathbb{Z}_N^n$ and output
$$\mathrm{MPK} := (\ \mathbb{G}, e(g_1,g_1)^{\alpha}, g_1, g_1^{\mathbf{w}}\ ) \quad \text{and} \quad \mathrm{MSK} := (\ \alpha, \mathbf{w}, g_2, g_3\ ).$$
- $\mathsf{Enc}(\mathrm{MPK}, \mathbf{x}, m)$: On input an attribute vector $\mathbf{x} := (x_1, \ldots, x_n) \in \{0,1\}^n$ and $m \in G_T$, output
$$\mathrm{CT}_{\mathbf{x}} := \Big(\ C_0 := g_1^{s},\ C_1 := g_1^{s\langle \mathbf{w},\mathbf{x}\rangle},\ C_2 := e(g_1,g_1)^{\alpha s} \cdot m\ \Big) \in G_N \times G_N \times G_T,$$
where $s \leftarrow_{\mathrm{R}} \mathbb{Z}_N$.
- $\mathsf{KeyGen}(\mathrm{MPK}, \mathrm{MSK}, \mathbb{A} := (\mathbf{M}, \rho))$: On input an access structure $\mathbb{A} := (\mathbf{M}, \rho)$, where $\mathbf{M} \in \mathbb{Z}_N^{\ell \times \ell'}$ and $\rho : [\ell] \to [n]$, pick a random vector $\mathbf{u} \leftarrow_{\mathrm{R}} \mathbb{Z}_N^{\ell'}$ such that $\mathbf{1u} = \alpha$ and set $\alpha_j := \mathbf{M}_j \mathbf{u}$, $j \in [\ell]$.[4] Output
$$\mathrm{SK}_{\mathbb{A}} := \Bigg(\ \mathbf{D}_j := g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot g_2^{r'_j \mathbf{w}} \cdot \mathbf{X}_j,\ D_{0,j} := g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j \;:\; j \in [\ell]\ \Bigg) \in (G_N^n \times G_N)^{\ell},$$
where $r_1, r'_1, \ldots, r_\ell, r'_\ell \leftarrow_{\mathrm{R}} \mathbb{Z}_N$; $\mathbf{X}_j \leftarrow_{\mathrm{R}} G_{p_3}^n$; $Z_j \leftarrow_{\mathrm{R}} G_{p_3}$, and $(\mathbf{e}_1, \ldots, \mathbf{e}_n)$ is the standard basis for $\mathbb{Z}_N^n$.
- $\mathsf{Dec}(\mathrm{MPK}, \mathrm{SK}_{\mathbb{A}}, \mathrm{CT}_{\mathbf{x}})$: If $\mathbf{x}$ satisfies $\mathbb{A}$, compute $\omega_1, \ldots, \omega_\ell \in \mathbb{Z}_N$ such that
$$\sum_{j : x_{\rho(j)} = 1} \omega_j \mathbf{M}_j = \mathbf{1}.$$
Then, compute[5]
$$e(g_1,g_1)^{\alpha s} \leftarrow \prod_{j : x_{\rho(j)} = 1} \Big(e(C_0^{\mathbf{x}}, \mathbf{D}_j) \cdot e(C_1, D_{0,j})^{-1} \Big)^{\omega_j},$$

[4] The α_j's do in fact correspond to LSSS secret shares of α, distributed across n parties, where the i'th party receive $|\rho^{-1}(i)|$ shares, given by $\{\alpha_j : \rho(j) = i\}$.

[5] It is easy to see that $e(C_0^{\mathbf{x}}, \mathbf{D}_j)$ can in fact be computed using only a single pairing.

and recover the message as $m \leftarrow C_2/e(g_1,g_1)^{\alpha s} \in G_T$.

Correctness. Observe that

$$\begin{aligned} e(C_0^{\mathbf{x}}, \mathbf{D}_j) \cdot e(C_1, D_{0,j})^{-1} &= e((g_1^s)^{\mathbf{x}}, g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot g_2^{r'_j \mathbf{w}} \cdot \mathbf{X}_j) \cdot e(g_1^{s\langle \mathbf{w},\mathbf{x}\rangle}, g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j)^{-1} \\ &= e(g_1,g_1)^{\alpha_j s \langle \mathbf{e}_{\rho(j)}, \mathbf{x}\rangle} \cdot e(g_1,g_1)^{r_j s \langle \mathbf{w},\mathbf{x}\rangle} \cdot e(g_1,g_1)^{-r_j s\langle \mathbf{w},\mathbf{x}\rangle} \\ &= e(g_1,g_1)^{\alpha_j s}. \end{aligned}$$

In addition, we have

$$\sum_{j: x_{\rho(j)}=1} \omega_j \alpha_j = \sum_{j: x_{\rho(j)}=1} \omega_j \mathbf{M}_j \mathbf{u} = \mathbf{1u} = \alpha.$$

This means

$$\prod_{j: x_{\rho(j)}=1} \Big(e(C_0^{\mathbf{x}}, \mathbf{D}_j) \cdot e(C_1, D_{0,j})^{-1} \Big)^{\omega_j} = \prod_{j: x_{\rho(j)}=1} e(g_1,g_1)^{\omega_j \alpha_j s} = e(g_1,g_1)^{\alpha s}.$$

Correctness follows readily.

3.2 Proof of Security

We prove the following theorem:

Theorem 1. *Under Assumptions 1, 2 and 3 (described in Section 2.4), our KP-ABE scheme defined in Section 3.1 is semi-adaptively secure (in the sense of Definition 2). More precisely, for any adversary $\mathcal{A}$ that makes at most q key queries against the KP-ABE scheme, there exist probabilistic algorithms $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ such that*

$$\mathsf{Adv}_{\mathcal{A}}^{\text{KP-ABE}}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}_1}^{\text{AS1}}(\lambda) + n \cdot \mathsf{Adv}_{\mathcal{B}_2}^{\text{AS3}}(\lambda) + \mathsf{Adv}_{\mathcal{B}_3}^{\text{AS2}}(\lambda) + 1/p_1 + (n+1)/p_2,$$

and

$$\max\{\mathsf{Time}(\mathcal{B}_1), \mathsf{Time}(\mathcal{B}_2), \mathsf{Time}(\mathcal{B}_3)\} \approx \mathsf{Time}(\mathcal{A}) + q \cdot \mathrm{poly}(\lambda, n),$$

where n is the size of universe attribute set and $\mathrm{poly}(\lambda,n)$ *is independent of* $\mathsf{Time}(\mathcal{A})$.

Overview. The proof follows via a series of games. To describe the games, we must first define semi-functional keys and ciphertexts. Fix random generators g_1, g_2, g_3, and let $\mathbf{x}^*$ denote the semi-adaptive challenge. We stress that unlike standard dual system encryption, we allow the semi-functional secret keys to depend on the semi-adaptive challenge $\mathbf{x}^*$ (this is okay because in the semi-adaptive security game, $\mathbf{x}^*$ is fixed before the adversary sees any secret keys). In the final transition (c.f. Lemma 3), we need to be able to simulate the secret keys given $g_1^\alpha X_2$ (as provided in Assumption 2) instead of g_1^α, so we define the semi-functional secret keys to have additional random G_{p_2}-components for the indices j corresponding to $x^*_{\rho(j)} = 0$ as captured by the term $\alpha'_j \mathbf{e}_{\rho(j)}$ below.

Semi-functional ciphertext.

$$\mathsf{CT}_{\mathbf{x}^*} := \left(g_1^s \cdot \boxed{g_2^{s'}},\ g_1^{s\langle \mathbf{w},\mathbf{x}^*\rangle} \cdot \boxed{g_2^{s'\langle \mathbf{w},\mathbf{x}^*\rangle}},\ e(g_1,g_1)^{\alpha s} \cdot m \right),$$

where $s' \leftarrow_{\mathrm{R}} \mathbb{Z}_N$.

Semi-functional secret key.

$$\mathrm{SK}_{\mathbb{A}} := \left(\begin{array}{lll} g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot g_2^{r'_j \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j & : x^*_{\rho(j)} = 1 \\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot g_2^{\boxed{\alpha'_j \mathbf{e}_{\rho(j)}} + r'_j \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j & : x^*_{\rho(j)} = 0 \end{array} \right),$$

where fresh $\alpha'_1, \ldots, \alpha'_\ell \leftarrow_{\mathrm{R}} \mathbb{Z}_N$ are chosen for each secret key (specifically, we pick fresh $\alpha'_j \leftarrow_{\mathrm{R}} Z_N$ for all j such that $x^*_{\rho(j)} = 0$).

Remark 1 (decryption capabilities). Fix $\mathbf{x}^*, \mathbb{A}$ such that $\mathbf{x}^*$ satisfies $\mathbb{A}$. Then,

- both semi-functional and normal secret key $\mathrm{SK}_{\mathbb{A}}$ can decrypt a normal ciphertext $\mathrm{CT}_{\mathbf{x}^*}$;
- a normal secret key $\mathrm{SK}_{\mathbb{A}}$ can decrypt a semi-functional ciphertext $\mathrm{CT}_{\mathbf{x}^*}$;
- a semi-functional secret key $\mathrm{SK}_{\mathbb{A}}$ *can* decrypt a semi-functional ciphertext $\mathrm{CT}_{\mathbf{x}^*}$; this is because the j'th subkey $(\mathbf{D}_j, D_{0,j})$ corresponding to $x^*_{\rho(j)} = 0$ is not used for decryption although it has an additional semi-functional component $g_2^{\alpha'_j}$. This is different from a standard dual system encryption argument, but is okay in our setting because $\mathbf{x}^*$ is fixed semi-adaptively *before* the adversary makes secret key queries.

Game sequence. We consider the following sequence of games:

- Game_0: is the real security game (c.f. Section 2.3).
- Game_1: is the same as Game_0 except that the challenge ciphertext is semi-functional.
- $\mathsf{Game}_{2,k}$, $k = 1, 2, \ldots, n$: we incrementally transform each normal secret key to a semi-functional one, i.e. $\mathsf{Game}_{2,k}$ is the same as Game_1 except that, for each secret key

  $$\mathrm{SK}_{\mathbb{A}} := \left(\; \mathbf{D}_j, \quad D_{0,j} \quad : j \in [\ell] \; \right),$$

  the j'th subkey $(\mathbf{D}_j, D_{0,j})$ is semi-functional if $\rho(j) \leq k$, and normal if $\rho(j) > k$. More precisely,

  $$\mathrm{SK}_{\mathbb{A}} := \left(\begin{array}{lll} g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot g_2^{r'_j \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j & : x^*_{\rho(j)} = 1 \\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot g_2^{r'_j \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j & : (x^*_{\rho(j)} = 0) \wedge (\rho(j) > k) \\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot g_2^{\alpha'_j \mathbf{e}_{\rho(j)} + r'_j \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j & : (x^*_{\rho(j)} = 0) \wedge (\rho(j) \leq k) \end{array} \right),$$

  where fresh $\alpha'_1, \ldots, \alpha'_\ell \leftarrow_{\mathrm{R}} \mathbb{Z}_N$ are chosen for each secret key. In other words, from $\mathsf{Game}_{2,k-1}$ to $\mathsf{Game}_{2,k}$, we modify the first component $\mathbf{D}_j$ of the j'th subkey for all j such that $\rho(j) = k$ (that is, corresponds to the variable x_k) as follows:

  - if $x^*_k = 1$, we leave it unchanged;
  - if $x^*_k = 0$, we change the semi-functional component from $g_2^{r'_j \mathbf{w}}$ to $g_2^{\alpha'_j \mathbf{e}_k + r'_j \mathbf{w}}$.

  Note that in $\mathsf{Game}_{2,n}$, all keys are semi-functional.
- Game_3: is the same as $\mathsf{Game}_{2,n}$ except that the challenge ciphertext is a semi-functional encryption of a random message in G_T.

Fix an adversary $\mathcal{A}$. We write $\mathsf{Adv}_{\mathrm{xx}}(\lambda)$ to denote the advantage of $\mathcal{A}$ in $\mathsf{Game}_{\mathrm{xx}}$. It is easy to see that $\mathsf{Adv}_3(\lambda) = 0$, because the view of the adversary is Game_3 is independent of the challenge bit β. We complete the proof by establishing the following sequence of lemmas.

Lemma 1 (Normal to semi-functional ciphertext). *There exists an adversary $\mathcal{B}_1$ such that:*

$$|\mathsf{Adv}_0(\lambda) - \mathsf{Adv}_1(\lambda)| \leq \mathsf{Adv}^{\mathrm{AS1}}_{\mathcal{B}_1}(\lambda) + 1/p_1 + 1/p_2$$

and $\mathsf{Time}(\mathcal{B}_1) \approx \mathsf{Time}(\mathcal{A}) + q \cdot \mathrm{poly}(\lambda, n)$ *where* $\mathrm{poly}(\lambda, n)$ *is independent of* $\mathsf{Time}(\mathcal{A})$.

Proof. We construct an adversary $\mathcal{B}_1$ for Assumption 1 using $\mathcal{A}$. Recall that in Assumption 1, the adversary is given $D := (\mathbb{G}; g_1, U_1U_2, g_3)$, along with T, where T is distributed as

$$g_1^s \quad \text{or} \quad g_1^s g_2^{s'}.$$

Here, $\mathcal{B}_1$ simulates Game_0 if $T := g_1^s$ and Game_1 if $T := g_1^s g_2^{s'}$. The quantity s, s' in the assumption will correspond the random exponents s, s' used in the ciphertext.

Specifically, $\mathcal{B}_1$ proceeds as follows:

Setup. $\mathcal{B}_1$ samples $\alpha \leftarrow_{\mathrm{R}} \mathbb{Z}_n$, $\mathbf{w} \leftarrow_{\mathrm{R}} \mathbb{Z}_N^n$ and outputs

$$\mathrm{MPK} := (\ e(g_1, g_1^\alpha), g_1, g_1^{\mathbf{w}}\ ).$$

We note that

$$(\ \alpha, \mathbf{w}, g_1, U_1U_2, g_3; T\ )$$

is known to $\mathcal{B}_1$. The adversary $\mathcal{A}$ outputs a challenge $\mathbf{x}^* := (x_1^*, \ldots, x_n^*)$.

