Extended Nested Dual System Groups, Revisited

Junqing Gong* Jie Chen† Xiaolei Dong‡ Zhenfu Cao§ Shaohua Tang¶

October 7, 2015

Abstract

The notion of extended nested dual system groups (ENDSG) was recently proposed by Hofheinz *et al.* [PKC 2015] for constructing almost-tight identity based encryptions (IBE) in the multi-instance, multi-ciphertext (MIMC) setting. However only a composite-order instantiation was proposed and more efficient prime-order instantiations are absent. The paper fills the blank by presenting two constructions.

We revise the definition of ENDSG and realize it using prime-order bilinear groups based on Chen and Wee's prime-order instantiation of nested dual system groups [CRYPTO 2013]. This yields the first almost-tight IBE in the prime-order setting achieving *weak* adaptive security in MIMC scenario under the d-linear (d-Lin) assumption. We further enhanced the revised ENDSG to capture stronger security notions for IBE, including *B-weak* adaptive security and *full* adaptive security. We show that our prime-order instantiation is readily B-weak adaptive secure and full adaptive secure without introducing extra assumption.

We then try to find better solution by fine-tuning ENDSG again and realizing it using the technique of Chen, Gay, and Wee [EUROCRYPT 2015]. This leads to an almost-tight secure IBE in the same setting with better performance than our first result, but the security relies on a non-standard assumption, d-linear assumption with auxiliary input (d-LinAI) for an even positive integer d. However we note that, the 2-LinAI assumption is implied by the external decisional linear (XDLIN) assumption. This concrete instantiation could also be realized using *symmetric* bilinear groups under standard decisional linear assumption.

Keywords: Identity based encryptions, Dual system groups, Tight security, Security model, Prime-order bilinear groups

*Department of Computer Science and Engineering, Shanghai Jiao Tong University. Email: gongjunqing@126.com

†Shanghai Key Laboratory of Multidimensional Information Processing and Shanghai Key Lab of Trustworthy Computing, East China Normal University. Email: S080001@e.ntu.edu.sg

‡Shanghai Key Lab for Trustworthy Computing, East China Normal University. Email: dongxiaolei@sei.ecnu.edu.cn

§Shanghai Key Lab for Trustworthy Computing, East China Normal University. Email: zfcao@sei.ecnu.edu.cn

¶School of Computer Science & Engineering, South China University of Technology. Email: shtang@IEEE.org

Contents

1 Introduction

1.1 Background and Problem

Recently we have witnessed a breakthrough of proof technique in the field of functional encryptions. In 2009, Waters [Wat09] proposed a new proof paradigm for identity based encryptions (IBE), called *dual system technique*, and obtained the first adaptively secure IBE with short public key in the standard model whose security relies on a static assumption and the security loss is $O(q)$ where q is the number of key extraction queries. From a high-level view, the dual system technique works with two copies of some target cryptographic primitive such as IBE. The first copy is put into the so-called *normal space* and acts as the real system, while the second copy is put into the so-called *semi-functional space* and only used in the proof. Furthermore, the independence of the two spaces (say, orthogonality under pairing operations) allows us to make some changes in the semi-functional space for proof but still maintain the correctness in the normal space. It is worth noting that the new technique permits the simulator to reply all queries made by the adversary and avoids the security loss caused by the classical partition technique [BF01, BB04a, Wat05].

The revolution was then spreading across the field of functional encryptions. In particular, the dual system technique has been applied for establishing adaptive security of various types of functional encryptions, ranging from simple functionality, such as IBE [BKP14, CW14, JR13, CW13, Lew12, CLL+12, RCS12] to expressive and complicated functionality, like ABE and IPE [LOS+10, LW12, OT12, Att14, CW14, Wee14, AY15, CGW15]. Some of them applied the dual system technique in a modular and abstract fashion such as Wee's predicate encoding [Wee14] and Attrapadung's pairing encoding [Att14].

The dual system technique also helped us to go further. Chen and Wee [CW13] combined the dual system technique with the proof idea underlying the Naor-Reingold pseudorandom function [NR04] and achieved the first almost-tight IBE from standard assumption in the standard model. The security loss is $O(n)$ where n the length of identities, and unrelated to the number of key extraction queries anymore. They established the real system in the normal space and a mirror one in the semi-functional space for proof as the original dual system technique [Wat09]. However, instead of dealing with key extraction queries (in the semi-functional space) separately as Waters [Wat09], they handled all (i.e., q) secret keys as a whole in the next step following the proof strategy of Naor and Reingold [NR04]. In detail, we may imagine the master secret key as a truly random function taking identities as input. Starting from the original master secret key whose domain is just $\{\epsilon\}$, the proof argues that we can double the domain size until it reaches the size of the identity space if identities are encoded in a bit-by-bit fashion [Wat05]. For identity space $\{0,1\}^n$, only n steps are required. Finally, the property of the random function allows us to information-theoretically hide the challenge message.

Recent work by Hofheinz *et al.* [HKS15] extended Chen and Wee's result [CW13] and achieved almost tightness in the *multi-instance, multi-ciphertext* (MIMC) setting where the adversary simultaneously attacks multiple challenge identities in multiple IBE instances. In Chen and Wee's paradigm [CW13], the ith step that increases the domain size from 2^{i-1} to 2^i can only handle the situation where all challenge ciphertexts share the same ith bit, which no longer holds in MIMC setting. The proposed solution [HKS15] is to further split the semi-functional space into two independent (in some sense) subspaces, labelled by $\wedge$ and $\sim$ respectively. The ith step starts from ciphertexts with $\wedge$-semi-functional component. We then move the semi-functional components in all ciphertexts for identities whose ith bit is 1 to the $\sim$-semi-functional space. At this moment, (1) in the $\wedge$-semi-functional space, all ciphertexts share the same ith bit 0; (2) in the $\sim$-semi-functional space, all ciphertexts share the same ith bit 1, which means that one can now applied Chen and Wee's proof strategy [CW13] in both subspaces separately.

Unfortunately, only an instantiation using *composite-order* bilinear groups was proposed in [HKS15]. Our goal is to realize a fully and almost-tightly secure IBE in MIMC setting using *prime-order* bilinear groups. We emphasize that it is not just a theoretical interest to pursue such a solution. Most schemes (including [HKS15]) using composite-order bilinear groups base their security on the *Subgroup Decision Assumption* [BWY11] which implies the hardness of factoring the group order. This forces us to work with elliptic curve groups with quite large, say 1024 bits, base field when implementing the scheme. In contrast, for constructions in the prime-order setting, we could employ smaller base field, say 160 bits, without sacrificing the security level. Although the construction now becomes complex in general, this still brings us a considerable advantage in both computation and space efficiency.

1.2 Motivation and Observation

Hofheinz *et al.*'s work [HKS15] roughly follows the style of [CW13]. In particular, they first extended the notion of *Nested Dual System Groups* (NDSG) proposed by Chen and Wee [CW13], then proposed a general IBE construction from the extended NDSG (ENDSG) in MIMC setting, and finally presented an instantiation of ENDSG using composite-order bilinear groups. Therefore it is sufficient for our purpose to realize ENDSG

using prime-order bilinear groups and apply the general transformation in [HKS15]. However we observe that the definition of ENDSG in [HKS15] sets too strong requirements on algebraic structure of underlying groups, which makes it hard to be instantiated using existing techniques for prime-order bilinear groups.

An ENDSG describes a set of abstract groups with a bunch of structural and computational requirements supporting Hofheinz *et al.*'s proof strategy. We roughly recall[1] that an ENDSG defined in [HKS15] consists of five algorithms: SampP, SampG, SampH, $\widehat{\mathsf{SampG}}$, and $\widetilde{\mathsf{SampG}}$. Informally, the first algorithm generates a set of groups $\mathbb{G}, \mathbb{H}, \mathbb{G}_T$ of order N (as well as other parameters) and the other four algorithms are used to sample random elements from some subgroup of $\mathbb{G}$ or $\mathbb{H}$ (which are associated with ciphertexts and secret keys, respectively, in the context of IBE). We emphasize that they required that

- Groups $\mathbb{G}$ and $\mathbb{H}$ are generated by some $g \in \mathbb{G}$ and $h \in \mathbb{H}$, respectively. (From the specification of group generator G.)
- The outputs of SampG, $\widehat{\mathsf{SampG}}$, and $\widetilde{\mathsf{SampG}}$ are distributed uniformly over the generators of different nontrivial subgroups of $\mathbb{G}^{n+1}$ of coprime order, respectively. (From the G-subgroups.)

However, nearly all techniques realizing dual system technique in the prime-order setting employs vector spaces over $\mathbb{F}_p$ (for a prime p) to simulate group $\mathbb{G}$ and $\mathbb{H}$ [Lew12, LW12, OT12, CW13, CW14, CGW15]. Meanwhile subgroups of $\mathbb{G}$ and $\mathbb{H}$ are naturally simulated by its subspaces. Firstly, since a vector space is an additive group but not cyclic in general, neither $\mathbb{G}$ nor $\mathbb{H}$ is cyclic. Secondly, any d-dimensional subspace has p^d vectors, thus the orders of the outputs of SampG, $\widehat{\mathsf{SampG}}$, and $\widetilde{\mathsf{SampG}}$ must share a common factor p. In a word, techniques based on vector spaces by no means meets the requirements shown above.

Fortunately, we observe that both requirements are applied nowhere but to provide random self-reducibility of computational requirements (including LS1, LS2, NH) when they proved "ENDSG implies IBE". For example, the *Left-subgroup indistinguishability 1* (LS1) said that, for any $(\mathrm{PP}, \mathrm{SP}) \leftarrow \mathsf{SampP}(k, n)$, the following two distributions are computationally indistinguishable.

$$\{\mathbf{g} : \mathbf{g} \leftarrow \mathsf{SampG}(\mathrm{PP})\} \quad \text{and} \quad \left\{\mathbf{g} \cdot \widehat{\mathbf{g}} : \mathbf{g} \leftarrow \mathsf{SampG}(\mathrm{PP}),\ \widehat{\mathbf{g}} \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})\right\}.$$

Given $\mathbf{T}$ which is either $\mathbf{g}$ or $\mathbf{g} \cdot \widehat{\mathbf{g}}$, the simulator (in the proof) can sample $s \leftarrow \mathbb{Z}_N^*$ and generate another independent problem instance $\mathbf{T}^s$ following the two requirements we have reviewed. We note that this property is crucial for achieving almost-tight reduction in MIMC setting where the adversary is able to enquire more than one challenge ciphertext. This suggests that, if we adapt the ENDSG to support such random self-reducibility explicitly, it will still imply an IBE in MIMC setting and the limitations on underlying groups may be removed. As this happens, many existing techniques in the prime-order setting can now be applied to realize ENDSG and finally derives an almost-tight IBE in MIMC setting using *prime-order* bilinear groups.

1.3 Contributions and Techniques

In this paper, we revise the definition of ENDSG, and show that the revised ENDSG not only implies an IBE in MIMC setting but also can be almost-tightly instantiated using prime-order bilinear groups. Putting them together, we obtain an almost-tight IBE in the same setting from prime-order bilinear groups. In particular, we proposed two instantiations: the first one is proven secure under the d-linear assumption (d-Lin), while the second one is proven secure under a stronger assumption, d-linear assumption with auxiliary input, d-LinAI for short, but achieves shorter keys and ciphertexts.

Revisiting Extended Nested Dual System Groups. Our ENDSG is defined mainly in the spirit of [HKS15] but with the difference that we provide (in requirements like LS1) enough independently-sampled subgroup elements directly instead of assuming some special algebraic structure. As an example, we define LS1 as: for any $(\mathrm{PP}, \mathrm{SP}) \leftarrow \mathsf{SampP}(k, n)$, the following two distributions are computationally indistinguishable.

$$\left\{\left\{\mathbf{g}_j\right\}_{j\in[q]} : \mathbf{g}_j \leftarrow \mathsf{SampG}(\mathrm{PP})\right\} \quad \text{and} \quad \left\{\left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j\right\}_{j\in[q]} : \mathbf{g}_j \leftarrow \mathsf{SampG}(\mathrm{PP}),\ \widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})\right\}.$$

Here the parameter q depends on the number of challenge ciphertexts. This makes the definition more general and allows us to realize the notion using diverse algebra frameworks, especially prime-order bilinear groups. On the other hand, it still almost-tightly implies a fully secure IBE in MIMC setting. The construction and the proof are nearly the same as [HKS15].

To be fair, Hofheinz *et al.*'s definition is more convenient in the sense that any instantiation of ENDSG immediately results in an *almost-tight* IBE in MIMC setting. In contrast, an instantiation of our definition with

[1] The notation is slightly different from [HKS15].

loose security reduction (say, with security loss $\mathcal{O}(q)$) clearly can not lead to *tightly secure* IBE. Hence, when working with our definition, we should not jump to the conclusion before checking the tightness. We also remark that we *do not* negate prime-order instantiations of Hofheinz *et al.*'s ENDSG.

Instantiation from d-Linear Assumption. We implement our revised ENDSG by extending the prime-order instantiation of NDSG by Chen and Wee [CW13]. The security only relies on the d-Lin assumption and the security loss is $\mathcal{O}(d)$ and independent of the number of samples, say q, given to the adversary. By the generic construction [HKS15] (also c.f. Appendix B), we obtain the first almost-tight IBE in MIMC setting in the prime-order setting and fill the blank left by Hofheinz *et al.* [HKS15].

Technically, we extend the basis from $2d \times 2d$ matrix used in [CW13] to $3d \times 3d$ matrix in order to accommodate the additional semi-functional space. In detail, the first d-dimension subspace is the normal space, the next d-dimension subspace is the $\wedge$-semi-functional space, and the last d-dimension subspace is the $\sim$-semi-functional space.

The main challenge is to realize the *Left Subgroup Indistinguishability 2 (LS2)* property (c.f. Section 3). Roughly, we must prove that $\mathbf{g} \cdot \widehat{\mathbf{g}}$ (sampled from the normal space and $\wedge$-semi-functional space of $\mathbb{G}$) and $\mathbf{g} \cdot \widetilde{\mathbf{g}}$ (sampled from the normal space and $\sim$-semi-functional space of $\mathbb{G}$) are computationally indistinguishable even when the adversary can access to $\widehat{h}^* \cdot \widetilde{h}^* \in \mathbb{H}$ where $\widehat{h}^* \in \mathbb{H}$ is orthogonal to the normal and $\sim$-semi-functional space of $\mathbb{G}$ and $\widetilde{h}^* \in \mathbb{H}$ to the normal and $\wedge$-semi-functional space of $\mathbb{G}$. To simulate $\widehat{h}^* \cdot \widetilde{h}^*$, we further extend the subspace of $\widehat{h}^*$ and $\widetilde{h}^*$ from 1-dimension to d-dimension which allows us to utilize the technique for proving *right subgroup indistinguishability* property of Chen-Wee's prime-order instantiation of dual system groups [CW14]. So as to support this technical extension and conform to our revision, we model the process of sampling $\widehat{h}^*$ and $\widetilde{h}^*$ as two algorithms $\widehat{\mathsf{SampH}}^*$ and $\widetilde{\mathsf{SampH}}^*$ respectively, and give adversary adequate samples in related computational requirements.

Achieving Stronger Security Guarantee. Hofheinz *et al.* [HKS15] achieved weak security from their ENDSG where the adversary is allowed to make single challenge query for each identity in each instance. They introduced a variant of the BDDH assumption (s-BDDH) and proved the full security of their original construction where the above restriction on the adversary is removed. This additional computational requirement is realized under the *dual system bilinear DDH assumption* (DS-BDDH).

The revisions we have made do not involve the s-BDDH assumption, and the resulting ENDSG only leads to weak security. Motivated by and based on our prime-order instantiation, we investigate two flavors of stronger security: *B-weak* and *full adaptive security*. The former model allows adversary to make at most B challenge queries for each identity in each instance where B is a prior bound, while the latter one sets no limitation on the number of challenge queries on a single identity, i.e., polynomially many queries are allowed.

For each of them, we follow Hofheinz *et al.*'s method. Concretely, to achieve stronger security, we only need to enhance the *non-degeneracy* property in our revised ENDSG (see Section 3) and updating the indistinguishability between Game_3 and Game_4 (reviewed in Section 6, c.f. Appendix B) in Hofheinz *et al.*'s proof to make it sound in stronger models, where the non-degeneracy property is applied. We then prove that our instantiation of ENDSG under the d-Lin assumption (see Section 4) satisfies the enhanced non-degeneracy property. The two results together imply an IBE with stronger security guarantee and almost-tight reduction in MIMC setting. In particular,

1. We enhance the non-degenerate property to *B-bounded* version which states that the non-degeneracy property holds even when a single $\widehat{h}^*$ works with B $\widehat{g}_0$'s where B is a prior bound. It is easy to show that our instantiation under the d-Lin assumption is d-bounded non-degenerated unconditionally.

2. We enhance the non-degeneracy property to *computational* version which is essentially similar to the s-BDDH assumption [HKS15] and states that the non-degeneracy property holds even when a single $\widehat{h}^*$ works with *polynomially many* $\widehat{g}_0$'s. Luckily, we can prove that our instantiation is computationally non-degenerated under the d-Lin assumption, and no additional assumption is required.

Towards More Efficient Instantiation. Having obtained the first construction, we continue to purse more efficient solutions. One possible method is to reduce the dimension of two semi-functional spaces. Because we hope to continue to base our construction on the standard d-Lin assumption, we found the attempt gives rise to two technical problems due to the lack of space.

– We can not prove *Left Subgroup Indistinguishability 2* (LS2) property using the technique provided by Chen and Wee in [CW14]. In particular, the simulator will need some elements in another source group to simulate $\widehat{h}^* \cdot \widetilde{h}^*$ which is not given in the standard d-Lin assumption.

– We can not prove *Computational Non-degeneracy* (ND) property as before since neither $\widehat{g}_0$ nor $\widehat{h}$ has enough space to program the d-Lin problem during the simulation.

The second issue is easy to solve by the observation that there are two semi-functional spaces and we only use one of them so far. We first define a variant of computational non-degeneracy property taking the $\sim$ semi-functional space into account. Then if two semi-functional spaces together has at least d dimensions, this computational non-degeneracy property should be proved as before. On the other hand, from the view of IBE, we could use the pseudo-randomness of $e(\widehat{g}_0 \cdot \widetilde{g}_0, \widehat{h}^* \cdot \widetilde{h}^*)$ to prove the security (from Game_3 to Game_4, c.f. Appendiex B) instead of just $e(\widehat{g}_0, \widehat{h}^*)$. To make the intuition explicit and general, we define three Left-subgroup indistinguishability (LS) requirements as: (1) LS1: $\mathbf{g} \approx \mathbf{g} \cdot \widehat{\mathbf{g}} \cdot \widetilde{\mathbf{g}}$; (2) LS2: $\mathbf{g} \cdot \widehat{\mathbf{g}} \cdot \widetilde{\mathbf{g}} \approx \mathbf{g} \cdot \widetilde{\mathbf{g}}$; (3) LS3: $\mathbf{g} \cdot \widehat{\mathbf{g}} \cdot \widetilde{\mathbf{g}} \approx \mathbf{g} \cdot \widehat{\mathbf{g}}$, where $\approx$ stands for "computationally indistinguishable".

In contrast, the first issue is seemingly hard to circumvent. Therefore, we decide to prove the LS2 property under an enhanced d-Lin assumption where we give adversary more elements on another source group for simulating $\widehat{h}^* \cdot \widetilde{h}^*$, which is called d-linear assumption with auxiliary input (d-LinAI, c.f. Section 8) for *an even positive integer* d. Even though this assumption is non-standard in general, we point out that the concrete assumption with $d = 2$ is implied by the external decision linear assumption (XDLIN) [ACD+12] (c.f. Section 9 and Appendix A.2), which has been formally introduced and used to build other cryptographic primitives.

We further fine-tune the ENDSG by hiding public parameters for SampH from the adversary when defining computational requirements, including LS1, LS2, LS3, NH, and ND. We argue that the absence of this part of public parameters will not arise difficulty in building IBE since they always correspond to the master secret key which is not necessary to be public according to the security model. Instead, we give the adversary enough samples from $\mathbb{H}^{n+1}$ which is sufficient for answering key extraction queries in the proof of "ENDSG implies IBE". We hope it will bring us a simple, clean and efficient solution.

In summary, we have fine-tuned the ENDSG in three aspects: (1) update non-degeneracy requirement; (2) re-define LS requirements; (3) hide parameters for SampH. We describe the fine-tuned ENDSG in Section 7 and verify in Appendix C that these modifications won't prevent ENDSG from almost-tightly deriving a fully secure IBE in MIMC setting.

The start point of instantiating the fine-tuned ENDSG is the prime-order instantiation of dual system groups recently proposed by Chen *et al.* [CGW15], which is quite simple due to a new basis randomizing technique. We technically work with $2d \times 2d$ matrix and generate the basis using the dual pairing vector space method [OT08, OT09, LOS+10]. The first d-dimension subspace is normal space, the remaining two $d/2$-dimension subspaces act as $\wedge$ semi-functional subspace and $\sim$ semi-functional subspace, respectively. Note that the latter two are now smaller but enough for our proof (the entire semi-functional space has d dimension). Finally, the basis is then randomized following [CGW15]. Here we always assume positive integer d is even.

The security of this instantiation is almost tightly reduced to the d-LinAI assumption, which leads to an almost-tightly secure IBE in MIMC setting with full security and higher efficiency than our first construction. As we have mentioned, the concrete IBE with $d = 2$ is based on the XDLIN [ACD+12]. This also suggests that this concrete construction can be further adapted to work with *symmetric* bilinear groups and the security is now based on the decisional linear assumption in the *symmetric* setting, which is well-established and has been extensively used in many sub-field of cryptography.

1.4 Comparison and Discussion

We make a comparison among existing almost-tightly secure IBE schemes in MIMC setting in terms of time and space efficiency. The details are shown in Table 1. Our comparison involves the composite-order construction by Hofheinz *et al.* [HKS15], the prime-order construction in Section 5 based on the decisional linear (DLIN, 2-Lin) and symmetric external Diffie-Hellman (SXDH, 1-Lin) assumption, and the prime-order construction from Section 9 based on the XDLIN (2-LinAI) assumption. As a base line, we also consider the efficiency of prime-order construction by Chen and Wee [CW13] and Blazy *et al.* [BKP14], which is *not* built for MIMC setting.

Hofheinz *et al.*'s construction (see the third row) works with a symmetric bilinear group whose order is the product of four distinct primes, the sizes of group elements are much larger, and exponentiation operations and pairing operations are much more expensive. Therefore the overall efficiency is not acceptable even though the numbers of group elements in MSK, SK and CT are smaller and Enc and Dec require less exponentiation operations and pairing operations.

When instantiating our first instantiation (see the fourth row) under the DLIN assumption, each group element in $\mathbb{G}$ and $\mathbb{H}$ is a 6-dimension vector over G_1 and G_2, respectively. When instantiating under the SXDH assumption, each group element in $\mathbb{G}$ and $\mathbb{H}$ is a 3-dimension vector over G_1 and G_2, respectively. Compared with Blazy *et al.*'s construction [BKP14], both size of MPK, SK and CT and cost of Enc and Dec are (at least) doubled in our construction. On the other hand, in our second instantiation based on the XDLIN assumption

Scheme	$\|G\|$	Assum.	$\|\mathsf{MPK}\|$		$\|\mathsf{SK}\|$	$\|\mathsf{CT}\|$		T_{Enc}		T_{Dec}	MIMC
			G_1/G	G_T	G_2/G	G_1/G	G_T	E_1/E	E_T	P	
[CW13]	P	d-Lin	$2d^2(2n+1)$	d	$4d$	$4d$	1	$4d^2$	d	$4d$	✗
		DLIN	$16n+8$	2	8	8	1	16	2	8	
		SXDH	$4n+2$	1	4	4	1	4	1	4	
[BKP14]	P	d-Lin	$(2n+1)d^2+d$	d	$2d+1$	$2d+1$	1	$2d^2+1$	d	$2d+1$	✗
		DLIN	$8n+6$	2	5	5	1	9	2	5	
		SXDH	$2n+2$	1	3	3	1	3	1	3	
[HKS15]	C	Static	$2n+1$	1	2	2	1	2	1	2	✓
Sec. 5	P	d-Lin	$3d^2(2n+1)$	d	$6d$	$6d$	1	$6d^2$	d	$6d$	✓
		DLIN	$24n+12$	2	12	12	1	24	2	12	
		SXDH	$6n+3$	1	6	6	1	6	1	6	
Sec. 9	P	d-LinAI	$2d^2(2n+1)$	d	$4d$	$4d$	1	$4d^2$	d	$4d$	✓
		XDLIN	$16n+8$	2	8	8	1	16	2	8	

Table 1: Comparing Efficiency among existing and proposed almost-tight IBE schemes. n is the length of identities. Column $|\mathsf{MPK}|$, $|\mathsf{SK}|$, and $|\mathsf{CT}|$ show the size of master public keys, user's secret keys and ciphertexts, respectively. Each sub-column contains the number of elements in G, G_1, G_2, and G_T. Column T_{Enc} and T_{Dec} show encryption and decryption cost, respectively. Each sub-column E, E_1, and E_T shows the number of exponentiations on group G, G_1, and G_T, respectively, and sub-column P shows the number of pairings. Column "Assum." shows the underlying assumption. "Static" means static assumptions in the composite-order bilinear group. Column "$|G|$" indicates the group order, "P" for prime and "C" for composite order, respectively.

(see the last row), each group element in $\mathbb{G}$ and $\mathbb{H}$ is a vector of 4-dimension over G. Although the resulting IBE is still less efficient than Blazy *et al.*'s construction [BKP14] under the DLIN assumption, the stronger computational assumption (i.e., XDLIN) helps us to narrow the gap. We may view this as a tradeoff between strength of security and efficiency without changing the security model. We leave it as an open problem to find more efficient fully secure IBE with tight reduction in MIMC setting, especially from standard d-linear assumption.

1.5 Related Work

Dual System Groups and Its Variants. Chen and Wee proposed the notion of dual system group [CW14], which captures key algebraic structure supporting the dual system technique. They used this abstract primitive to obtain an HIBE scheme with constant-size ciphertext using prime-order bilinear groups. The nested dual system group, an variant of dual system groups, was proposed by Chen and Wee [CW13] to reach almost-tight adaptively secure IBE in the standard model. Chen, Gay, and Wee [CGW15] combined the dual system group with predicate encodings and obtained a more general framework leading to a lot of constructions in the prime-order setting. Very recent work by Gong *et al.* [GCTC15] extended the concept of dual system group to build an unbounded HIBE [LW11, Lew12] with shorter ciphertexts in the prime-order setting.

Identity Based Encryption. The notion of identity based encryptions was introduced by Shamir [Sha84] in 1984. The first practical realization was proposed by Boneh and Franklin [BF01] using bilinear groups and Cocks [Coc01] using quadratic residue. Both of them rely on the heuristic random oracle model. Since then several practical solutions in the standard model were proposed, including Boneh-Boyen's IBE [BB04b, BB04a], Waters' IBE [Wat05], and Gentry's IBE [Gen06]. In 2009, Waters [Wat09] proposed a new proof methodology, the dual system encryption, and presented an IBE scheme with short public key and proved its security under several simple assumptions in the standard model. Recently, Chen and Wee [CW13] achieved almost-tight IBE by utilizing the dual system technique in a novel way. Very recently, Blazy *et al.*[BKP14] built the connection between IBEs and affine message authentication code which is a symmetric primitive. IBE can also be realized using other algebra framework such as lattices [GPV08, ABB10a, ABB10b].