Challenge Ciphertext. Upon receiving two equal-length messages m_0 and m_1 from $\mathcal{A}$, $\mathcal{B}_1$ picks $\beta \leftarrow_{\mathrm{R}} \{0,1\}$ and outputs the semi-functional challenge ciphertext as:

$$\mathrm{CT}_{\mathbf{x}^*} := \Big(\ T,\ T^{\langle \mathbf{w}, \mathbf{x}^* \rangle},\ e(T, g_1^\alpha) \cdot m_\beta \Big).$$

Now, suppose $T = g_1^s \cdot g_2^{s'}$, then,

$$\begin{aligned} T^{\langle \mathbf{w}, \mathbf{x}^* \rangle} &:= (g_1^s \cdot g_2^{s'})^{\langle \mathbf{w}, \mathbf{x}^* \rangle} = g_1^{s\langle \mathbf{w}, \mathbf{x}^* \rangle} g_2^{s'\langle \mathbf{w}, \mathbf{x}^* \rangle}, \\ e(T, g_1^\alpha) &:= e(g_1^s \cdot g_2^{s'}, g_1^\alpha) = e(g_1, g_1)^{\alpha s}. \end{aligned}$$

Now, if $s' = 0$ (i.e., $T = g_1^s$), this would indeed be a normal encryption. On the other hand, if $s' \leftarrow_{\mathrm{R}} \mathbb{Z}_N$ instead, this would indeed be a semi-functional encryption.

Key Queries. On input $\mathbb{A} := (\mathbf{M}, \rho)$, $\mathcal{B}_1$ needs to generate a normal key $\mathrm{SK}_{\mathbb{A}}$, which has the distribution

$$\Big(\ \mathbf{D}_j := g_1^{\alpha_j \mathbf{e}_{\rho(j)}} \cdot (g_1^{r_j} \cdot g_2^{r'_j})^{\mathbf{w}} \cdot \mathbf{X}_j,\ D_{0,j} := (g_1^{r_j} \cdot g_2^{r'_j}) \cdot Z_j \quad : j \in [\ell]\ \Big).$$

$\mathcal{B}_1$ picks $\tilde{r}_j \leftarrow_{\mathrm{R}} \mathbb{Z}_N$ for $j \in [\ell]$ and replaces $g_1^{r_j} \cdot g_2^{r'_j}$ with $(U_1U_2)^{\tilde{r}_j}$; then, it outputs

$$\mathrm{SK}_{\mathbb{A}} := \Big(\ g_1^{\alpha_j \mathbf{e}_{\rho(j)}} \cdot (U_1U_2)^{\tilde{r}_j \mathbf{w}} \cdot \mathbf{X}_j,\ (U_1U_2)^{\tilde{r}_j} \cdot Z_j \quad : j \in [\ell]\ \Big).$$

Observe that $(U_1U_2)^{\tilde{r}_j}$ is properly distributed as long as U_1U_2 is a generator of $G_{p_1p_2}$ (by the Chinese Remainder Theorem), which occurs with probability $1 - 1/p_1 - 1/p_2$.

We may therefore conclude that: $|\mathsf{Adv}_0(\lambda) - \mathsf{Adv}_1(\lambda)| \leq \mathsf{Adv}_{\mathcal{B}_1}^{\mathrm{AS1}}(\lambda) + 1/p_1 + 1/p_2$. □

Lemma 2 (Normal to semi-functional keys). *For $k = 1, \ldots, n$, there exists an adversary $\mathcal{B}_2$ such that:*

$$|\mathsf{Adv}_{2,k-1}(\lambda) - \mathsf{Adv}_{2,k}(\lambda)| \leq \mathsf{Adv}_{\mathcal{B}_2}^{\mathrm{AS3}}(\lambda) + 1/p_2$$

and $\mathsf{Time}(\mathcal{B}_2) \approx \mathsf{Time}(\mathcal{A}) + q \cdot \mathrm{poly}(\lambda, n)$ *where* $\mathrm{poly}(\lambda, n)$ *is independent of* $\mathsf{Time}(\mathcal{A})$. *(We note that* $\mathsf{Game}_{2,0}$ *is identical to* Game_1.*)*

Overview of proof. Fix k. We want to modify j'th subkey $(\mathbf{D}_j, D_{0,j})$ for all j such that $\rho(j) = k$ (that is, corresponds to the variable x_k) as follows:

- if $x_k^* = 1$, we leave it unchanged (in this case, $\mathsf{Game}_{2,k-1}$ and $\mathsf{Game}_{2,k}$ are identical);
- if $x_k^* = 0$, we change the semi-functional component in $\mathbf{D}_j$ from $g_2^{r'_j\mathbf{w}}$ to $g_2^{\alpha'_j\mathbf{e}_k+r'_j\mathbf{w}}$ using Assumption 3.

In the rest of the overview, we focus on the case $x_k^* = 0$. Roughly speaking, we rely on the fact that $w_k \pmod{p_2}$ is statistically hidden given MPK to obtain computational entropy as captured by $\{g_2^{\alpha'_j} : \rho(j) = k\}$. For simplicity, we first consider a single subkey $(\mathbf{D}_j, D_{0,j})$ for which $\rho(j) = k$. Recall that $(\mathbf{D}_j, D_{0,j})$ in $\mathsf{Game}_{2,k-1}$ and $\mathsf{Game}_{2,k}$ are of the form:

$$(g_1^{\alpha_j\mathbf{e}_k+r_j\mathbf{w}} \cdot \boxed{g_2^{r'_j\mathbf{w}}} \cdot \mathbf{X}_j,\ g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j) \quad\text{and}\quad (g_1^{\alpha_j\mathbf{e}_k+r_j\mathbf{w}} \cdot \boxed{g_2^{\alpha'_j\mathbf{e}_k+r'_j\mathbf{w}}} \cdot \mathbf{X}_j,\ g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j)$$

Roughly speaking, it suffices to show that:

$$(g_1^{\mathbf{w}},\ \boxed{g_2^{r'_j\mathbf{w}}} \cdot \mathbf{X}_j,\ g_2^{r'_j} \cdot Z_j) \quad\text{and}\quad (g_1^{\mathbf{w}},\ \boxed{g_2^{\alpha'_j\mathbf{e}_k+r'_j\mathbf{w}}} \cdot \mathbf{X}_j,\ g_2^{r'_j} \cdot Z_j)$$

are computationally indistinguishable, where $g_1^{\mathbf{w}}$ is provided in MPK. We may further simplify this to show that:

$$(g_1^{w_k},\ \boxed{g_2^{r'_jw_k}} \cdot X_j,\ g_2^{r'_j} \cdot Z_j) \quad\text{and}\quad (g_1^{w_k},\ \boxed{g_2^{\alpha'_j+r'_jw_k}} \cdot X_j,\ g_2^{r'_j} \cdot Z_j)$$

are computationally indistinguishable, where $X_j, Z_j \leftarrow_{\mathrm{R}} G_{p_3}$. This follows essentially from Assumption 3, which tells us that

$$(\boxed{g_2^{r'_jw_k}} \cdot X_j,\ g_2^{r'_j} \cdot Z_j,\ g_2^{w_k} \cdot Y_3) \quad\text{and}\quad (\boxed{g_2^{\alpha'_j+r'_jw_k}} \cdot X_j,\ g_2^{r'_j} \cdot Z_j,\ g_2^{w_k} \cdot Y_3)$$

are computationally indistinguishable, where $X_j, Z_j, Y_3 \leftarrow G_{p_3}$. Here, we rely crucially on the fact that $w_k \pmod{p_2}$ is completely random given $g_1^{w_k}$. To handle multiple subkeys $\{(\mathbf{D}_j, D_{0,j}) : j \in \rho^{-1}(k)\}$, we can proceed via a hybrid argument, but that would yield a security loss of $|\rho^{-1}(k)|$. To avoid this loss, we rely on the rerandomization trick from [28]. Finally, note that we cannot generate a semi-functional ciphertext for $\mathbf{x}^*$ such that $x_k^* = 1$ since we are only given $g_2^{w_k}Y_3$ and not $g_2^{w_k}$. (For the proof, it suffices to simulate a semi-functional ciphertext for which $x_k^* = 0$.)

Proof. We construct an adversary $\mathcal{B}_2$ (which gets as additional input $k \in [n]$) for Assumption 3 using $\mathcal{A}$. We note that the case $x_k^* = 1$ is straight-forward since $\mathsf{Game}_{2,k}$ is identical to $\mathsf{Game}_{2,k-1}$, which means

$$|\mathsf{Adv}_{2,k-1}(\lambda) - \mathsf{Adv}_{2,k}(\lambda)| = 0 \leq \mathsf{Adv}^{\mathsf{AS3}}_{\mathcal{B}_2}(\lambda).$$

This leaves us with k such that $x_k^* = 0$. Recall that in Assumption 3, the adversary is given $D := (\mathbb{G}; g_1, U_1U_2, g_2^xX_3, g_2^yY_3, g_2U_3, g_3)$, along with T, where T is distributed as

$$g_2^{xy}W_3 \quad\text{or}\quad g_2^{xy+z}W_3.$$

Here, we assume that $z \leftarrow_{\mathrm{R}} \mathbb{Z}_{p_2}^*$, which yields a $1/p_2$ negligible difference from Assumption 3 in the advantage; $\mathcal{B}_2$ simulates $\mathsf{Game}_{2,k-1}$ if $T = g_2^{xy}W_3$ and $\mathsf{Game}_{2,k}$ if $T = g_2^{xy+z}W_3$. Moreover, we use a "trick" from [28] to get a tight security reduction and avoid losing a factor of ℓ.

Specifically, $\mathcal{B}_2$ proceeds as follows:

Setup. $\mathcal{B}_2$ samples $\alpha \leftarrow_{\mathrm{R}} \mathbb{Z}_N$, $\tilde{\mathbf{w}} \leftarrow_{\mathrm{R}} \mathbb{Z}_N^n$ and implicitly sets the parameter $\mathbf{w} := \tilde{\mathbf{w}} \bmod p_1p_3$ (whereas $\mathbf{w} \bmod p_2$ is undetermined at this point). $\mathcal{B}_2$ outputs

$$\text{MPK} := (\ e(g_1, g_1^{\alpha}), g_1, g_1^{\tilde{\mathbf{w}}}\ ).$$

Observe that this is indeed the correct distribution since $g_1^{\mathbf{w}} = g_1^{\tilde{\mathbf{w}}}$. Moreover, we note that

$$(\ \alpha, \tilde{\mathbf{w}}, g_3; U_1U_2, g_2^xX_3, g_2^yY_3, g_2U_3; T\ )$$

is known to $\mathcal{B}_2$. Upon receiving a challenge $\mathbf{x}^* := (x_1^*, \ldots, x_n^*)$ for which $x_k^* = 0$, $\mathcal{B}_2$ implicitly sets the parameter $\mathbf{w} = \tilde{\mathbf{w}} + y \cdot \mathbf{e}_k \bmod p_2$.

Challenge Ciphertext. Upon receiving two equal-length messages m_0 and m_1 from $\mathcal{A}$, $\mathcal{B}_2$ picks $\beta \leftarrow_{\mathrm{R}} \{0,1\}$ and outputs the semi-functional challenge ciphertext as:

$$\Big(U_1U_2,\ (U_1U_2)^{\langle \tilde{\mathbf{w}}, \mathbf{x}^* \rangle},\ e(g_1^\alpha, U_1U_2) \cdot m_\beta \Big).$$

Observe that this is indeed the correct distribution since $\langle \tilde{\mathbf{w}}, \mathbf{x}^* \rangle = \langle \mathbf{w}, \mathbf{x}^* \rangle \bmod p_1p_2$.

Key Queries. On input $\mathbb{A} := (\mathbf{M}, \rho)$, $\mathcal{B}_2$ needs to generate a secret key $\mathrm{SK}_{\mathbb{A}}$ of the form:

$$\left(\begin{array}{lll} g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot g_2^{r_j' \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r_j'} \cdot Z_j & : x_{\rho(j)}^* = 1 \\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot g_2^{r_j' \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r_j'} \cdot Z_j & : (x_{\rho(j)}^* = 0) \wedge (\rho(j) > k) \\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot g_2^{\alpha_j' \mathbf{e}_{\rho(j)} + r_j' \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r_j'} \cdot Z_j & : (x_{\rho(j)}^* = 0) \wedge (\rho(j) < k) \\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot \boxed{g_2^{r_j' \mathbf{w}}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r_j'} \cdot Z_j & : (x_{\rho(j)}^* = 0) \wedge (\rho(j) = k) \wedge (T = g_2^{xy} W_3) \\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \mathbf{w}} \cdot \boxed{g_2^{\alpha_j' \mathbf{e}_{\rho(j)} + r_j' \mathbf{w}}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r_j'} \cdot Z_j & : (x_{\rho(j)}^* = 0) \wedge (\rho(j) = k) \wedge (T = g_2^{xy+z} W_3) \end{array} \right)$$

where $\alpha_1', \ldots, \alpha_\ell' \leftarrow_{\mathrm{R}} \mathbb{Z}_N$. Note that we know α and can therefore compute $\alpha_j := \mathbf{M}_j \mathbf{u}$ as in the normal KeyGen. We proceed via a case analysis for j. The first three cases are straight-forward, observe that

$$g_1^{\tilde{\mathbf{w}}} = g_1^{\mathbf{w}} \quad \text{and} \quad g_2^{\mathbf{w}} = g_2^{\tilde{\mathbf{w}}} \cdot (g_2^y)^{\mathbf{e}_k}.$$

We simply use g_2U_3 and $g_2^yY_3$ in place of g_2 and g_2^y respectively and pick $r_j, r_j', \alpha_j' \leftarrow_{\mathrm{R}} \mathbb{Z}_N$.