1.6 Independent Work

The independent work by Attrapadung, Hanaoka, and Yamada [AHY15] also involves several constructions of almost-tight IBE in MIMC setting. They developed an elegant framework for building almost-tight IBE in MIMC setting from the so-called *broadcast encoding*, which is a special form of Attrapadung's pairing encoding [Att14], and obtained a series of almost-tight IBE schemes with various properties (including sub-linear size master public key and anonymous version) using both composite-order and prime-order bilinear groups. Their results and ours partially overlap. Their scheme with constant-size ciphertext in prime-order group (i.e.,

Φ_{cc}^{prime}) is similar to our second construction based on the XDLIN assumption shown in Section 9. In fact, they share the same performance in terms of the size of ciphertexts and secret keys and running time of Enc and Dec. However we note that we also provide an generalization of this construction but proven secure under the non-standard d-LinAI assumption. Furthermore, our first construction in Section 5 is full adaptively secure under the d-Lin assumption, which is a more general and weaker assumption than the XDLIN used by both Attrapadung *et al.*'s and our second constructions.

1.7 Outline

The paper is organized as follows: Section 2 presents necessary background. Section 3 gives our revised definition of ENDSG. We realize our revised ENDSG in the prime-order setting in Section 4 and investigate how to update our ENDSG and its prime-order instantiation to achieve higher security level in Section 6. A derived concrete IBE is presented in Section 5. The next two sections are devoted to gain more efficient solutions. We fine-tune the notion of ENDSG in Section 7 and present a prime-order realization in Section 8. The resulting concrete IBE is shown in Section 9.

2 Preliminaries

2.1 Notations

For a finite set S, we use $s \leftarrow S$ to denote the process of picking s from S at random. For any $n \in \mathbb{Z}^+$, we take $[n]$ as the brief representation of set $\{1,\ldots,n\}$. For a probabilistic algorithm Alg, $y \leftarrow \mathsf{Alg}(x;r)$ means that we run the algorithm Alg on input x and randomness r, and then assign the result to variable y. We may omit r for brevity when it is clear from the context. $\mathsf{Alg}^{\lambda}(x)$ means we run Alg for λ times using independent random coins. Fixed an input x, we may view $\mathsf{Alg}(x;r)$ as a random variable and use $[\mathsf{Alg}(x;r)]$ to indicate its support, i.e., the set of all possible outputs of algorithm Alg on input x. "p.p.t." stands for "probabilistic polynomial time".

We use $\mathrm{ord}(G)$ to denote the order of group G. Let $\mathbf{e}_i$ denote the vector with 1 on the ith position and 0 elsewhere and $h^{\mathbf{e}_i}$ with $h \in G$ be a vector over G with h on the ith position and 1 elsewhere. For two vectors $\mathbf{g} := (g_1,\ldots,g_n) \in G^n$ and $\mathbf{g}' := (g'_1,\ldots,g'_n) \in G^n$, we define $\mathbf{g}\cdot\mathbf{g}' = (g_1\cdot g'_1,\ldots,g_n\cdot g'_n) \in G^n$ where "$\cdot$" on the right-hand side is the group operation of G. For any vector $\mathbf{x} = (x_1,\ldots,x_n)$ and $i \in [n]$, we define $\mathbf{x}_{-i}$ as a vector $(x_1,\ldots,x_{i-1},\perp,x_{i+1},\ldots,x_n)$ whose ith position is unknown (we take $\perp$ as a placeholder), and $\mathbf{x}|_i$ as its prefix of length i, i.e., $\mathbf{x}|_i := (x_1,\ldots,x_i)$.

2.2 Identity Based Encryptions

Algorithms. An IBE scheme in the multi-instance setting consists of five p.p.t. algorithms defined as follows[2]. (1) The *parameter generation algorithm* $\mathsf{Param}(1^k, \mathrm{SYS})$ takes as input a security parameter $k \in \mathbb{Z}^+$ in its unary form and a system-level parameter SYS, and outputs a global parameter GP. (2) The *setup algorithm* $\mathsf{Setup}(\mathrm{GP})$ takes as input a global parameter GP, and outputs a master public/secret key pair $(\mathrm{MPK},\mathrm{MSK})$. (3) The *key generation algorithm* $\mathsf{KeyGen}(\mathrm{MPK},\mathrm{MSK},\mathbf{y})$ takes as input a master public key MPK, a master secret key MSK and an identity $\mathbf{y}$, and outputs a secret key $\mathrm{SK}_{\mathbf{y}}$ for the identity. (4) The *encryption algorithm* $\mathsf{Enc}(\mathrm{MPK},\mathbf{x},\mathrm{M})$ takes as input a master public key MPK, an identity $\mathbf{x}$ and a message M, outputs a ciphertext $\mathrm{CT}_{\mathbf{x}}$ for the message under the identity. (5) The *decryption algorithm* $\mathsf{Dec}(\mathrm{MPK},\mathrm{SK},\mathrm{CT})$ takes as input a master public key MPK, a secret key SK and a ciphertext CT, outputs a message M or a failure symbol $\perp$.

The so-called "multi-instance setting" indicates that we are considering a collection of IBE instances established under the same global parameter GP. We leave the system-level parameter SYS undefined for generality. It may depend on concrete constructions or application scenarios.

Correctness. Roughly speaking, the correctness says that, for any IBE instance equipped with a legal master public/secret key pair, any secret key honestly generated using the master secret key for some identity should be able to recover the message from a ciphertext for the same identity under the master public key. Formally, for any parameter $k \in \mathbb{Z}^+$, any SYS, any $\mathrm{GP} \in [\mathsf{Param}(1^k,\mathrm{SYS})]$, any $(\mathrm{MPK},\mathrm{MSK}) \in [\mathsf{Setup}(\mathrm{GP})]$, any identity $\mathbf{x}$, and any message M, it holds that

$$\Pr\left[\mathsf{Dec}(\mathrm{MPK},\mathsf{KeyGen}(\mathrm{MPK},\mathrm{MSK},\mathbf{x}),\mathsf{Enc}(\mathrm{MPK},\mathbf{x},\mathrm{M})) = \mathrm{M}\right] \geqslant 1 - 2^{-\Omega(k)}.$$

[2]The definition shown here is slightly different from that in [HKS15]. We combine the (system-level) public parameter pp and secret parameter sp in [HKS15] as a global parameter GP. This global parameter is only fed to algorithm Setup to create fresh master public/secret key pairs. And all the other algorithms just take MPK, a local parameter, as input instead of pp, a global one. The adaptation is purely conceptual and made for clarity. The security model (given below) is tuned accordingly.

The probability space is defined by the random coins consumed by algorithm KeyGen and Enc.

Adaptive Security in the Multi-instance, Multi-ciphertext Setting. Roughly, the adaptive security in the multi-instance, multi-ciphertext setting extends the traditional adaptive security model for IBE [BF01] in the sense that the adversary can access to multiple IBE instances (obtaining master public key and users' keys) and attack multiple ciphertexts (i.e., challenge ciphertexts), which is formalized by Hofheinz *et al.* [HKS15]. Ideally, the adversary is free to choose the challenge instance, the challenge identity and the challenge message pair. Hofheinz *et al.* [HKS15] also identified a weaker variant in which only one challenge ciphertext is allowed for each challenge identity in each challenge instance, and called the ideal one *full security*.

We review the experiment $\mathbf{Exp}^{\mathrm{IBE}}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R)$ between a challenger $\mathscr{C}$ and an adversary $\mathscr{A}$ [HKS15], which captures both the weaker and full security notion.

Setup. $\mathscr{C}$ gets $\mathrm{GP} \leftarrow \mathsf{Param}(1^k,\mathrm{SYS})$ and creates $(\mathrm{MPK}_\iota,\mathrm{MSK}_\iota) \leftarrow \mathsf{Setup}(\mathrm{GP})$ for $\iota \in [\lambda]$. All master public keys $\{\mathrm{MPK}_\iota\}_{\iota\in[\lambda]}$ are sent to $\mathscr{A}$. $\mathscr{C}$ also chooses a secret random bit $\beta \in \{0,1\}$ and initializes Q_K and Q_C as empty sets.

Query. $\mathscr{A}$ is allowed to make two types of queries: key extraction queries and challenge queries. $\mathscr{C}$ answers every queries as follows:

- For each key extraction query $(\iota,\mathbf{y})$, $\mathscr{C}$ returns $\mathrm{SK} \leftarrow \mathsf{KeyGen}(\mathrm{MPK}_\iota,\mathrm{MSK}_\iota,\mathbf{y})$ and updates $Q_K := Q_K \cup \{(\iota,\mathbf{y})\}$;
- For each challenge query $(\iota^*,\mathbf{x}^*,\mathrm{M}_0^*,\mathrm{M}_1^*)$, $\mathscr{C}$ returns $\mathrm{CT}^* \leftarrow \mathsf{Enc}(\mathrm{MPK}_{\iota^*},\mathbf{x}^*,\mathrm{M}_\beta^*)$ and updates $Q_C := Q_C \cup \{(\iota^*,\mathbf{x}^*)\}$.

Guess. $\mathscr{A}$ outputs its guess $\beta' \in \{0,1\}$.

We say an adversary $\mathscr{A}$ wins experiment $\mathbf{Exp}^{\mathrm{IBE}}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R)$, denoted by $\mathbf{Exp}^{\mathrm{IBE}}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R)=1$, if and only if (1) $\beta=\beta'$, (2) $Q_K \cap Q_C = \emptyset$, (3) $\mathscr{A}$ made at most q_K key extraction queries, (4) there are at most q_C challenge identities, and (5) for each of them, there exist at most q_R challenge ciphertexts. We define the advantage of $\mathscr{A}$ as

$$\mathsf{Adv}^{\mathrm{IBE}}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R) = \left|\Pr[\mathbf{Exp}^{\mathrm{IBE}}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R)=1]-1/2\right|.$$

The probability space is defined by random coins consumed by both $\mathscr{C}$ and $\mathscr{A}$. An IBE is (λ,q_K,q_C,q_R)-*adaptively-secure* if, for any p.p.t. adversary $\mathscr{A}$ the advantage $\mathsf{Adv}^{\mathrm{IBE}}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_K)$ is bounded by $2^{-\Omega(k)}$. Clearly, the (λ,q_k,q_C,q_R)-adaptive security with unbounded q_R is consistent with the full security, while the $(\lambda,q_k,q_C,1)$-adaptive security is exactly the weak security. Furthermore, we define *B-weak adaptive security*, an intermediate security notion between them, as (λ,q_K,q_C,B)-adaptive security for a priori bound $B \geqslant 1$.

3 Revisiting Extended Nested Dual System Groups

This section revises the ENDSG proposed by Hofheinz *et al.* [HKS15]. Our main goal is to reduce the dependence on some special algebraic structure which hinders the development of more instantiations, especially those using prime-order bilinear groups. (See Section 1.) We show our revised ENDSG followed by a series of remarks clarifying motivations and reasons behind several technical decisions. As discussed in Section 1, key points are: (1) removing special group requirements; (2) explicitly providing samples in each computational assumption; (3) generalizing subgroup of $\widetilde{h}^*$ and $\widetilde{h}^*$.

Syntax. Our revised ENDSG consists of eight p.p.t. algorithms defined as follows:

- $\mathsf{SampP}(1^k,n)$: Output: (1) PP containing (a) group description $(\mathbb{G},\mathbb{H},\mathbb{G}_T)$ and an admissible bilinear map $e:\mathbb{G}\times\mathbb{H}\to\mathbb{G}_T$; (b) an efficient linear map μ defined on $\mathbb{H}$; (c) an efficient sampler for $\mathbb{H}$ and $\mathbb{Z}_{\mathrm{ord}(\mathbb{H})}$, respectively; (d) public parameters for SampG and SampH. (2) SP containing secret parameters for $\widehat{\mathsf{SampG}}$, $\widetilde{\mathsf{SampG}}$, $\widetilde{\mathsf{SampH}}^*$ and $\widehat{\mathsf{SampH}}^*$.
- SampGT: $\mathrm{Im}(\mu)\to\mathbb{G}_T$.
- $\mathsf{SampG}(\mathrm{PP})$: Output $\mathbf{g}=(g_0,g_1,\ldots,g_n)\in\mathbb{G}^{n+1}$.
- $\mathsf{SampH}(\mathrm{PP})$: Output $\mathbf{h}=(h_0,h_1,\ldots,h_n)\in\mathbb{H}^{n+1}$.
- $\widehat{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$: Output $\widehat{\mathbf{g}}=(\widehat{g}_0,\widehat{g}_1,\ldots,\widehat{g}_n)\in\mathbb{G}^{n+1}$.

– $\widetilde{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$: Output $\widetilde{\mathbf{g}} = (\widetilde{g}_0, \widetilde{g}_1, \ldots, \widetilde{g}_n) \in \mathbb{G}^{n+1}$.

– $\widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$: Output $\widehat{h}^* \in \mathbb{H}$.

– $\widetilde{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$: Output $\widetilde{h}^* \in \mathbb{H}$.

The first four algorithms are used in the real system, while the remaining ones are defined for the proof. The notation SampG_0 refers to the first element in the output of SampG, i.e., g_0. The notational convention also applies to SampH, $\widehat{\mathsf{SampG}}$, and $\widetilde{\mathsf{SampG}}$.

Correctness. For all $k, n \in \mathbb{Z}^+$ and all $(\mathrm{PP}, \mathrm{SP}) \in [\mathsf{SampP}(1^k, n)]$, it is required that:

(Projective.) For all $h \in \mathbb{H}$ and all possible randomness s, $\mathsf{SampGT}(\mu(h); s) = e(\mathsf{SampG}_0(\mathrm{PP}; s), h)$.

(Associative.) For all $(g_0, g_1, \ldots, g_n) \in [\mathsf{SampG}(\mathrm{PP})]$ and all $(h_0, h_1, \ldots, h_n) \in [\mathsf{SampH}(\mathrm{PP})]$, $e(g_0, h_i) = e(g_i, h_0)$ for $i \in [n]$.

Security. For all $k, n \in \mathbb{Z}^+$ and all $(\mathrm{PP}, \mathrm{SP}) \in [\mathsf{SampP}(1^k, n)]$, it is required that:

(Orthogonality.) For all $\widehat{h}^* \in [\widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})]$ and all $\widetilde{h}^* \in [\widetilde{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})]$,

1. $\mu(\widehat{h}^*) = \mu(\widetilde{h}^*) = 1$;
2. $e(\widehat{g}_0, \widetilde{h}^*) = 1$ for all $\widehat{g}_0 \in [\widehat{\mathsf{SampG}}_0(\mathrm{PP}, \mathrm{SP})]$;
3. $e(\widetilde{g}_0, \widehat{h}^*) = 1$ for all $\widetilde{g}_0 \in [\widetilde{\mathsf{SampG}}_0(\mathrm{PP}, \mathrm{SP})]$;

The first requirement implies that $e(g_0, \widetilde{h}^*) = e(g_0, \widehat{h}^*) = 1$ for all $g_0 \in [\mathsf{SampG}_0(\mathrm{PP})]$ by the *projective* property (c.f. Section 3.2 in [CW13]).

(Non-degeneracy.) Over the probability space defined by $\widehat{g}_0 \leftarrow \widehat{\mathsf{SampG}}_0(\mathrm{PP}, \mathrm{SP})$, with overwhelming probability $1 - 2^{-\Omega(k)}$, $e(\widehat{g}_0, \widehat{h}^*)$ is distributed uniformly over $\mathbb{G}_T$ when sampling $\widehat{h}^* \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$.

($\mathbb{H}$-subgroup.) The output of $\mathsf{SampH}(\mathrm{PP})$ is distributed uniformly over some subgroup of $\mathbb{H}^{n+1}$, while those of $\widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$ and $\widetilde{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$ are distributed uniformly over some subgroup of $\mathbb{H}$.

(Left subgroup indistinguishability 1 (LS1).) For any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,

$$\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k, q) := \left|\Pr[\mathscr{A}(D, T_0) = 1] - \Pr[\mathscr{A}(D, T_1) = 1]\right|,$$

where

$$D := (\mathrm{PP}),\ T_0 := \left\{\mathbf{g}_j\right\}_{j\in[q]},\ T_1 := \left\{\mathbf{g}_j \cdot \boxed{\widehat{\mathbf{g}}_j}\right\}_{j\in[q]}$$

and $\mathbf{g}_j \leftarrow \mathsf{SampG}(\mathrm{PP})$ and $\widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$.

(Left subgroup indistinguishability 2 (LS2).) For any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,

$$\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k, q, q') := \left|\Pr[\mathscr{A}(D, T_0) = 1] - \Pr[\mathscr{A}(D, T_1) = 1]\right|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}^*_j \cdot \widetilde{h}^*_j\right\}_{j\in[q+q']}, \left\{\mathbf{g}'_j \cdot \widehat{\mathbf{g}}'_j\right\}_{j\in[q]}\right),\ T_0 := \left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j\right\}_{j\in[q]},\ T_1 := \left\{\mathbf{g}_j \cdot \boxed{\widetilde{\mathbf{g}}_j}\right\}_{j\in[q]}$$

and $\widehat{h}^*_j \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$, $\widetilde{h}^*_j \leftarrow \widetilde{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$, $\mathbf{g}'_j \leftarrow \mathsf{SampG}(\mathrm{PP})$, $\widehat{\mathbf{g}}'_j \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$, $\mathbf{g}_j \leftarrow \mathsf{SampG}(\mathrm{PP})$, $\widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$, and $\widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$.

(Nested-hiding indistinguishability (NH).) For all $\eta \in [\lfloor n/2 \rfloor]$ and any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,

$$\mathsf{Adv}^{\mathrm{NH}(\eta)}_{\mathscr{A}}(k, q, q') := \left|\Pr[\mathscr{A}(D, T_0) = 1] - \Pr[\mathscr{A}(D, T_1) = 1]\right|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}_j^*\right\}_{j\in[q+q']}, \left\{\widetilde{h}_j^*\right\}_{j\in[q+q']}, \left\{(\widehat{\mathbf{g}}_j)_{-(2\eta-1)}\right\}_{j\in[q]}, \left\{(\widetilde{\mathbf{g}}_j)_{-2\eta}\right\}_{j\in[q]}\right),$$

$$T_0 := \left\{\mathbf{h}_j\right\}_{j\in[q']},\ T_1 := \left\{\mathbf{h}_j \cdot \boxed{(\widehat{h}_j^{**})^{\mathbf{e}_{2\eta-1}} \cdot (\widetilde{h}_j^{**})^{\mathbf{e}_{2\eta}}}\right\}_{j\in[q']}$$

and $\widehat{h}_j^* \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\widetilde{h}_j^* \leftarrow \widetilde{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, $\widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, $\mathbf{h}_j \leftarrow \mathsf{SampH}(\mathrm{PP})$, $\widehat{h}_j^{**} \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\widetilde{h}_j^{**} \leftarrow \widetilde{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$. We further define

$$\mathsf{Adv}^{\mathrm{NH}}_{\mathscr{A}}(k,q,q') := \max_{\eta\in[\lfloor n/2\rfloor]} \left\{\mathsf{Adv}^{\mathrm{NH}(\eta)}_{\mathscr{A}}(k,q,q')\right\}.$$

Remark 1 (notations) *The ENDSG is mainly defined for building IBE. We remark that, in the description of LS1, LS2, and NH, the parameter q and q' roughly corresponds to the maximum number of challenge ciphertexts and key extraction queries, respectively.*

Remark 2 (sampling $\widehat{h}^*$ and $\widetilde{h}^*$, and $\mathbb{H}$-subgroup) *We model the process of sampling over subgroup generated by $\widehat{h}^*$ and $\widetilde{h}^*$ (in [HKS15]) as algorithm $\widehat{\mathsf{SampH}}^*$ and $\widetilde{\mathsf{SampH}}^*$, respectively. This allows us to employ more complex algebraic structure (say, extending the subspaces from one dimension to higher one), which is crucial for our prime-order instantiation in Section 4. Due to its generality, the $\mathbb{H}$-subgroup property must be extended to take both $\widehat{\mathsf{SampH}}^*$ and $\widetilde{\mathsf{SampH}}^*$ into account.*

Remark 3 ($\mathbb{G}$-subgroup and $\mathbb{H}$-subgroup) *Since we provide adequate samples of $\mathbb{G}^{n+1}$ directly in the last three computational security requirements and further re-randomization is not necessary in the proof, the $\mathbb{G}$-subgroup in the original definition could be safely removed. However this won't let the revised ENDSG free from $\mathbb{H}$-subgroup property. The simulator still need the property to re-randomize T_0 or T_1 in NH(η) using $\mathsf{SampH}(\mathrm{PP})$ to maintain the consistency of truly random functions on two identities sharing the same η-bit prefix.*

On one hand, our revised definition for ENDSG is essentially consistent with Hofheinz *et al.*'s definition [HKS15]. In particular, it is not hard to see that one may use Hofheinz *et al.*'s ENDSG [HKS15] to realize this revised version. Therefore their instantiation using composite-order bilinear groups can also be taken as an instantiation of the revised version above. On the other hand, although our revised definition is more general, it still almost-tightly implies an IBE in MIMC setting. In fact, the construction, the security result and its proof are nearly the same as those presented in [HKS15]. One may consider them as rewriting Hofheinz *et al.*'s results [HKS15] in the language of our revised ENDSG. We present the construction and sketch of the proof in Appendix B for completeness. It is worth noting that the construction only achieves weak adaptive security. We will show how to enhance *non-degeneracy* to reach full adaptive security in Section 6.

4 Instantiating ENDSG from d-Linear Assumption

We give an instantiation of our revised ENDSG (defined in Section 3) using prime-order bilinear groups based on the technique of Chen and Wee [CW13]. Following the generic construction proposed in [HKS15] (c.f. Appendix B), this yields the first fully and almost-tight secure IBE in MIMC setting using prime-order bilinear groups. (See Section 5.)

4.1 Prime-order Bilinear Groups and Computational Assumptions

A prime-order (asymmetric) bilinear group generator $\mathsf{GrpGen}(1^k)$ takes security parameter 1^k as input and outputs $\mathscr{G} := (p, G_1, G_2, G_T, e)$, where G_1, G_2 and G_T are finite cyclic groups of prime order p, and $e : G_1 \times G_2 \to G_T$ is a non-degenerated and efficiently computable bilinear map. We let g_1, g_2 and $g_T := e(g_1, g_2)$ be a generator of G_1, G_2 and G_T, respectively. We state the (standard) d-linear assumption (d-Lin for short) in G_1 (see Assumption 1), the analogous assumption in G_2 can be defined by exchanging the role of G_1 and G_2.

Assumption 1 (d-Linear Assumption in G_1) *For any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,*

$$\mathsf{Adv}^{d\text{-Lin}}_{\mathscr{A}}(k) := \left|\Pr[\mathscr{A}(D, T_0) = 1] - \Pr[\mathscr{A}(D, T_1) = 1]\right|,$$

where

$$D := \left(\mathscr{G}, g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, g_1^{a_{d+1}}, g_1^{a_1 s_1}, \ldots, g_1^{a_d s_d}\right),$$

$$T_0 := g_1^{a_{d+1}(s_1+\cdots+s_d)},\ T_1 := g_1^{a_{d+1}(s_1+\cdots+s_d)+\boxed{s_{d+1}}}$$

and $\mathcal{G} := (p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$, $s_1, \ldots, s_d \leftarrow \mathbb{Z}_p$, *and* $a_1, \ldots, a_d, a_{d+1}, s_{d+1} \leftarrow \mathbb{Z}_p^*$.

"Matrix-in-the-exponent" Notation. For an $m \times n$ matrix $\mathbf{X} = (x_{i,j})$ over $\mathbb{Z}_p$ and a group element g of G (which may be G_1, G_2 or G_T), we define $g^{\mathbf{X}} := (g^{x_{i,j}})$ which is an $m \times n$ matrix over G. We naturally extend the domain of pairing e: given two matrices $\mathbf{A}$ and $\mathbf{B}$ over $\mathbb{Z}_p$ whose multiplication is well-defined, we define $e(g_1^{\mathbf{A}}, g_2^{\mathbf{B}}) := e(g_1, g_2)^{\mathbf{A}^\top \mathbf{B}}$, which is a matrix (of proper size) over G_T. As a special case, for vectors $\mathbf{x}$ and $\mathbf{y}$ over $\mathbb{Z}_p$ of the same length, we have $e(g_1^{\mathbf{x}}, g_2^{\mathbf{y}}) := e(g_1, g_2)^{\mathbf{x}^\top \mathbf{y}} \in G_T$, the standard inner product $\langle \mathbf{x}, \mathbf{y} \rangle$ in the exponent. We will use $\mathbf{0}$ to denote both vectors and matrices with only zero entries when it's size is clear from the context; if necessary, we may give out its dimension or size in the subscript.

An extended version of d-Lifted Linear Assumption. We describe an extension of the d-Lifted Linear (d-LLin) assumption [JR14] (c.f. Appendix A.1) for improving the readability of our proofs, which is called (d, ℓ, q)-Lifted Linear Assumption, and (d, ℓ, q)-LLin for short. As usual, we just show the assumption in G_1 and the counterpart in G_2 is readily derived. The extension is made in two steps: we first consider ℓ correlated challenges, and then consider q independent copies of it, i.e., its q-fold [EHK+13].

Assumption 2 ((d, ℓ, q)-Lifted Linear Assumption in G_1) *For any p.p.t. adversary $\mathcal{A}$, the following advantage function is negligible in k,*

$$\mathsf{Adv}_{\mathcal{A}}^{(d,\ell,q)\text{-LLin}}(k) := \left|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\right|,$$

where

$$D := \left(\mathcal{G}, g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, \left\{g_1^{b_{i,j}}\right\}_{i\in[\ell], j\in[d]}, \left\{g_1^{a_1 s_{1,j}}, \ldots, g_1^{a_d s_{d,j}}\right\}_{j\in[q]}\right),$$

$$T_0 := \left\{g_1^{b_{i,1}s_{1,j}+\cdots+b_{i,d}s_{d,j}}\right\}_{i\in[\ell], j\in[q]},\ T_1 := \left\{g_1^{b_{i,1}s_{1,j}+\cdots+b_{i,d}s_{d,j}+\boxed{s_{d+i,j}}}\right\}_{i\in[\ell], j\in[q]}$$

and $\mathcal{G} := (p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$, $a_1, \ldots, a_d, b_{i,j} \leftarrow \mathbb{Z}_p^*$, $s_{1,j}, \ldots, s_{d,j} \leftarrow \mathbb{Z}_p$, $s_{d+i,j} \leftarrow \mathbb{Z}_p^*$.

We show that the (d, ℓ, q)-LLin assumption is tightly implied by the standard d-Lin assumption (see Lemma 1). We remark that, since ℓ corresponds to a relatively small parameter, say 2, in our construction and q corresponds to the number of adversary's queries which may be 2^{30}, we will prove Lemma 1 under the assumption that $\ell < q$ for simplicity.

Lemma 1 (d-Lin $\Rightarrow$ (d, ℓ, q)-LLin) *For any p.p.t. adversary $\mathcal{A}$, there exists an adversary $\mathcal{B}$ such that*

$$\mathsf{Adv}_{\mathcal{A}}^{(d,\ell,q)\text{-LLin}}(k) \leqslant \ell \cdot \mathsf{Adv}_{\mathcal{B}}^{d\text{-Lin}}(k) + 1/(p-1),$$

and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + \ell^2 d \cdot \mathsf{poly}(k)$ *where* $\mathsf{poly}(k)$ *is independent of* $\mathsf{Time}(\mathcal{A})$.