This leaves us with j such that $(x_{\rho(j)}^* = 0) \wedge (\rho(j) = k)$. Here, $\mathcal{B}_2$ picks $\delta_j, \delta_j' \leftarrow_{\mathrm{R}} \mathbb{Z}_N$ and implicitly sets

$$r_j' := x\delta_j + \delta_j'.$$

We can then rewrite the j'th normal subkey as:

$$\big(\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \tilde{\mathbf{w}}} \cdot \boxed{(g_2^{x\delta_j} \cdot g_2^{\delta_j'})^{\tilde{\mathbf{w}}} \cdot (g_2^{xy\delta_j} \cdot g_2^{y\delta_j'})^{\mathbf{e}_{\rho(j)}}} \cdot \mathbf{X}_j,\ g_1^{r_j} \cdot (g_2^{x\delta_j} \cdot g_2^{\delta_j'}) \cdot Z_j\ \big).$$

Here, we want to replace $g_2, g_2^x, g_2^y, g_2^{xy}$ with $g_2U_3, g_2^xX_3, g_2^yY_3, T$ respectively. First, $\mathcal{B}_2$ computes

$$R_j := (g_2^xX_3)^{\delta_j} \cdot (g_2U_3)^{\delta_j'} = g_2^{r_j'} \cdot (X_3^{\delta_j} U_3^{\delta_j'}),$$

and outputs as the j'th subkey

$$\big(\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \tilde{\mathbf{w}}} \cdot \boxed{R_j^{\tilde{\mathbf{w}}} \cdot \big(T^{\delta_j} \cdot (g_2^yY_3)^{\delta_j'}\big)^{\mathbf{e}_{\rho(j)}}} \cdot \mathbf{X}_j,\ g_1^{r_j} \cdot R_j \cdot Z_j\ \big).$$

Now, suppose $T = g_2^{xy+z}W_3$. Then,

$$R_j^{\tilde{\mathbf{w}}} \cdot \big(T^{\delta_j} \cdot (g_2^yY_3)^{\delta_j'}\big)^{\mathbf{e}_{\rho(j)}} = g_2^{z\delta_j \mathbf{e}_{\rho(j)} + r_j' \mathbf{w}} \cdot \mathbf{X}_j'$$

for some $\mathbf{X}_j' \in G_{p_3}^n$. Now, if $z = 0$ (i.e., $T = g_2^{xy}W_3$), this would indeed be a normal subkey. On the other hand, if $z \leftarrow_{\mathrm{R}} \mathbb{Z}_{p_2}^*$, this would be a semi-functional subkey, with $\alpha_j' := z\delta_j$, and where (r_j', δ_j) are pairwise-independent modulo p_2.

In summary, $\mathcal{B}_2$ outputs as $\textsc{sk}_{\mathbb{A}}$:

$$\begin{pmatrix} g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \tilde{\mathbf{w}}} \cdot \mathbf{S}_j \cdot \mathbf{X}_j, & g_1^{r_j} \cdot (g_2 U_3)^{r'_j} \cdot Z_j & : x^*_{\rho(j)} = 1 \\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \tilde{\mathbf{w}}} \cdot \mathbf{S}_j \cdot \mathbf{X}_j, & g_1^{r_j} \cdot (g_2 U_3)^{r'_j} \cdot Z_j & : (x^*_{\rho(j)} = 0) \wedge (\rho(j) > k) \\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \tilde{\mathbf{w}}} \cdot (g_2 U_3)^{\alpha'_j \mathbf{e}_{\rho(j)}} \cdot \mathbf{S}_j \cdot \mathbf{X}_j, & g_1^{r_j} \cdot (g_2 U_3)^{r'_j} \cdot Z_j & : (x^*_{\rho(j)} = 0) \wedge (\rho(j) < k) \\ g_1^{\alpha_j \mathbf{e}_{\rho(j)} + r_j \tilde{\mathbf{w}}} \cdot \boxed{R_j^{\tilde{\mathbf{w}}} \cdot \left(T^{\delta_j} \cdot (g_2^y Y_3)^{\delta'_j}\right)^{\mathbf{e}_{\rho(j)}}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot R_j \cdot Z_j & : (x^*_{\rho(j)} = 0) \wedge (\rho(j) = k) \end{pmatrix}$$

where $\mathbf{S}_j := (g_2^y Y_3)^{r'_j \mathbf{e}_k} \cdot (g_2 U_3)^{r'_j \tilde{\mathbf{w}}} \in G^n_{p_2 p_3}$, $R_j := (g_2^x X_3)^{\delta_j} \cdot (g_2 U_3)^{\delta'_j} \in G_{p_2 p_3}$.

We may therefore conclude that: $|\mathsf{Adv}_{2,k-1}(\lambda) - \mathsf{Adv}_{2,k}(\lambda)| \le \mathsf{Adv}^{\mathrm{AS3}}_{\mathcal{B}_2}(\lambda) + 1/p_2$. □

Lemma 3 (Final transition). *There exists an adversary $\mathcal{B}_3$ such that:*

$$|\mathsf{Adv}_{2,n}(\lambda) - \mathsf{Adv}_3(\lambda)| \le \mathsf{Adv}^{\mathrm{AS2}}_{\mathcal{B}_3}(\lambda)$$

and $\mathsf{Time}(\mathcal{B}_3) \approx \mathsf{Time}(\mathcal{A}) + q \cdot \mathrm{poly}(\lambda, n)$ *where* $\mathrm{poly}(\lambda, n)$ *is independent of* $\mathsf{Time}(\mathcal{A})$.

Overview of proof. Following the final transitions in [25, 27], we use Assumption 2, in which we are given $(g_1, g_1^\alpha X_2, g_1^s Y_2, g_2, g_3, T)$ where T is either $e(g_1, g_1)^{\alpha s}$ or drawn uniformly from G_T to blind the challenge message m_β. The main challenge in our setting lies in simulating a semi-functional key $\textsc{sk}_{\mathbb{A}}$ given $g_1^\alpha X_2$ and not α itself. Recall that a semi-functional key $\textsc{sk}_{\mathbb{A}}$ has the same distribution

$$\begin{pmatrix} \boxed{g_1^{\alpha_j \mathbf{e}_{\rho(j)}}} \cdot g_1^{r_j \mathbf{w}} \cdot g_2^{r'_j \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j & : x^*_{\rho(j)} = 1 \\ \boxed{g_1^{\alpha_j \mathbf{e}_{\rho(j)}}} \cdot \boxed{g_2^{\alpha'_j \mathbf{e}_{\rho(j)}}} \cdot g_1^{r_j \mathbf{w}} \cdot g_2^{r'_j \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j & : x^*_{\rho(j)} = 0 \end{pmatrix}$$

in both $\mathsf{Game}_{2,n}$ and Game_3. Specifically, we need to simulate (given $g_1, g_2, g_1^\alpha X_2$)

$$\begin{pmatrix} \boxed{g_1^{\alpha_j}} & : x^*_{\rho(j)} = 1 \\ \boxed{g_1^{\alpha_j}} \cdot \boxed{g_2^{\alpha'_j}} & : x^*_{\rho(j)} = 0 \end{pmatrix}$$

where $\alpha_1, \ldots, \alpha_\ell$ are LSSS shares of α according to $\mathbb{A} = (\mathbf{M}, \rho)$ and $\alpha'_1, \ldots, \alpha'_\ell$ are independently random values. Roughly speaking, we proceed as follows:

- simulate the terms $(g_1^{\alpha_j} : x^*_{\rho(j)} = 1)$ by raising g_1 to the power of random LSSS shares of 0 (as determined by $\mathbf{M}\tilde{\mathbf{u}}_0$ below);
- simulate the terms $(g_1^{\alpha_j} \cdot g_2^{\alpha'_j} : x^*_{\rho(j)} = 0)$ by doing a LSSS share of $g_1^\alpha X_2$ "in the exponent" (as determined by $\alpha \mathbf{M} \tilde{\mathbf{u}}_1$ below), multiplying by the shares of 0 from the previous step, then re-randomizing the G_{p_2}-components.

We exploit the fact that $\mathbf{x}^*$ does not satisfy $\mathbb{A}$ to argue that we can choose $\tilde{\mathbf{u}}_1$ so that $\mathbf{M}_{\mathbf{x}^*} \tilde{\mathbf{u}}_1 = \mathbf{0}$.

Proof. We construct an adversary $\mathcal{B}_3$ for Assumption 2 using $\mathcal{A}$. Recall that in Assumption 2, the adversary is given $D := (\mathbb{G}; g_1, g_1^\alpha X_2, g_1^s Y_2, g_2, g_3)$, along with T, where T equals $e(g_1, g_1)^{\alpha s}$ or is drawn uniformly from G_T. Here, $\mathcal{B}_3$ simulates $\mathsf{Game}_{2,n}$ if $T := e(g_1, g_1)^{\alpha s}$ and Game_3 if $T \leftarrow_{\mathrm{R}} G_T$. The quantity α in the assumption will correspond exactly to α in $\textsc{msk}$, and the quantity s in the assumption will correspond the random exponents s used in the (semi-functional) ciphertext.

Specifically, $\mathcal{B}_3$ proceeds as follows:

Setup. $\mathcal{B}_3$ samples $\mathbf{w} \leftarrow_{\mathrm{R}} \mathbb{Z}_N^n$ and output the public parameters

$$\textsc{mpk} := (\ e(g_1, g_1^\alpha X_2), g_1, g_1^{\mathbf{w}}\ ).$$

We note that

$$(\ \mathbf{w}, g_2, g_3; g_1^{\alpha} X_2, g_1^{s} Y_2; T\ )$$

is known to $\mathcal{B}_3$. The adversary $\mathcal{A}$ outputs a challenge $\mathbf{x}^* := (x_1^*, \ldots, x_n^*)$.

Challenge Ciphertext. Upon receiving two equal-length messages m_0 and m_1 from $\mathcal{A}$, $\mathcal{B}_3$ picks $\beta \leftarrow_{\mathrm{R}} \{0,1\}$ and outputs the semi-functional challenge ciphertext as:

$$\mathrm{CT}_{\mathbf{x}^*} := \Big(\ g_1^{s} Y_2,\ (g_1^{s} Y_2)^{\langle \mathbf{w}, \mathbf{x}^* \rangle},\ T \cdot m_\beta\ \Big).$$

Now, if T is distributed as distributed as $e(g_1, g_1)^{\alpha s}$, this would indeed be a properly distributed semi-functional encryption of m_β. On the other hand, if $T \leftarrow_{\mathrm{R}} G_T$, instead, then the challenge ciphertext is a properly distributed semi-functional encryption of a random message in G_T.

Key Queries. On input $\mathbb{A} := (\mathbf{M}, \rho)$, $\mathcal{B}_3$ needs to generate a semi-functional key $\mathrm{SK}_{\mathbb{A}}$, which has the distribution

$$\mathrm{SK}_{\mathbb{A}} := \left(\begin{array}{lll} \boxed{g_1^{\alpha_j \mathbf{e}_{\rho(j)}}} \cdot g_1^{r_j \mathbf{w}} \cdot g_2^{r'_j \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j & : x^*_{\rho(j)} = 1 \\ \boxed{g_1^{\alpha_j \mathbf{e}_{\rho(j)}}} \cdot \boxed{g_2^{\alpha'_j \mathbf{e}_{\rho(j)}}} \cdot g_1^{r_j \mathbf{w}} \cdot g_2^{r'_j \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j & : x^*_{\rho(j)} = 0 \end{array} \right),$$

where $\alpha'_1, \ldots, \alpha'_\ell \leftarrow_{\mathrm{R}} \mathbb{Z}_N$. The main challenge lies in simulating the terms $g_1^{\alpha_j}$ since $\mathcal{B}_3$ is only given $g_1^{\alpha} X_2$ and not α itself. By definition of the KP-ABE security game, $\mathbf{x}^*$ does not satisfy $\mathbb{A}$, so $\mathbf{1} \notin \mathsf{span}\langle \mathbf{M}_{\mathbf{x}^*} \rangle$. (Refer to Definition 1 for the notation.) Therefore, we can efficiently compute $\tilde{\mathbf{u}}_1 \in \mathbb{Z}_N^{\ell'}$ such that

$$\mathbf{M}_{\mathbf{x}^*} \tilde{\mathbf{u}}_1 = \mathbf{0} \quad \text{and} \quad \mathbf{1} \tilde{\mathbf{u}}_1 = 1.$$

$\mathcal{B}_3$ samples $\tilde{\mathbf{u}}_0 \leftarrow_{\mathrm{R}} \mathbb{Z}_N^{\ell'}$ such that $\mathbf{1} \tilde{\mathbf{u}}_0 = 0$, and implicitly sets

$$\mathbf{u} := \alpha \cdot \tilde{\mathbf{u}}_1 + \tilde{\mathbf{u}}_0.$$

Observe that $\mathbf{u}$ has indeed the correct distribution. Recall that we set $\alpha_j := \mathbf{M}_j \mathbf{u}$, which yields

$$\alpha_j = \begin{cases} \mathbf{M}_j \tilde{\mathbf{u}}_0 & \text{if } x^*_{\rho(j)} = 1 \\ \alpha \cdot \mathbf{M}_j \tilde{\mathbf{u}}_1 + \mathbf{M}_j \tilde{\mathbf{u}}_0 & \text{if } x^*_{\rho(j)} = 0 \end{cases}$$

where both $\tilde{\mathbf{u}}_1$ and $\tilde{\mathbf{u}}_0$ are known to $\mathcal{B}_3$. The case j such that $x^*_{\rho(j)} = 1$ is straight-forward; $\mathcal{B}_3$ simply picks $r_j, r'_j \leftarrow_{\mathrm{R}} \mathbb{Z}_N$. For the case j such that $x^*_{\rho(j)} = 0$, we can then rewrite $g_1^{\alpha_j} \cdot g_2^{\alpha'_j}$ as a function of $\tilde{\mathbf{u}}_0, \tilde{\mathbf{u}}_1$, and $g_1^{\alpha} X_2$:

$$g_1^{\alpha_j} \cdot g_2^{\alpha'_j} = g_1^{\alpha \cdot \mathbf{M}_j \tilde{\mathbf{u}}_1 + \mathbf{M}_j \tilde{\mathbf{u}}_0} \cdot g_2^{\alpha'_j} = (g_1^{\alpha} X_2)^{\mathbf{M}_j \tilde{\mathbf{u}}_1} \cdot g_1^{\mathbf{M}_j \tilde{\mathbf{u}}_0} \cdot g_2^{\tilde{\alpha}'_j},$$

where $\mathcal{B}_3$ picks $\tilde{\alpha}'_j \leftarrow_{\mathrm{R}} \mathbb{Z}_N$ and implicitly sets $g_2^{\alpha'_j} := X_2^{\mathbf{M}_j \tilde{\mathbf{u}}_1} \cdot g_2^{\tilde{\alpha}'_j}$. $\mathcal{B}_3$ then outputs

$$\mathrm{SK}_{\mathbb{A}} := \left(\begin{array}{lll} \boxed{g_1^{\mathbf{M}_j \tilde{\mathbf{u}}_0 \mathbf{e}_{\rho(j)}}} \cdot g_1^{r_j \mathbf{w}} \cdot g_2^{r'_j \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j & : x^*_{\rho(j)} = 1 \\ \boxed{\big((g_1^{\alpha} X_2)^{\mathbf{M}_j \tilde{\mathbf{u}}_1} \cdot g_1^{\mathbf{M}_j \tilde{\mathbf{u}}_0} \cdot g_2^{\tilde{\alpha}'_j}\big)^{\mathbf{e}_{\rho(j)}}} \cdot g_1^{r_j \mathbf{w}} \cdot g_2^{r'_j \mathbf{w}} \cdot \mathbf{X}_j, & g_1^{r_j} \cdot g_2^{r'_j} \cdot Z_j & : x^*_{\rho(j)} = 0 \end{array} \right).$$

We may therefore conclude that: $|\mathsf{Adv}_{2,n}(\lambda) - \mathsf{Adv}_3(\lambda)| \leq \mathsf{Adv}_{\mathcal{B}_3}^{\mathrm{AS2}}(\lambda)$. □

Acknowledgments. We thank Allison Lewko and the reviewers for helpful discussions and feedback.

References

[1] N. Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In *EUROCRYPT*, pages 557–577, 2014.

[2] N. Attrapadung, B. Libert, and E. de Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In *Public Key Cryptography*, pages 90–108, 2011.

[3] A. Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. Ph.D., Technion - Israel Institute of Technology, 1996.

[4] S. Benabbas, R. Gennaro, and Y. Vahlis. Verifiable delegation of computation over large datasets. In *CRYPTO*, page 110, 2011. Cryptology ePrint Archive, Report 2011/132.

[5] D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003.

[6] D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC*, pages 535–554, 2007. Also Cryptology ePrint Archive, Report 2006/287.

[7] D. Boneh, E.-J. Goh, and K. Nissim. Evaluating 2-DNF formulas on ciphertexts. In *TCC*, pages 325–341, 2005.

[8] J. Chen and H. Wee. Fully, (almost) tightly secure IBE and dual system groups. In *CRYPTO (2)*, pages 435–460, 2013.

[9] J. Chen and H. Wee. Fully, (almost) tightly secure IBE from standard assumptions. IACR Cryptology ePrint Archive, Report 2013/803, 2013. Preliminary version in [8].

[10] J. Chen and H. Wee. Dual system groups and its applications — compact HIBE and more. IACR Cryptology ePrint Archive, Report 2014/265, 2014. Preliminary version in [8].

[11] J. Chen, H. W. Lim, S. Ling, H. Wang, and H. Wee. Shorter IBE and signatures via asymmetric pairings. In *Pairing*, pages 122–140, 2012.

[12] L. Cheung and C. C. Newport. Provably secure ciphertext policy ABE. In *ACM Conference on Computer and Communications Security*, pages 456–465, 2007.

[13] C. Cocks. An identity based encryption scheme based on quadratic residues. In *IMA Int. Conf.*, pages 360–363, 2001.

[14] D. Fiore and R. Gennaro. Publicly verifiable delegation of large polynomials and matrix computations, with applications. In *ACM Conference on Computer and Communications Security*, pages 501–512, 2012.

[15] D. M. Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In *EUROCRYPT*, pages 44–61, 2010.

[16] R. Gennaro, C. Gentry, and B. Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In *CRYPTO*, pages 465–482, 2010.

[17] R. Gennaro, C. Gentry, B. Parno, and M. Raykova. Quadratic span programs and succinct NIZKs without PCPs. In *EUROCRYPT*, pages 626–645, 2013. Also Cryptology ePrint Archive, Report 2012/215.

[18] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum. Delegating computation: interactive proofs for muggles. In *STOC*, pages 113–122, 2008.

[19] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM Conference on Computer and Communications Security*, pages 89–98, 2006.

[20] V. Goyal, A. Jain, O. Pandey, and A. Sahai. Bounded ciphertext policy attribute based encryption. In *ICALP (2)*, pages 579–591, 2008.

[21] M. Karchmer and A. Wigderson. On span programs. In *Structure in Complexity Theory Conference*, pages 102–111, 1993.

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

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

[24] A. B. Lewko. *Functional Encryption: New Proof Techniques and Advancing Capabilities*. Ph.D., The University of Texas at Austin, 2012.

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

[26] 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.

[27] A. B. 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.

[28] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004.

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

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

[31] T. Okamoto and K. Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *CRYPTO*, pages 191–208, 2010. Also, Cryptology ePrint Archive, Report 2010/563.

[32] B. Parno, M. Raykova, and V. Vaikuntanathan. How to delegate and verify in public: Verifiable computation from attribute-based encryption. In *TCC*, pages 422–439, 2012.

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

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

[35] K. Takashima. Expressive attribute-based encryption with constant-size ciphertexts from the decisional linear assumption. *IACR Cryptology ePrint Archive*, 2014:207, 2014.

[36] B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In *CRYPTO*, pages 619–636, 2009.

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

[38] H. Wee. Dual system encryption via predicate encodings. In *TCC*, pages 616–637, 2014.

A Semi-Adaptive ABE in Prime-Order Groups

This is based on embedding the ABE scheme of Goyal et al. [19] (see also [27]).

A.1 Dual Pairing Vector Spaces

Our second construction is based on dual pairing vector spaces proposed by Okamoto and Takashima [29, 30]. In this paper, we concentrate on the asymmetric version [31]. We only briefly describe how to generate random dual orthonormal bases. See [29, 30, 31] for a full definition of dual pairing vector spaces.

We first introduce asymmetric bilinear pairing groups. A generator $\mathcal{G}$ which takes as input a security parameter 1^λ and outputs a description $\mathbb{G} := (q, G_1, G_2, G_T, e)$, where q is a prime of $\Theta(\lambda)$ bits, G_1, G_2, and G_T are cyclic groups of order q, and $e : G_1 \times G_2 \rightarrow G_T$ is a map with the following properties:

1. (Bilinearity) $\forall h_1 \in G_1, h_2 \in G_2, a, b \in \mathbb{Z}_q, e(h_1^a, h_2^b) = e(h_1, h_2)^{ab}$.
2. (Non-degeneracy) $\exists g_1 \in G_1, g_2 \in G_2$ such that $e(g_1, g_2)$ has order q in G_T.

We require that the group operations in G_1, G_2, and G_T as well the bilinear map e are computable in deterministic polynomial time with respect to λ. Furthermore, the group descriptions of G_1, G_2, and G_T include generators of the respective cyclic groups.

We extend the pairing e to vectors of group elements over G_1 and G_2 by defining

$$e(g_1^{\mathbf{v}}, g_2^{\mathbf{w}}) := e(g_1, g_2)^{\langle \mathbf{v}, \mathbf{w} \rangle}.$$

Dual Pairing Vector Spaces. For a fixed (constant) dimension n, we will choose random bases

$$(\mathbf{d}_1, \ldots, \mathbf{d}_n) \quad \text{and} \quad (\mathbf{d}_1^*, \ldots, \mathbf{d}_n^*)$$

of $\mathbb{Z}_q^n$, subject to the constraint that they are "dual orthonormal", meaning that

$$\langle \mathbf{d}_i^*, \mathbf{d}_j \rangle = 0 \mod q$$

whenever $i \neq j$, and

$$\langle \mathbf{d}_i, \mathbf{d}_i^* \rangle = 1 \mod q$$

for all i. We denote such algorithm as

$$(\mathbf{d}_1, \ldots, \mathbf{d}_n, \mathbf{d}_1^*, \ldots, \mathbf{d}_n^*) \leftarrow_{\mathrm{R}} \mathsf{Dual}(\mathbb{Z}_q^n) \quad \text{or} \quad (\mathbf{d}_1, \mathbf{d}_1^*, \ldots, \mathbf{d}_n, \mathbf{d}_n^*) \leftarrow_{\mathrm{R}} \mathsf{Dual}(\mathbb{Z}_q^n).$$

Then for generators $g_1 \in G_1$ and $g_2 \in G_2$, we have

$$e(g_1^{\mathbf{d}_i}, g_2^{\mathbf{d}_j^*}) = 1$$

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

Computational Assumptions. We now state the decisional Diffie-Hellman (DDH) and Subspace assumptions that are required in our security proof. We stress that all these assumptions hold under the symmetric external Diffie-Hellman (SXDH) assumption is hard.

Assumption 4 (DDH1: Decisional Diffie-Hellman Assumption in G_1) *Given a group generator $\mathcal{G}$, we define the following distribution:*

$$\begin{aligned}
&\mathbb{G} := (q, G_1, G_2, G_T, g_1, g_2, e) \xleftarrow{\text{R}} \mathcal{G},\\
&x, y, z \xleftarrow{\text{R}} \mathbb{Z}_q,\\
&T_0 := g_1^{xy},\ T_1 := g_1^{xy+z},\\
&D := (\mathbb{G}; g_1, g_2, g_1^x, g_1^y).
\end{aligned}$$

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

$$\mathsf{Adv}^{\mathsf{DDH1}}_{\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(D, T_0) - \Pr[\mathcal{A}(D, T_1)]|$$

is negligible in the security parameter λ.

The dual of above assumption is decisional Diffie-Hellman assumption in G_2 (denoted as DDH2), which is identical to Definitions 4 with the roles of G_1 and G_2 reversed. We say that:

Definition 3. *The symmetric external Diffie-Hellman (SXDH) assumption holds if DDH problems are intractable in both G_1 and G_2.*

The Subspace assumptions were introduced in [23, 11, 31]. In this paper, specifically, we require the following Subspace assumption in G_1, which is similar with the SXDH-based Subspace assumptions of [11] but involves $n+1$ bases pairs chosen independently at random (see also [24]).

Assumption 5 (DS1: Decisional Subspace Assumption in G_1) *Given a group generator $\mathcal{G}(\cdot)$, define the following distribution:*

$$\begin{aligned}
&\mathbb{G} := (q, G_1, G_2, G_T, g_1, g_2, e) \xleftarrow{\text{R}} \mathcal{G}(1^\lambda),\\
&(\mathbf{d}_0, \mathbf{d}_0^*, \mathbf{f}_0, \mathbf{f}_0^*), \ldots, (\mathbf{d}_n, \mathbf{d}_n^*, \mathbf{f}_n, \mathbf{f}_n^*) \xleftarrow{\text{R}} \mathsf{Dual}(\mathbb{Z}_q^2),\\
&\tau_1, \tau_2, \mu_1, \mu_2 \xleftarrow{\text{R}} \mathbb{Z}_q,\\
&U_0 := g_2^{\mu_1 \mathbf{d}_0^* + \mu_2 \mathbf{f}_0^*}, \ldots, U_n := g_2^{\mu_1 \mathbf{d}_n^* + \mu_2 \mathbf{f}_n^*},\\
&V_0 := g_1^{\tau_1 \mathbf{d}_0}, \ldots, V_n := g_1^{\tau_1 \mathbf{d}_n},\\
&W_0 := g_1^{\tau_1 \mathbf{d}_0 + \tau_2 \mathbf{f}_0}, \ldots, W_n := g_1^{\tau_1 \mathbf{d}_n + \tau_2 \mathbf{f}_n},\\
D := &\Big(\ \mathbb{G}; g_1^{\mathbf{d}_0}, g_1^{\mathbf{f}_0}, g_2^{\mathbf{d}_0^*}, \ldots, g_1^{\mathbf{d}_n}, g_1^{\mathbf{f}_n}, g_2^{\mathbf{d}_n^*}, U_0, \ldots, U_n, \mu_2\ \Big).
\end{aligned}$$

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

$$\mathsf{Adv}^{\mathsf{DS1}}_{\mathcal{A}}(\lambda) := |\Pr[\mathcal{A}(D, V_0, \ldots, V_n) = 1] - \Pr[\mathcal{A}(D, W_0, \ldots, W_n) = 1]|$$

is negligible in the security parameter λ.

Lemma 4. *If the DDH assumption in G_1 holds, then the Subspace assumption in G_1 stated in Definition 5 also holds. More precisely, for any adversary $\mathcal{A}$ against the Subspace assumption in G_1, there exist probabilistic algorithms $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\mathsf{DS1}}_{\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{DDH1}}_{\mathcal{B}}(\lambda)$$

and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + \mathrm{poly}(\lambda, n)$ *where* $\mathrm{poly}(\lambda, n)$ *is independent of* $\mathsf{Time}(\mathcal{A})$.

The proof of above lemma is essentially the same as given in [11], but for completeness, we include a proof in Section D.

A.2 Construction

$\mathsf{Setup}(1^\lambda, [n])$: On input an attribute universe $[n]$, generate $(\mathbf{d}_i, \mathbf{d}_i^*, \mathbf{f}_i, \mathbf{f}_i^*) \leftarrow_{\mathrm{R}} \mathsf{Dual}(2)$, for $i = 0, 1, \ldots, n$. Pick $\alpha, \zeta \leftarrow_{\mathrm{R}} \mathbb{Z}_q$. Output

$$\textsc{mpk} := (\; e(g_1, g_2)^\alpha, g_1^{\mathbf{d}_0}, \ldots, g_1^{\mathbf{d}_n} \;) \quad \text{and} \quad \textsc{msk} := (\; \alpha, \zeta, g_1^{\mathbf{d}_0^*}, \ldots, g_1^{\mathbf{d}_n^*}, g_1^{\mathbf{f}_0^*}, \ldots, g_1^{\mathbf{f}_n^*} \;).$$

$\mathsf{Enc}(\textsc{mpk}, \mathbf{x}, m)$: On input an attribute vector $\mathbf{x} \in \{0,1\}^n$ and $m \in G_T$, output

$$\textsc{ct}_{\mathbf{x}} := \Big(\; C_0 := g_1^{s\mathbf{d}_0},\; C_1 := g_1^{x_1 s\mathbf{d}_1},\; \ldots,\; C_n := g_1^{x_n s\mathbf{d}_n},\; C_{n+1} := e(g_1, g_2)^{\alpha s} \cdot m \;\Big) \in (G_1^2)^{1+n} \times G_T,$$

where $s \leftarrow_{\mathrm{R}} \mathbb{Z}_q$.

$\mathsf{KeyGen}(\textsc{mpk}, \textsc{msk}, \mathbb{A} := (\mathbf{M}, \rho))$: On input an access structure $\mathbb{A} := (\mathbf{M}, \rho)$, where $\mathbf{M} \in \mathbb{Z}_q^{\ell \times \ell'}$ and $\rho : [\ell] \to [n]$, pick random vectors $\mathbf{u}, \mathbf{v} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{\ell'}$ such that $\mathbf{1u} = \alpha$, $\mathbf{1v} = \zeta$ and set $\alpha_j := \mathbf{M}_j\mathbf{u}$, $\zeta_j := \mathbf{M}_j\mathbf{v}$, for $j \in [\ell]$. Output

$$\textsc{sk}_{\mathbb{A}} := \Bigg(\; D_j := g_2^{r_j \mathbf{d}_0^* + r_j' \mathbf{f}_0^*},\; D_j' := g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^* + (\zeta_j - r_j')\mathbf{f}_{\rho(j)}^*} \;:\; j \in [\ell] \;\Bigg) \in (G_2^2)^{2\ell}$$

where $r_1, r_1' \ldots, r_\ell, r_\ell' \leftarrow_{\mathrm{R}} \mathbb{Z}_q$.