Proof. We may prove the lemma in two steps. Following [JR14] and [CW13], one can prove that, for any p.p.t. adversary $\mathcal{A}$, there exists an adversary $\mathcal{B}$ with $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + \ell d \cdot \mathsf{poly}(k)$ such that

$$\mathsf{Adv}_{\mathcal{A}}^{(d,\ell,1)\text{-LLin}}(k) \leqslant \mathsf{Adv}_{\mathcal{B}}^{d\text{-Lin}}(k).$$

Applying Lemma 1 in [EHK+13], we have that, for any p.p.t. adversary $\mathcal{A}$, there exists an adversary $\mathcal{B}$ with $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + \ell^2 d \cdot \mathsf{poly}(k)$ such that

$$\mathsf{Adv}_{\mathcal{A}}^{(d,\ell,q)\text{-LLin}}(k) \leqslant \ell \cdot \mathsf{Adv}_{\mathcal{B}}^{(d,\ell,1)\text{-LLin}}(k) + 1/(p-1).$$

Putting them together, one may deduce the lemma immediately.

We now give the proof of the first claim. For simplicity, we discard the subscript j related to parameter q when considering $(d, \ell, 1)$-LLin. Given a d-Lin problem instance

$$\left(g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, g_1^{a_{d+1}}, g_1^{a_1 s_1}, \ldots, g_1^{a_d s_d}, g_1^{a_{d+1}(s_1+\cdots+s_d)+s_{d+1}}\right)$$

as input where s_{d+1} are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathcal{B}$ works as follows:

Simulating $a_1, \ldots, a_d$ and $s_1, \ldots, s_d$. We implicitly define $a_i := a_i$ and $s_i := s_i$ for all $i \in [d]$. The a_i and s_i on the left-hand side refers to those in the $(d, \ell, 1)$-LLin problem instance we want to simulate, while those on the right-hand side come from the d-Lin problem instance.

Simulating $g_1^{b_{i,j}}$ for $i \in [\ell],\ j \in [d]$. Sample $\gamma_i, \delta_{i,j} \leftarrow \mathbb{Z}_p^*$ for all $i \in [\ell]$ and $j \in [d]$ and implicitly set $b_{i,j} := \gamma_i a_{d+1} + \delta_{i,j} a_j$. Therefore we can simulate

$$g_1^{b_{i,j}} := \left(g_1^{a_{d+1}}\right)^{\gamma_i} \cdot \left(g_1^{a_j}\right)^{\delta_{i,j}}.$$

Simulating the Challenge. For any $i \in [\ell]$, we observe that

$$\begin{aligned} b_{i,1}s_1 + \cdots + b_{i,d}s_d &= (\gamma_i a_{d+1} + \delta_{i,1}a_1)s_1 + \cdots + (\gamma_i a_{d+1} + \delta_{i,d}a_d)s_d \\ &= \gamma_i a_{d+1}(s_1 + \cdots + s_d) + (\delta_{i,1}a_1 s_1 + \cdots + \delta_{i,d}a_d s_d). \end{aligned}$$

We can simulate the challenge as

$$g_1^{b_{i,1}s_1 + \cdots + b_{i,d}s_d + s_{d+i}} := \left(g_1^{a_{d+1}(s_1+\cdots+s_d)+s_{d+1}}\right)^{\gamma_i} \cdot \left(g_1^{a_1 s_1}\right)^{\delta_{i,1}} \cdots \left(g_1^{a_d s_d}\right)^{\delta_{i,d}}.$$

Analysis. Thanks to the entropy of $\delta_{i,j}$, the simulation of $b_{i,j}$ is perfect. If $s_{d+1} = 0$, we can see that $s_{d+i} = 0$ for all $i \in [\ell]$; if $s_{d+1} \leftarrow \mathbb{Z}_p^*$, we implicitly set $s_{d+i} := s_{d+1}\gamma_i$. (The a_{d+1} on the left-hand side and the right-hand side is for the $(d, \ell, 1)$-LLin and d-Lin assumption, respectively.) Observe that, since all γ_i are hidden by $\delta_{i,j}$, all s_{d+i} are independently and uniformly distributed over $\mathbb{Z}_p$. Therefore we proved our first claim. □

4.2 Construction

Our construction is based on the prime-order instantiation of NDSG by Chen and Wee [CW13] and works with $3d \times 3d$ matrices under the d-Lin assumption. (For more motivation, see Section 1). We let $\pi_{\mathrm{L}}(\cdot)$, $\pi_{\mathrm{M}}(\cdot)$, and $\pi_{\mathrm{R}}(\cdot)$ be functions mapping from a $3d \times 3d$ matrix to its left-most d columns, the middle d columns, and the right-most d columns, respectively. Algorithms of our revised ENDSG are shown as follows.

- $\mathsf{SampP}(1^k, n)$: Generate $(p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$ and define $(\mathbb{G}, \mathbb{H}, \mathbb{G}_T, e) := (G_1^{3d}, G_2^{3d}, G_T, e)$. Sample $\mathbf{B}, \mathbf{R} \leftarrow \mathrm{GL}_{3d}(\mathbb{Z}_p)$ and $\mathbf{A}_1, \ldots, \mathbf{A}_n \leftarrow \mathbb{Z}_p^{3d \times 3d}$. Set $\mathbf{B}^* := (\mathbf{B}^{-1})^\top$. Define

$$\begin{array}{llll} \mathbf{D} := \pi_{\mathrm{L}}(\mathbf{B}), & \mathbf{D}_i = \pi_{\mathrm{L}}(\mathbf{B}\mathbf{A}_i); & \mathbf{E} := \pi_{\mathrm{M}}(\mathbf{B}), & \mathbf{E}_i = \pi_{\mathrm{M}}(\mathbf{B}\mathbf{A}_i); \\ \mathbf{D}^* := \mathbf{B}^*\mathbf{R}, & \mathbf{D}_i^* = \mathbf{B}^*\mathbf{A}_i^\top\mathbf{R}; & \mathbf{F} := \pi_{\mathrm{R}}(\mathbf{B}), & \mathbf{F}_i = \pi_{\mathrm{R}}(\mathbf{B}\mathbf{A}_i); \end{array}$$

  where $i \in [n]$. Define $\mu(g_2^{\mathbf{k}}) := e(g_1^{\mathbf{D}}, g_2^{\mathbf{k}}) = e(g_1, g_2)^{\mathbf{D}^\top \mathbf{k}}$ for all $\mathbf{k} \in \mathbb{Z}_p^{3d}$. Output

$$\mathsf{PP} := \begin{pmatrix} g_1^{\mathbf{D}}, & g_1^{\mathbf{D}_1}, & \cdots, & g_1^{\mathbf{D}_n} \\ g_2^{\mathbf{D}^*}, & g_2^{\mathbf{D}_1^*}, & \cdots, & g_2^{\mathbf{D}_n^*} \end{pmatrix} \text{ and } \mathsf{SP} := \begin{pmatrix} g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)}, g_1^{\mathbf{E}}, & g_1^{\mathbf{E}_1}, & \cdots, & g_1^{\mathbf{E}_n} \\ g_2^{\pi_{\mathrm{R}}(\mathbf{B}^*)}, g_1^{\mathbf{F}}, & g_1^{\mathbf{F}_1}, & \cdots, & g_1^{\mathbf{F}_n} \end{pmatrix}.$$

  We assume PP always contains $\mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \mu$.

- $\mathsf{SampGT}(g_T^{\mathbf{p}})$: Sample $\mathbf{s} \leftarrow \mathbb{Z}_p^d$ and output $g_T^{\mathbf{s}^\top \mathbf{p}} \in G_T$.

- $\mathsf{SampG}(\mathsf{PP})$: Sample $\mathbf{s} \leftarrow \mathbb{Z}_p^d$ and output $\left(g_1^{\mathbf{D}\mathbf{s}}, g_1^{\mathbf{D}_1\mathbf{s}}, \ldots, g_1^{\mathbf{D}_n\mathbf{s}}\right) \in (G_1^{3d})^{n+1}$.

- $\mathsf{SampH}(\mathsf{PP})$: Sample $\mathbf{r} \leftarrow \mathbb{Z}_p^{3d}$ and output $\left(g_2^{\mathbf{D}^*\mathbf{r}}, g_2^{\mathbf{D}_1^*\mathbf{r}}, \ldots, g_2^{\mathbf{D}_n^*\mathbf{r}}\right) \in (G_2^{3d})^{n+1}$.

- $\widehat{\mathsf{SampG}}(\mathsf{PP}, \mathsf{SP})$: Sample $\widehat{\mathbf{s}} \leftarrow \mathbb{Z}_p^d$ and output $\left(g_1^{\mathbf{F}\widehat{\mathbf{s}}}, g_1^{\mathbf{E}_1\widehat{\mathbf{s}}}, \ldots, g_1^{\mathbf{E}_n\widehat{\mathbf{s}}}\right) \in (G_1^{3d})^{n+1}$.

- $\widetilde{\mathsf{SampG}}(\mathsf{PP}, \mathsf{SP})$: Sample $\widetilde{\mathbf{s}} \leftarrow \mathbb{Z}_p^d$ and output $\left(g_1^{\mathbf{F}\widetilde{\mathbf{s}}}, g_1^{\mathbf{F}_1\widetilde{\mathbf{s}}}, \ldots, g_1^{\mathbf{F}_n\widetilde{\mathbf{s}}}\right) \in (G_1^{3d})^{n+1}$.

- $\widehat{\mathsf{SampH}}^*(\mathsf{PP}, \mathsf{SP})$: Sample $\widehat{\mathbf{r}} \leftarrow \mathbb{Z}_p^d$ and output $g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}} \in G_2^{3d}$.

- $\widetilde{\mathsf{SampH}}^*(\mathsf{PP}, \mathsf{SP})$: Sample $\widetilde{\mathbf{r}} \leftarrow \mathbb{Z}_p^d$ and output $g_2^{\pi_{\mathrm{R}}(\mathbf{B}^*)\widetilde{\mathbf{r}}} \in G_2^{3d}$.

Correctness. We may check all correctness requirements as follows:

(Projective.) For all $\mathbf{k} \in \mathbb{Z}_p^{3d}$ and all $\mathbf{s} \in \mathbb{Z}_p^d$, we have that

$$\mathsf{SampGT}(\mu(g_2^{\mathbf{k}}); \mathbf{s}) = e(g_1, g_2)^{\mathbf{s}^\top(\mathbf{D}^\top \mathbf{k})} = e(g_1^{\mathbf{Ds}}, g_2^{\mathbf{k}}) = e(\mathsf{SampG}_0(\mathrm{PP}; \mathbf{s}), g_2^{\mathbf{k}}).$$

The second equality follows the fact that $\mathbf{s}^\top(\mathbf{D}^\top \mathbf{k}) = (\mathbf{Ds})^\top \mathbf{k}$.

(Associative.) For all $\mathbf{s} \in \mathbb{Z}_p^d$ and all $\mathbf{r} \in \mathbb{Z}_p^{3d}$, we have, for all $i \in [n]$,

$$e(g_1^{\mathbf{Ds}}, g_2^{\mathbf{D}_i^* \mathbf{r}}) = e(g_1, g_2)^{\bar{\mathbf{s}}^\top \mathbf{B}^\top (\mathbf{B}^* \mathbf{A}_i^\top \mathbf{R}) \mathbf{r}} = e(g_1, g_2)^{\bar{\mathbf{s}}^\top (\mathbf{BA}_i)^\top (\mathbf{B}^* \mathbf{R}) \mathbf{r}} = e(g_1^{\mathbf{D}_i \mathbf{s}}, g_2^{\mathbf{D}^* \mathbf{r}}),$$

where $\bar{\mathbf{s}} := \left(\begin{smallmatrix} \mathbf{s} \\ \mathbf{0} \end{smallmatrix}\right) \in \mathbb{Z}_p^{3d}$. The first and the last equality follow the definition of $\pi_{\mathrm{L}}(\cdot)$ while the second equality uses the fact that $\mathbf{B}^\top(\mathbf{B}^* \mathbf{A}_i^\top \mathbf{R}) = (\mathbf{B}^\top \mathbf{B}^*) \mathbf{A}_i^\top \mathbf{R} = \mathbf{A}_i^\top \mathbf{R} = \mathbf{A}_i^\top (\mathbf{B}^\top \mathbf{B}^*) \mathbf{R} = (\mathbf{BA}_i)^\top (\mathbf{B}^* \mathbf{R})$.

Security. We may check the following security requirements:

(Orthogonality.) For all $\widehat{\mathbf{r}}, \widetilde{\mathbf{r}}, \widehat{\mathbf{s}}, \widetilde{\mathbf{s}} \in \mathbb{Z}_p^d$, we check that

1. $\mu(g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}}) = e(g_1, g_2)^{\pi_{\mathrm{L}}(\mathbf{B})^\top \pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}} = e(g_1, g_2)^{\mathbf{0}_{d\times d}\widehat{\mathbf{r}}} = (1, \ldots, 1)^\top \in G_T^d$;
2. $\mu(g_2^{\pi_{\mathrm{R}}(\mathbf{B}^*)\widetilde{\mathbf{r}}}) = e(g_1, g_2)^{\pi_{\mathrm{L}}(\mathbf{B})^\top \pi_{\mathrm{R}}(\mathbf{B}^*)\widetilde{\mathbf{r}}} = e(g_1, g_2)^{\mathbf{0}_{d\times d}\widetilde{\mathbf{r}}} = (1, \ldots, 1)^\top \in G_T^d$;
3. $e(g_1^{\pi_{\mathrm{M}}(\mathbf{B})\widehat{\mathbf{s}}}, g_2^{\pi_{\mathrm{R}}(\mathbf{B}^*)\widetilde{\mathbf{r}}}) = e(g_1, g_2)^{\widehat{\mathbf{s}}^\top \pi_{\mathrm{M}}(\mathbf{B})^\top \pi_{\mathrm{R}}(\mathbf{B}^*)\widetilde{\mathbf{r}}} = e(g_1, g_2)^{\widehat{\mathbf{s}}^\top \mathbf{0}_{d\times d}\widetilde{\mathbf{r}}} = 1_{G_T}$;
4. $e(g_1^{\pi_{\mathrm{R}}(\mathbf{B})\widetilde{\mathbf{s}}}, g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}}) = e(g_1, g_2)^{\widetilde{\mathbf{s}}^\top \pi_{\mathrm{R}}(\mathbf{B})^\top \pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}} = e(g_1, g_2)^{\widetilde{\mathbf{s}}^\top \mathbf{0}_{d\times d}\widehat{\mathbf{r}}} = 1_{G_T}$.

The second equality of them follows the fact that $\pi_{\mathrm{L}}(\mathbf{B})^\top \pi_{\mathrm{M}}(\mathbf{B}^*) = \pi_{\mathrm{L}}(\mathbf{B})^\top \pi_{\mathrm{R}}(\mathbf{B}^*) = \pi_{\mathrm{M}}(\mathbf{B})^\top \pi_{\mathrm{R}}(\mathbf{B}^*) = \pi_{\mathrm{R}}(\mathbf{B})^\top \pi_{\mathrm{M}}(\mathbf{B}^*) = \mathbf{0}_{d\times d}$.

(Non-degeneracy.) For all $\widehat{\mathbf{s}} \in \mathbb{Z}_p^d$ and $\widehat{\mathbf{r}} \in \mathbb{Z}_p^d$, we have that

$$e(g_1^{\mathbf{E}\widehat{\mathbf{s}}}, g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}}) = e(g_1, g_2)^{\widehat{\mathbf{s}}^\top \pi_{\mathrm{M}}(\mathbf{B})^\top \pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}} = e(g_1, g_2)^{\widehat{\mathbf{s}}^\top \widehat{\mathbf{r}}}.$$

With probability $1 - 1/p^d$, sampling $\widehat{\mathbf{s}} \leftarrow \mathbb{Z}_p^d$ results in $\widehat{\mathbf{s}} \neq \mathbf{0}$, in which case the inner product $\widehat{\mathbf{s}}^\top \widehat{\mathbf{r}}$ is distributed uniformly over $\mathbb{Z}_p$ and therefore $e(g_1^{\mathbf{E}\widehat{\mathbf{s}}}, g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}})$ is distributed over G_T when picking $\widehat{\mathbf{r}} \leftarrow \mathbb{Z}_p^d$.

($\mathbb{H}$-subgroup.) This follows from the fact that $\mathbb{Z}_p^{3d}$ (for algorithm SampH) and $\mathbb{Z}_p^d$ (for algorithm $\widehat{\mathsf{SampH}}^*$ and $\widetilde{\mathsf{SampH}}^*$) are additive groups.

We check the remaining security properties (LS1, LS2, and NH) in the following subsections.

4.3 Left Subgroup Indistinguishability 1

We may rewrite the LS1 advantage function $\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k, q)$ as follows:

$$\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k, q) := \left| \Pr[\mathscr{A}(D, T_0) = 1] - \Pr[\mathscr{A}(D, T_1) = 1] \right|,$$

where

$$D := (\mathrm{PP}), \quad T_0 := \left\{ \mathbf{g}_j \right\}_{j \in [q]}, \quad T_1 := \left\{ \mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \right\}_{j \in [q]},$$

and

$$\begin{aligned}
\mathrm{PP} &:= \begin{pmatrix} g_1^{\pi_{\mathrm{L}}(\mathbf{B})}, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_1)}, & \ldots, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_n)} \\ g_2^{\mathbf{B}^*\mathbf{R}}, & g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}}, & \ldots, & g_2^{\mathbf{B}^*\mathbf{A}_n^\top\mathbf{R}} \end{pmatrix}; \\
\mathbf{g}_j &:= (g_1^{\mathbf{B}\left(\begin{smallmatrix} \mathbf{s}_j \\ \mathbf{0}_d \\ \mathbf{0}_d \end{smallmatrix}\right)}, g_1^{\mathbf{BA}_1\left(\begin{smallmatrix} \mathbf{s}_j \\ \mathbf{0}_d \\ \mathbf{0}_d \end{smallmatrix}\right)}, \ldots, g_1^{\mathbf{BA}_n\left(\begin{smallmatrix} \mathbf{s}_j \\ \mathbf{0}_d \\ \mathbf{0}_d \end{smallmatrix}\right)}); \\
\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j &:= (g_1^{\mathbf{B}\left(\begin{smallmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \mathbf{0}_d \end{smallmatrix}\right)}, g_1^{\mathbf{BA}_1\left(\begin{smallmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \mathbf{0}_d \end{smallmatrix}\right)}, \ldots, g_1^{\mathbf{BA}_n\left(\begin{smallmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \mathbf{0}_d \end{smallmatrix}\right)});
\end{aligned}$$

for $\mathbf{s}_j, \widehat{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^d$.

Lemma 2 ((d,d,q)-LLin ⇒ LS1) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k,q) \leqslant \mathsf{Adv}^{(d,d,q)\text{-LLin}}_{\mathscr{B}}(k),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + qd^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Proof. Given an instance of (d,d,q)-LLin problem (i.e., set $\ell = d$)

$$\left(g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, \left\{g_1^{b_{i,j}}\right\}_{i,j\in[d]}, \left\{g_1^{a_1 s_{1,j}}, \ldots, g_1^{a_d s_{d,j}}\right\}_{j\in[q]}, \left\{g_1^{b_{i,1}s_{1,j}+\cdots+b_{i,d}s_{d,j}+s_{d+i,j}}\right\}_{i\in[d],j\in[q]}\right)$$

as input where all $s_{d+i,j}$ are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathscr{B}$ works as follows:

Programming $\mathbf{s}_j$ and $\widehat{\mathbf{s}}_j$ for $j \in [q]$. Adversary $\mathscr{B}$ implicitly sets

$$\mathbf{s}_j = (s_{1,j}, \ldots, s_{d,j})^\top \text{ and } \widehat{\mathbf{s}}_j = (s_{d+1,j}, \ldots, s_{2d,j})^\top.$$

Programming $\mathbf{B}, \mathbf{B}^*, \mathbf{A}_1, \ldots, \mathbf{A}_n, \mathbf{R}$. Define $\mathbf{W} \in \mathbb{Z}_p^{3d\times 3d}$ as

$$\mathbf{W} := \left(\begin{array}{ccc|ccc|ccc} a_1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & a_d & & & & & & \\ \hline b_{1,1} & \cdots & b_{1,d} & 1 & & & & & \\ \vdots & & \vdots & & \ddots & & & & \\ b_{d,1} & \cdots & b_{d,d} & & & 1 & & & \\ \hline & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{array}\right)$$

and set $\mathbf{W}^* := (\mathbf{W}^{-1})^\top$. Sample $\bar{\mathbf{B}}, \bar{\mathbf{R}} \leftarrow \mathrm{GL}_{3d}(\mathbb{Z}_p)$ and set $\bar{\mathbf{B}}^* := (\bar{\mathbf{B}}^{-1})^\top$. Also sample $\bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_n \leftarrow \mathbb{Z}_p^{3d\times 3d}$, and implicitly set

$$(\mathbf{B}, \mathbf{B}^*) := (\bar{\mathbf{B}}\mathbf{W}, \bar{\mathbf{B}}^*\mathbf{W}^*), \quad \mathbf{R} := \mathbf{W}^\top\bar{\mathbf{R}}, \quad \mathbf{A}_i := \mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W},$$

for $i \in [n]$. Observe that $\mathbf{B}, \mathbf{B}^*, \mathbf{R}$ and all $\mathbf{A}_i$ are distributed properly, and we have

$$\mathbf{B}\mathbf{A}_i = \bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}, \quad \mathbf{B}^*\mathbf{R} = \bar{\mathbf{B}}^*\bar{\mathbf{R}}, \quad \mathbf{B}^*\mathbf{A}_i^\top\mathbf{R} = \bar{\mathbf{B}}^*\bar{\mathbf{A}}_i^\top\bar{\mathbf{R}}.$$

Simulating pp. Algorithm $\mathscr{B}$ can simulate

$$\begin{array}{lll} g_1^{\pi_{\mathrm{L}}(\mathbf{B})} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{B}}\mathbf{W})} = g_1^{\bar{\mathbf{B}}\pi_{\mathrm{L}}(\mathbf{W})} & \text{and} & g_1^{\pi_{\mathrm{L}}(\mathbf{B}\mathbf{A}_i)} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W})} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\pi_{\mathrm{L}}(\mathbf{W})}, \\ g_2^{\mathbf{B}^*\mathbf{R}} = g_2^{\bar{\mathbf{B}}^*\bar{\mathbf{R}}} & \text{and} & g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}} = g_2^{\bar{\mathbf{B}}^*\bar{\mathbf{A}}_i^\top\bar{\mathbf{R}}}, \end{array}$$

for $i \in [n]$ using the knowledge of $g_1^{\pi_{\mathrm{L}}(\mathbf{W})}$ and $\bar{\mathbf{B}}, \bar{\mathbf{B}}^*, \bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_n, \bar{\mathbf{R}}$.

Simulating the challenge. Algorithm $\mathscr{B}$ simulate the challenge as

$$g_1^{\mathbf{B}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}} = g_1^{\bar{\mathbf{B}}\mathbf{W}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}} \text{ and } g_1^{\mathbf{B}\mathbf{A}_i\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}}$$

for $i \in [n]$ using the knowledge of $\bar{\mathbf{B}}, \bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_n$ and

$$g_1^{\mathbf{W}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}} = g_1^{\begin{pmatrix} a_1 s_{1,j} \\ \vdots \\ a_d s_{d,j} \\ b_{1,1}s_{1,j}+\cdots+b_{1,d}s_{d,j}+s_{d+1,j} \\ \vdots \\ b_{d,1}s_{1,j}+\cdots+b_{d,d}s_{d,j}+s_{2d,j} \\ \mathbf{0}_d \end{pmatrix}}.$$

Analysis. Observe that if all $s_{d+i,j} = 0$, then all $\widehat{\mathbf{s}}_j = \mathbf{0}$ and the output challenge is distributed as $\left\{\mathbf{g}_j\right\}_{j\in[q]}$; otherwise, if all $s_{d+i,j} \leftarrow \mathbb{Z}_p^*$, then all $\widehat{\mathbf{s}}_j \leftarrow (\mathbb{Z}_p^*)^d$ and the output challenge is distributed as $\left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j\right\}_{j\in[q]}$. Therefore we may conclude that $\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k,q) \leqslant \mathsf{Adv}^{(d,d,q)\text{-LLin}}_{\mathscr{B}}(k)$. □

Corollary 1 (d-Lin ⇒ LS1) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k,q) \leqslant d \cdot \mathsf{Adv}^{d\text{-Lin}}_{\mathscr{B}}(k) + 1/(p-1),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + qd^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

4.4 Left Subgroup Indistinguishability 2

We may rewrite the LS2 advantage function $\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k,q,q')$ as follows:

$$\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k,q,q') := \left|\Pr[\mathscr{A}(D,T_0)=1] - \Pr[\mathscr{A}(D,T_1)=1]\right|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}_j^* \cdot \widetilde{h}_j^*\right\}_{j\in[q+q']}, \left\{\mathbf{g}'_j \cdot \widehat{\mathbf{g}}'_j\right\}_{j\in[q]}\right), \quad T_0 := \left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j\right\}_{j\in[q]}, \quad T_1 := \left\{\mathbf{g}_j \cdot \widetilde{\mathbf{g}}_j\right\}_{j\in[q]}.$$

and

$$\begin{aligned}
\mathrm{PP} &:= \begin{pmatrix} g_1^{\pi_{\mathrm{L}}(\mathbf{B})}, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_1)}, & \dots, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_n)} \\ g_2^{\mathbf{B}^*\mathbf{R}}, & g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}}, & \dots, & g_2^{\mathbf{B}^*\mathbf{A}_n^\top\mathbf{R}} \end{pmatrix}; \\
\widehat{h}_j^* \cdot \widetilde{h}_j^* &:= g_2^{\mathbf{B}^*\begin{pmatrix}\mathbf{0}_d \\ \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j\end{pmatrix}}; \\
\mathbf{g}'_j \cdot \widehat{\mathbf{g}}'_j &:= (g_1^{\mathbf{B}\begin{pmatrix}\mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \mathbf{0}_d\end{pmatrix}}, g_1^{\mathbf{BA}_1\begin{pmatrix}\mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \mathbf{0}_d\end{pmatrix}}, \dots, g_1^{\mathbf{BA}_n\begin{pmatrix}\mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \mathbf{0}_d\end{pmatrix}}); \\
\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j &:= (g_1^{\mathbf{B}\begin{pmatrix}\mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \mathbf{0}_d\end{pmatrix}}, g_1^{\mathbf{BA}_1\begin{pmatrix}\mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \mathbf{0}_d\end{pmatrix}}, \dots, g_1^{\mathbf{BA}_n\begin{pmatrix}\mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \mathbf{0}_d\end{pmatrix}}); \\
\mathbf{g}_j \cdot \widetilde{\mathbf{g}}_j &:= (g_1^{\mathbf{B}\begin{pmatrix}\mathbf{s}_j \\ \mathbf{0}_d \\ \widetilde{\mathbf{s}}_j\end{pmatrix}}, g_1^{\mathbf{BA}_1\begin{pmatrix}\mathbf{s}_j \\ \mathbf{0}_d \\ \widetilde{\mathbf{s}}_j\end{pmatrix}}, \dots, g_1^{\mathbf{BA}_n\begin{pmatrix}\mathbf{s}_j \\ \mathbf{0}_d \\ \widetilde{\mathbf{s}}_j\end{pmatrix}});
\end{aligned}$$

for $\widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j, \mathbf{s}'_j, \widehat{\mathbf{s}}'_j, \mathbf{s}_j, \widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^d$.

Lemma 3 ((d,d,q)-LLin ⇒ LS2) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k,q,q') \leqslant 2 \cdot \mathsf{Adv}^{(d,d,q)\text{-LLin}}_{\mathscr{B}}(k),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Overview of the Proof. We will prove Lemma 3 using hybrid argument consisting of two steps with the help of an auxiliary distribution $T_{1/2} = \left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j\right\}_{j\in[q]}$ where

$$\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j := (g_1^{\mathbf{B}\begin{pmatrix}\mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j\end{pmatrix}}, g_1^{\mathbf{BA}_1\begin{pmatrix}\mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j\end{pmatrix}}, \dots, g_1^{\mathbf{BA}_n\begin{pmatrix}\mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j\end{pmatrix}}).$$

In particular, we prove that, given D, distribution T_0 and $T_{1/2}$ are computational indistinguishable under the (d,d,q)-LLin assumption (see Lemma 4), and so do $T_{1/2}$ and T_1 (see Lemma 5). These immediately prove Lemma 3.