$\mathsf{Dec}(\textsc{mpk}, \textsc{sk}_{\mathbb{A}}, \textsc{ct}_{\mathbf{x}})$: If $\mathbf{x}$ satisfies $\mathbb{A}$, compute $\omega_1, \ldots, \omega_\ell \in \mathbb{Z}_q$ such that

$$\sum_{j : x_{\rho(j)} = 1} \omega_j \mathbf{M}_j = \mathbf{1}.$$

Then, compute

$$e(g_1, g_2)^{\alpha s} \leftarrow \prod_{j : x_{\rho(j)} = 1} \Big(e(C_0, D_j) \cdot e(C_{\rho(j)}, D_j') \Big)^{\omega_j},$$

and recover the message as $m \leftarrow C_{n+1} / e(g_1, g_2)^{\alpha s} \in G_T$.

Correctness. Observe that

$$\begin{aligned} e(C_0, D_j) \cdot e(C_{\rho(j)}, D_j') &= e(g_1^{s\mathbf{d}_0}, g_2^{r_j \mathbf{d}_0^* + r_j' \mathbf{f}_0^*}) \cdot e(g_1^{x_{\rho(j)} s\mathbf{d}_{\rho(j)}}, g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^* + (\zeta_j - r_j')\mathbf{f}_{\rho(j)}^*}) \\ &= e(g_1, g_2)^{r_j s\mathbf{d}_0 \cdot \mathbf{d}_0^*} \cdot e(g_1, g_2)^{(\alpha_j s - r_j s)\mathbf{d}_{\rho(j)} \cdot \mathbf{d}_{\rho(j)}^*} \\ &= e(g_1, g_2)^{\alpha_j s}. \end{aligned}$$

In addition, we have

$$\sum_{j : x_{\rho(j)} = 1} \omega_j \alpha_j = \sum_{j : x_{\rho(j)} = 1} \omega_j \mathbf{M}_j \mathbf{u} = \mathbf{1u} = \alpha.$$

This means

$$\prod_{j : x_{\rho(j)} = 1} \Big(e(C_{\rho(j)}', D_j) \cdot e(C_{\rho(j)}, D_j') \Big)^{\omega_j} = \prod_{j : x_{\rho(j)} = 1} e(g_1, g_2)^{\omega_j \alpha_j s} = e(g_1, g_2)^{\alpha s}.$$

Correctness follows readily.

A.3 Proof of Security

We prove the following theorem:

Theorem 2. *Under the SXDH assumption (described in Section A.1), our KP-ABE scheme defined in Section A.2 is semi-adaptively secure (in the sense of Definition 2). More precisely, for any adversary $\mathcal{A}$ that makes at most q key queries against the KP-ABE scheme, there exist probabilistic algorithms $\mathcal{B}_1, \mathcal{B}_2$ such that*

$$\mathsf{Adv}_{\mathcal{A}}^{\text{KP-ABE}}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}_1}^{\text{DDH1}}(\lambda) + \mathsf{Adv}_{\mathcal{B}_2}^{\text{DDH2}}(\lambda) + 3/q,$$

and

$$\max\{\mathsf{Time}(\mathcal{B}_1), \mathsf{Time}(\mathcal{B}_2)\} \approx \mathsf{Time}(\mathcal{A}) + q \cdot \mathrm{poly}(\lambda, n),$$

where $\mathrm{poly}(\lambda, n)$ *is independent of* $\mathsf{Time}(\mathcal{A})$.

The proof follows via a series of games. To describe the games, we must first define semi-functional keys and ciphertexts. Fix random generators g_1, g_2, and let $\mathbf{x}^*$ denote the semi-adaptive challenge.

Semi-functional ciphertext.

$$\mathrm{CT}_{\mathbf{x}} := \left(g_1^{s\mathbf{d}_0 + \boxed{s'\mathbf{f}_0}},\ g_1^{x_i s\mathbf{d}_i + \boxed{x_i s'\mathbf{f}_i}},\ e(g_1, g_2)^{\alpha s + \boxed{\zeta s'}} \cdot m \right),$$

where $s' \leftarrow_{\mathrm{R}} \mathbb{Z}_q$.

Semi-functional secret key.

$$\mathrm{SK}_{\mathbb{A}} := \left(\begin{array}{lll} g_2^{r_j\mathbf{d}_0^* + r_j'\mathbf{f}_0^*},\ g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^* + (\zeta_j - r_j')\mathbf{f}_{\rho(j)}^*} & & : x_{\rho(j)}^* = 1 \\ g_2^{r_j\mathbf{d}_0^* + r_j'\mathbf{f}_0^*},\ g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^* + (\zeta_j - r_j' - \boxed{\alpha_j'})\mathbf{f}_{\rho(j)}^*} & & : x_{\rho(j)}^* = 0 \end{array} \right),$$

where fresh $\alpha_1', \ldots, \alpha_\ell' \leftarrow_{\mathrm{R}} \mathbb{Z}_q$ are chosen for each secret key.

Game Sequence. We consider the following sequence of games:

- Game_0: is the real security game (c.f. Section 2.3).
- Game_1: is the same as Game_0 except that the challenge ciphertext is semi-functional.
- Game_2: is the same as Game_1 except that all keys are semi-functional. (This step is simpler than the corresponding transition in the composite-order setting.)
- Game_3: is the same as Game_2 except that the challenge ciphertext is a semi-functional encryption of a random message in G_T.

Fix an adversary $\mathcal{A}$. We write $\mathsf{Adv}_{\mathsf{x}}(\lambda)$ to denote the advantage of $\mathcal{A}$ in $\mathsf{Game}_{\mathsf{x}}$. It is easy to see that $\mathsf{Adv}_3(\lambda) = 0$, because the view of the adversary is Game_3 is independent of the challenge bit β. We complete the proof by establishing the following sequence of lemmas.

Lemma 5 (Normal to semi-functional ciphertext). *There exists an adversary $\mathcal{B}_1$ such that:*

$$|\mathsf{Adv}_0(\lambda) - \mathsf{Adv}_1(\lambda)| \leq \mathsf{Adv}_{\mathcal{B}_1}^{\text{DDH1}}(\lambda)$$

and $\mathsf{Time}(\mathcal{B}_1) \approx \mathsf{Time}(\mathcal{A}) + q \cdot \mathrm{poly}(\lambda, n)$ *where* $\mathrm{poly}(\lambda, n)$ *is independent of* $\mathsf{Time}(\mathcal{A})$.

Proof. We construct an adversary $\mathcal{B}_1$ for Subspace assumption in G_1 using $\mathcal{A}$. Recall that in Subspace assumption in G_1, the adversary is given

$$D := (\ \mathbb{G}; g_1^{\mathbf{d}_0}, g_1^{\mathbf{f}_0}, g_2^{\mathbf{d}_0^*}, \ldots, g_1^{\mathbf{d}_n}, g_1^{\mathbf{f}_n}, g_2^{\mathbf{d}_n^*}, U_0, \ldots, U_n\ ),$$

where $U_0, \dots, U_n$ are distributed as

$$g_2^{\alpha \mathbf{d}_0^* + \zeta \mathbf{f}_0^*}, \dots, g_2^{\alpha \mathbf{d}_n^* + \zeta \mathbf{f}_n^*},$$

along with $T_0, \dots, T_n$, where $T_0, \dots, T_n$ are distributed as

$$g_1^{s\mathbf{d}_0}, \dots, g_1^{s\mathbf{d}_n} \quad \text{or} \quad g_1^{s\mathbf{d}_0 + s'\mathbf{f}_0}, \dots, g_1^{s\mathbf{d}_n + s'\mathbf{f}_n}.$$

Here, $\mathcal{B}_1$ simulates Game_0 if $T_0, \dots, T_n$ are distributed as the former and Game_1 if $T_0, \dots, T_n$ are distributed as the latter. We have named the exponents in the Assumption so that the quantity α, ζ in the assumption will correspond exactly to α, ζ in MSK, and the quantity s, s' in the assumption will correspond the random exponents s, s' used in the (semi-functional) ciphertext.

Specifically, $\mathcal{B}_1$ proceeds as follows:

Setup. $\mathcal{B}_1$ outputs

$$\mathrm{MPK} := (\ e(g_1^{\mathbf{d}_0}, U_0), g_1^{\mathbf{d}_0}, \dots, g_1^{\mathbf{d}_n}\ ).$$

We note that

$$(\ g_2^{\mathbf{d}_0^*}, \dots, g_2^{\mathbf{d}_n^*}, U_0, \dots, U_n; T_0, \dots, T_n\ )$$

is known to $\mathcal{B}_1$. The adversary $\mathcal{A}$ outputs a challenge $\mathbf{x}^* := (x_1^*, \dots, x_n^*)$.

Challenge Ciphertext. Upon receiving two equal-length messages m_0 and m_1 from $\mathcal{A}$, $\mathcal{B}_1$ picks $\beta \leftarrow_{\mathrm{R}} \{0,1\}$ and outputs the semi-functional challenge ciphertext as:

$$\mathrm{CT}_{\mathbf{x}^*} := \Big(\ T_0,\ \ T_i^{x_i^*},\ \ e(T_0, U_0) \cdot m_\beta\ \Big).$$

Now, suppose $T_0, \dots, T_n$ are distributed as $g_1^{s\mathbf{d}_0 + s'\mathbf{f}_0}, \dots, g_1^{s\mathbf{d}_n + s'\mathbf{f}_n}$. Then,

$$T_i^{x_i^*} = g_1^{x_i^* s \mathbf{d}_i + x_i^* s' \mathbf{f}_i}, \ \ e(T_0, U_0) \cdot m_\beta = e(g_1, g_2)^{\alpha s + \zeta s'} \cdot m_\beta.$$

Now, if $s' = 0$ (i.e., $T_0, \dots, T_n$ are distributed as $g_1^{s\mathbf{d}_0}, \dots, g_1^{s\mathbf{d}_n}$), this would indeed be a normal encryption. On the other hand, if $s' \leftarrow_{\mathrm{R}} \mathbb{Z}_q$ instead, this would indeed be a semi-functional encryption.

Key Queries. On input $\mathbb{A} := (\mathbf{M}, \rho)$, $\mathcal{B}_1$ needs to generate a normal key $\mathrm{SK}_{\mathbb{A}}$, which has the distribution

$$\left(\ D_j := g_2^{r_j \mathbf{d}_0^* + r_j' \mathbf{f}_0^*},\ \ D_j' := \boxed{g_2^{\alpha_j \mathbf{d}_{\rho(j)}^* + \zeta_j \mathbf{f}_{\rho(j)}^*}} \cdot g_2^{-(r_j \mathbf{d}_{\rho(j)}^* + r_j' \mathbf{f}_{\rho(j)}^*)} \quad : j \in [\ell]\ \right).$$

The main challenge lies in simulating the term $g_2^{\alpha_j \mathbf{d}_{\rho(j)}^* + \zeta_j \mathbf{f}_{\rho(j)}^*}$. We begin with the remaining terms. Given $(g_2^{\mathbf{d}_0^*}, \dots, g_2^{\mathbf{d}_n^*})$ and $U_0, \dots, U_n$, along with some $j \in [\ell]$, we can easily sample $U_{j,0}', \dots, U_{j,n}'$ which are distributed as:

$$g_2^{r_j \mathbf{d}_0^* + r_j' \mathbf{f}_0^*}, \dots, g_2^{r_j \mathbf{d}_n^* + r_j' \mathbf{f}_n^*},$$

where $r_j, r_j' \leftarrow_{\mathrm{R}} \mathbb{Z}_q$. Namely, pick $\tilde{r}_j, \tilde{r}_j' \leftarrow_{\mathrm{R}} \mathbb{Z}_q$, and output

$$(g_2^{\mathbf{d}_0^*})^{\tilde{r}_j} \cdot U_0^{\tilde{r}_j'}, \dots, (g_2^{\mathbf{d}_n^*})^{\tilde{r}_j} \cdot U_n^{\tilde{r}_j'}.$$

This allows us to rewrite $\mathrm{SK}_{\mathbb{A}}$ as:

$$\left(\ U_{j,0}',\ \ \boxed{g_2^{\alpha_j \mathbf{d}_{\rho(j)}^* + \zeta_j \mathbf{f}_{\rho(j)}^*}} \cdot (U_{j,\rho(j)}')^{-1} \quad : j \in [\ell]\ \right).$$

Now, we just have to simulate the term $g_2^{\alpha_j \mathbf{d}^*_{\rho(j)} + \zeta_j \mathbf{f}^*_{\rho(j)}}$ given $U_{\rho(j)} := g_2^{\alpha \mathbf{d}^*_{\rho(j)} + \zeta \mathbf{f}^*_{\rho(j)}}$. To do this, $\mathcal{B}_1$ picks $\tilde{\mathbf{u}}, \tilde{\mathbf{v}} \in \mathbb{Z}_q^{\ell'}$ such that

$$\mathbf{1}\tilde{\mathbf{u}} = 0 \quad \text{and} \quad \mathbf{1}\tilde{\mathbf{v}} = 1,$$

and implicitly sets

$$\mathbf{u} := \alpha\tilde{\mathbf{v}} + \tilde{\mathbf{u}} \quad \text{and} \quad \mathbf{v} := \zeta\tilde{\mathbf{v}}.$$

Observe that $\mathbf{u}, \mathbf{v}$ have indeed the correct distributions. Recall that we set $\alpha_j := \mathbf{M}_j\mathbf{u}$ and $\zeta_j := \mathbf{M}_j\mathbf{v}$, which means $\alpha_j = \alpha \cdot \mathbf{M}_j\tilde{\mathbf{v}} + \mathbf{M}_j\tilde{\mathbf{u}}$ and $\zeta_j = \zeta \cdot \mathbf{M}_j\tilde{\mathbf{v}}$, and thus

$$\alpha_j \mathbf{d}^*_{\rho(j)} + \zeta_j \mathbf{f}^*_{\rho(j)} = \mathbf{M}_j\tilde{\mathbf{v}} \cdot (\alpha\mathbf{d}^*_{\rho(j)} + \zeta\mathbf{f}^*_{\rho(j)}) + \mathbf{M}_j\tilde{\mathbf{u}} \cdot \mathbf{d}^*_{\rho(j)}.$$

We can then rewrite $g_2^{\alpha_j \mathbf{d}^*_{\rho(j)} + \zeta_j \mathbf{f}^*_{\rho(j)}}$ as a function of $\tilde{\mathbf{u}}, \tilde{\mathbf{v}}$ and $U_{\rho(j)}$. That is, $\mathcal{B}_1$ simply outputs:

$$\text{SK}_{\mathbb{A}} := \left(\; U'_{j,0}, \;\; \boxed{U_{\rho(j)}^{\mathbf{M}_j\tilde{\mathbf{v}}} \cdot (g_2^{\mathbf{d}^*_{\rho(j)}})^{\mathbf{M}_j\tilde{\mathbf{u}}}} \cdot (U'_{j,\rho(j)})^{-1} \quad : j \in [\ell] \; \right),$$

which would indeed be a normal secret key.

We may therefore conclude that: $|\mathsf{Adv}_0(\lambda) - \mathsf{Adv}_1(\lambda)| \leq \mathsf{Adv}_{\mathcal{B}_1}^{\mathrm{DDH1}}(\lambda)$. □

Lemma 6 (Normal to semi-functional keys). *There exists an adversary $\mathcal{B}_2$ whose running time is essentially the same as that of $\mathcal{A}$ such that:*

$$|\mathsf{Adv}_1(\lambda) - \mathsf{Adv}_2(\lambda)| \leq \mathsf{Adv}_{\mathcal{B}_2}^{\mathrm{DDH2}}(\lambda) + 2/q$$

and $\mathsf{Time}(\mathcal{B}_2) \approx \mathsf{Time}(\mathcal{A}) + q \cdot \mathrm{poly}(\lambda, n)$ *where* $\mathrm{poly}(\lambda, n)$ *is independent of* $\mathsf{Time}(\mathcal{A})$.

Proof. We construct an adversary $\mathcal{B}_2$ for DDH assumption in G_2 using $\mathcal{A}$. Recall that in DDH assumption in G_2, the adversary is given $D := (\mathbb{G}; g_1, g_2, g_2^x, g_2^y)$ along with T, where T is distributed as g_2^{xy} or g_2^{xy+z}. Here, we assume that $y, z \leftarrow_{\mathrm{R}} \mathbb{Z}_{p_2}^*$, which yields a $2/q$ negligible difference in the advantage; $\mathcal{B}_2$ simulates Game_1 if $T = g_2^{xy}$ and Game_2 if $T = g_2^{xy+z}$.

Specifically, $\mathcal{B}_2$ proceeds as follows:

Setup. $\mathcal{B}_2$ samples $\alpha, \zeta \leftarrow_{\mathrm{R}} \mathbb{Z}_q$, and $(\mathbf{d}_i, \mathbf{d}_i^*, \tilde{\mathbf{f}}_i, \tilde{\mathbf{f}}_i^*) \leftarrow_{\mathrm{R}} \mathsf{Dual}(2)$, for $i = 0, 1, \ldots, n$. $\mathcal{B}_2$ outputs

$$\text{MPK} := (\; e(g_1, g_2)^{\alpha}, g_1^{\mathbf{d}_0}, \ldots, g_1^{\mathbf{d}_n} \;).$$

Moreover, we note that

$$(\; \alpha, \zeta, g_2^{\mathbf{d}_0^*}, \ldots, g_2^{\mathbf{d}_n^*}; g_1, g_2, g_2^x, g_2^y, \tilde{\mathbf{f}}_0, \tilde{\mathbf{f}}_0^*, \ldots, \tilde{\mathbf{f}}_n, \tilde{\mathbf{f}}_n^*; T \;)$$

is known to $\mathcal{B}_2$.

Challenge Attribute. Upon receiving a challenge $\mathbf{x}^* := (x_1^*, \ldots, x_n^*)$ where $x_k^* = 0$, $\mathcal{B}_2$ implicitly sets the parameters

$$\begin{aligned} (\; \mathbf{f}_0, \; \mathbf{f}_0^* \;) &:= (\; \tilde{\mathbf{f}}_0, \; \tilde{\mathbf{f}}_0^* \;), & \\ (\; \mathbf{f}_i, \; \mathbf{f}_i^* \;) &:= (\; \tilde{\mathbf{f}}_i, \; \tilde{\mathbf{f}}_i^* \;) & : x_i^* = 1, \\ (\; \mathbf{f}_i, \; \mathbf{f}_i^* \;) &:= (\; y^{-1}\tilde{\mathbf{f}}_i, \; y\tilde{\mathbf{f}}_i^* \;) & : x_i^* = 0. \end{aligned}$$

Note that all the bases are properly distributed.

Challenge Ciphertext. Upon receiving two equal-length messages m_0 and m_1 from $\mathcal{A}$, $\mathcal{B}_2$ picks $\beta \leftarrow_{\mathrm{R}} \{0,1\}$ and outputs the semi-functional challenge ciphertext as:

$$\mathrm{CT}_{\mathbf{x}^*} := \Big(\; g_1^{s\mathbf{d}_0 + s'\tilde{\mathbf{f}}_0}, \;\; g_1^{x_i^*(s\mathbf{d}_i + s'\tilde{\mathbf{f}}_i)}, \;\; e(g_1^{\alpha s + \zeta s'}, g_2) \cdot m_\beta \; \Big).$$

Key Queries. On input $\mathbb{A} := (\mathbf{M}, \rho)$, $\mathcal{B}_2$ needs to generate a secret key $\mathrm{SK}_{\mathbb{A}}$ of the form:

$$\left(\begin{array}{lll} g_2^{r_j\mathbf{d}_0^* + r_j'\mathbf{f}_0^*}, & g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^* + (\zeta_j - r_j')\mathbf{f}_{\rho(j)}^*} & : x_{\rho(j)}^* = 1 \\ g_2^{r_j\mathbf{d}_0^* + r_j'\mathbf{f}_0^*}, & g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^*} \cdot \boxed{g_2^{(\zeta_j - r_j')\mathbf{f}_{\rho(j)}^*}} & : (x_{\rho(j)}^* = 0) \wedge (T = g_2^{xy}) \\ g_2^{r_j\mathbf{d}_0^* + r_j'\mathbf{f}_0^*} & g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^*} \cdot \boxed{g_2^{(\zeta_j - r_j' - \alpha_j')\mathbf{f}_{\rho(j)}^*}} & : (x_{\rho(j)}^* = 0) \wedge (T = g_2^{xy+z}) \end{array} \right),$$

where $\alpha_1', \ldots, \alpha_\ell' \leftarrow_{\mathrm{R}} \mathbb{Z}_q$.
Note that we know α, ζ and can therefore compute $\alpha_j := \mathbf{M}_j\mathbf{u}, \zeta_j := \mathbf{M}_j\mathbf{v}$ as in the normal KeyGen. We proceed via a case analysis for j. The case $x_{\rho(j)}^* = 1$ is straight-forward; we know $g_2^{\mathbf{f}_0^*} = g_2^{\tilde{\mathbf{f}}_0^*}$, $g_2^{\mathbf{f}_{\rho(j)}^*} = g_2^{\tilde{\mathbf{f}}_{\rho(j)}^*}$. This leaves us with j such that $x_{\rho(j)}^* = 0$. Here, $\mathcal{B}_2$ picks $\delta_j, \delta_j' \leftarrow_{\mathrm{R}} \mathbb{Z}_q$ and implicitly sets

$$r_j' := x\delta_j + \delta_j'.$$

We can then rewrite the j'th normal subkey as:

$$\big(\; g_2^{r_j\mathbf{d}_0^*} \cdot (g_2^x)^{\delta_j\tilde{\mathbf{f}}_0^*} \cdot g_2^{\delta_j'\tilde{\mathbf{f}}_0^*}, \quad g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^*} \cdot \boxed{(g_2^{xy})^{-\delta_j\tilde{\mathbf{f}}_{\rho(j)}^*} \cdot (g_2^y)^{(\zeta_j - \delta_j')\tilde{\mathbf{f}}_{\rho(j)}^*}} \; \big).$$

Here, we want to replace g_2^{xy} with T. $\mathcal{B}_2$ outputs as the j'th subkey

$$\big(\; g_2^{r_j\mathbf{d}_0^*} \cdot (g_2^x)^{\delta_j\tilde{\mathbf{f}}_0^*} \cdot g_2^{\delta_j'\tilde{\mathbf{f}}_0^*}, \quad g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^*} \cdot \boxed{T^{-\delta_j\tilde{\mathbf{f}}_{\rho(j)}^*} \cdot (g_2^y)^{(\zeta_j - \delta_j')\tilde{\mathbf{f}}_{\rho(j)}^*}} \; \big).$$

Now, suppose $T = g_2^{xy+z}$. Then,

$$\begin{aligned} g_2^{r_j\mathbf{d}_0^*} \cdot (g_2^x)^{\delta_j\tilde{\mathbf{f}}_0^*} \cdot g_2^{\delta_j'\tilde{\mathbf{f}}_0^*} &= g_2^{r_j\mathbf{d}_0^* + r_j'\mathbf{f}_0^*}, \\ g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^*} \cdot T^{-\delta_j\tilde{\mathbf{f}}_{\rho(j)}^*} \cdot (g_2^y)^{(\zeta_j - \delta_j')\tilde{\mathbf{f}}_{\rho(j)}^*} &= g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^*} \cdot (g_2^{xy+z})^{-\delta_j\tilde{\mathbf{f}}_{\rho(j)}^*} \cdot (g_2^y)^{(\zeta_j - \delta_j')\tilde{\mathbf{f}}_{\rho(j)}^*} \\ &= g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^* + (\zeta_j - r_j' - zy^{-1}\delta_j)\mathbf{f}_{\rho(j)}^*}, \end{aligned}$$

where $\mathcal{B}_2$ implicitly sets $\alpha_j' := zy^{-1}\delta_j$. Now, if $z = 0$ (i.e., $T = g_2^{xy}$), this would indeed be a normal subkey. On the other hand, if $z \leftarrow_{\mathrm{R}} \mathbb{Z}_q^*$, this would be a semi-functional subkey, and where (r_j', α_j') are pairwise-independent. In summary, $\mathcal{B}_2$ outputs

$$\mathrm{SK}_{\mathbb{A}} := \left(\begin{array}{lll} g_2^{r_j\mathbf{d}_0^* + r_j'\tilde{\mathbf{f}}_0^*}, & g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^* + (\zeta_j - r_j')\tilde{\mathbf{f}}_{\rho(j)}^*} & : x_{\rho(j)}^* = 1 \\ g_2^{r_j\mathbf{d}_0^*} \cdot (g_2^x)^{\delta_j\tilde{\mathbf{f}}_0^*} \cdot g_2^{\delta_j'\tilde{\mathbf{f}}_0^*}, & g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^*} \cdot \boxed{T^{-\delta_j\tilde{\mathbf{f}}_{\rho(j)}^*} \cdot (g_2^y)^{(\zeta_j - \delta_j')\tilde{\mathbf{f}}_{\rho(j)}^*}} & : x_{\rho(j)}^* = 0 \end{array} \right).$$

We may therefore conclude that: $|\mathsf{Adv}_1(\lambda) - \mathsf{Adv}_2(\lambda)| \leq \mathsf{Adv}_{\mathcal{B}_2}^{\mathrm{DDH2}}(\lambda) + 2/q$. □

Lemma 7 (Final transition). *For any adversary $\mathcal{A}$:*

$$|\mathsf{Adv}_2(\lambda) - \mathsf{Adv}_3(\lambda)| \leq 1/q.$$

Proof. To prove this lemma, it suffices to show that ζ is completely random, even given all of the semi-functional keys in Game_2. This would mean that the exponent $\zeta s'$ in the semi-functional ciphertext is distributed as a random value in $\mathbb{Z}_q$ (if $s' \neq 0$). This would in turn imply that challenge ciphertext in Game_2 is statistically close to a semi-functional encryption of a random message in G_T.

For each key query $\mathbb{A} := (\mathbf{M}, \rho)$, a semi-functional key has the distribution:

$$\mathrm{SK}_{\mathbb{A}} := \begin{pmatrix} g_2^{r_j \mathbf{d}_0^* + r_j' \mathbf{f}_0^*}, & g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^* + (\zeta_j - r_j')\mathbf{f}_{\rho(j)}^*} & : x_{\rho(j)}^* = 1 \\ g_2^{r_j \mathbf{d}_0^* + r_j' \mathbf{f}_0^*}, & g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^* + (\zeta_j - r_j' - \alpha_j')\mathbf{f}_{\rho(j)}^*} & : x_{\rho(j)}^* = 0 \end{pmatrix},$$

where fresh $\alpha_1', \ldots, \alpha_\ell' \leftarrow_{\mathrm{R}} \mathbb{Z}_q$ are chosen for each secret key. It suffices to show that the subkeys corresponding to $x_{\rho(j)}^* = 1$ do not reveal any information about ζ, since the share ζ_j in the remaining subkeys are completely masked by α_j'.

By definition of the KP-ABE security game, $\mathbf{x}^*$ does not satisfy $\mathbb{A}$, so $\mathbf{1} \notin \mathsf{span}\langle \mathbf{M}_{\mathbf{x}^*} \rangle$. (Refer to Definition 1 for the notation.) Therefore, we can efficiently compute $\tilde{\mathbf{v}}_1 \in \mathbb{Z}_q^{\ell'}$ such that

$$\mathbf{M}_{\mathbf{x}^*} \tilde{\mathbf{v}}_1 = \mathbf{0} \quad \text{and} \quad \mathbf{1}\tilde{\mathbf{v}}_1 = 1.$$

Next, we sample $\tilde{\mathbf{v}}_0 \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{\ell'}$ such that $\mathbf{1}\tilde{\mathbf{v}}_0 = 0$, and implicitly set

$$\mathbf{v} := \zeta \cdot \tilde{\mathbf{v}}_1 + \tilde{\mathbf{v}}_0.$$

Observe that $\mathbf{v}$ has indeed the correct distribution. Recall that we set $\zeta_j := \mathbf{M}_j \mathbf{v}$, which yields

$$\zeta_j = \begin{cases} \mathbf{M}_j \tilde{\mathbf{v}}_0 & \text{if } x_{\rho(j)}^* = 1 \\ \zeta \cdot \mathbf{M}_j \tilde{\mathbf{v}}_1 + \mathbf{M}_j \tilde{\mathbf{v}}_0 & \text{if } x_{\rho(j)}^* = 0 \end{cases}.$$

This means that the subkeys

$$\begin{pmatrix} g_2^{r_j \mathbf{d}_0^* + r_j' \mathbf{f}_0^*}, & g_2^{(\alpha_j - r_j)\mathbf{d}_{\rho(j)}^* + (\mathbf{M}_j \tilde{\mathbf{v}}_0 - r_j')\mathbf{f}_{\rho(j)}^*} & : x_{\rho(j)}^* = 1 \end{pmatrix}$$

reveal no information about ζ since the distribution of $\tilde{\mathbf{v}}_0$ is independent of ζ. Hence, ζ is statistically hidden for the semi-functional key $\mathrm{SK}_{\mathbb{A}}$. Therefore, Game_2 and Game_3 are statistically indistinguishable except with probability $1/q$ (namely, the case $s' = 0$).