Lemma 4 (from T_0 to $T_{1/2}$) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\left|\Pr[\mathscr{A}(D,T_0)=1] - \Pr[\mathscr{A}(D,T_{1/2})=1]\right| \leqslant \mathsf{Adv}^{(d,d,q)\text{-LLin}}_{\mathscr{B}}(k),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Proof. Given an instance of (d,d,q)-LLin problem (i.e., set $\ell = d$)

$$\left(g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, \left\{g_1^{b_{i,j}}\right\}_{i,j\in[d]}, \left\{g_1^{a_1 s_{1,j}}, \ldots, g_1^{a_d s_{d,j}}\right\}_{j\in[q]}, \left\{g_1^{b_{i,1}s_{1,j}+\cdots+b_{i,d}s_{d,j}+s_{d+i,j}}\right\}_{i\in[d],j\in[q]}\right)$$

as input where all $s_{d+i,j}$ are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathscr{B}$ works as follows:

Programming $\widehat{\mathbf{s}}_j$ and $\widetilde{\mathbf{s}}_j$ for $j \in [q]$. Adversary $\mathscr{B}$ implicitly sets

$$\widehat{\mathbf{s}}_j = (s_{1,j}, \ldots, s_{d,j})^\top \text{ and } \widetilde{\mathbf{s}}_j = (s_{d+1,j}, \ldots, s_{2d,j})^\top.$$

Programming $\mathbf{B}, \mathbf{B}^*, \mathbf{A}_1, \ldots, \mathbf{A}_n, \mathbf{R}$. We define $\mathbf{W} \in \mathbb{Z}_p^{3d\times 3d}$ as

$$\mathbf{W} := \left(\begin{array}{ccc|ccc|ccc} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ \hline & & & a_1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & a_d & & & \\ \hline & & & b_{1,1} & \cdots & b_{1,d} & 1 & & \\ & & & \vdots & & \vdots & & \ddots & \\ & & & b_{d,1} & \cdots & b_{d,d} & & & 1 \end{array}\right)$$

and set $\mathbf{W}^* := (\mathbf{W}^{-1})^\top$. Sample[3] $\bar{\mathbf{B}}, \bar{\mathbf{R}} \leftarrow \mathrm{GL}_{3d}(\mathbb{Z}_p)$ and set $\bar{\mathbf{B}}^* := (\bar{\mathbf{B}}^{-1})^\top$. Also sample $\bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_n \leftarrow \mathbb{Z}_p^{3d\times 3d}$, and implicitly set

$$(\mathbf{B}, \mathbf{B}^*) := (\bar{\mathbf{B}}\mathbf{W}, \bar{\mathbf{B}}^*\mathbf{W}^*), \quad \mathbf{R} := \mathbf{W}^\top\bar{\mathbf{R}}, \quad \mathbf{A}_i := \mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W},$$

for $i \in [n]$. Observe that $\mathbf{B}, \mathbf{B}^*, \mathbf{R}$ and all $\mathbf{A}_i$ are distributed properly, and we have

$$\mathbf{B}\mathbf{A}_i = \bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}, \quad \mathbf{B}^*\mathbf{R} = \bar{\mathbf{B}}^*\bar{\mathbf{R}}, \quad \mathbf{B}^*\mathbf{A}_i^\top\mathbf{R} = \bar{\mathbf{B}}^*\bar{\mathbf{A}}_i^\top\bar{\mathbf{R}}.$$

Simulating PP. $\mathscr{B}$ can simulate

$$\begin{array}{lll} g_1^{\pi_{\mathrm{L}}(\mathbf{B})} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{B}}\mathbf{W})} = g_1^{\bar{\mathbf{B}}\pi_{\mathrm{L}}(\mathbf{W})} & \text{and} & g_1^{\pi_{\mathrm{L}}(\mathbf{B}\mathbf{A}_i)} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W})} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\pi_{\mathrm{L}}(\mathbf{W})}, \\ g_2^{\mathbf{B}^*\mathbf{R}} = g_2^{\bar{\mathbf{B}}^*\bar{\mathbf{R}}} & \text{and} & g_2^{\mathbf{B}^*\mathbf{A}_i^\top\mathbf{R}} = g_2^{\bar{\mathbf{B}}^*\bar{\mathbf{A}}_i^\top\bar{\mathbf{R}}}, \end{array}$$

for $i \in [n]$ using the knowledge of $\pi_{\mathrm{L}}(\mathbf{W})$ and $\bar{\mathbf{B}}, \bar{\mathbf{B}}^*, \bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_n, \bar{\mathbf{R}}$.

Simulating $\widehat{h}_j^* \cdot \widetilde{h}_j^*$ for $j \in [q+q']$. It is not hard to compute $\mathbf{W}^* \in \mathbb{Z}_p^{3d\times 3d}$ as

$$\mathbf{W}^* := \left(\begin{array}{ccc|ccc|ccc} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ \hline & & & a_1^{-1} & & & -a_1^{-1}b_{1,1} & \cdots & -a_1^{-1}b_{d,1} \\ & & & & \ddots & & \vdots & & \vdots \\ & & & & & a_d^{-1} & -a_d^{-1}b_{1,d} & \cdots & -a_d^{-1}b_{d,d} \\ \hline & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{array}\right).$$

For all $j \in [q+q']$, we sample $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^{2d}$ and implicitly set

$$\begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j \end{pmatrix} = (\mathbf{W}^*)^{-1}\begin{pmatrix} \mathbf{0}_d \\ \bar{\mathbf{r}}_j \end{pmatrix} = \mathbf{W}^\top\begin{pmatrix} \mathbf{0}_d \\ \bar{\mathbf{r}}_j \end{pmatrix}.$$

[3] In our symbol system, a variable with a bar on the top, say $\bar{\mathbf{B}}$, is sampled by the simulator (i.e., $\mathscr{B}$) and is completely known to it.

Since the right-bottom $2d \times 2d$ sub-matrix of $\mathbf{W}^*$ is full-rank with overwhelming probability, both $\widehat{\mathbf{r}}_j$ and $\widetilde{\mathbf{r}}_j$ are distributed properly and $\mathscr{B}$ can simulate

$$\widehat{h}_j^* \cdot \widetilde{h}_j^* = g_2^{\mathbf{B}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j \end{pmatrix}} = g_2^{\bar{\mathbf{B}}^* \mathbf{W}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j \end{pmatrix}} = g_2^{\bar{\mathbf{B}}^* \begin{pmatrix} \mathbf{0}_d \\ \bar{\mathbf{r}}_j \end{pmatrix}}$$

using the knowledge of $\bar{\mathbf{B}}^*$ and $\bar{\mathbf{r}}_j$.

Simulating $\mathbf{g}'_j \cdot \widehat{\mathbf{g}}'_j$ for $j \in [q]$. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\mathbf{B} \begin{pmatrix} \mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \mathbf{0}_d \end{pmatrix}} = g_1^{\bar{\mathbf{B}} \mathbf{W} \begin{pmatrix} \mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \mathbf{0}_d \end{pmatrix}} \text{ and } g_1^{\mathbf{B} \mathbf{A}_i \begin{pmatrix} \mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \mathbf{0}_d \end{pmatrix}} = g_1^{\bar{\mathbf{B}} \bar{\mathbf{A}}_i \mathbf{W} \begin{pmatrix} \mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \mathbf{0}_d \end{pmatrix}}$$

for $i \in [n]$ by sampling $\mathbf{s}'_j, \widehat{\mathbf{s}}'_j \leftarrow \mathbb{Z}_p^d$ and using the knowledge of $g_1^{\mathbf{W}}$ and $\bar{\mathbf{B}}, \bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_n$.

Simulating the challenge. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\mathbf{B} \begin{pmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j \end{pmatrix}} = g_1^{\bar{\mathbf{B}} \mathbf{W} \begin{pmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j \end{pmatrix}} \text{ and } g_1^{\mathbf{B} \mathbf{A}_i \begin{pmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j \end{pmatrix}} = g_1^{\bar{\mathbf{B}} \bar{\mathbf{A}}_i \mathbf{W} \begin{pmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j \end{pmatrix}}$$

using the knowledge of $\bar{\mathbf{B}}, \bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_n$ and

$$g_1^{\mathbf{W} \begin{pmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j \end{pmatrix}} = g_1^{\begin{pmatrix} \mathbf{s}_j \\ a_1 s_{1,j} \\ \vdots \\ a_d s_{d,j} \\ b_{1,1} s_{1,j} + \cdots + b_{1,d} s_{d,j} + s_{d+1,j} \\ \vdots \\ b_{d,1} s_{1,j} + \cdots + b_{d,d} s_{d,j} + s_{2d,j} \end{pmatrix}} \text{ where } \mathbf{s}_j \leftarrow \mathbb{Z}_p^d.$$

Analysis. Observe that if all $s_{d+i,j} = 0$, then all $\widetilde{\mathbf{s}}_j = \mathbf{0}$ and the output challenge is distributed as $\left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j\right\}_{j \in [q]}$; in the other case, if all $s_{d+i,j} \leftarrow \mathbb{Z}_p^*$, then all $\widetilde{\mathbf{s}}_j \leftarrow (\mathbb{Z}_p^*)^d$ and the output challenge is distributed as $\left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j\right\}_{j \in [q]}$. Therefore we may conclude that $\left|\Pr[\mathscr{A}(D, T_0) = 1] - \Pr[\mathscr{A}(D, T_{1/2}) = 1]\right| \leqslant \mathsf{Adv}_{\mathscr{B}}^{(d,d,q)\text{-LLin}}(k)$. □

Lemma 5 (from $T_{1/2}$ to T_1) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\left|\Pr[\mathscr{A}(D, T_{1/2}) = 1] - \Pr[\mathscr{A}(D, T_1) = 1]\right| \leqslant \mathsf{Adv}_{\mathscr{B}}^{(d,d,q)\text{-LLin}}(k),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q + q')d^2 \cdot \mathsf{poly}(k, n)$ *where* $\mathsf{poly}(k, n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Proof. The proof is similar to that for Lemma 4. Given an instance of (d, d, q)-LLin problem (i.e., set $\ell = d$)

$$\left(g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, \left\{g_1^{b_{i,j}}\right\}_{i,j \in [d]}, \left\{g_1^{a_1 s_{1,j}}, \ldots, g_1^{a_d s_{d,j}}\right\}_{j \in [q]}, \left\{g_1^{b_{i,1} s_{1,j} + \cdots + b_{i,d} s_{d,j} + s_{d+i,j}}\right\}_{i \in [d], j \in [q]}\right)$$

as input where all $s_{d+i,j}$ are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathscr{B}$ behaves as in the proof of Lemma 4 with the differences that:

Programming $\widehat{\mathbf{s}}_j$ and $\widetilde{\mathbf{s}}_j$ for $j \in [q]$. Adversary $\mathscr{B}$ implicitly sets

$$\widehat{\mathbf{s}}_j = (s_{2d,j}, \ldots, s_{d+1,j})^\top \text{ and } \widetilde{\mathbf{s}}_j = (s_{d,j}, \ldots, s_{1,j})^\top.$$

Defining $\mathbf{W}$. Adversary $\mathscr{B}$ defines $\mathbf{W} \in \mathbb{Z}_p^{3d \times 3d}$ as

$$\mathbf{W} := \left(\begin{array}{ccc|ccc} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ \hline & & & 1 & b_{d,d} & \cdots & b_{d,1} \\ & & & & \vdots & & \vdots \\ & & & & b_{1,d} & \cdots & b_{1,1} \\ \hline & & & & a_d & & \\ & & & & & \ddots & \\ & & & & & & a_1 \end{array}\right).$$

Then algoirthm $\mathscr{B}$ can program $\mathbf{B}, \mathbf{B}^*, \mathbf{A}_1, \ldots, \mathbf{A}_n, \mathbf{R}$ and simulate all entries in PP, $\left\{\widehat{h}_j^* \cdot \widetilde{h}_j^*\right\}_{j\in[q+q']}$, $\left\{\mathbf{g}'_j \cdot \widehat{\mathbf{g}}'_j\right\}_{j\in[q]}$ as well as the challenge following the strategies proving Lemma 4. Observe that, if all $s_{d+i,j} = 0$, then all $\widehat{\mathbf{s}}_j = \mathbf{0}$ and the output challenge is distributed as $\left\{\mathbf{g}_j \cdot \widetilde{\mathbf{g}}_j\right\}_{j\in[q]}$; in the other case, if all $s_{d+i,j} \leftarrow \mathbb{Z}_p^*$, then all $\widetilde{\mathbf{s}}_j \leftarrow (\mathbb{Z}_p^*)^d$ and the output challenge is distributed as $\left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j\right\}_{j\in[q]}$. □

Corollary 2 (d-Lin ⇒ LS2) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k, q, q') \leqslant 2d \cdot \mathsf{Adv}^{d\text{-Lin}}_{\mathscr{B}}(k) + 2/(p-1),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k, n)$ *where* $\mathsf{poly}(k, n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

4.5 Generalized Many-Tuple Lemma

The proof of the nested-hiding indistinguishability property requires a generalized version of many-tuple lemma shown in [CW13]. Instead of the d-Lin assumption, this subsection is going to establish a generalized version from the (d, d, d)-LLin assumption.

Lemma 6 (Generalized Many-Tuple Lemma) *There exists an efficient algorithm that on input $q \in \mathbb{Z}^+$, a finite cyclic group G generated by $g \in G$ and*

$$\left(g, g^{a_1}, \ldots, g^{a_d}, \left\{g^{b_{i,j}}\right\}_{i,j\in[d]}, \left\{g^{a_1 r_{1,j}}, \ldots, g^{a_d r_{d,j}}\right\}_{j\in[d]}, \left\{g^{b_{i,1}r_{1,j}+\cdots+b_{i,d}r_{d,j}+r_{d+i,j}}\right\}_{i,j\in[d]}\right),$$

outputs $\left(g^{\mathbf{VZ}}, g^{\mathbf{Z}}\right)$ for some matrix $\mathbf{V} \in \mathbb{Z}_p^{d\times d}$ along with

$$\left\{\left(g^{\mathbf{t}_j}, g^{\mathbf{V}\mathbf{t}_j+\boldsymbol{\tau}_j}\right)\right\}_{j\in[q]},$$

where $\mathbf{t}_j \leftarrow \mathbb{Z}_p^d$, $\mathbf{Z}$ is an invertible diagonal matrix, and all $\boldsymbol{\tau}_j$ are either $\mathbf{0}_d$ or uniformly distributed over $\mathbb{Z}_p^d$ depending on whether all $r_{d+i,j}$ are 0 or uniformly distributed over $\mathbb{Z}_p$.

Proof. The algorithm works as follows:

Programming V and Z. We implicitly define $\mathbf{V}, \mathbf{Z} \in \mathbb{Z}_p^{d\times d}$ and $\mathbf{P} \in \mathbb{Z}_p^{d\times 2d}$ as follows

$$\mathbf{V} := \begin{pmatrix} r_{1,1} & \cdots & r_{d,1} \\ \vdots & & \vdots \\ r_{1,d} & \cdots & r_{d,d} \end{pmatrix} \text{ and } \mathbf{Z} := \begin{pmatrix} a_1 & & \\ & \ddots & \\ & & a_d \end{pmatrix},$$

and

$$\mathbf{P} := \left(\begin{array}{ccc|ccc} a_1 & & & b_{1,1} & \cdots & b_{d,1} \\ & \ddots & & \vdots & & \vdots \\ & & a_d & b_{1,d} & \cdots & b_{d,d} \end{array}\right).$$

It is not hard to see that we can compute $g^{\mathbf{VZ}}$, $g^{\mathbf{Z}}$, $g^{\mathbf{P}}$, and

$$g^{\mathbf{C}} := \left(\begin{array}{ccc|ccc} g^{a_1 r_{1,1}} & \cdots & g^{a_d r_{d,1}} & g^{b_{1,1}r_{1,1}+\cdots+b_{1,d}r_{d,1}+r_{d+1,1}} & \cdots & g^{b_{d,1}r_{1,1}+\cdots+b_{d,d}r_{d,1}+r_{2d,1}} \\ \vdots & & \vdots & \vdots & & \vdots \\ g^{a_1 r_{1,d}} & \cdots & g^{a_d r_{d,d}} & g^{b_{1,1}r_{1,d}+\cdots+b_{1,d}r_{d,d}+r_{d+1,d}} & \cdots & g^{b_{d,1}r_{1,d}+\cdots+b_{d,d}r_{d,d}+r_{2d,d}} \end{array}\right).$$

Generating q tuples. For each $j \in [q]$, sample $\bar{\mathbf{t}}_j \leftarrow \mathbb{Z}_p^{2d}$ and output

$$\left(g^{\mathbf{t}_j}, g^{\mathbf{V}\mathbf{t}_j+\boldsymbol{\tau}_j}\right) := \left(g^{\mathbf{P}\bar{\mathbf{t}}_j}, g^{\mathbf{C}\bar{\mathbf{t}}_j}\right).$$

Analysis. Observe that, if all $r_{d+i,j} = 0$, we have known that $g^{\mathbf{C}} = g^{\mathbf{VP}}$ and thus $\boldsymbol{\tau}_j = \mathbf{0}_d$; otherwise, when all $r_{d+i,j} \leftarrow \mathbb{Z}_p$, we may write $g^{\mathbf{C}} = g^{\mathbf{VP}+\mathbf{T}}$ where

$$\mathbf{T} = \left(\begin{array}{ccc|ccc} 0 & \cdots & 0 & r_{d+1,1} & \cdots & r_{2d,1} \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & r_{d+1,d} & \cdots & r_{2d,d} \end{array}\right) \in \mathbb{Z}_p^{d\times 2d},$$

and thus implicitly set $\boldsymbol{\tau}_j = \mathbf{T}\bar{\mathbf{t}}_j$. Clearly, we have

$$\begin{pmatrix} \mathbf{t}_j \\ \boldsymbol{\tau}_j \end{pmatrix} = \begin{pmatrix} \mathbf{P} \\ \mathbf{T} \end{pmatrix} \bar{\mathbf{t}}_j.$$

Since $\left(\begin{smallmatrix}\mathbf{P}\\ \mathbf{T}\end{smallmatrix}\right)$ is full-rank with overwhelming probability, $\mathbf{t}_j$ and $\boldsymbol{\tau}_j$ are distributed independently. □

4.6 Nested-hiding Indistinguishability

We may rewrite the NH advantage function $\mathsf{Adv}^{\mathrm{NH}(\eta)}_{\mathscr{A}}(k,q,q')$ for all $\eta \in [\lfloor n/2 \rfloor]$ as follows:

$$\mathsf{Adv}^{\mathrm{NH}(\eta)}_{\mathscr{A}}(k,q,q') := |\Pr[\mathscr{A}(D,T_0)=1] - \Pr[\mathscr{A}(D,T_1)=1]|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}^*_j\right\}_{j\in[q+q']}, \left\{\widetilde{h}^*_j\right\}_{j\in[q+q']}, \left\{\left(\widehat{\mathbf{g}}_j\right)_{-(2\eta-1)}\right\}_{j\in[q]}, \left\{\left(\widetilde{\mathbf{g}}_j\right)_{-2\eta}\right\}_{j\in[q]}\right),$$

$$T_0 := \left\{\mathbf{h}_j\right\}_{j\in[q']},\ T_1 := \left\{\mathbf{h}'_j\right\}_{j\in[q']}.$$

and

$$\begin{aligned}
\mathrm{PP} &:= \begin{pmatrix} g_1^{\pi_{\mathrm{L}}(\mathbf{B})}, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_1)}, & \cdots, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_n)} \\ g_2^{\mathbf{B}^*\mathbf{R}}, & g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}}, & \cdots, & g_2^{\mathbf{B}^*\mathbf{A}_n^\top\mathbf{R}} \end{pmatrix};\\
\widehat{h}^*_j &:= g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}_j},\quad \widetilde{h}^*_j := g_2^{\pi_{\mathrm{R}}(\mathbf{B}^*)\widetilde{\mathbf{r}}_j};\\
\widehat{\mathbf{g}}_j &:= \left(g_1^{\pi_{\mathrm{M}}(\mathbf{B})\widehat{\mathbf{s}}_j}, g_1^{\pi_{\mathrm{M}}(\mathbf{BA}_1)\widehat{\mathbf{s}}_j}, \cdots, g_1^{\pi_{\mathrm{M}}(\mathbf{BA}_n)\widehat{\mathbf{s}}_j}\right);\\
\widetilde{\mathbf{g}}_j &:= \left(g_1^{\pi_{\mathrm{R}}(\mathbf{B})\widetilde{\mathbf{s}}_j}, g_1^{\pi_{\mathrm{R}}(\mathbf{BA}_1)\widetilde{\mathbf{s}}_j}, \ldots, g_1^{\pi_{\mathrm{R}}(\mathbf{BA}_n)\widetilde{\mathbf{s}}_j}\right);\\
\mathbf{h}_j &:= \left(g_2^{\mathbf{B}^*\mathbf{R}\mathbf{r}_j}, g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}\mathbf{r}_j}, \ldots, g_2^{\mathbf{B}^*\mathbf{A}_{2\eta-1}^\top\mathbf{R}\mathbf{r}_j}, g_2^{\mathbf{B}^*\mathbf{A}_{2\eta}^\top\mathbf{R}\mathbf{r}_j}, \ldots, g_2^{\mathbf{B}^*\mathbf{A}_n^\top\mathbf{R}\mathbf{r}_j}\right);\\
\mathbf{h}'_j &:= \left(g_2^{\mathbf{B}^*\mathbf{R}\mathbf{r}_j}, g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}\mathbf{r}_j}, \ldots, g_2^{\mathbf{B}^*\mathbf{A}_{2\eta-1}^\top\mathbf{R}\mathbf{r}_j+\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\boldsymbol{\gamma}}_j}, g_2^{\mathbf{B}^*\mathbf{A}_{2\eta}^\top\mathbf{R}\mathbf{r}_j+\pi_{\mathrm{R}}(\mathbf{B}^*)\widetilde{\boldsymbol{\gamma}}_j}, \ldots, g_2^{\mathbf{B}^*\mathbf{A}_n^\top\mathbf{R}\mathbf{r}_j}\right);
\end{aligned}$$

where $\widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j, \widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j, \widehat{\boldsymbol{\gamma}}_j, \widetilde{\boldsymbol{\gamma}}_j \leftarrow \mathbb{Z}_p^d$ and $\mathbf{r}_j \leftarrow \mathbb{Z}_p^{3d}$.

Lemma 7 ((d,d,d)-LLin $\Rightarrow$ NH) *For any $\eta \in [\lfloor n/2 \rfloor]$ and for any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{NH}(\eta)}_{\mathscr{A}}(k,q,q') \leqslant \mathsf{Adv}^{(d,d,d)\text{-LLin}}_{\mathscr{B}}(k),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Proof. Given an instance of (d,d,d)-LLin problem (on G_2)

$$\left(g_1, g_2, g_2^{a_1}, \ldots, g_2^{a_d}, \left\{g_2^{b_{i,j}}\right\}_{i,j\in[d]}, \left\{g_2^{a_1 r_{1,j}}, \ldots, g_2^{a_d r_{d,j}}\right\}_{j\in[d]}, \left\{g_2^{b_{i,1}r_{1,j}+\cdots+b_{i,d}r_{d,j}+r_{d+i,j}}\right\}_{i,j\in[d]}\right),$$

where all $r_{d+i,j}$ are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathscr{B}$ works as follows:

Generating $2q'$ tuples. Algorithm $\mathscr{B}$ runs the algorithm described in Lemma 6 on the input $2q'$, group G_2, and the (d,d,d)-LLin instance, and obtains

$$\left(g_2^{\mathbf{VZ}}, g_2^{\mathbf{Z}}\right) \text{ and } \left\{\left(g_2^{\mathbf{t}_j}, g_2^{\mathbf{V}\mathbf{t}_j+\boldsymbol{\tau}_j}\right)\right\}_{j\in[2q']}.$$

Programming $\mathbf{B}, \mathbf{B}^*, \mathbf{R}, \mathbf{A}_1, \ldots, \mathbf{A}_n$. Sample $\mathbf{B} \leftarrow \mathrm{GL}_{3d}(\mathbb{Z}_p)$ and set $\mathbf{B}^* := (\mathbf{B}^{-1})^\top$. Sample $\mathbf{A}_i \leftarrow \mathbb{Z}_p^{3d\times 3d}$ for all $i \in [n] \setminus \{2\eta-1, 2\eta\}$. Sample $\bar{\mathbf{A}}_{2\eta-1}, \bar{\mathbf{A}}_{2\eta} \leftarrow \mathbb{Z}_p^{3d\times 3d}$, $\bar{\mathbf{R}} \leftarrow \mathrm{GL}_{3d}(\mathbb{Z}_p)$ and implicitly set

$$\mathbf{A}_{2\eta-1} := \bar{\mathbf{A}}_{2\eta-1} + \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{V}^\top & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix},\quad \mathbf{A}_{2\eta} := \bar{\mathbf{A}}_{2\eta} + \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{V}^\top \end{pmatrix},\quad \text{and}\quad \mathbf{R} := \begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Z} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{Z} \end{pmatrix}\bar{\mathbf{R}},$$

where $\mathbf{I}$ is the d-by-d identity matrix and $\mathbf{0}$ is the d-by-d zero matrix.

Simulating PP. Algorithm $\mathscr{B}$ can simulate $g_1^{\pi_{\mathrm{L}}(\mathbf{B})}$ and $g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_i)} = g_1^{\mathbf{B}\pi_{\mathrm{L}}(\mathbf{A}_i)}$ for $i \in [n]$ using the knowledge of $\mathbf{B}, \pi_{\mathrm{L}}(\mathbf{A}_1), \ldots, \pi_{\mathrm{L}}(\mathbf{A}_n)$. Note that we have $\pi_{\mathrm{L}}(\mathbf{A}_{2\eta-1}) = \pi_{\mathrm{L}}(\bar{\mathbf{A}}_{2\eta-1})$ and $\pi_{\mathrm{L}}(\mathbf{A}_{2\eta}) = \pi_{\mathrm{L}}(\bar{\mathbf{A}}_{2\eta})$, which are known to $\mathscr{B}$. Algorithm $\mathscr{B}$ can also simulate $g_2^{\mathbf{B}^*\mathbf{R}}$ and $g_2^{\mathbf{B}^*\mathbf{A}_i^\top\mathbf{R}}$ for $i \in [n] \setminus \{2\eta-1, 2\eta\}$ using the knowledge of $g_2^{\mathbf{Z}}$ and $\mathbf{B}^*, \bar{\mathbf{R}}$ as well as $\mathbf{A}_i$. Finally, it can simulate

$$g_2^{\mathbf{B}^*\mathbf{A}_{2\eta-1}^\top\mathbf{R}} = g_2^{\mathbf{B}^*\bar{\mathbf{A}}_{2\eta-1}^\top\left(\begin{smallmatrix}\mathbf{I}&\mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{Z}&\mathbf{0}\\ \mathbf{0}&\mathbf{0}&\mathbf{Z}\end{smallmatrix}\right)\bar{\mathbf{R}} + \mathbf{B}^*\left(\begin{smallmatrix}\mathbf{0}&\mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{V}\mathbf{Z}&\mathbf{0}\\ \mathbf{0}&\mathbf{0}&\mathbf{0}\end{smallmatrix}\right)\bar{\mathbf{R}}} \quad \text{and} \quad g_2^{\mathbf{B}^*\mathbf{A}_{2\eta}^\top\mathbf{R}} = g_2^{\mathbf{B}^*\bar{\mathbf{A}}_{2\eta}^\top\left(\begin{smallmatrix}\mathbf{I}&\mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{Z}&\mathbf{0}\\ \mathbf{0}&\mathbf{0}&\mathbf{Z}\end{smallmatrix}\right)\bar{\mathbf{R}} + \mathbf{B}^*\left(\begin{smallmatrix}\mathbf{0}&\mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{0}&\mathbf{V}\mathbf{Z}\end{smallmatrix}\right)\bar{\mathbf{R}}}$$

using the knowledge of $\left(g_2^{\mathbf{VZ}}, g_2^{\mathbf{Z}}\right)$ and $\mathbf{B}^*, \bar{\mathbf{A}}_{2\eta-1}, \bar{\mathbf{A}}_{2\eta}$ and $\bar{\mathbf{R}}$.