We may then conclude that: $|\mathsf{Adv}_2(\lambda) - \mathsf{Adv}_3(\lambda)| \leq 1/q$. □

B Publicly Verifiable Computation

As an application, we obtain improved publicly verifiable delegation schemes for Boolean formula with *semi-adaptive* soundness, where correctness of the computation is guaranteed even if the client's input is chosen adaptively depending on its public key. Most of the definitions here are taken almost verbatim from [32].

B.1 Definition for Publicly Verifiable Computation

Definition 4 (Publicly Verifiable Computation [32, 16]). *A publicly verifiable computation protocol (with preprocessing) consists of four algorithms* $(\mathsf{KeyGen}, \mathsf{ProbGen}, \mathsf{Compute}, \mathsf{Verify})$*:*

$\mathsf{Setup}(1^\lambda, \mathsf{F}) \to (\mathsf{PK}_\mathsf{F}, \mathsf{EK}_\mathsf{F})$. *The randomized key generation algorithm takes in a security parameter* 1^λ *and a function* F*. It outputs a public key* PK_F *and an evaluation key* EK_F.

$\mathsf{ProbGen}(\mathsf{PK}_\mathsf{F}, \mathbf{x}) \to (\sigma_\mathbf{x}, \mathsf{VK}_\mathsf{F})$. *The randomized problem generation algorithm uses the public key* PK_F *to encode an input* $\mathbf{x}$ *into public values* $\sigma_\mathbf{x}$ *and* $\mathsf{VK}_\mathbf{x}$*. The value* $\sigma_\mathbf{x}$ *is given to the worker to compute with, whereas* $\mathsf{VK}_\mathbf{x}$ *is made public, and later used for verification.*

$\mathsf{Compute}(\mathsf{EK_F}, \sigma_{\mathbf{x}}) \rightarrow \sigma_{\mathsf{out}}$. *The deterministic worker algorithm uses the evaluation key* $\mathsf{EK_F}$ *together with the value* $\sigma_{\mathbf{x}}$ *to compute a value* σ_{out}.

$\mathsf{Verify}(\mathsf{VK}_{\mathbf{x}}, \sigma_{\mathsf{out}}) \rightarrow \mathbf{y}$. *The deterministic verification algorithm uses the verification key* $\mathsf{VK}_{\mathbf{x}}$ *and the worker's output* σ_{out} *to compute a string* $\mathbf{y} \in \{0,1\}^* \cup \{\perp\}$. *Here, the special symbol* $\perp$ *signifies that the verification algorithm rejects the worker's answer* σ_{out}.

Correctness. A publicly verifiable computation protocol is correct for a class of functions $\mathcal{F}$ if for any $\mathsf{F} \in \mathcal{F}$, any pair of keys $(\mathsf{PK_F}, \mathsf{EK_F}) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{F})$, any $\mathbf{x} \in \mathsf{Domain}(\mathsf{F})$, any $(\sigma_{\mathbf{x}}, \mathsf{VK_F}) \leftarrow \mathsf{ProbGen}(\mathsf{PK_F}, \mathbf{x})$, and any $\sigma_{\mathsf{out}} \leftarrow \mathsf{Compute}(\mathsf{EK_F}, \sigma_{\mathbf{x}}))$, the verification algorithm Verify on input $\mathsf{VK}_{\mathbf{x}}$ and σ_{out} outputs $\mathbf{y} = \mathsf{F}(\mathbf{x})$.

B.2 Security for Publicly Verifiable Computation

There are three notions of security (soundness) for publicly verifiable computation, depending on the level of adaptivity the client has in choosing the instance $\mathbf{x}^*$ with respect to $\mathsf{PK_F}$ and $\mathsf{EK_F}$:

- the weakest notion requires that $\mathbf{x}^*$ be chosen independently of $\mathsf{PK_F}, \mathsf{EK_F}$. This is the notion achieved in [32] based on the GPSW KP-ABE.
- an intermediate notion (introduced in this work) requires that $\mathbf{x}^*$ by chosen independently of $\mathsf{EK_F}$, but may potentially depend on $\mathsf{PK_F}$.
- the strongest notion allows $\mathbf{x}^*$ to depend on both $\mathsf{PK_F}$ and $\mathsf{EK_F}$.

We remind the reader $\mathsf{PK_F}$ is public and reused over computation on many instances. For this reason, we believe that it is important that we allow client's input $\mathbf{x}^*$ to depend on $\mathsf{PK_F}$ in order to achieve any meaningful notion of security. On the other hand, $\mathsf{EK_F}$ is only known to the server carrying out the computation; as such, it seems reasonable to consider relaxed scenarios where the client's input does not depend on the server's private evaluation key $\mathsf{EK_F}$. Indeed, both of these are captured in the intermediate notion, which we formalize in the next paragraph.

Semi-adaptive soundness. Let a publicly verifiable computation scheme be for a class of functions $\mathcal{F}$, and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be a stateful adversary. Consider the experiment $\mathbf{Exp}_{\mathcal{A}}^{\textsc{PubVerif}}[\mathsf{F}, \lambda]$ for any $\mathsf{F} \in \mathcal{F}$ below:

```
Experiment $\mathbf{Exp}_{\mathcal{A}}^{\textsc{PubVerif}}[\mathsf{F}, \lambda]$
  $(\mathsf{PK_F}, \mathsf{EK_F}) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{F})$;
  $(\mathbf{x}^*, \mathsf{state}) \leftarrow \mathcal{A}_1(\mathsf{PK_F})$;
  $(\sigma_{\mathbf{x}^*}, \mathsf{VK}_{\mathbf{x}^*}) \leftarrow \mathsf{ProbGen}(\mathsf{PK_F}, \mathbf{x}^*)$;
  $(\sigma^*_{\mathsf{out}}) \leftarrow \mathcal{A}_2(\mathsf{state}, \sigma_{\mathbf{x}^*}, \mathsf{VK_F}, \mathsf{EK_F})$;
  $\mathbf{y}^* \leftarrow \mathsf{Verify}(\mathsf{VK}_{\mathbf{x}^*}, \sigma^*_{\mathsf{out}})$;
If $\mathbf{y}^* \neq \perp$ and $\mathbf{y}^* \neq \mathsf{F}(\mathbf{x}^*)$, outputs "1", else outputs "0";
```

The advantage of an adversary $\mathcal{A}$ is defined to be $\Pr[\mathbf{Exp}_{\mathcal{A}}^{\textsc{PubVerif}}[\mathsf{F}, \lambda] = 1]$.

Definition 5. *A publicly verifiable computation protocol is* secure *for a class of functions* $\mathcal{F}$ *if all PPT adversaries* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *achieve at most a negligible advantage in the above security game for every function* $\mathsf{F} \in \mathcal{F}$.

Remark 2. Informally, we can think of $\mathcal{A}_1$ as a client choosing the input for delegation, and $\mathcal{A}_2$ as the cheating server. While we do not think of the client as being necessarily adversarial, we want to allow an adversary $\mathcal{A}$ to choose the input of the client in order to guarantee soundness over inputs that may be selected from some "worst-case" distributions, possibly correlated with the public parameters $\mathsf{PK_F}$.

Remark 3. The security definition "composes" in the sense that security as defined for a single instances implies security for multiple instances $\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots$.

B.3 Efficiency for Publicly Verifiable Computation

Definition 6 (Efficiency). *A publicly verifiable computation protocol is efficient for a class of functions $\mathcal{F}$ that act on $n = n(\lambda)$ bits if there is a polynomial p such that:*[6]

- *the running time of* $\mathsf{ProbGen}$ *and* Verify *together is at most $p(n, \lambda)$, the rest of the algorithms are probabilistic polynomial-time, and*
- *there exists a function $\mathsf{F} \in \mathcal{F}$ whose running time is $\omega(p(n, \lambda))$.*[7]

In a similar vein, a publicly verifiable computation protocol is depth-efficient if the computation depth of $\mathsf{ProbGen}$ *and Verify combined (written as Boolean circuits) is at most $p(n, \lambda)$, whereas there is a function $\mathsf{F} \in \mathcal{F}$ whose computation depth is $\omega(p(n, \lambda))$.*

B.4 Publicly Verifiable Computation from KP-ABE

We recall the Main Theorem and the construction of a publicly verifiable computation protocol from a KP-ABE scheme from [32]. In the remaining of this section, we will use the notion of Boolean function (implemented by a family of circuits $\mathcal{C}$), which is equivalent to access structure in KP-ABE schemes. In fact, it is sufficient for the KP-ABE to be secure against an adversary that only requests for a single secret key.

Theorem 3 (implicit in [32, Theorem 2]). *Let $\mathcal{F}$ be a class of Boolean functions (implemented by a family of circuits $\mathcal{C}$), and let $\bar{\mathcal{F}} = \{\bar{\mathsf{F}} | \mathsf{F} \in \mathcal{F}\}$ where $\bar{\mathsf{F}}$ denotes the complement of the function* F. *Let a KP-ABE scheme be semi-adaptively secure for $\mathcal{F} \cup \bar{\mathcal{F}}$, and let H be any one-way function.*

Then, there is a publicly verifiable computation protocol (secure under Definition 5) for $\mathcal{F}$. If the circuit family $\mathcal{C}$ is unbounded (resp. depth-unbounded), then the protocol is efficient (resp. depth-efficient) in the sense of Definition 6.

For completeness, we present the publicly verifiable computation protocol from [32]. Let ($\mathsf{ABE.Setup}$, $\mathsf{ABE.KeyGen}$, $\mathsf{ABE.Enc}$, $\mathsf{ABE.Dec}$) be four algorithms of a KP-ABE scheme for the class of functions $\mathcal{F} \cup \bar{\mathcal{F}}$, then the publicly verifiable computation protocol consisting of four algorithms (Setup, $\mathsf{ProbGen}$, $\mathsf{Compute}$, Verify) for $\mathcal{F}$ works as follows:

- $\mathsf{Setup}(1^\lambda, \mathsf{F})$: On input a function $\mathsf{F} \in \mathcal{F}$ with input length n, run the KP-ABE setup algorithm twice, to generate two independent key-pairs

  $$(\textsc{mpk}_0, \textsc{msk}_0) \leftarrow \mathsf{ABE.Setup}(1^\lambda, [n]) \quad \text{and} \quad (\textsc{mpk}_1, \textsc{msk}_1) \leftarrow \mathsf{ABE.Setup}(1^\lambda, [n]).$$

  Generate two secret keys

  $$\textsc{sk}_{\bar{\mathsf{F}}} \leftarrow \mathsf{ABE.KeyGen}(\textsc{mpk}_0, \textsc{msk}_0, \bar{\mathsf{F}}) \text{ (corresponding to } \bar{\mathsf{F}})$$

[6] To be completely precise, one has to talk about a family $\mathcal{F} = \{\mathcal{F}_n\}_{n \in \mathbb{N}}$ parameterized by the input length n. We simply speak of F to implicitly mean F_n whenever there is no cause for confusion.

[7] This condition is to rule out trivial protocols, e.g., for a class of functions that can be computed in time less than $p(\lambda)$.

and

$$\textsc{sk}_{\mathsf{F}} \leftarrow \mathsf{ABE.KeyGen}(\textsc{mpk}_1, \textsc{msk}_1, \mathsf{F}) \text{ (corresponding to F).}$$

Output the public key $\mathsf{PK}_{\mathsf{F}} := (\textsc{mpk}_0, \textsc{mpk}_1)$ and the evaluation key $\mathsf{EK}_{\mathsf{F}} := (\textsc{sk}_{\bar{\mathsf{F}}}, \textsc{sk}_{\mathsf{F}})$.

- $\mathsf{ProbGen}(\mathsf{PK}_{\mathsf{F}}, \mathbf{x})$: On input $\mathbf{x}$ and the public key PK_{F}, sample two uniformly messages m_0 and m_1 of equal length from the message space, compute the ciphertexts

$$\textsc{ct}_{\mathbf{x},0} \leftarrow \mathsf{ABE.Enc}(\textsc{mpk}_0, \mathbf{x}, m_0) \quad \text{and} \quad \textsc{ct}_{\mathbf{x},1} \leftarrow \mathsf{ABE.Enc}(\textsc{mpk}_1, \mathbf{x}, m_1).$$

  Output the value $\sigma_{\mathbf{x}} := (\textsc{ct}_{\mathbf{x},0}, \textsc{ct}_{\mathbf{x},1})$ (to be sent to the worker), and the verification key $\mathsf{VK}_{\mathbf{x}} := (H(m_0), H(m_1))$, where H is the one-way function.

- $\mathsf{Compute}(\mathsf{EK}_{\mathsf{F}}, \sigma_{\mathbf{x}})$: On input the value $\sigma_{\mathbf{x}} := (\textsc{ct}_{\mathbf{x},0}, \textsc{ct}_{\mathbf{x},1})$ and the evaluation key $\mathsf{EK}_{\mathsf{F}} := (\textsc{sk}_{\bar{\mathsf{F}}}, \textsc{sk}_{\mathsf{F}})$, compute

$$m_0' \leftarrow \mathsf{ABE.Dec}(\textsc{msk}_0, \textsc{sk}_{\bar{\mathsf{F}}}, \textsc{ct}_{\mathbf{x},0}) \quad \text{and} \quad m_1' \leftarrow \mathsf{ABE.Dec}(\textsc{msk}_1, \textsc{sk}_{\mathsf{F}}, \textsc{ct}_{\mathbf{x},1}).$$

  Output $\sigma_{\mathsf{out}} := (m_0', m_1')$.

- $\mathsf{Verify}(\mathsf{VK}_{\mathbf{x}}, \sigma_{\mathsf{out}})$: On input $\mathsf{VK}_{\mathbf{x}} := (H(m_0), H(m_1))$ and $\sigma_{\mathsf{out}} := (m_0', m_1')$. Output

$$\mathbf{y} := \begin{cases} 0 & \text{if } H(m_0') = H(m_0) \text{ and } H(m_1') \neq H(m_1) \\ 1 & \text{if } H(m_1') = H(m_1) \text{ and } H(m_0') \neq H(m_0) \\ \perp & \text{otherwise} \end{cases}$$

C On Assumption 3

In this section, we provide more discussion on Assumption 3. We first show that Assumption 3 is implied by the following assumption, which is essentially the 3PDH assumption from [6] except that we consider three (instead of two) subgroups.

Assumption 6 (3PDH: Composite 3-party Diffie-Hellman Assumption) *Given a group generator $\mathcal{G}$, we define the following distribution:*

$$\begin{aligned}
&\mathbb{G} := (N = p_1p_2p_3, G_N, G_T, e) \leftarrow_{\mathrm{R}} \mathcal{G}, \\
&a, b, c \leftarrow_{\mathrm{R}} \mathbb{Z}_N, \\
&g_1 \leftarrow_{\mathrm{R}} G_{p_1},\ g_2 \leftarrow_{\mathrm{R}} G_{p_2},\ g_3, R_3, S_3, T_3 \leftarrow_{\mathrm{R}} G_{p_3}, \\
&T_0 := g_2^c T_3,\ T_1 := g_2^{c+d} T_3, \\
&D := (\mathbb{G}; g_1, g_2, g_2^a, g_2^b, g_2^{ab} R_3, g_2^{abc} S_3, g_3).
\end{aligned}$$

We assume that for any PPT algorithm $\mathcal{A}$,

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

is negligible in the security parameter λ.

Lemma 8 (3PDH Implies AS3). *For any adversary $\mathcal{A}$, there exists an adversary $\mathcal{B}$ such that:*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathrm{AS3}}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}}^{\mathrm{3PDH}}(\lambda).$$

and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + \mathrm{poly}(\lambda)$ *where* $\mathrm{poly}(\lambda)$ *is independent of* $\mathsf{Time}(\mathcal{A})$.

Proof. We construct an adversary $\mathcal{B}$ for 3PDH assumption using $\mathcal{A}$. Recall that in 3PDH assumption, the adversary is given

$$\Big(\ \mathbb{G}; g_1, g_2, g_2^a, g_2^b, g_2^{ab}R_3, g_2^{abc}S_3, g_3, g_2^{c+d}T_3\ \Big),$$

where either $d = 0$ or $d \leftarrow_{\mathrm{R}} \mathbb{Z}_p$; $\mathcal{B}$ picks

$$U_1 \leftarrow_{\mathrm{R}} G_{p_1},\ U_2 \leftarrow_{\mathrm{R}} G_{p_2},\ U_3 \leftarrow_{\mathrm{R}} G_{p_3},$$

implicitly set

$$\begin{aligned} x &:= ab, \quad & y &:= c + d, \quad & z &:= -abd,\\ X_3 &:= R_3, \quad & Y_3 &:= T_3, \quad & W_3 &:= S_3, \end{aligned}$$

and output

$$\Big(\ \mathbb{G}; g_1, U_1U_2, g_2^{ab}R_3, g_2^{c+d}T_3, g_2U_3, g_3, g_2^{abc}S_3\ \Big).$$

Now, observe that $xy + z = ab(c + d) - abc = abc$ and

- if $d = 0$, the output is distributed as $(\mathbb{G}; g_1, U_1U_2, g_2^xX_3, g_2^yY_3, g_2U_3, g_3, g_2^{xy}W_3)$;
- if $d \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, the output is distributed as $(\mathbb{G}; g_1, U_1U_2, g_2^xX_3, g_2^yY_3, g_2U_3, g_3, g_2^{xy+z}W_3)$.

The lemma then follows readily. □

Next, we prove that Assumption 3 holds in the generic group model. Instead of working directly with Assumption 3, we introduce Assumption 7, which is simpler to state and trivially implies Assumption 3, and then show that Assumption 7 holds in the generic group model.

Assumption 7 *Given a group generator $\mathcal{G}$, we define the following distribution:*

$$\begin{aligned} &\mathbb{G} := (N = p_1p_2p_3, G, G_T, e) \leftarrow_{\mathrm{R}} \mathcal{G},\\ &x, y, z \leftarrow_{\mathrm{R}} \mathbb{Z}_N,\\ &g_1 \leftarrow_{\mathrm{R}} G_{p_1},\ g_2 \leftarrow_{\mathrm{R}} G_{p_2},\ g_3, X_3, Y_3 \leftarrow_{\mathrm{R}} G_{p_3},\\ &T_0 = g_2^{xy},\ T_1 = g_2^z,\\ &D := (\mathbb{G}; g_1, g_2, g_3, g_2^xX_3, g_2^yY_3). \end{aligned}$$

We assume that for any PPT algorithm $\mathcal{A}$,

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{AS6}}(\lambda) := \big|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\big|$$

is negligible in the security parameter λ.

Lemma 9. *Assumption 7 holds in the generic group model under the assumption that finding a non-trivial factor of N (the group order) is hard.*

Proof. We appeal to the general framework introduced in [22, Theorem A.2] for proving assumptions in the generic group model. In the framework, Assumption 7 may be written as:

$$A_1 = (1, 0, 0),\ A_2 = (0, 1, 0),\ A_3 = (0, 0, 1),\ A_4 = (0, X, X_3),\ A_5 = (0, Y, Y_3),$$
$$T_0 = (0, XY, 0),\ T_1 = (0, Z, 0).$$

We have $S := \{i \mid e(T_0, A_i) \neq e(T_1, A_i)\} = \{2, 4, 5\}$. We need to verify that the following conditions holds:

1. Each of T_0 and T_1 is independent of $\{A_i\}$.

2. For $k \in S$ it holds that $e(T_0, A_k)$ is independent of $\{e(A_i, A_j)\} \bigcup \{e(T_0, A_i)\}_{i \neq k}$ and $e(T_1, A_k)$ is independent of $\{e(A_i, A_j)\} \bigcup \{e(T_1, A_i)\}_{i \neq k}$.

It is easy to see that Condition 1 holds since the second component of T_0 (resp. T_1) contains XY (resp. Z) that is not present amongst A_1, A_2, A_3, A_4, A_5. Next, we proceed to condition 2. We start with T_0 and obtain the following tuples:

$$C_{0,2} := e(T_0, A_2) = [0, XY, 0],\ C_{0,4} := e(T_0, A_4) = [0, X^2Y, 0],\ C_{0,5} := e(T_0, A_5) = [0, XY^2, 0],$$
$$A_{2,4} := e(A_2, A_4) = [0, X, 0],\ A_{2,5} := e(A_2, A_5) = [0, Y, 0],$$
$$A_{3,4} := e(A_3, A_4) = [0, 0, X_3],\ A_{3,5} := e(A_3, A_5) = [0, 0, Y_3],$$
$$A_{4,5} := e(A_4, A_5) = [0, XY, X_3Y_3].$$

Here, we omit all $[0, 0, 0]$ terms. We need to verify the following three statements:

- $C_{0,2} = [0, XY, 0]$ is independent of $C_{0,4} = [0, X^2Y, 0]$, $C_{0,5} = [0, XY^2, 0]$, $A_{2,4} = [0, X, 0]$, $A_{2,5} = [0, Y, 0]$, $A_{3,4} = [0, 0, X_3]$, $A_{3,5} = [0, 0, Y_3]$, and $A_{4,5} = [0, XY, X_3Y_3]$ since the only other way to obtain an element whose second component contains XY is from $A_{4,5}$, which yields the element $[0, XY, X_3Y_3]$. But there is no other way to generate an element whose third component is X_3Y_3, and hence no way to cancel that term.
- $C_{0,4} = [0, X^2Y, 0]$ is independent of $C_{0,2} = [0, XY, 0]$, $C_{0,5} = [0, XY^2, 0]$, $A_{2,4} = [0, X, 0]$, $A_{2,5} = [0, Y, 0]$, $A_{3,4} = [0, 0, X_3]$, $A_{3,5} = [0, 0, Y_3]$, and $A_{4,5} = [0, XY, X_3Y_3]$ since the second component contains X^2Y that cannot be generated any other way.
- $C_{0,5} = [0, XY^2, 0]$ is independent of $C_{0,2} = [0, XY, 0]$, $C_{0,4} = [0, X^2Y, 0]$, $A_{2,4} = [0, X, 0]$, $A_{2,5} = [0, Y, 0]$, $A_{3,4} = [0, 0, X_3]$, $A_{3,5} = [0, 0, Y_3]$, and $A_{4,5} = [0, XY, X_3Y_3]$ since the second component contains XY^2 that cannot be generated any other way.

Finally, we need to verify Condition 2 for T_1. Here, we obtain the following tuples:

$$C_{1,2} := e(T_1, A_2) = [0, Z, 0],\ C_{1,4} := e(T_1, A_4) = [0, ZX, 0],\ C_{1,5} := e(T_1, A_5) = [0, ZY, 0].$$

Observe that each of these three tuples are independent of the other two tuples. Moreover, they are independent of the $\{e(A_i, A_j)\}$ terms, since the latter do not contain a Z-term. We conclude that Assumption 7 holds in the generic group model. □

D Proof of Lemma 4

We assume there exists a PPT algorithm $\mathcal{A}$ breaking the Subspace assumption with non-negligible advantage $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DS1}}(\lambda)$. We create a PPT algorithm $\mathcal{B}$ which breaks the DDH assumption in G_1 with non-negligible advantage $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{DS1}}(\lambda)$. $\mathcal{B}$ is given $D := (\mathbb{G}; g_1, g_2, g_1^x, g_1^y)$ along with T, where T is distributed as g_1^{xy} or g_1^{xy+z}.

$\mathcal{B}$ first samples $n+1$ pairs of random dual orthonormal bases, denoted by

$$(\tilde{\mathbf{d}}_0, \tilde{\mathbf{d}}_0^*, \tilde{\mathbf{f}}_0, \tilde{\mathbf{f}}_0^*), \ldots, (\tilde{\mathbf{d}}_n, \tilde{\mathbf{d}}_n^*, \tilde{\mathbf{f}}_n, \tilde{\mathbf{f}}_n^*).$$

Then, $\mathcal{B}$ implicitly sets:

$$\mathbf{d}_0 := \tilde{\mathbf{d}}_0 + x\tilde{\mathbf{f}}_0, \ldots, \mathbf{d}_n := \tilde{\mathbf{d}}_n + x\tilde{\mathbf{f}}_n,$$
$$\mathbf{f}_0 := \tilde{\mathbf{f}}_0, \ldots, \mathbf{f}_n := \tilde{\mathbf{f}}_n,$$

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

$$\mathbf{d}_0^* := \tilde{\mathbf{d}}_0, \ldots, \mathbf{d}_n^* := \tilde{\mathbf{d}}_n,$$
$$\mathbf{f}_0^* := \tilde{\mathbf{f}}_0^* - x\tilde{\mathbf{d}}_0^*, \ldots, \mathbf{f}_n^* := \tilde{\mathbf{f}}_n^* - x\tilde{\mathbf{d}}_n^*.$$

It is clear that $(\mathbf{d}_0, \mathbf{d}_0^*, \mathbf{f}_0, \mathbf{f}_0^*), \ldots, (\mathbf{d}_n, \mathbf{d}_n^*, \mathbf{f}_n, \mathbf{f}_n^*)$ are properly distributed. We note that $\mathcal{B}$ can produce all of $g_1^{\mathbf{d}_0}, g_1^{\mathbf{f}_0}, \ldots, g_1^{\mathbf{d}_n}, g_1^{\mathbf{f}_n}$ (given g_1, g_1^x) as well as $g_2^{\mathbf{d}_0^*}, \ldots, g_2^{\mathbf{d}_n^*}$ (given g_2). However, $\mathcal{B}$ cannot produce $g_2^{\mathbf{f}_0^*}, \ldots, g_2^{\mathbf{f}_n^*}$ (these require knowledge of g_2^x). $\mathcal{B}$ proceeds as follows:

- $\mathcal{B}$ creates $U_0, \ldots, U_n$ by choosing random values $\mu_1', \mu_2' \in \mathbb{Z}_q$ and setting:

$$U_0 := g_2^{\mu_1'\mathbf{d}_0^* + \mu_2'\tilde{\mathbf{f}}_0^*} = g_2^{(\mu_1' + x\mu_2')\mathbf{d}_0^* + \mu_2'\mathbf{f}_0^*},$$
$$\vdots$$
$$U_n := g_2^{\mu_1'\mathbf{d}_n^* + \mu_2'\tilde{\mathbf{f}}_n^*} = g_2^{(\mu_1' + x\mu_2')\mathbf{d}_n^* + \mu_2'\mathbf{f}_n^*},$$

  where $\mathcal{B}$ has implicitly set $\mu_1 := \mu_1' + x\mu_2'$ and $\mu_2 := \mu_2'$.

- Next, $\mathcal{B}$ computes:

$$T_0 := T^{\tilde{\mathbf{f}}_0} \cdot (g_1^y)^{\tilde{\mathbf{d}}_0}, \ldots, T_n := T^{\tilde{\mathbf{f}}_n} \cdot (g_1^y)^{\tilde{\mathbf{d}}_n}.$$

  Now, suppose $T = g_1^{xy+z}$. Then

$$T_0 = (g_1^{xy+z})^{\tilde{\mathbf{f}}_0} \cdot (g_1^y)^{\tilde{\mathbf{d}}_0} = (g_1^{xy+z})^{\mathbf{f}_0} \cdot (g_1^y)^{(\mathbf{d}_0 - x\mathbf{f}_0)} = g_1^{y\mathbf{d}_0 + z\mathbf{f}_0},$$
$$\vdots$$
$$T_n = (g_1^{xy+z})^{\tilde{\mathbf{f}}_n} \cdot (g_1^y)^{\tilde{\mathbf{d}}_n} = (g_1^{xy+z})^{\mathbf{f}_n} \cdot (g_1^y)^{(\mathbf{d}_n - x\mathbf{f}_n)} = g_1^{y\mathbf{d}_n + z\mathbf{f}_n},$$

  where $\mathcal{B}$ implicitly sets $\tau_1 := y, \tau_2 := z$. Observe that if $z = 0$ (namely, $T = g_1^{xy}$), $T_0, \ldots, T_n$ would indeed be distributed as $V_0, \ldots, V_n$. On the other hand, if $z \leftarrow_{\mathrm{R}} \mathbb{Z}_q$ instead, $T_0, \ldots, T_n$ would indeed be distributed as $W_0, \ldots, W_n$.

- Finally, $\mathcal{B}$ gives

$$D := \Big(\mathbb{G}; g_1^{\mathbf{d}_0}, g_1^{\mathbf{f}_0}, g_2^{\mathbf{d}_0^*}, \ldots, g_1^{\mathbf{d}_n}, g_1^{\mathbf{f}_n}, g_2^{\mathbf{d}_n^*}, U_0, \ldots, U_n, \mu_2 \Big)$$

  to $\mathcal{A}$, along with $T_0, \ldots, T_n$.

We may then conclude that: $\mathsf{Adv}_{\mathcal{A}}^{\mathtt{DS1}}(\lambda) \leq \mathsf{Adv}_{\mathcal{B}}^{\mathtt{DDH1}}(\lambda)$.