Simulating $\widehat{h}_j^*$ and $\widetilde{h}_j^*$ for $j \in [q+q']$. Algorithm $\mathscr{B}$ can simulate

$$\widehat{h}_j^* = g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}_j} \quad \text{and} \quad \widetilde{h}_j^* = g_2^{\pi_{\mathrm{R}}(\mathbf{B}^*)\widetilde{\mathbf{r}}_j}$$

by sampling $\widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d$ and using the knowledge of $\mathbf{B}^*$.

Simulating $\left\{\left(\widehat{\mathbf{g}}_j\right)_{-(2\eta-1)}\right\}_{j\in[q]}$ and $\left\{\left(\widetilde{\mathbf{g}}_j\right)_{-2\eta}\right\}_{j\in[q]}$ for $j \in [q]$. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\pi_{\mathrm{M}}(\mathbf{B})\widehat{\mathbf{s}}_j} \quad \text{and} \quad g_1^{\pi_{\mathrm{M}}(\mathbf{B}\mathbf{A}_i)\widehat{\mathbf{s}}_j} = g_1^{\mathbf{B}\pi_{\mathrm{M}}(\mathbf{A}_i)\widehat{\mathbf{s}}_j},$$

for $i \in [n] \setminus \{2\eta-1\}$ and $j \in [q]$ by sampling $\widehat{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^d$ and using the knowledge of $\mathbf{B}$ and $\pi_{\mathrm{M}}(\mathbf{A}_i)$ for $i \in [n] \setminus \{2\eta-1\}$. We note that $\pi_{\mathrm{M}}(\mathbf{A}_{2\eta}) = \pi_{\mathrm{M}}(\bar{\mathbf{A}}_{2\eta})$ is know to $\mathscr{B}$, but $\pi_{\mathrm{M}}(\mathbf{A}_{2\eta-1})$ containing secret matrix $\mathbf{V}$ is not. In a similar manner, algorithm $\mathscr{B}$ can also simulate

$$g_1^{\pi_{\mathrm{R}}(\mathbf{B})\widetilde{\mathbf{s}}_j} \quad \text{and} \quad g_1^{\pi_{\mathrm{R}}(\mathbf{B}\mathbf{A}_i)\widetilde{\mathbf{s}}_j} = g_1^{\mathbf{B}\pi_{\mathrm{R}}(\mathbf{A}_i)\widetilde{\mathbf{s}}_j},$$

for $i \in [n] \setminus \{2\eta\}$ and $j \in [q]$ by sampling $\widetilde{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^d$ and using the knowledge of $\mathbf{B}$ and $\pi_{\mathrm{R}}(\mathbf{A}_i)$ for $i \in [n] \setminus \{2\eta\}$. We note that $\pi_{\mathrm{R}}(\mathbf{A}_{2\eta-1}) = \pi_{\mathrm{R}}(\bar{\mathbf{A}}_{2\eta-1})$ is know to $\mathscr{B}$, but $\pi_{\mathrm{R}}(\mathbf{A}_{2\eta})$ containing secret matrix $\mathbf{V}$ is not.

Simulating the challenge. For each $j \in [q']$, $\mathscr{B}$ samples $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d$ and implicitly sets $\mathbf{r}_j$ by

$$\mathbf{R}\mathbf{r}_j := \begin{pmatrix} \bar{\mathbf{r}}_j \\ \mathbf{t}_{2j-1} \\ \mathbf{t}_{2j} \end{pmatrix}.$$

Adversary $\mathscr{B}$ can simulate

$$g_2^{\mathbf{B}^*\mathbf{R}\mathbf{r}_j} = g_2^{\mathbf{B}^*\begin{pmatrix} \bar{\mathbf{r}}_j \\ \mathbf{t}_{2j-1} \\ \mathbf{t}_{2j} \end{pmatrix}} \quad \text{and} \quad g_2^{\mathbf{B}^*\mathbf{A}_i^\top\mathbf{R}\mathbf{r}_j} = g_2^{\mathbf{B}^*\mathbf{A}_i^\top\begin{pmatrix} \bar{\mathbf{r}}_j \\ \mathbf{t}_{2j-1} \\ \mathbf{t}_{2j} \end{pmatrix}},$$

for $i \in [n] \setminus \{2\eta-1, 2\eta\}$ using the knowledge of $g_2^{\mathbf{t}_{2j-1}}$, $g_2^{\mathbf{t}_{2j}}$, $\mathbf{B}^*$ and $\mathbf{A}_i$, and simulate

$$\begin{aligned} g_2^{\mathbf{B}^*\mathbf{A}_{2\eta-1}^\top\mathbf{R}\mathbf{r}_j + \pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\boldsymbol{\gamma}}_j} &= g_2^{\mathbf{B}^*\bar{\mathbf{A}}_{2\eta-1}^\top\begin{pmatrix} \bar{\mathbf{r}}_j \\ \mathbf{t}_{2j-1} \\ \mathbf{t}_{2j} \end{pmatrix} + \pi_{\mathrm{M}}(\mathbf{B}^*)(\mathbf{V}\mathbf{t}_{2j-1} + \boldsymbol{\tau}_{2j-1})} \\ g_2^{\mathbf{B}^*\mathbf{A}_{2\eta}^\top\mathbf{R}\mathbf{r}_j + \pi_{\mathrm{R}}(\mathbf{B}^*)\widetilde{\boldsymbol{\gamma}}_j} &= g_2^{\mathbf{B}^*\bar{\mathbf{A}}_{2\eta}^\top\begin{pmatrix} \bar{\mathbf{r}}_j \\ \mathbf{t}_{2j-1} \\ \mathbf{t}_{2j} \end{pmatrix} + \pi_{\mathrm{R}}(\mathbf{B}^*)(\mathbf{V}\mathbf{t}_{2j} + \boldsymbol{\tau}_{2j})} \end{aligned}$$

using the knowledge of $\left(g_2^{\mathbf{t}_{2j-1}}, g_2^{\mathbf{V}\mathbf{t}_{2j-1}+\boldsymbol{\tau}_{2j-1}}\right)$ and $\left(g_2^{\mathbf{t}_{2j}}, g_2^{\mathbf{V}\mathbf{t}_{2j}+\boldsymbol{\tau}_{2j}}\right)$ as well as $\mathbf{B}^*, \bar{\mathbf{A}}_{2\eta-1}, \bar{\mathbf{A}}_{2\eta}$. Here we implicitly set $\widehat{\boldsymbol{\gamma}}_j = \boldsymbol{\tau}_{2j-1}$ and $\widetilde{\boldsymbol{\gamma}}_j = \boldsymbol{\tau}_{2j}$.

Analysis. Observe that if all $r_{d+i,j} = 0$, then $\widehat{\boldsymbol{\gamma}}_j = \widetilde{\boldsymbol{\gamma}}_j = \mathbf{0}$ and the output challenge is distributed as $\left\{\mathbf{h}_j\right\}_{j\in[q']}$; otherwise, if all $r_{d+i,j} \leftarrow \mathbb{Z}_p^*$, then $\widehat{\boldsymbol{\gamma}}_j, \widetilde{\boldsymbol{\gamma}}_j \leftarrow (\mathbb{Z}_p^*)^d$ and the output challenge is distributed as $\left\{\mathbf{h}'_j\right\}_{j\in[q']}$. Therefore we may conclude that $\mathsf{Adv}_{\mathscr{A}}^{\mathrm{NH}(\eta)}(k,q,q') \leqslant \mathsf{Adv}_{\mathscr{B}}^{(d,d,d)\text{-LLin}}(k)$. □

Corollary 3 (d-Lin ⇒ NH) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}_{\mathscr{A}}^{\mathrm{NH}(\eta)}(k,q,q') \leqslant d \cdot \mathsf{Adv}_{\mathscr{B}}^{d\text{-LLin}}(k) + 1/(p-1),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

5 Concrete IBE from d-Linear Assumption

This section describe the concrete IBE scheme derived from our prime-order instantiation in Section 4 following Hofheinz *et al.*'s framework (c.f. Appendix B). Let GrpGen be the bilinear group generator described in Section 4.1 and $\pi_{\mathrm{L}}(\cdot)$ be the function mapping from a $3d \times 3d$ matrix to its left-most d columns.

– $\mathsf{Param}(1^k, n)$: Run $(p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$. Sample $\mathbf{B}, \mathbf{R} \leftarrow \mathrm{GL}_{3d}(\mathbb{Z}_p)$ and $\mathbf{A}_1, \ldots, \mathbf{A}_{2n} \leftarrow \mathbb{Z}_p^{3d\times 3d}$, and set $\mathbf{B}^* := (\mathbf{B}^{-1})^\top$. Output

$$\mathrm{GP} := \left(p, G_1^{3d}, G_2^{3d}, G_T, e; \begin{matrix} g_1^{\pi_{\mathrm{L}}(\mathbf{B})}, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_1)}, & \cdots, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_{2n})} \\ g_2^{\mathbf{B}^*\mathbf{R}}, & g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}}, & \cdots, & g_2^{\mathbf{B}^*\mathbf{A}_{2n}^\top\mathbf{R}} \end{matrix} \right).$$

– $\mathsf{Setup}(\mathrm{GP})$: Sample $\mathbf{k} \leftarrow \mathbb{Z}_p^{3d}$ and output[4]

$$\begin{aligned} \mathrm{MPK} &:= \left(p, G_1^{3d}, G_2^{3d}, G_T, e; e(g_1, g_2)^{\pi_{\mathrm{L}}(\mathbf{B})^\top\mathbf{k}}, g_1^{\pi_{\mathrm{L}}(\mathbf{B})}, g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_1)}, \ldots, g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_{2n})} \right); \\ \mathrm{MSK} &:= \left(g_2^{\mathbf{k}}, g_2^{\mathbf{B}^*\mathbf{R}}, g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}}, \ldots, g_2^{\mathbf{B}^*\mathbf{A}_{2n}^\top\mathbf{R}} \right). \end{aligned}$$

– $\mathsf{KeyGen}(\mathrm{MPK}, \mathrm{MSK}, \mathbf{y})$: Let $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$. Sample $\mathbf{r} \leftarrow \mathbb{Z}_p^{3d}$ and output

$$\mathrm{SK}_{\mathbf{y}} := \left(K_0 := g_2^{\mathbf{B}^*\mathbf{R}\mathbf{r}},\ K_1 := g_2^{\mathbf{k}+\mathbf{B}^*(\mathbf{A}_{2-y_1}+\cdots+\mathbf{A}_{2n-y_n})^\top\mathbf{R}\mathbf{r}} \right).$$

– $\mathsf{Enc}(\mathrm{MPK}, \mathbf{x}, \mathrm{M})$: Let $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$ and $\mathrm{M} \in \mathbb{G}_T$. Sample $\mathbf{s} \leftarrow \mathbb{Z}_p^d$ and output

$$\mathrm{CT}_{\mathbf{x}} := \left(C_0 := g_1^{\pi_{\mathrm{L}}(\mathbf{B})\mathbf{s}},\ C_1 := g_1^{\pi_{\mathrm{L}}(\mathbf{B}(\mathbf{A}_{2-x_1}+\cdots+\mathbf{A}_{2n-x_n}))\mathbf{s}},\ C_2 := e(g_1, g_2)^{\mathbf{s}^\top\pi_{\mathrm{L}}(\mathbf{B})^\top\mathbf{k}} \cdot \mathrm{M} \right).$$

– $\mathsf{Dec}(\mathrm{MPK}, \mathrm{SK}, \mathrm{CT})$: Let $\mathrm{SK} = (K_0, K_1)$ and $\mathrm{CT} = (C_0, C_1, C_2)$. Output $\mathrm{M} := C_2 \cdot e(C_1, K_0)/e(C_0, K_1)$.

6 Achieving Stronger Security Guarantee

This section will investigate two flavors of stronger adaptive security: *B-weak* and *full* adaptive security (see Section 2) by enhancing the *non-degeneracy* property and updating the indistinguishability between Game_3 and Game_4 in Hofheinz *et al.*'s proof. Before we begin to work, we first recall Game_3 and Game_4 in Hofheinz *et al.*'s proof (c.f. Appendix B) where the non-degeneracy is utilized. In Game_3, challenger $\mathscr{C}$ answers all key extraction queries and all challenge queries using type-n semi-functional secret keys and type-$(\wedge, n)$ semi-functional ciphertexts, respectively. More formally, the experiment between challenger $\mathscr{C}$ and adversary $\mathscr{A}$ proceeds as follows:

Setup. $\mathscr{C}$ samples $(\mathrm{PP}, \mathrm{SP}) \leftarrow \mathsf{SampP}(1^k, 2n)$ and $\mathrm{MSK}_\iota \leftarrow \mathbb{H}$ (using PP) for all $\iota \in [\lambda]$, and returns

$$\{\mathrm{MPK}_\iota := (\mathrm{PP}, \mu(\mathrm{MSK}_\iota)\}_{\iota\in[\lambda]}$$

to adversary $\mathscr{A}$. Then pick $\beta \leftarrow \{0,1\}$. During the experiment, $\mathscr{C}$ maintains two random functions

$$\widehat{\mathsf{R}}_n : [\lambda] \times \{0,1\}^n \to [\widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})] \quad \text{and} \quad \widetilde{\mathsf{R}}_n : [\lambda] \times \{0,1\}^n \to [\widetilde{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})].$$

Key extraction queries. On query $(\iota, \mathbf{y})$, let $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$. $\mathscr{C}$ samples $\mathbf{h} := (h_0, h_1, \ldots, h_{2n}) \leftarrow \mathsf{SampH}(\mathrm{PP})$ and outputs

$$\mathrm{SK} := \left(h_0,\ \mathrm{MSK}_\iota \cdot \widehat{\mathsf{R}}_n(\iota, \mathbf{y}) \cdot \widetilde{\mathsf{R}}_n(\iota, \mathbf{y}) \cdot \prod_{i=1}^{n} h_{2i-y_i} \right)$$

and updates $Q_K := Q_K \cup \{(\iota, \mathbf{y})\}$.

Challenge queries. On query $(\iota^*, \mathbf{x}^*, \mathrm{M}_0^*, \mathrm{M}_1^*)$, let $\mathbf{x}^* = (x_1^*, \ldots, x_n^*) \in \{0,1\}^n$. $\mathscr{C}$ samples $(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\mathrm{PP})$ and $(\widehat{g}_0, \widehat{g}_1, \ldots, \widehat{g}_{2n}) \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$ and outputs

$$\mathrm{CT}^* := \left(g_0 \cdot \widehat{g}_0,\ \prod_{i=1}^{n} \left(g_{2i-x_i^*} \cdot \widehat{g}_{2i-x_i^*} \right),\ \boxed{e(g_0 \cdot \widehat{g}_0, \mathrm{MSK}_{\iota^*}) \cdot e(\widehat{g}_0, \widehat{\mathsf{R}}_n(\iota^*, \mathbf{x}^*)) \cdot \mathrm{M}_\beta^*} \right),$$

and updates $Q_C := Q_C \cup \{(\iota^*, \mathbf{x}^*)\}$.

Guess. $\mathscr{A}$ outputs its guess $\beta' \in \{0,1\}$.

Note that the boxed term have been re-written following the *orthogonality* property of our revised NDSG. Game_4 is identical to Game_3 except that the boxed term is independently and uniformly distributed over $\mathbb{G}_T$ for each challenge ciphertext.

[4] We only put necessary entries for Enc into MPK, while entries from GP (or PP) for running KeyGen are put into MSK.

6.1 Warmup: Achieving B-weak Adaptive Security

Recall that the original non-degeneracy property said that:

(Non-degeneracy (Recalled).) Over the probability space defined by $\widehat{g}_0 \leftarrow \widehat{\mathsf{SampG}_0}(\mathrm{PP}, \mathrm{SP})$, with overwhelming probability $1-2^{-\Omega(k)}$, $e(\widehat{g}_0, \widehat{h}^*)$ is distributed uniformly over $\mathbb{G}_T$ when sampling $\widehat{h}^* \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$.

We observe that $\widehat{h}^*$ in our prime-order instantiation (see Section 4) actually contains higher entropy than those in Hofheinz *et al.*'s composite-order instantiation [HKS15]. In particular, $\widehat{h}^*$ is uniformly distributed over a d-dimension subspace of G_2^{3d} containing p^d elements (vectors), while $e(\widehat{g}_0, \widehat{h}^*)$ is an element in G_T containing just p elements. This suggests that, given $e(\widehat{g}_0, \widehat{h}^*)$, there may be leftover entropy in $\widehat{h}^*$, and our prime-order instantiation may achieve stronger non-degeneracy even relying on no computational assumption.

To formally investigate the above idea, we describe the notion of *B-bounded non-degeneracy* which roughly ensures the non-degeneracy even when a single $\widehat{h}^*$ is paired with at most B $\widehat{g}_0$'s.

(B-bounded non-degeneracy.) Over the probability space defined by $(\widehat{g}_{0,1}, \ldots, \widehat{g}_{0,B}) \leftarrow \widehat{\mathsf{SampG}_0}^B(\mathrm{PP}, \mathrm{SP})$, with overwhelming probability $1-2^{-\Omega(k)}$, $(e(\widehat{g}_{0,1}, \widehat{h}^*), \ldots, e(\widehat{g}_{0,B}, \widehat{h}^*))$ is distributed uniformly over $\mathbb{G}_T^B$ when sampling $\widehat{h}^* \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$.

It is obvious that the ENDSG with B-bounded non-degeneracy almost-tightly implies a B-weak adaptively secure IBE in MIMC setting. We now prove that our prime-order instantiation in Section 4 indeed reaches this stronger version of non-degeneracy.

Lemma 8 *Our prime-order instantiation of ENDSG in Section 4 based on the d-Lin assumption is d-bounded non-degenerated.*

Proof. The proof is just a simple statistical argument extended from the proof for the original non-degeneracy. For $\widehat{\mathbf{s}}_1, \ldots, \widehat{\mathbf{s}}_d \leftarrow \mathbb{Z}_p^d$ and $\widehat{\mathbf{r}} \leftarrow \mathbb{Z}_p^d$, we have that

$$\begin{pmatrix} e(g_1^{\mathbf{E}\widehat{\mathbf{s}}_1}, g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}}) \\ \vdots \\ e(g_1^{\mathbf{E}\widehat{\mathbf{s}}_d}, g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}}) \end{pmatrix} = \begin{pmatrix} e(g_1,g_2)^{\widehat{\mathbf{s}}_1^\top \pi_{\mathrm{M}}(\mathbf{B})^\top \pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}} \\ \vdots \\ e(g_1,g_2)^{\widehat{\mathbf{s}}_d^\top \pi_{\mathrm{M}}(\mathbf{B})^\top \pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}} \end{pmatrix} = \begin{pmatrix} e(g_1,g_2)^{\widehat{\mathbf{s}}_1^\top \widehat{\mathbf{r}}} \\ \vdots \\ e(g_1,g_2)^{\widehat{\mathbf{s}}_d^\top \widehat{\mathbf{r}}} \end{pmatrix} = e(g_1,g_2)^{\begin{pmatrix} \widehat{\mathbf{s}}_1^\top \\ \vdots \\ \widehat{\mathbf{s}}_d^\top \end{pmatrix} \widehat{\mathbf{r}}}.$$

With probability at least $1-1/(p-1)$, the matrix $(\widehat{\mathbf{s}}_1, \ldots, \widehat{\mathbf{s}}_d)^\top$ is full-rank over $\mathbb{Z}_p$, in which case $(\widehat{\mathbf{s}}_1, \ldots, \widehat{\mathbf{s}}_d)^\top \widehat{\mathbf{r}}$ is distributed uniformly over $\mathbb{Z}_p^d$ when picking $\widehat{\mathbf{r}} \leftarrow \mathbb{Z}_p^d$. □

Therefore, when we build our instantiation with parameter $d > 1$, we obtain an IBE with strictly stronger security guarantee which ensures the confidentiality of at most d ciphertexts for single identity. As a special case, if we set $d = 1$ (i.e., the SXDH assumption), the resulting IBE is still weak adaptive secure.

6.2 Computational Non-degeneracy and Full Adaptive Security

The attempt in the previous subsection more or less suggests that it is probably inevitable to introduce additional computational argument in order to achieve fully adaptive security where a single $\widehat{h}^*$ can be paired with polynomially many $\widehat{g}_0$'s without violating the non-degeneracy property.

As a first step, we describe a computational version of non-degeneracy which is essentially similar to the s-BDDH assumption [HKS15]. Following the style of our revised ENDSG (in Section 3), our definition is more general than the s-BDDH assumption.

(Computational non-degeneracy (ND).) For any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,

$$\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{A}}(k, q, q', q'') := \left|\Pr[\mathscr{A}(D, T_0) = 1] - \Pr[\mathscr{A}(D, T_1) = 1]\right|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}_j^* \cdot \widetilde{h}_j^*\right\}_{j\in[q']}, \left\{\widehat{\mathbf{g}}_{j,j'}\right\}_{j\in[q],j'\in[q'']}\right),$$

$$T_0 := \left\{e(\widehat{g}_{0,j,j'}, \widehat{h}_j^{**})\right\}_{j\in[q],j'\in[q'']},\ T_1 := \left\{R_{j,j'}\right\}_{j\in[q],j'\in[q'']}$$

and $\widehat{h}_j^* \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$, $\widetilde{h}_j^* \leftarrow \widetilde{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$, $\widehat{h}_j^{**} \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$, $\widehat{\mathbf{g}}_{j,j'} = \left(\widehat{g}_{0,j,j'}, \widehat{g}_{1,j,j'}, \ldots, \widehat{g}_{n,j,j'}\right) \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$ and $R_{j,j'} \leftarrow \mathbb{G}_T$.

Let $\mathsf{Adv}^{\mathsf{Game}_3}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R)$ and $\mathsf{Adv}^{\mathsf{Game}_4}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R)$ be the advantage function of adversary $\mathscr{A}$ in Game_3 and Game_4, respectively, in the full adaptive security model. We prove Lemma 9 showing that the computational non-degeneracy property implies that Game_3 and Game_4 are indistinguishable. It suffice to conclude that an ENDSG with computational non-degeneracy property tightly implies a fully adaptively secure IBE in MIMC setting.

Lemma 9 (from Game_3 to Game_4) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\left|\mathsf{Adv}^{\mathsf{Game}_3}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R)-\mathsf{Adv}^{\mathsf{Game}_4}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R)\right|\leqslant \mathsf{Adv}^{\mathrm{ND}}_{\mathscr{B}}(k,q_K,q_C,q_R),$$

and $\mathsf{Time}(\mathscr{B})\approx\mathsf{Time}(\mathscr{A})+(q_K+q_Cq_R)\cdot\mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Proof. Given

$$\left(\mathrm{PP},\left\{\widehat{h}_j^*\cdot\widetilde{h}_j^*\right\}_{j\in[q_K]},\left\{\widehat{\mathbf{g}}_{j,j'}\right\}_{j\in[q_C],j'\in[q_R]},\left\{T_{j,j'}\right\}_{j\in[q_C],j'\in[q_R]}\right)$$

where $\widehat{\mathbf{g}}_{j,j'}=\left(\widehat{g}_{0,j,j'},\widehat{g}_{1,j,j'},\ldots,\widehat{g}_{2n,j,j'}\right)$ and each $T_{j,j'}$ is either $e(\widehat{g}_{0,j,j'},\widehat{h}_j^{**})$ or uniformly distributed over $\mathbb{G}_T$, algorithm $\mathscr{B}$ does:

Setup. Sample $\mathrm{MSK}_\iota\leftarrow\mathbb{H}$ (using PP) for all $\iota\in[\lambda]$, and output

$$\left\{\mathrm{MPK}_\iota:=(\mathrm{PP},\mu(\mathrm{MSK}_\iota))\right\}_{\iota\in[\lambda]}.$$

Then algorithm $\mathscr{B}$ picks a secret random bit $\beta\leftarrow\{0,1\}$.

Key extraction queries. On the jth query $(\iota,\mathbf{y})$, sample $\mathbf{h}:=(h_0,h_1,\ldots,h_{2n})\leftarrow\mathsf{SampH}(\mathrm{PP})$. If query $(\iota,\mathbf{y})$ has been made before, say the j'th query ($j'<j$) for the first time, set $\overline{\mathrm{MSK}}=\mathrm{MSK}_\iota\cdot\widehat{h}_{j'}^*\cdot\widetilde{h}_{j'}^*$; otherwise, set $\overline{\mathrm{MSK}}=\mathrm{MSK}_\iota\cdot\widehat{h}_j^*\cdot\widetilde{h}_j^*$. Output

$$\mathrm{SK}:=\left(h_0,\ \overline{\mathrm{MSK}}\cdot\prod_{i=1}^{n}h_{2i-y_i}\right).$$

Here we implicitly set $\widehat{\mathsf{R}}_n(\iota,\mathbf{y}):=\widehat{h}_j^*$ and $\widetilde{\mathsf{R}}_n(\iota,\mathbf{y}):=\widetilde{h}_j^*$ if the query has not been made yet.

Challenge queries. On input $(\iota^*,\mathbf{x}^*,\mathrm{M}_0^*,\mathrm{M}_1^*)$, we let the query be the j'th occurrence of pair $(\iota^*,\mathbf{x}^*)$, which is the jth distinct pair we have met, and $\mathbf{x}^*=(x_1^*,\ldots,x_n^*)\in\{0,1\}^n$. $\mathscr{B}$ samples $(g_0,g_1,\ldots,g_{2n})\leftarrow\mathsf{SampG}(\mathrm{PP})$ and outputs

$$\mathrm{CT}^*=\left(g_0\cdot\widehat{g}_{0,j,j'},\ \prod_{i=1}^{n}(g_{2i-x_i^*}\cdot\widehat{g}_{2i-x_i^*,j,j'}),\ \boxed{e(g_0\cdot\widehat{g}_{0,j,j'},\mathrm{MSK}_{\iota^*})\cdot T_{j,j'}\cdot\mathrm{M}_\beta^*}\right).$$

Here we implicitly set $\widehat{\mathsf{R}}_n(\iota^*,\mathbf{x}^*):=\widehat{h}_j^{**}$.

Guess. $\mathscr{B}$ outputs 1 if $\mathscr{A}$'s guess equals β, and outputs 0 in the other case.

Analysis. Observe that, if $T_{j,j'}=e(\widehat{g}_{0,j,j'},\widehat{h}_j^{**})$, then the boxed term equals $e(g_0\cdot\widehat{g}_{0,j,j'},\mathrm{MSK}_{\iota^*})\cdot e(\widehat{g}_{0,j,j'},\widehat{h}_j^{**})\cdot\mathrm{M}_\beta^*$, the simulation is identical to Game_3; otherwise, if all $T_{j,j'}$ are uniformly distributed over $\mathbb{G}_T$, then the boxed term is independently and uniformly distributed over $\mathbb{G}_T$ and the simulation is identical to Game_4. Therefore we may conclude that $\left|\mathsf{Adv}^{\mathsf{Game}_3}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R)-\mathsf{Adv}^{\mathsf{Game}_4}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R)\right|\leqslant\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{B}}(k,q_K,q_C,q_R)$. □

6.3 Computational Non-degeneracy from d-Linear Assumption

We now prove that the prime-order instantiation proposed in Section 4 has realized the computational non-degeneracy. And this immediately implies that the concrete IBE scheme shown in Section 5 is fully adaptively secure in MIMC setting with almost-tight reduction.

We rewrite the ND advantage function $\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{A}}(k,q,q',q'')$ as follows:

$$\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{A}}(k,q,q',q''):=\left|\Pr[\mathscr{A}(D,T_0)=1]-\Pr[\mathscr{A}(D,T_1)=1]\right|,$$

where

$$D:=\left(\mathrm{PP},\left\{\widehat{h}_j^*\cdot\widetilde{h}_j^*\right\}_{j\in[q']},\left\{\widehat{\mathbf{g}}_{j,j'}\right\}_{j\in[q],j'\in[q'']}\right),$$

$$T_0 := \left\{e(\widehat{g}_{0,j,j'}, \widehat{h}_j^{**})\right\}_{j\in[q],j'\in[q'']},\ T_1 := \left\{e(\widehat{g}_{0,j,j'}, \widehat{h}_j^{**})\cdot \widehat{R}_{j,j'}\right\}_{j\in[q],j'\in[q'']}$$

and

$$\begin{aligned}
\mathrm{PP} &:= \begin{pmatrix} g_1^{\pi_{\mathrm{L}}(\mathbf{B})}, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_1)}, & \cdots, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_n)} \\ g_2^{\mathbf{B}^*\mathbf{R}}, & g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}}, & \cdots, & g_2^{\mathbf{B}^*\mathbf{A}_n^\top\mathbf{R}} \end{pmatrix};\\
\widehat{h}_j^* \cdot \widetilde{h}_j^* &:= g_2^{\mathbf{B}^*\begin{pmatrix}\mathbf{0}_d\\ \widetilde{\mathbf{r}}'_j\\ \widetilde{\mathbf{r}}_j\end{pmatrix}};\\
\widehat{\mathbf{g}}_{j,j'} &:= \left(g_1^{\pi_{\mathrm{M}}(\mathbf{B})\widehat{\mathbf{s}}_{j,j'}}, g_1^{\pi_{\mathrm{M}}(\mathbf{BA}_1)\widehat{\mathbf{s}}_{j,j'}}, \ldots, g_1^{\pi_{\mathrm{M}}(\mathbf{BA}_n)\widehat{\mathbf{s}}_{j,j'}}\right);\\
e(\widehat{g}_{0,j,j'}, \widehat{h}_j^{**}) &:= e(g_1^{\pi_{\mathrm{M}}(\mathbf{B})\widehat{\mathbf{s}}_{j,j'}}, g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}_j}) = e(g_1,g_2)^{\widehat{\mathbf{s}}_{j,j'}^\top\widehat{\mathbf{r}}_j};\\
\widehat{R}_{j,j'} &:= e(g_1,g_2)^{\widehat{\gamma}_{j,j'}};
\end{aligned}$$

where $\widehat{\mathbf{r}}'_j, \widetilde{\mathbf{r}}'_j, \widehat{\mathbf{r}}_j, \widehat{\mathbf{s}}_{j,j'} \leftarrow \mathbb{Z}_p^d$ and $\widehat{\gamma}_{j,j'} \leftarrow \mathbb{Z}_p$.

Lemma 10 ($(d,1,qq'')$-LLin ⇒ ND) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{A}}(k,q,q',q'') \leqslant \mathsf{Adv}^{(d,1,qq'')\text{-LLin}}_{\mathscr{B}}(k),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (qq''+q')d^2\cdot\mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Overview of the Proof. From the observation that all $\widehat{h}_j^{**} = g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}_j}$ are independently distributed and will never be given to $\mathscr{A}$ individually, we essentially prove a stronger result:

"Given D, $g_1^{\widehat{\mathbf{s}}_{j,j'}^\top\widehat{\mathbf{r}}_j}$ are computationally indistinguishable from $g_1^{\widehat{\mathbf{s}}_{j,j'}^\top\widehat{\mathbf{r}}_j+\widehat{\gamma}_{j,j'}}$."

It is direct to based the pseudo-randomness of the challenge terms on the (d,q,q'')-LLin assumption. However the assumption is reduced to d-Lin assumption with reduction loss $\mathscr{O}(q)$. In order to obtain a tight reduction, we further rewrite the challenge term as

$$g_1^{\widehat{\mathbf{s}}_{j,j'}^\top\widehat{\mathbf{r}}_j} = g_1^{\widehat{\mathbf{s}}_{j,j'}^\top\mathbf{V}^\top\bar{\mathbf{r}}_j} = g_1^{\bar{\mathbf{r}}_j^\top\mathbf{V}\widehat{\mathbf{s}}_{j,j'}}$$

where $\mathbf{V}$ is a $(d+1)\times d$ matrix over $\mathbb{Z}_p$ of rank d and $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^{d+1}$. Clearly, we implicitly define $\widehat{\mathbf{r}}_j := \mathbf{V}^\top\bar{\mathbf{r}}_j$. Since the matrix $\mathbf{V}$ is shared by all $\widehat{\mathbf{r}}_j$'s in challenge terms, we could now deal with polynomially many distinct $\widehat{\mathbf{r}}_j$'s *uniformly* which results in a proof with constant security loss.

Proof. Given an instance of $(d,1,qq'')$-LLin problem (i.e., set $\ell=1$ and $q=qq''$)

$$\left(g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, \left\{g_1^{b_i}\right\}_{i\in[d]}, \left\{g_1^{a_1 s_{1,j,j'}}, \ldots, g_1^{a_d s_{d,j,j'}}\right\}_{j\in[q],j'\in[q'']}, \left\{g_1^{b_1 s_{1,j,j'}+\cdots+b_d s_{d,j,j'}+s_{d+1,j,j'}}\right\}_{j\in[q],j'\in[q'']}\right)$$

as input where all $s_{d+1,j,j'}$ are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathscr{B}$ works as follows:

Programming $\widehat{\mathbf{s}}_{j,j'}$ for $j\in[q], j'\in[q']$. Adversary $\mathscr{B}$ implicitly sets $\widehat{\mathbf{s}}_{j,j'} = (s_{1,j,j'}, \ldots, s_{d,j,j'})^\top$.

Programming $\mathbf{B}, \mathbf{B}^*, \mathbf{A}_1, \ldots, \mathbf{A}_n, \mathbf{R}$. Define $\mathbf{W}\in\mathbb{Z}_p^{3d\times 3d}$ as

$$\mathbf{W} := \left(\begin{array}{ccc|ccc|ccc} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ \hline & & & a_1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & a_d & & & \\ \hline & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{array}\right)$$

and set $\mathbf{W}^* := (\mathbf{W}^{-1})^\top$. Sample $\bar{\mathbf{B}}, \bar{\mathbf{R}} \leftarrow \mathrm{GL}_{3d}(\mathbb{Z}_p)$ and set $\bar{\mathbf{B}}^* := (\bar{\mathbf{B}}^{-1})^\top$. Also sample $\bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_n \leftarrow \mathbb{Z}_p^{3d\times 3d}$, and implicitly set

$$(\mathbf{B}, \mathbf{B}^*) := (\bar{\mathbf{B}}\mathbf{W}, \bar{\mathbf{B}}^*\mathbf{W}^*), \quad \mathbf{R} := \mathbf{W}^\top \bar{\mathbf{R}}, \quad \mathbf{A}_i := \mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W},$$

for $i \in [n]$. Observe that $\mathbf{B}, \mathbf{B}^*, \mathbf{R}$ and all $\mathbf{A}_i$ are distributed properly, and we have

$$\mathbf{B}\mathbf{A}_i = \bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}, \quad \mathbf{B}^*\mathbf{R} = \bar{\mathbf{B}}^*\bar{\mathbf{R}}, \quad \mathbf{B}^*\mathbf{A}_i^\top\mathbf{R} = \bar{\mathbf{B}}^*\bar{\mathbf{A}}_i^\top\bar{\mathbf{R}}.$$

Simulating PP. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\pi_{\mathrm{L}}(\mathbf{B})} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{B}}\mathbf{W})} = g_1^{\bar{\mathbf{B}}\pi_{\mathrm{L}}(\mathbf{W})} \quad \text{and} \quad g_1^{\pi_{\mathrm{L}}(\mathbf{B}\mathbf{A}_i)} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W})} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\pi_{\mathrm{L}}(\mathbf{W})},$$
$$g_2^{\mathbf{B}^*\mathbf{R}} = g_2^{\bar{\mathbf{B}}^*\bar{\mathbf{R}}} \quad \text{and} \quad g_2^{\mathbf{B}^*\mathbf{A}_i^\top\mathbf{R}} = g_2^{\bar{\mathbf{B}}^*\bar{\mathbf{A}}_i^\top\bar{\mathbf{R}}},$$

for $i \in [n]$ using the knowledge of $\pi_{\mathrm{L}}(\mathbf{W})$ and $\bar{\mathbf{B}}, \bar{\mathbf{B}}^*, \bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_n, \bar{\mathbf{R}}$.

Simulating $\widehat{h}_j^* \cdot \widetilde{h}_j^*$ for $j \in [q']$. It is not hard to compute $\mathbf{W}^* \in \mathbb{Z}_p^{3d\times 3d}$ as

$$\mathbf{W}^* := \left(\begin{array}{ccc|ccc} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ \hline & & & a_1^{-1} & & \\ & & & & \ddots & \\ & & & & & a_d^{-1} \\ \hline & & & 1 & & \\ & & & & \ddots & \\ & & & & & 1 \end{array}\right).$$

For all $j \in [q+q']$, we sample $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^{2d}$ and implicitly set

$$\begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j \end{pmatrix} = (\mathbf{W}^*)^{-1} \begin{pmatrix} \mathbf{0}_d \\ \bar{\mathbf{r}}_j \end{pmatrix} = \mathbf{W}^\top \begin{pmatrix} \mathbf{0}_d \\ \bar{\mathbf{r}}_j \end{pmatrix}.$$

Since the right-bottom $2d \times 2d$ sub-matrix of $\mathbf{W}^*$ is full-rank with overwhelming probability, both $\widehat{\mathbf{r}}_j$ and $\widetilde{\mathbf{r}}_j$ are distributed properly and $\mathscr{B}$ can simulate

$$\widehat{h}_j^* \cdot \widetilde{h}_j^* = g_2^{\mathbf{B}^*\begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j \end{pmatrix}} = g_2^{\bar{\mathbf{B}}^*\mathbf{W}^*\begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j \end{pmatrix}} = g_2^{\bar{\mathbf{B}}^*\begin{pmatrix} \mathbf{0}_d \\ \bar{\mathbf{r}}_j \end{pmatrix}}$$

using the knowledge of $\bar{\mathbf{B}}^*$ and $\bar{\mathbf{r}}_j$.

Simulating $\widehat{\mathbf{g}}_{j,j'}$ for $j \in [q], j' \in [q']$. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\mathbf{B}\begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{s}}_{j,j'} \\ \mathbf{0}_d \end{pmatrix}} = g_1^{\bar{\mathbf{B}}\mathbf{W}\begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{s}}_{j,j'} \\ \mathbf{0}_d \end{pmatrix}} \quad \text{and} \quad g_1^{\mathbf{B}\mathbf{A}_i\begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{s}}_{j,j'} \\ \mathbf{0}_d \end{pmatrix}} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}\begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{s}}_{j,j'} \\ \mathbf{0}_d \end{pmatrix}}$$

for $i \in [n]$ using the knowledge of $\bar{\mathbf{B}}, \bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_n$ and

$$g_1^{\mathbf{W}\begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{s}}_{j,j'} \\ \mathbf{0}_d \end{pmatrix}} = g_1^{\begin{pmatrix} \mathbf{0}_d \\ a_1 s_{1,j,j'} \\ \vdots \\ a_d s_{d,j,j'} \\ \mathbf{0}_d \end{pmatrix}}.$$

Simulating the challenge. Define an additional matrix $\mathbf{V} \in \mathbb{Z}_p^{(d+1)\times d}$ of rank d as

$$\mathbf{V} := \begin{pmatrix} a_1 & & \\ & \ddots & \\ & & a_d \\ b_1 & \cdots & b_d \end{pmatrix}.$$

For all $j \in [q]$, algorithm $\mathscr{B}$ samples $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^{d+1}$ and implicitly set $\widehat{\mathbf{r}}_j^\top := \bar{\mathbf{r}}_j^\top \mathbf{V}$. Algorithm $\mathscr{B}$ simulates

$$g_1^{\widehat{\mathbf{r}}_j^\top \widehat{\mathbf{s}}_{j,j'} + \widehat{\gamma}_{j,j'}} = g_1^{\bar{\mathbf{r}}_j^\top \begin{pmatrix} a_1 s_{1,j,j'} \\ \vdots \\ a_d s_{d,j,j'} \\ b_1 s_{1,j,j'} + \cdots + b_d s_{d,j,j'} + s_{d+1,j,j'} \end{pmatrix}}$$

using the knowledge of $\left\{ g_1^{a_1 s_{1,j,j'}}, \ldots, g_1^{a_d s_{d,j,j'}}, g_1^{b_1 s_{1,j,j'} + \cdots + b_d s_{d,j,j'} + s_{d+1,j,j'}} \right\}$, and outputs $e(g_1^{\widehat{\mathbf{r}}_j^\top \widehat{\mathbf{s}}_{j,j'} + \widehat{\gamma}_{j,j'}}, g_2)$.

Analysis. Observe that, if $s_{d+1,j,j'} = 0$, then the output challenge is distributed as T_0 where $\widehat{\gamma}_{j,j'} = 0$; if $s_{d+1,j,j'} \leftarrow \mathbb{Z}_p^*$, then the output challenge is distributed as

$$e(g_1^{\bar{\mathbf{r}}_j^\top (\mathbf{V}\widehat{\mathbf{s}}_{j,j'} + \mathbf{e}_{d+1} s_{d+1,j,j'})}, g_2) = e(g_1, g_2)^{\widehat{\mathbf{s}}_{j,j'}^\top \widehat{\mathbf{r}}_j} \cdot \boxed{e(g_1, g_2)^{s_{d+1,j,j'} \mathbf{e}_{d+1}^\top \bar{\mathbf{r}}_j}}$$

which is identical to T_1 where $\widehat{\gamma}_{j,j'} := s_{d+1,j,j'} \mathbf{e}_{d+1}^\top \bar{\mathbf{r}}_j$ (in the box) is uniformly distributed over $\mathbb{Z}_p$. Therefore we may conclude that $\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{A}}(k, q, q', q'') \leqslant \mathsf{Adv}^{(d,1,qq'')\text{-LLin}}_{\mathscr{B}}(k)$. □

Corollary 4 (d-Lin ⇒ ND) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{A}}(k, q, q', q'') \leqslant \mathsf{Adv}^{d\text{-Lin}}_{\mathscr{B}}(k) + 1/(p-1),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (qq'' + q')d^2 \cdot \mathsf{poly}(k, n)$ *where* $\mathsf{poly}(k, n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

7 Fine-Tuning Extended Nested Dual System Groups from Section 3

In this section, we begin our work on exploring more efficient instantiation of ENDSG leading to more efficient IBE. Our fine-tuning is based on the ENDSG shown in Section 3 equipped with computational non-degeneracy defined in Section 6. We show in Appendix C that the ENDSG after fine-tuning tightly implies an IBE in MIMC setting by showing the construction and the sketch of the proof.

Syntax. The fine-tuned ENDSG consists of eight p.p.t. algorithms defined as follows:

- $\mathsf{SampP}(1^k, n)$: Output (1) PP containing (a) group description $(\mathbb{G}, \mathbb{H}, \mathbb{G}_T)$ and an admissible bilinear map $e : \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$; (b) an efficient linear map μ defined on $\mathbb{H}$; (c) an efficient sampler for $\mathbb{H}$ and $\mathbb{Z}_{\mathrm{ord}(\mathbb{H})}$, respectively; (d) public parameters for SampG; (2) HP containing parameters for SampH; (3) SP containing secret parameters for $\widehat{\mathsf{SampG}}$, $\widetilde{\mathsf{SampG}}$, $\widehat{\mathsf{SampH}}^*$ and $\widetilde{\mathsf{SampH}}^*$.
- SampGT: $\mathrm{Im}(\mu) \to \mathbb{G}_T$.
- $\mathsf{SampG}(\mathrm{PP})$: Output $\mathbf{g} = (g_0, g_1, \ldots, g_n) \in \mathbb{G}^{n+1}$.
- $\mathsf{SampH}(\mathrm{PP}, \mathrm{HP})$: Output $\mathbf{h} = (h_0, h_1, \ldots, h_n) \in \mathbb{H}^{n+1}$.
- $\widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$: Output $\widehat{\mathbf{g}} = (\widehat{g}_0, \widehat{g}_1, \ldots, \widehat{g}_n) \in \mathbb{G}^{n+1}$.
- $\widetilde{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$: Output $\widetilde{\mathbf{g}} = (\widetilde{g}_0, \widetilde{g}_1, \ldots, \widetilde{g}_n) \in \mathbb{G}^{n+1}$.
- $\widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$: Output $\widehat{h}^* \in \mathbb{H}$.
- $\widetilde{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$: Output $\widetilde{h}^* \in \mathbb{H}$.

Correctness and Security. For all $k, n \in \mathbb{Z}^+$ and all $(\mathrm{PP}, \mathrm{HP}, \mathrm{SP}) \in [\mathsf{SampP}(1^k, n)]$, the *projective*, *associative*, *orthogonality*, and $\mathbb{H}$-*subgroup* requirement are identical to those defined in Section 3.

(Left subgroup indistinguishability 1 (LS1).) For any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,

$$\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k, q, q') := \left| \Pr[\mathscr{A}(D, T_0) = 1] - \Pr[\mathscr{A}(D, T_1) = 1] \right|,$$

where

$$D := \left(\mathrm{PP}, \left\{ \mathbf{h}_j \right\}_{j \in [q']} \right), \quad T_0 := \left\{ \mathbf{g}_j \right\}_{j \in [q]}, \quad T_1 := \left\{ \mathbf{g}_j \cdot \boxed{\widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j} \right\}_{j \in [q]}$$

and $\mathbf{g}_j \leftarrow \mathsf{SampG}(\mathrm{PP})$, $\widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$, $\widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$, and $\mathbf{h}_j \leftarrow \mathsf{SampH}(\mathrm{PP}, \mathrm{HP})$.

(Left subgroup indistinguishability 2 (LS2).) For any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,

$$\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k,q,q') := \left|\Pr[\mathscr{A}(D,T_0)=1]-\Pr[\mathscr{A}(D,T_1)=1]\right|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}^*_j\cdot\widetilde{h}^*_j\right\}_{j\in[q+q']}, \left\{\mathbf{g}'_j\cdot\widehat{\mathbf{g}}'_j\cdot\widetilde{\mathbf{g}}'_j\right\}_{j\in[q]}, \left\{\mathbf{h}_j\right\}_{j\in[q']}\right),$$

$$T_0 := \left\{\mathbf{g}_j\cdot\boxed{\widehat{\mathbf{g}}_j}\cdot\widetilde{\mathbf{g}}_j\right\}_{j\in[q]},\ T_1 := \left\{\mathbf{g}_j\cdot\widetilde{\mathbf{g}}_j\right\}_{j\in[q]},$$

and $\widehat{h}^*_j \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\widetilde{h}^*_j \leftarrow \widetilde{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\mathbf{g}'_j \leftarrow \mathsf{SampG}(\mathrm{PP})$, $\widehat{\mathbf{g}}'_j \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, $\widetilde{\mathbf{g}}'_j \leftarrow \widetilde{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, $\mathbf{g}_j \leftarrow \mathsf{SampG}(\mathrm{PP})$, $\widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, $\widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, and $\mathbf{h}_j \leftarrow \mathsf{SampH}(\mathrm{PP},\mathrm{HP})$.

(Left subgroup indistinguishability 3 (LS3).) For any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,

$$\mathsf{Adv}^{\mathrm{LS3}}_{\mathscr{A}}(k,q,q') := \left|\Pr[\mathscr{A}(D,T_0)=1]-\Pr[\mathscr{A}(D,T_1)=1]\right|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}^*_j\cdot\widetilde{h}^*_j\right\}_{j\in[q+q']}, \left\{\mathbf{g}'_j\cdot\widetilde{\mathbf{g}}'_j\right\}_{j\in[q]}, \left\{\mathbf{h}_j\right\}_{j\in[q']}\right),$$

$$T_0 := \left\{\mathbf{g}_j\cdot\widehat{\mathbf{g}}_j\cdot\boxed{\widetilde{\mathbf{g}}_j}\right\}_{j\in[q]},\ T_1 := \left\{\mathbf{g}_j\cdot\widehat{\mathbf{g}}_j\right\}_{j\in[q]},$$

and $\widehat{h}^*_j \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\widetilde{h}^*_j \leftarrow \widetilde{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\mathbf{g}'_j \leftarrow \mathsf{SampG}(\mathrm{PP})$, $\widetilde{\mathbf{g}}'_j \leftarrow \widetilde{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, $\mathbf{g}_j \leftarrow \mathsf{SampG}(\mathrm{PP})$, $\widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, $\widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, and $\mathbf{h}_j \leftarrow \mathsf{SampH}(\mathrm{PP},\mathrm{HP})$.

(Nested-hiding indistinguishability (NH).) For all $\eta \in [\lfloor n/2\rfloor]$ and any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,

$$\mathsf{Adv}^{\mathrm{NH}(\eta)}_{\mathscr{A}}(k,q,q') := \left|\Pr[\mathscr{A}(D,T_0)=1]-\Pr[\mathscr{A}(D,T_1)=1]\right|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}^*_j\right\}_{j\in[q+q']}, \left\{\widetilde{h}^*_j\right\}_{j\in[q+q']}, \left\{(\widehat{\mathbf{g}}_j)_{-(2\eta-1)}\right\}_{j\in[q]}, \left\{(\widetilde{\mathbf{g}}_j)_{-2\eta}\right\}_{j\in[q]}, \left\{\mathbf{h}'_j\right\}_{j\in[q']}\right),$$

$$T_0 := \left\{\mathbf{h}_j\right\}_{j\in[q']},\quad T_1 := \left\{\mathbf{h}_j\cdot\boxed{(\widehat{h}^{**}_j)^{\mathbf{e}_{2\eta-1}}\cdot(\widetilde{h}^{**}_j)^{\mathbf{e}_{2\eta}}}\right\}_{j\in[q']}$$

and $\widehat{h}^*_j \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\widetilde{h}^*_j \leftarrow \widetilde{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, $\widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, $\mathbf{h}_j \leftarrow \mathsf{SampH}(\mathrm{PP},\mathrm{HP})$, $\widehat{h}^{**}_j \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\widetilde{h}^{**}_j \leftarrow \widetilde{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, and $\mathbf{h}'_j \leftarrow \mathsf{SampH}(\mathrm{PP},\mathrm{HP})$. We further define

$$\mathsf{Adv}^{\mathrm{NH}}_{\mathscr{A}}(k,q,q') := \max_{\eta\in[\lfloor n/2\rfloor]}\left\{\mathsf{Adv}^{\mathrm{NH}(\eta)}_{\mathscr{A}}(k,q,q')\right\}.$$

(Computational non-degeneracy (ND).) For any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,

$$\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{A}}(k,q,q',q'') := \left|\Pr[\mathscr{A}(D,T_0)=1]-\Pr[\mathscr{A}(D,T_1)=1]\right|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}^*_j\cdot\widetilde{h}^*_j,\ \mathbf{h}_j\right\}_{j\in[q']}, \left\{\widehat{\mathbf{g}}_{j,j'}\cdot\widetilde{\mathbf{g}}_{j,j'}\right\}_{j\in[q],j'\in[q'']}\right),$$

$$T_0 := \left\{e(\widehat{g}_{0,j,j'}\cdot\widetilde{g}_{0,j,j'},\widehat{h}^{**}_j\cdot\widetilde{h}^{**}_j)\right\}_{j\in[q],j'\in[q'']},\quad T_1 := \left\{R_{j,j'}\right\}_{j\in[q],j'\in[q'']}.$$

and $\widehat{h}^*_j \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\widetilde{h}^*_j \leftarrow \widetilde{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\mathbf{h}_j \leftarrow \mathsf{SampH}(\mathrm{PP},\mathrm{HP})$, $\widehat{h}^{**}_j \leftarrow \widehat{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\widetilde{h}^{**}_j \leftarrow \widetilde{\mathsf{SampH}}^*(\mathrm{PP},\mathrm{SP})$, $\widehat{\mathbf{g}}_{j,j'} = \left(\widehat{g}_{0,j,j'},\widehat{g}_{1,j,j'},\ldots,\widehat{g}_{n,j,j'}\right) \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, $\widetilde{\mathbf{g}}_{j,j'} = \left(\widetilde{g}_{0,j,j'},\widetilde{g}_{1,j,j'},\ldots,\widetilde{g}_{n,j,j'}\right) \leftarrow \widetilde{\mathsf{SampG}}(\mathrm{PP},\mathrm{SP})$, and $R_{j,j'} \leftarrow G_T$.

8 Instantiating ENDSG from d-Linear Assumption with Auxiliary Input

The section present an instantiation of fine-tuned ENDSG in Section 7 using prime-order bilinear groups. This yields an almost-tight IBE in MIMC setting using prime-order bilinear groups following the generic construction shown in Appendix C. We describe this IBE and its variants in Section 9.

8.1 d-Linear Assumption with Auxiliary Input

We assume a prime-order bilinear group generator $\mathsf{GrpGen}(1^k)$ as defined in Section 4, which takes security parameter 1^k as input and outputs group description $\mathscr{G} := (p, G_1, G_2, G_T, e)$. The *$d$-linear assumption in G_1 with auxiliary input in G_2* (d-LinAI) is defined as follows, the analogous assumption in G_2 can be defined by exchanging the role of G_1 and G_2. We prove the assumption holds in the generic model [Sho97] in Section A.4. *Note that we always let d be an even positive integer.*

Assumption 3 (d-Linear Assumption in G_1 with Auxiliary Input in G_2) *For any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,*

$$\mathsf{Adv}^{d\text{-LinAI}}_{\mathscr{A}}(k) := \left|\Pr[\mathscr{A}(D, \mathrm{AUX}, T_0) = 1] - \Pr[\mathscr{A}(D, \mathrm{AUX}, T_1) = 1]\right|,$$

where

$$D := \left(\mathscr{G}, g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, g_1^{a_{d+1}}, g_1^{a_1 s_1}, \ldots, g_1^{a_d s_d}\right)$$

$$\mathrm{AUX} := \left(g_2^{a a_1^{-1} a_{d+1}}, \ldots, g_2^{a a_{d/2}^{-1} a_{d+1}}, g_2^{a}\right)$$

$$T_0 := g_1^{a_{d+1}(s_1+\cdots+s_d)}, \quad T_1 := g_1^{a_{d+1}(s_1+\cdots+s_d)+\boxed{s_{d+1}}}$$

and $\mathscr{G} := (p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$, $s_1, \ldots, s_d \leftarrow \mathbb{Z}_p$, $a_1, \ldots, a_d, a_{d+1}, s_{d+1} \leftarrow \mathbb{Z}_p^*$ *and* $a := a_1 \cdots a_{d/2}$.

As we have done in Section 4, we also define an natural extension of the above assumption, i.e., (d, ℓ, q)-*Lifted Linear Assumption in G_1 with Auxiliary Input in G_2* ((d, ℓ, q)-LLinAI). The relation between them is shown in Lemma 11. We point out that this assumption implies the (d, ℓ, q)-LLin assumption.

Assumption 4 ((d, ℓ, q)-Lifted Linear Assumption in G_1 with Auxiliary Input in G_2) *For any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,*

$$\mathsf{Adv}^{(d,\ell,q)\text{-LLinAI}}_{\mathscr{A}}(k) := \left|\Pr[\mathscr{A}(D, \mathrm{AUX}, T_0) = 1] - \Pr[\mathscr{A}(D, \mathrm{AUX}, T_1) = 1]\right|,$$

where

$$D := \left(\mathscr{G}, g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, \left\{g_1^{b_{i,j}}\right\}_{i\in[\ell], j\in[d]}, \left\{g_1^{a_1 s_{1,j}}, \ldots, g_1^{a_d s_{d,j}}\right\}_{j\in[q]}\right),$$

$$\mathrm{AUX} := \left(\left\{g_2^{a a_j^{-1} b_{i,j}}\right\}_{i\in[\ell], j\in[d/2]}, g_2^{a}\right)$$

$$T_0 := \left\{g_1^{b_{i,1}s_{1,j}+\cdots+b_{i,d}s_{d,j}}\right\}_{i\in[\ell], j\in[q]}, \quad T_1 := \left\{g_1^{b_{i,1}s_{1,j}+\cdots+b_{i,d}s_{d,j}+\boxed{s_{d+i,j}}}\right\}_{i\in[\ell], j\in[q]}$$

and $\mathscr{G} := (p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$, $a_1, \ldots, a_d, b_{i,j} \leftarrow \mathbb{Z}_p^*$, $a := a_1 \cdots a_{d/2}$, $s_{1,j}, \ldots, s_{d,j} \leftarrow \mathbb{Z}_p$, $s_{d+i,j} \leftarrow \mathbb{Z}_p^*$.

Lemma 11 (d-LinAI $\Rightarrow$ (d, ℓ, q)-LLinAI) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{(d,\ell,q)\text{-LLinAI}}_{\mathscr{A}}(k) \leqslant \ell \cdot \mathsf{Adv}^{d\text{-LinAI}}_{\mathscr{B}}(k) + 1/(p-1),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + \ell^2 d \cdot \mathsf{poly}(k)$ *where* $\mathsf{poly}(k)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Proof. The proof is similar to that for Lemma 1. We first prove that, for any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ with $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + \ell d \cdot \mathsf{poly}(k)$ such that

$$\mathsf{Adv}^{(d,\ell,1)\text{-LLinAI}}_{\mathscr{A}}(k) \leqslant \mathsf{Adv}^{d\text{-LinAI}}_{\mathscr{B}}(k).$$

We can deduce the lemma by applying the idea of Lemma 1 in [EHK+13].

We now give the proof of the claim. For simplicity, we discard the subscript j related to parameter q when considering $(d, \ell, 1)$-LLinAI. Given a d-LinAI problem instance

$$\left(g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, g_1^{a_{d+1}}, g_1^{a_1 s_1}, \ldots, g_1^{a_d s_d}, g_2^{a a_1^{-1} a_{d+1}}, \ldots, g_2^{a a_{d/2}^{-1} a_{d+1}}, g_2^{a}, g_1^{a_{d+1}(s_1+\cdots+s_d)+s_{d+1}}\right)$$

as input where s_{d+1} are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathscr{B}$ works as follows:

Simulating $a_1, \ldots, a_d$ and $s_1, \ldots, s_d$. Set $a_i := a_i$ and implicitly define $s_i := s_i$ for all $i \in [d]$.

Simulating $g_1^{b_{i,j}}$ for $i \in [\ell],\ j \in [d]$. Sample $\gamma_i, \delta_{i,j} \leftarrow \mathbb{Z}_p^*$ for all $i \in [\ell]$ and $j \in [d]$ and simulate

$$g_1^{b_{i,j}} := \left(g_1^{a_{d+1}}\right)^{\gamma_i} \cdot \left(g_1^{a_j}\right)^{\delta_{i,j}}.$$

Simulating Aux. Algorithm $\mathscr{B}$ can simulate g_2^a directly. Observe that, for all $i \in [\ell], j \in [d/2]$, we have

$$aa_j^{-1}b_{i,j} = aa_j^{-1}(\gamma_i a_{d+1} + \delta_{i,j}a_j) = \gamma_i aa_j^{-1}a_{d+1} + \delta_{i,j}a.$$

Therefore we can simulate

$$g_2^{aa_j^{-1}b_{i,j}} = \left(g_2^{aa_j^{-1}a_{d+1}}\right)^{\gamma_i} \cdot \left(g_2^a\right)^{\delta_{i,j}}.$$

Simulating the Challenge. Similar to Lemma 11, we can simulate the challenge as

$$g_1^{b_{i,1}s_1+\cdots+b_{i,d}s_d+s_{d+i}} := \left(g_1^{a_{d+1}(s_1+\cdots+s_d)+s_{d+1}}\right)^{\gamma_i} \cdot \left(g_1^{a_1 s_1}\right)^{\delta_{i,1}} \cdots \left(g_1^{a_d s_d}\right)^{\delta_{i,d}}.$$

Note that Aux is simulated perfectly and also reveal no information on γ_i. Therefore we have prove the claim following the analysis of Lemma 1. □

8.2 Construction

The construction is based on Okamoto and Takashima's DPVS [OT08, OT09, LOS+10] and Chen, Gay and Wee's new method for randomizing the basis [CGW15]. We let $\pi_{\mathrm{L}}(\cdot)$, $\pi_{\mathrm{M}}(\cdot)$, and $\pi_{\mathrm{R}}(\cdot)$ be functions mapping from a $2d \times 2d$ matrix to its left-most d columns, the next $d/2$ columns, and the right-most $d/2$ columns, respectively. *Note that we always consider d as an even positive integer.* Algorithms of the fine-tuned ENDSG are shown as follows.

- $\mathsf{SampP}(1^k, n)$: Generate $(p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$ and define $(\mathbb{G}, \mathbb{H}, \mathbb{G}_T, e) := (G_1^{2d}, G_2^{2d}, G_T, e)$. Sample $\mathbf{D} \leftarrow \mathrm{GL}_{2d}(\mathbb{Z}_p)$ and set $\mathbf{D}^* := (\mathbf{D}^{-1})^\top$. Define

$$\begin{array}{lll} \mathbf{A} = \pi_{\mathrm{L}}(\mathbf{D}), & \widehat{\mathbf{A}} = \pi_{\mathrm{M}}(\mathbf{D}), & \widetilde{\mathbf{A}} = \pi_{\mathrm{R}}(\mathbf{D}); \\ \mathbf{B} = \pi_{\mathrm{L}}(\mathbf{D}^*), & \widehat{\mathbf{B}} = \pi_{\mathrm{M}}(\mathbf{D}^*), & \widetilde{\mathbf{B}} = \pi_{\mathrm{R}}(\mathbf{D}^*); \end{array}$$

  Define $\mu(g_2^{\mathbf{k}}) := e(g_1^{\mathbf{A}}, g_2^{\mathbf{k}}) = e(g_1, g_2)^{\mathbf{A}^\top \mathbf{k}}$ for all $\mathbf{k} \in \mathbb{Z}_p^{2d}$. Sample $\mathbf{W}_1, \ldots, \mathbf{W}_n \leftarrow \mathbb{Z}_p^{2d \times 2d}$ and output

$$\begin{aligned} \mathrm{PP} &:= \left(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \mu; g_1^{\mathbf{A}}, g_1^{\mathbf{W}_1^\top \mathbf{A}}, \ldots, g_1^{\mathbf{W}_n^\top \mathbf{A}}\right) \\ \mathrm{HP} &:= \left(g_2^{\mathbf{B}}, g_2^{\mathbf{W}_1 \mathbf{B}}, \ldots, g_2^{\mathbf{W}_n \mathbf{B}}\right) \\ \mathrm{SP} &:= \begin{pmatrix} g_2^{\widehat{\mathbf{B}}}, & g_1^{\widehat{\mathbf{A}}}, & g_1^{\mathbf{W}_1^\top \widehat{\mathbf{A}}}, & \ldots, & g_1^{\mathbf{W}_n^\top \widehat{\mathbf{A}}} \\ g_2^{\widetilde{\mathbf{B}}}, & g_1^{\widetilde{\mathbf{A}}}, & g_1^{\mathbf{W}_1^\top \widetilde{\mathbf{A}}}, & \ldots, & g_1^{\mathbf{W}_n^\top \widetilde{\mathbf{A}}} \end{pmatrix}. \end{aligned}$$

  We assume PP always contains $\mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \mu$.

- $\mathsf{SampGT}(g_T^{\mathbf{p}})$: Sample $\mathbf{s} \leftarrow \mathbb{Z}_p^d$ and output $g_T^{\mathbf{s}^\top \mathbf{p}} \in G_T$.
- $\mathsf{SampG}(\mathrm{PP})$: Sample $\mathbf{s} \leftarrow \mathbb{Z}_p^d$ and output $\left(g_1^{\mathbf{As}}, g_1^{\mathbf{W}_1^\top \mathbf{As}}, \ldots, g_1^{\mathbf{W}_n^\top \mathbf{As}}\right) \in (G_1^{2d})^{n+1}$.
- $\mathsf{SampH}(\mathrm{PP}, \mathrm{HP})$: Sample $\mathbf{r} \leftarrow \mathbb{Z}_p^d$ and output $\left(g_2^{\mathbf{Br}}, g_2^{\mathbf{W}_1 \mathbf{Br}}, \ldots, g_2^{\mathbf{W}_n \mathbf{Br}}\right) \in (G_2^{2d})^{n+1}$.
- $\widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$: Sample $\widehat{\mathbf{s}} \leftarrow \mathbb{Z}_p^{d/2}$ and output $\left(g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}}, g_1^{\mathbf{W}_1^\top \widehat{\mathbf{A}}\widehat{\mathbf{s}}}, \ldots, g_1^{\mathbf{W}_n^\top \widehat{\mathbf{A}}\widehat{\mathbf{s}}}\right) \in (G_1^{2d})^{n+1}$.
- $\widetilde{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$: Sample $\widetilde{\mathbf{s}} \leftarrow \mathbb{Z}_p^{d/2}$ and output $\left(g_1^{\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}}, g_1^{\mathbf{W}_1^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}}, \ldots, g_1^{\mathbf{W}_n^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}}\right) \in (G_1^{2d})^{n+1}$.
- $\widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$: Sample $\widehat{\mathbf{r}} \leftarrow \mathbb{Z}_p^{d/2}$ and output $g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}} \in G_2^{2d}$.
- $\widetilde{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})$: Sample $\widetilde{\mathbf{r}} \leftarrow \mathbb{Z}_p^{d/2}$ and output $g_2^{\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}} \in G_2^{2d}$.

Correctness and Security. We may check several correctness and security properties as follows:

(Projective.) For all $\mathbf{k} \in \mathbb{Z}_p^{2d}$ and all $\mathbf{s} \in \mathbb{Z}_p^d$, we have that

$$\mathsf{SampGT}(\mu(g_2^{\mathbf{k}}); \mathbf{s}) = e(g_1, g_2)^{\mathbf{s}^\top(\mathbf{A}^\top \mathbf{k})} = e(g_1, g_2)^{(\mathbf{As})^\top \mathbf{k}} = e(g_1^{\mathbf{As}}, g_2^{\mathbf{k}}) = e(\mathsf{SampG}_0(\mathrm{pp}; \mathbf{s}), g_2^{\mathbf{k}}).$$

(Associative.) For all $\mathbf{s} \in \mathbb{Z}_p^d$ and all $\mathbf{r} \in \mathbb{Z}_p^d$, we have that

$$e(g_1^{\mathbf{As}}, g_2^{\mathbf{W}_i \mathbf{Br}}) = e(g_1, g_2)^{\mathbf{s}^\top \mathbf{A}^\top \mathbf{W}_i \mathbf{Br}} = e(g_1, g_2)^{(\mathbf{W}_i^\top \mathbf{As})^\top \mathbf{Br}} = e(g_1^{\mathbf{W}_i^\top \mathbf{As}}, g_2^{\mathbf{Br}}), \quad \forall i \in [n].$$

(Orthogonality.) For all $\widehat{\mathbf{r}}, \widetilde{\mathbf{r}}, \widehat{\mathbf{s}}, \widetilde{\mathbf{s}} \in \mathbb{Z}_p^{d/2}$, we check that

1. $\mu(g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}}) = e(g_1, g_2)^{\mathbf{A}^\top \widehat{\mathbf{B}}\widehat{\mathbf{r}}} = e(g_1, g_2)^{\mathbf{0}_{d \times (d/2)} \widehat{\mathbf{r}}} = (1, \ldots, 1)^\top \in G_T^d$;
2. $\mu(g_2^{\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}}) = e(g_1, g_2)^{\mathbf{A}^\top \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}} = e(g_1, g_2)^{\mathbf{0}_{d \times (d/2)} \widetilde{\mathbf{r}}} = (1, \ldots, 1)^\top \in G_T^d$;
3. $e(g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}}, g_2^{\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}}) = e(g_1, g_2)^{\widehat{\mathbf{s}}^\top \widehat{\mathbf{A}}^\top \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}} = e(g_1, g_2)^{\widehat{\mathbf{s}}^\top \mathbf{0}_{(d/2) \times (d/2)} \widetilde{\mathbf{r}}} = 1_{G_T}$;
4. $e(g_1^{\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}}, g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}}) = e(g_1, g_2)^{\widetilde{\mathbf{s}}^\top \widetilde{\mathbf{A}}^\top \widehat{\mathbf{B}}\widehat{\mathbf{r}}} = e(g_1, g_2)^{\widetilde{\mathbf{s}}^\top \mathbf{0}_{(d/2) \times (d/2)} \widehat{\mathbf{r}}} = 1_{G_T}$.

($\mathbb{H}$-subgroup.) This follows from the fact that $\mathbb{Z}_p^d$ (for algorithm SampH) and $\mathbb{Z}_p^{d/2}$ (for algorithm $\widehat{\mathsf{SampH}}^*$ and $\widetilde{\mathsf{SampH}}^*$) are additive groups.

We check the remaining security properties (LS1, LS2, LS3, NH and ND) in the following subsections. We prove LS2 and LS3 property under the (d, ℓ, q)-LLinAI assumption, while others are still proven under the (d, ℓ, q)-LLin assumption which is further implied by the (d, ℓ, q)-LLinAI assumption.

8.3 Left subgroup indistinguishability 1

We may rewrite the LS1 advantage function $\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k, q, q')$ as follows:

$$\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k, q, q') := \left| \Pr[\mathscr{A}(D, T_0) = 1] - \Pr[\mathscr{A}(D, T_1) = 1] \right|,$$

where

$$D := \left(\mathrm{pp}, \left\{ \mathbf{h}_j \right\}_{j \in [q']} \right), \; T_0 := \left\{ \mathbf{g}_j \right\}_{j \in [q]}, \; T_1 := \left\{ \mathbf{g}_j \cdot \boxed{\widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j} \right\}_{j \in [q]}$$

and

$$\begin{aligned}
\mathrm{pp} &:= \left(g_1^{\mathbf{A}}, g_1^{\mathbf{W}_1^\top \mathbf{A}}, \ldots, g_1^{\mathbf{W}_n^\top \mathbf{A}} \right); \\
\mathbf{h}_j &:= \left(g_2^{\mathbf{Br}_j}, g_2^{\mathbf{W}_1 \mathbf{Br}_j}, \ldots, g_2^{\mathbf{W}_n \mathbf{Br}_j} \right); \\
\mathbf{g}_j &:= \left(g_1^{\mathbf{As}_j}, g_1^{\mathbf{W}_1^\top \mathbf{As}_j}, \ldots, g_1^{\mathbf{W}_n^\top \mathbf{As}_j} \right); \\
\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j &:= \left(g_1^{\mathbf{As}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j}, g_1^{\mathbf{W}_1^\top (\mathbf{As}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j)}, \ldots, g_1^{\mathbf{W}_n^\top (\mathbf{As}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j)} \right);
\end{aligned}$$

for $\mathbf{r}_j, \mathbf{s}_j \leftarrow \mathbb{Z}_p^d$ and $\widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^{d/2}$.

Lemma 12 ((d, d, q)-LLin $\Rightarrow$ LS1) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k, q, q') \leqslant \mathsf{Adv}^{(d,d,q)\text{-LLin}}_{\mathscr{B}}(k),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q + q')d^2 \cdot \mathsf{poly}(k, n)$ *where* $\mathsf{poly}(k, n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Proof. Given an instance of (d, d, q)-LLin problem (i.e., set $\ell = d$)

$$\left(g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, \left\{ g_1^{b_{i,j}} \right\}_{i,j \in [d]}, \left\{ g_1^{a_1 s_{1,j}}, \ldots, g_1^{a_d s_{d,j}} \right\}_{j \in [q]}, \left\{ g_1^{b_{i,1} s_{1,j} + \cdots + b_{i,d} s_{d,j} + s_{d+i,j}} \right\}_{i \in [d], j \in [q]} \right)$$

as input where all $s_{d+i,j}$ are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathscr{B}$ works as follows:

Programming $\mathbf{s}_j$, $\widehat{\mathbf{s}}_j$ and $\widetilde{\mathbf{s}}_j$ for $j \in [q]$. Adversary $\mathscr{B}$ implicitly sets

$$\mathbf{s}_j = \left(s_{1,j}, \ldots, s_{d,j}\right)^\top,\ \widehat{\mathbf{s}}_j = \left(s_{d+1,j}, \ldots, s_{3d/2,j}\right)^\top,\ \widetilde{\mathbf{s}}_j = \left(s_{(3d/2+1),j}, \ldots, s_{2d,j}\right)^\top.$$

Programming $\mathbf{D}, \mathbf{D}^*, \mathbf{W}_1, \cdots, \mathbf{W}_n$. Define $\mathbf{V} \in \mathbb{Z}_p^{2d \times 2d}$ as

$$\mathbf{V} := \left(\begin{array}{ccc|ccc} a_1 & & & & & \\ & \ddots & & & & \\ & & a_d & & & \\ \hline b_{1,1} & \cdots & b_{1,d} & 1 & & \\ \vdots & & \vdots & & \ddots & \\ b_{d/2,1} & \cdots & b_{d/2,d} & & & 1 \\ \hline b_{d/2+1,1} & \cdots & b_{d/2+1,d} & & & \\ \vdots & & \vdots & & & \\ b_{d,1} & \cdots & b_{d,d} & & & \end{array}\right) \left(\begin{array}{ccc} & & \\ & & \\ & & \\ \hline & & \\ & & \\ & & \\ \hline 1 & & \\ & \ddots & \\ & & 1 \end{array}\right).$$

Sample $\bar{\mathbf{D}} \leftarrow \mathrm{GL}_{2d}(\mathbb{Z}_p)$ and let $\bar{\mathbf{D}}^* := (\bar{\mathbf{D}}^{-1})^\top$. Define

$$\mathbf{D} := \bar{\mathbf{D}}\mathbf{V} \ \text{ and } \ \mathbf{D}^* := \bar{\mathbf{D}}^*\mathbf{V}^*.$$

Sample $\mathbf{W}_1, \ldots, \mathbf{W}_n \leftarrow \mathbb{Z}_p^{2d \times 2d}$. Observe that $\mathbf{D}, \mathbf{D}^*$ and all $\mathbf{W}_i$ for $i \in [n]$ are distributed properly.

Simulating PP. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\mathbf{A}} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{V})} = g_1^{\bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{V})} \ \text{ and } \ g_1^{\mathbf{W}_i^\top \mathbf{A}} = g_1^{\mathbf{W}_i^\top \pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{V})} = g_1^{\mathbf{W}_i^\top \bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{V})}$$

for $i \in [n]$ using the knowledge of $g_1^{\pi_{\mathrm{L}}(\mathbf{V})}$ and $\bar{\mathbf{D}}, \mathbf{W}_1, \ldots, \mathbf{W}_n$.

Simulating $\mathbf{h}_j$ for $j \in [q']$. It is not hard to compute $\mathbf{V}^* \in \mathbb{Z}_p^{2d \times 2d}$ as

$$\mathbf{V}^* := \left(\begin{array}{ccc|ccc|ccc} a_1^{-1} & & & -a_1^{-1}b_{1,1} & \cdots & -a_1^{-1}b_{d/2,1} & -a_1^{-1}b_{d/2+1,1} & \cdots & -a_1^{-1}b_{d,1} \\ & \ddots & & \vdots & & \vdots & \vdots & & \vdots \\ & & a_d^{-1} & -a_d^{-1}b_{1,d} & \cdots & -a_d^{-1}b_{d/2,d} & -a_d^{-1}b_{d/2+1,d} & \cdots & -a_d^{-1}b_{d,d} \\ \hline & & & 1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & 1 & & & \\ \hline & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{array}\right).$$

For all $j \in [q']$, we sample $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d$ and implicitly set

$$\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j = \begin{pmatrix} \bar{\mathbf{r}}_j \\ \mathbf{0}_d \end{pmatrix}.$$

Since the upper $d \times d$ sub-matrix of $\pi_{\mathrm{L}}(\mathbf{V}^*)$ is full-rank with overwhelming probability, all $\mathbf{r}_j$ are distributed properly and $\mathscr{B}$ can simulate

$$g_2^{\mathbf{B}\mathbf{r}_j} = g_2^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}^*\mathbf{V}^*)\mathbf{r}_j} = g_2^{\bar{\mathbf{D}}^*\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j} \ \text{ and } \ g_2^{\mathbf{W}_i\mathbf{B}\mathbf{r}_j} = g_2^{\mathbf{W}_i\pi_{\mathrm{L}}(\bar{\mathbf{D}}^*\mathbf{V}^*)\mathbf{r}_j} = g_2^{\mathbf{W}_i\bar{\mathbf{D}}^*\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j}$$

using the knowledge of $\bar{\mathbf{D}}^*$, $\mathbf{W}_1, \ldots, \mathbf{W}_n$ and $\bar{\mathbf{r}}_j$.

Simulating the challenge. Algorithm $\mathscr{B}$ computes the challenge

$$g_1^{\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j} = g_1^{\bar{\mathbf{D}}\mathbf{V}\begin{pmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j \end{pmatrix}} \ \text{ and } \ g_1^{\mathbf{W}_i^\top\left(\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j\right)} = g_1^{\mathbf{W}_i^\top\bar{\mathbf{D}}\mathbf{V}\begin{pmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j \end{pmatrix}}$$

for $i \in [n]$ using the knowledge of $\bar{\mathbf{D}}, \mathbf{W}_1, \ldots, \mathbf{W}_n$ as well as

$$g_1^{\mathbf{v}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \widetilde{\mathbf{s}}_j\end{pmatrix}} = g_1^{\begin{pmatrix} a_1 s_{1,j} \\ \vdots \\ a_d s_{d,j} \\ b_{1,1}s_{1,j}+\cdots+b_{1,d}s_{d,j}+s_{d+1,j} \\ \vdots \\ b_{d,1}s_{1,j}+\cdots+b_{d,d}s_{d,j}+s_{2d,j}\end{pmatrix}}.$$

Analysis. Observe that if $s_{d+i,j} = 0$, then $\widehat{\mathbf{s}}_j = \widetilde{\mathbf{s}}_j = \mathbf{0}_{d/2}$ and the output challenge is distributed as $\left\{\mathbf{g}_j\right\}_{j\in[q]}$; otherwise, if $s_{d+i,j} \leftarrow \mathbb{Z}_p^*$, then $\widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j \leftarrow (\mathbb{Z}_p^*)^{d/2}$ and the output challenge is distributed as $\left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j\right\}_{j\in[q]}$. Therefore we may conclude that $\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k,q,q') \leqslant \mathsf{Adv}^{(d,d,q)\text{-LLin}}_{\mathscr{B}}(k)$. □

Corollary 5 (d-Lin ⇒ LS1) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{A}}(k,q,q') \leqslant d \cdot \mathsf{Adv}^{d\text{-Lin}}_{\mathscr{B}}(k) + 1/(p-1),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

8.4 Left subgroup indistinguishability 2

We may rewrite the LS2 advantage function $\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k,q,q')$ as follows:

$$\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k,q,q') := \left|\Pr[\mathscr{A}(D,T_0)=1] - \Pr[\mathscr{A}(D,T_1)=1]\right|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}_j^* \cdot \widetilde{h}_j^*\right\}_{j\in[q+q']}, \left\{\mathbf{g}'_j \cdot \widehat{\mathbf{g}}'_j \cdot \widetilde{\mathbf{g}}'_j\right\}_{j\in[q]}, \left\{\mathbf{h}_j\right\}_{j\in[q']}\right),$$

$$T_0 := \left\{\mathbf{g}_j \cdot \boxed{\widehat{\mathbf{g}}_j} \cdot \widetilde{\mathbf{g}}_j\right\}_{j\in[q]},\ T_1 := \left\{\mathbf{g}_j \cdot \widetilde{\mathbf{g}}_j\right\}_{j\in[q]}$$

and

$$\begin{aligned}
\mathrm{PP} &:= \left(g_1^{\mathbf{A}}, g_1^{\mathbf{W}_1^\top \mathbf{A}}, \ldots, g_1^{\mathbf{W}_n^\top \mathbf{A}}\right);\\
\widehat{h}_j^* \cdot \widetilde{h}_j^* &:= g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}_j + \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j};\\
\mathbf{g}'_j \cdot \widehat{\mathbf{g}}'_j \cdot \widetilde{\mathbf{g}}'_j &:= \left(g_1^{\mathbf{A}\mathbf{s}'_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}'_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j}, g_1^{\mathbf{W}_1^\top\left(\mathbf{A}\mathbf{s}'_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}'_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j\right)}, \ldots, g_1^{\mathbf{W}_n^\top\left(\mathbf{A}\mathbf{s}'_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}'_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j\right)}\right);\\
\mathbf{h}_j &:= \left(g_2^{\mathbf{B}\mathbf{r}_j}, g_2^{\mathbf{W}_1\mathbf{B}\mathbf{r}_j}, \ldots, g_2^{\mathbf{W}_n\mathbf{B}\mathbf{r}_j}\right);\\
\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j &:= \left(g_1^{\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j}, g_1^{\mathbf{W}_1^\top\left(\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j\right)}, \ldots, g_1^{\mathbf{W}_n^\top\left(\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j\right)}\right);\\
\mathbf{g}_j \cdot \widetilde{\mathbf{g}}_j &:= \left(g_1^{\mathbf{A}\mathbf{s}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j}, g_1^{\mathbf{W}_1^\top\left(\mathbf{A}\mathbf{s}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j\right)}, \ldots, g_1^{\mathbf{W}_n^\top\left(\mathbf{A}\mathbf{s}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j\right)}\right);
\end{aligned}$$

for $\widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j, \widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j, \widehat{\mathbf{s}}'_j, \widetilde{\mathbf{s}}'_j \leftarrow \mathbb{Z}_p^{d/2}$ and $\mathbf{r}_j, \mathbf{s}_j, \mathbf{s}'_j \leftarrow \mathbb{Z}_p^d$.

Lemma 13 ($(d, d/2, q)$-LLinAI ⇒ LS2) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k,q,q') \leqslant \mathsf{Adv}^{(d,d/2,q)\text{-LLinAI}}_{\mathscr{B}}(k),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Proof. Given an instance of $(d, d/2, q)$-LLinAI problem (i.e., set $\ell = d/2$)

$$\left(g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, \left\{g_1^{b_{i,j}}\right\}_{i\in[d/2], j\in[d]}, \left\{g_1^{a_1 s_{1,j}}, \ldots, g_1^{a_d s_{d,j}}\right\}_{j\in[q]}, \left\{g_1^{b_{i,1}s_{1,j}+\cdots+b_{i,d}s_{d,j}+s_{d+i,j}}\right\}_{i\in[d/2], j\in[q]}\right)$$

along with auxiliary input

$$\mathrm{AUX} = \left(\left\{g_2^{aa_1^{-1}b_{i,1}}, \ldots, g_2^{aa_{d/2}^{-1}b_{i,d/2}}\right\}_{i\in[d/2]}, g_2^a\right), \quad \text{where } a = a_1 \cdots a_{d/2}$$

as input where all $s_{d+i,j}$ are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathscr{B}$ works as follows:

Programming $\mathbf{s}_j$, $\widehat{\mathbf{s}}_j$ and $\widetilde{\mathbf{s}}_j$ for $j \in [q]$. Sample $\bar{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^{d/2}$. Adversary $\mathscr{B}$ implicitly sets

$$\mathbf{s}_j = \left(\bar{\mathbf{s}}_j^\top, s_{1,j}, \ldots, s_{(d/2),j}\right)^\top,\ \widehat{\mathbf{s}}_j = \left(s_{(d+1),j}, \ldots, s_{(3d/2),j}\right)^\top,\ \widetilde{\mathbf{s}}_j = \left(s_{(d/2+1),j}, \ldots, s_{d,j}\right)^\top.$$

Programming $\mathbf{D}, \mathbf{D}^*, \mathbf{W}_1, \cdots, \mathbf{W}_n$. Define $\mathbf{V} \in \mathbb{Z}_p^{2d \times 2d}$ as

$$\mathbf{V} := \left(\begin{array}{ccc|ccc|ccc} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ \hline & & & a_1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & a_{d/2} & & & \\ \hline & & & b_{1,1} & \cdots & b_{1,d/2} & 1 & & \\ & & & \vdots & & \vdots & & \ddots & \\ & & & b_{d/2,1} & \cdots & b_{d/2,d/2} & & & 1 \\ \hline & & & & & & a_{d/2+1} & & \\ & & & & & & & \ddots & \\ & & & & & & & & a_d \end{array}\right) \begin{array}{c} \\ \\ \\ \\ \\ \\ b_{1,d/2+1} \\ \vdots \\ b_{d/2,d/2+1} \\ \\ \\ \end{array} \begin{array}{c} \\ \\ \\ \\ \\ \\ \cdots \\ \\ \cdots \\ \\ \\ \end{array} \begin{array}{c} \\ \\ \\ \\ \\ \\ b_{1,d} \\ \vdots \\ b_{d/2,d} \\ \\ \\ \end{array}.$$

Sample $\bar{\mathbf{D}} \leftarrow \mathrm{GL}_{2d}(\mathbb{Z}_p)$ and let $\bar{\mathbf{D}}^* := (\bar{\mathbf{D}}^{-1})^\top$. Define

$$\mathbf{D} := \bar{\mathbf{D}}\mathbf{V} \ \text{ and } \ \mathbf{D}^* := \bar{\mathbf{D}}^*\mathbf{V}^*.$$

Sample $\mathbf{W}_1, \ldots, \mathbf{W}_n \leftarrow \mathbb{Z}_p^{2d \times 2d}$. Observe that $\mathbf{D}, \mathbf{D}^*$ and all $\mathbf{W}_i$ for $i \in [n]$ are distributed properly.

Simulating PP. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\mathbf{A}} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{V})} = g_1^{\bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{V})} \ \text{ and } \ g_1^{\mathbf{W}_i^\top \mathbf{A}} = g_1^{\mathbf{W}_i^\top \pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{V})} = g_1^{\mathbf{W}_i^\top \bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{V})},$$

for $i \in [n]$ using the knowledge of $g_1^{\pi_{\mathrm{L}}(\mathbf{V})}$ and $\bar{\mathbf{D}}, \mathbf{W}_1, \ldots, \mathbf{W}_n$.

Simulating $\widehat{h}_j^* \cdot \widetilde{h}_j^*$ for $j \in [q+q']$. It is not hard to compute $\mathbf{V}^* \in \mathbb{Z}_p^{2d \times 2d}$ as

$$\mathbf{V}^* := \left(\begin{array}{ccc|ccc|ccc} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ \hline & & & a_1^{-1} & & & -a_1^{-1}b_{1,1} & \cdots & -a_1^{-1}b_{d/2,1} \\ & & & & \ddots & & \vdots & & \vdots \\ & & & & & a_{d/2}^{-1} & -a_{d/2}^{-1}b_{1,d/2} & \cdots & -a_{d/2}^{-1}b_{d/2,d/2} \\ \hline & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \\ \hline & & & & & & -a_{d/2+1}^{-1}b_{1,d/2+1} & \cdots & -a_{d/2+1}^{-1}b_{d/2,d/2+1} \\ & & & & & & \vdots & & \vdots \\ & & & & & & -a_d^{-1}b_{1,d} & \cdots & -a_d^{-1}b_{d/2,d} \end{array}\right) \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \\ a_{d/2+1}^{-1} \\ \\ \\ a_d^{-1} \end{array}.$$

Observe that distribution $\left\{\pi_{\mathrm{M}}(\mathbf{V}^*)\widehat{\mathbf{r}}_j + \pi_{\mathrm{R}}(\mathbf{V}^*)\widetilde{\mathbf{r}}_j : \widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^{d/2}\right\}$ is identical to $\left\{\bar{\mathbf{V}}^*\bar{\mathbf{r}}_j : \bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d\right\}$ where

$\bar{\mathbf{V}}^* \in \mathbb{Z}_p^{2d\times d}$ is defined as

$$\bar{\mathbf{V}}^* = \left(\begin{array}{ccc|c} & & & \\ & & & \\ & & & \\ \hline -aa_1^{-1}b_{1,1} & \cdots & -aa_1^{-1}b_{d/2,1} & \\ \vdots & & \vdots & \\ -aa_{d/2}^{-1}b_{1,d/2} & \cdots & -aa_{d/2}^{-1}b_{d/2,d/2} & \\ \hline a & & & \\ & \ddots & & \\ & & a & \\ \hline & & & 1 \\ & & & & \ddots \\ & & & & & 1 \end{array}\right).$$

Algorithm $\mathscr{B}$ can simulate

$$\widehat{h}_j^* \cdot \widetilde{h}_j^* = g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}_j + \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j} = g_2^{\bar{\mathbf{D}}^*\left(\pi_{\mathrm{M}}(\mathbf{V}^*)\widehat{\mathbf{r}}_j + \pi_{\mathrm{R}}(\mathbf{V}^*)\widetilde{\mathbf{r}}_j\right)} = g_2^{\bar{\mathbf{D}}^*\bar{\mathbf{V}}^*\bar{\mathbf{r}}_j}$$

by sampling $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d$ and using the knowledge of $\bar{\mathbf{D}}^*$ and $g_2^{\bar{\mathbf{V}}^*}$, i.e., AUX.

Simulating $\mathbf{h}_j$ for $j \in [q']$. For all $j \in [q']$, we sample $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d$ and implicitly set

$$\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j = \begin{pmatrix} \bar{\mathbf{r}}_j \\ \mathbf{0}_d \end{pmatrix}.$$

Since the upper $d \times d$ sub-matrix of $\pi_{\mathrm{L}}(\mathbf{V}^*)$ is full-rank with overwhelming probability, all $\mathbf{r}_j$ are distributed properly and $\mathscr{B}$ can simulate

$$g_2^{\mathbf{B}\mathbf{r}_j} = g_2^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}^*\mathbf{V}^*)\mathbf{r}_j} = g_2^{\bar{\mathbf{D}}^*\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j} \text{ and } g_2^{\mathbf{W}_i\mathbf{B}\mathbf{r}_j} = g_2^{\mathbf{W}_i\pi_{\mathrm{L}}(\bar{\mathbf{D}}^*\mathbf{V}^*)\mathbf{r}_j} = g_2^{\mathbf{W}_i\bar{\mathbf{D}}^*\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j}$$

using the knowledge of $\bar{\mathbf{D}}^*, \mathbf{W}_1, \ldots, \mathbf{W}_n$ and $\bar{\mathbf{r}}_j$.

Simulating $\mathbf{g}'_j \cdot \widehat{\mathbf{g}}'_j \cdot \widetilde{\mathbf{g}}'_j$ for $j \in [q]$. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\mathbf{A}\mathbf{s}'_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}'_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j} = g_1^{\bar{\mathbf{D}}\mathbf{V}\begin{pmatrix}\mathbf{s}'_j\\ \widehat{\mathbf{s}}'_j\\ \widetilde{\mathbf{s}}'_j\end{pmatrix}} \text{ and } g_1^{\mathbf{W}_i^\top(\mathbf{A}\mathbf{s}'_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}'_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j)} = g_1^{\mathbf{W}_i^\top\bar{\mathbf{D}}\mathbf{V}\begin{pmatrix}\mathbf{s}'_j\\ \widehat{\mathbf{s}}'_j\\ \widetilde{\mathbf{s}}'_j\end{pmatrix}},$$

for $i \in [n]$ by sampling $\mathbf{s}'_j \leftarrow \mathbb{Z}_p^d$, $\widehat{\mathbf{s}}'_j, \widetilde{\mathbf{s}}'_j \leftarrow \mathbb{Z}_p^{d/2}$ for all $j \in [q]$ and using the knowledge of $g_1^{\mathbf{V}}$ and $\bar{\mathbf{D}}, \mathbf{W}_1, \ldots, \mathbf{W}_n$.

Simulating the challenge. Algorithm $\mathscr{B}$ computes

$$g_1^{\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j} = g_1^{\bar{\mathbf{D}}\mathbf{V}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \widetilde{\mathbf{s}}_j\end{pmatrix}} \text{ and } g_1^{\mathbf{W}_i^\top\left(\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j\right)} = g_1^{\mathbf{W}_i^\top\bar{\mathbf{D}}\mathbf{V}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \widetilde{\mathbf{s}}_j\end{pmatrix}}$$

for $i \in [n]$ using the knowledge of $\bar{\mathbf{D}}, \mathbf{W}_1, \ldots, \mathbf{W}_n$ and

$$g_1^{\mathbf{V}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \widetilde{\mathbf{s}}_j\end{pmatrix}} = g_1^{\begin{pmatrix} \bar{\mathbf{s}}_j \\ a_1 s_{1,j} \\ \vdots \\ a_{d/2}s_{d/2,j} \\ b_{1,1}s_{1,j} + \cdots + b_{1,d}s_{d,j} + s_{d+1,j} \\ \vdots \\ b_{d/2,1}s_{1,j} + \cdots + b_{d/2,d}s_{d,j} + s_{3d/2,j} \\ a_{d/2+1}s_{(d/2+1),j} \\ \vdots \\ a_d s_{d,j} \end{pmatrix}}.$$

Analysis. Observe that if $s_{d+i,j} = 0$, then $\widehat{\mathbf{s}}_j = \mathbf{0}_{d/2}$ and the output challenge is distributed as $\left\{\mathbf{g}_j \cdot \widetilde{\mathbf{g}}_j\right\}_{j\in[q]}$; in the other case, if $s_{d+i,j} \leftarrow \mathbb{Z}_p^*$, then $\widehat{\mathbf{s}}_j \leftarrow (\mathbb{Z}_p^*)^{d/2}$ and the output challenge is distributed as $\left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j\right\}_{j\in[q]}$. Therefore we may conclude that $\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k,q,q') \leqslant \mathsf{Adv}^{(d,d/2,q)\text{-LLinAI}}_{\mathscr{B}}(k)$. □

Corollary 6 (d-LLinAI ⇒ LS2) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k,q,q') \leqslant d/2 \cdot \mathsf{Adv}^{d\text{-LinAI}}_{\mathscr{B}}(k) + 1/(p-1),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

8.5 Left subgroup indistinguishability 3

We may rewrite the LS3 advantage function $\mathsf{Adv}^{\mathrm{LS3}}_{\mathscr{A}}(k,q,q')$ as follows:

$$\mathsf{Adv}^{\mathrm{LS3}}_{\mathscr{A}}(k,q,q') := \left|\Pr[\mathscr{A}(D,T_0)=1] - \Pr[\mathscr{A}(D,T_1)=1]\right|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}_j^* \cdot \widetilde{h}_j^*\right\}_{j\in[q+q']}, \left\{\mathbf{g}'_j \cdot \widetilde{\mathbf{g}}'_j\right\}_{j\in[q]}, \left\{\mathbf{h}_j\right\}_{j\in[q']}\right),$$

$$T_0 := \left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \boxed{\widetilde{\mathbf{g}}_j}\right\}_{j\in[q]},\ T_1 := \left\{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j\right\}_{j\in[q]}$$

and

$$\begin{aligned}
\mathrm{PP} &:= \left(g_1^{\mathbf{A}}, g_1^{\mathbf{W}_1^\top \mathbf{A}}, \ldots, g_1^{\mathbf{W}_n^\top \mathbf{A}}\right);\\
\widehat{h}_j^* \cdot \widetilde{h}_j^* &:= g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}_j + \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j};\\
\mathbf{g}'_j \cdot \widetilde{\mathbf{g}}'_j &:= \left(g_1^{\mathbf{A}\mathbf{s}'_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j}, g_1^{\mathbf{W}_1^\top\left(\mathbf{A}\mathbf{s}'_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j\right)}, \ldots, g_1^{\mathbf{W}_n^\top\left(\mathbf{A}\mathbf{s}'_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j\right)}\right);\\
\mathbf{h}_j &:= \left(g_2^{\mathbf{B}\mathbf{r}_j}, g_2^{\mathbf{W}_1\mathbf{B}\mathbf{r}_j}, \ldots, g_2^{\mathbf{W}_n\mathbf{B}\mathbf{r}_j}\right);\\
\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j &:= \left(g_1^{\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j}, g_1^{\mathbf{W}_1^\top\left(\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j\right)}, \ldots, g_1^{\mathbf{W}_n^\top\left(\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j\right)}\right);\\
\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j &:= \left(g_1^{\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j}, g_1^{\mathbf{W}_1^\top\left(\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j\right)}, \ldots, g_1^{\mathbf{W}_n^\top\left(\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j\right)}\right);
\end{aligned}$$

for $\widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j, \widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j, \widetilde{\mathbf{s}}'_j \leftarrow \mathbb{Z}_p^{d/2}$ and $\mathbf{r}_j, \mathbf{s}_j, \mathbf{s}'_j \leftarrow \mathbb{Z}_p^d$.

Lemma 14 ($(d,d/2,q)$-LLinAI ⇒ LS3) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{LS3}}_{\mathscr{A}}(k,q,q') \leqslant \mathsf{Adv}^{(d,d/2,q)\text{-LLinAI}}_{\mathscr{B}}(k),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Proof. The proof is similar to that for Lemma 13. Given an instance of $(d,d/2,q)$-LLinAI problem

$$\left(g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, \left\{g_1^{b_{i,j}}\right\}_{i\in[d/2],j\in[d]}, \left\{g_1^{a_1 s_{1,j}}, \ldots, g_1^{a_d s_{d,j}}\right\}_{j\in[q]}, \left\{g_1^{b_{i,1}s_{1,j} + \cdots + b_{i,d}s_{d,j} + s_{d+i,j}}\right\}_{i\in[d/2],j\in[q]}\right)$$

along with auxiliary input

$$\mathrm{AUX} = \left(\left\{g_2^{a a_1^{-1} b_{i,1}}, \ldots, g_2^{a a_{d/2}^{-1} b_{i,d/2}}\right\}_{i\in[d/2]}, g_2^a\right), \quad \text{where } a = a_1 \cdots a_{d/2}$$

as input where all $s_{d+i,j}$ are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathscr{B}$ behaves as in the proof of Lemma 13 with the differences that:

Programming $\mathbf{s}_j$, $\widehat{\mathbf{s}}_j$ and $\widetilde{\mathbf{s}}_j$ for $j \in [q]$. Sample $\bar{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^{d/2}$. Adversary $\mathscr{B}$ implicitly sets

$$\mathbf{s}_j = \left(\bar{\mathbf{s}}_j^\top, s_{1,j}, \ldots, s_{d/2,j}\right)^\top,\ \widehat{\mathbf{s}}_j = \left(s_{d/2+1,j}, \ldots, s_{d,j}\right)^\top,\ \widetilde{\mathbf{s}}_j = \left(s_{d+1,j}, \ldots, s_{3d/2,j}\right)^\top.$$

Programming V. Define matrix $\mathbf{V} \in \mathbb{Z}_p^{2d \times 2d}$ as

$$\mathbf{V} := \left(\begin{array}{c|ccc|ccc|c} 1 & & & & & & & \\ & \ddots & & & & & & \\ & & 1 & & & & & \\ \hline & a_1 & & & & & & \\ & & \ddots & & & & & \\ & & & a_{d/2} & & & & \\ \hline & & & & a_{d/2+1} & & & \\ & & & & & \ddots & & \\ & & & & & & a_d & \\ \hline & b_{1,1} & \cdots & b_{1,d/2} & b_{1,d/2+1} & \cdots & b_{1,d} & 1 \\ & \vdots & & \vdots & \vdots & & \vdots & \ddots \\ & b_{d/2,1} & \cdots & b_{d/2,d/2} & b_{d/2,d/2+1} & \cdots & b_{d/2,d} & 1 \end{array}\right).$$

Algorithm $\mathscr{B}$ may program $\mathbf{D}, \mathbf{D}^*$ and $\mathbf{W}_1, \ldots, \mathbf{W}_n$, then simulate $\mathrm{PP}, \left\{\widehat{h}_j^* \cdot \widetilde{h}_j^*\right\}, \left\{\mathbf{g}'_j \cdot \widetilde{\mathbf{g}}'_j\right\}, \left\{\mathbf{h}_j\right\}$ as well as the challenge by the strategies used in the proof of Lemma 13. □

Corollary 7 (d-LinAI ⇒ LS3) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{LS3}}_{\mathscr{A}}(k,q,q') \leqslant d/2 \cdot \mathsf{Adv}^{d\text{-LinAI}}_{\mathscr{B}}(k) + 1/(p-1),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

8.6 Nested-hiding indistinguishability

We may rewrite the NH advantage function $\mathsf{Adv}^{\mathrm{NH}(\eta)}_{\mathscr{A}}(k,q,q')$ for all $\eta \in [\lfloor n/2 \rfloor]$ as follows:

$$\mathsf{Adv}^{\mathrm{NH}(\eta)}_{\mathscr{A}}(k,q,q') := \left|\Pr[\mathscr{A}(D,T_0)=1] - \Pr[\mathscr{A}(D,T_1)=1]\right|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}_j^*\right\}_{j\in[q+q']}, \left\{\widetilde{h}_j^*\right\}_{j\in[q+q']}, \left\{(\widehat{\mathbf{g}}_j)_{-(2\eta-1)}\right\}_{j\in[q]}, \left\{(\widetilde{\mathbf{g}}_j)_{-2\eta}\right\}_{j\in[q]}, \{\mathbf{h}'_j\}_{j\in[q']}\right),$$

$$T_0 := \left\{\mathbf{h}_j\right\}_{j\in[q']}, \quad T_1 := \left\{\mathbf{h}_j \cdot (\widehat{h}_j^{**})^{\mathbf{e}_{2\eta-1}} \cdot (\widetilde{h}_j^{**})^{\mathbf{e}_{2\eta}}\right\}_{j\in[q']}$$

and

$$\begin{aligned}
\mathrm{PP} &:= \left(g_1^{\mathbf{A}}, g_1^{\mathbf{W}_1^\top \mathbf{A}}, \ldots, g_1^{\mathbf{W}_n^\top \mathbf{A}}\right);\\
\widehat{h}_j^* &:= g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}'_j}, \quad \widetilde{h}_j^* := g_2^{\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}'_j};\\
\widehat{\mathbf{g}}_j &:= \left(g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j}, g_1^{\mathbf{W}_1^\top \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j}, \ldots, g_1^{\mathbf{W}_n^\top \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j}\right);\\
\widetilde{\mathbf{g}}_j &:= \left(g_1^{\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j}, g_1^{\mathbf{W}_1^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j}, \ldots, g_1^{\mathbf{W}_n^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j}\right);\\
\mathbf{h}'_j &:= \left(g_2^{\mathbf{B}\mathbf{r}'_j}, g_2^{\mathbf{W}_1\mathbf{B}\mathbf{r}'_j}, \ldots, g_2^{\mathbf{W}_n\mathbf{B}\mathbf{r}'_j}\right);\\
\mathbf{h}_j \cdot (\widehat{h}_j^{**})^{\mathbf{e}_{2\eta-1}} \cdot (\widetilde{h}_j^{**})^{\mathbf{e}_{2\eta}} &:= \left(g_2^{\mathbf{B}\mathbf{r}_j}, g_2^{\mathbf{W}_1\mathbf{B}\mathbf{r}_j}, \ldots, g_2^{\mathbf{W}_{2\eta-1}\mathbf{B}\mathbf{r}_j + \widehat{\mathbf{B}}\widehat{\mathbf{r}}_j}, g_2^{\mathbf{W}_{2\eta}\mathbf{B}\mathbf{r}_j + \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j}, \ldots, g_2^{\mathbf{W}_n\mathbf{B}\mathbf{r}_j}\right);
\end{aligned}$$

for $\widehat{\mathbf{r}}'_j, \widetilde{\mathbf{r}}'_j, \widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j, \widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^{d/2}$ and $\mathbf{r}_j, \mathbf{r}'_j \leftarrow \mathbb{Z}_p^d$.

Lemma 15 ((d,d,d)-LLin ⇒ NH) *For any $\eta \in [\lfloor n/2 \rfloor]$ and for any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{NH}(\eta)}_{\mathscr{A}}(k,q,q') \leqslant \mathsf{Adv}^{(d,d,d)\text{-LLin}}_{\mathscr{B}}(k),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Proof. Given an instance of (d,d,d)-LLin problem (on G_2)

$$\left(g_1, g_2, g_2^{a_1}, \ldots, g_2^{a_d}, \left\{g_2^{b_{i,j}}\right\}_{i,j\in[d]}, \left\{g_2^{a_1 r_{1,j}}, \ldots, g_2^{a_d r_{d,j}}\right\}_{j\in[d]}, \left\{g_2^{b_{i,1}r_{1,j}+\cdots+b_{i,d}r_{d,j}+r_{d+i,j}}\right\}_{i,j\in[d]}\right),$$

where all $r_{d+i,j}$ are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathscr{B}$ works as follows:

Generating q' tuples. Algorithm $\mathscr{B}$ runs the algorithm described in Lemma 6 on the input q', group G_2, and the (d,d,d)-LLin instance, and obtains

$$\left(g_2^{\mathbf{VZ}}, g_2^{\mathbf{Z}}\right) \text{ and } \left\{\left(g_2^{\mathbf{t}_j}, g_2^{\mathbf{V}\mathbf{t}_j+\boldsymbol{\tau}_j}\right)\right\}_{j\in[q']}.$$

Recall that $\mathbf{V},\mathbf{Z}\in\mathbb{Z}_p^{d\times d}$ and $\mathbf{t}_j,\boldsymbol{\tau}_j\in\mathbb{Z}_p^d$. Then sample q' additional tuples

$$\left\{\left(g_2^{\mathbf{t}'_j}, g_2^{\mathbf{V}\mathbf{t}'_j}\right)\right\}_{j\in[q']}$$

where $\mathbf{t}'_j$ is randomly distributed over $\mathbb{Z}_p^d$ using $\left(g_2^{\mathbf{VZ}}, g_2^{\mathbf{Z}}\right)$.

Programming $\mathbf{D},\mathbf{D}^*$ and $\mathbf{W}_1,\ldots,\mathbf{W}_n$. Algorithm $\mathscr{B}$ samples $(\mathbf{D},\mathbf{D}^*)\leftarrow \mathrm{GL}_{2d}(\mathbb{Z}_p)$ such that $\mathbf{D}^\top\mathbf{D}^*=\mathbf{I}$. Sample $\mathbf{W}_1,\ldots,\mathbf{W}_{2(\eta-1)},\bar{\mathbf{W}}_{2\eta-1},\bar{\mathbf{W}}_{2\eta},\mathbf{W}_{2(\eta+1)-1}\ldots,\mathbf{W}_n\leftarrow\mathbb{Z}_p^{2d\times 2d}$ and implicitly set

$$\mathbf{W}_{2\eta-1}=\bar{\mathbf{W}}_{2\eta-1}+\left(\widehat{\mathbf{B}}\middle|\mathbf{0}_{2d\times(d/2)}\right)\left(\mathbf{V}\middle|\mathbf{0}_{d\times d}\right) \text{ and } \mathbf{W}_{2\eta}=\bar{\mathbf{W}}_{2\eta}+\left(\mathbf{0}_{2d\times(d/2)}\middle|\widetilde{\mathbf{B}}\right)\left(\mathbf{V}\middle|\mathbf{0}_{d\times d}\right)$$

We note that the resulting $\mathbf{W}_{2\eta-1}$ and $\mathbf{W}_{2\eta}$ are uniformly distributed over $\mathbb{Z}_p^{2d\times 2d}$.

Programming PP. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\mathbf{A}}=g_1^{\pi_{\mathrm{L}}(\mathbf{D})} \text{ and } g_1^{\mathbf{W}_i^\top\mathbf{A}}=g_1^{\mathbf{W}_i^\top\pi_{\mathrm{L}}(\mathbf{D})}$$

for $i\in[n]\setminus\{2\eta-1,2\eta\}$ using the knowledge of $\mathbf{D}$ and $\mathbf{W}_1,\ldots,\mathbf{W}_{2\eta-2},\mathbf{W}_{2\eta+1},\ldots,\mathbf{W}_n$. Observe that

$$\begin{aligned}\mathbf{W}_{2\eta-1}^\top\mathbf{A} &= \bar{\mathbf{W}}_{2\eta-1}^\top\mathbf{A}+\begin{pmatrix}\mathbf{V}^\top\\ \mathbf{0}_{d\times d}\end{pmatrix}\begin{pmatrix}\widehat{\mathbf{B}}^\top\\ \mathbf{0}_{(d/2)\times 2d}\end{pmatrix}\mathbf{A}=\bar{\mathbf{W}}_{2\eta-1}^\top\mathbf{A},\\ \mathbf{W}_{2\eta}^\top\mathbf{A} &= \bar{\mathbf{W}}_{2\eta}^\top\mathbf{A}+\begin{pmatrix}\mathbf{V}^\top\\ \mathbf{0}_{d\times d}\end{pmatrix}\begin{pmatrix}\mathbf{0}_{(d/2)\times 2d}\\ \widetilde{\mathbf{B}}^\top\end{pmatrix}\mathbf{A}=\bar{\mathbf{W}}_{2\eta}^\top\mathbf{A},\end{aligned}$$

following the fact that $\widehat{\mathbf{B}}^\top\mathbf{A}=\widetilde{\mathbf{B}}^\top\mathbf{A}=\mathbf{0}_{(d/2)\times d}$. Hence $\mathscr{B}$ can also simulate

$$g_1^{\mathbf{W}_{2\eta-1}^\top\mathbf{A}}=g_1^{\bar{\mathbf{W}}_{2\eta-1}^\top\pi_{\mathrm{L}}(\mathbf{D})} \text{ and } g_1^{\mathbf{W}_{2\eta}^\top\mathbf{A}}=g_1^{\bar{\mathbf{W}}_{2\eta}^\top\pi_{\mathrm{L}}(\mathbf{D})}$$

just using the knowledge of $\bar{\mathbf{W}}_{2\eta-1}$, $\bar{\mathbf{W}}_{2\eta}$ and $\mathbf{D}$.

Simulating $\widehat{h}_j^*$ and $\widetilde{h}_j^*$ for $j\in[q+q']$. Algorithm $\mathscr{B}$ can simulate

$$\widehat{h}_j^*=g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}'_j}=g_2^{\pi_{\mathrm{M}}(\mathbf{D}^*)\widehat{\mathbf{r}}'_j} \text{ and } \widetilde{h}_j^*=g_2^{\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}'_j}=g_2^{\pi_{\mathrm{R}}(\mathbf{D}^*)\widetilde{\mathbf{r}}'_j},$$

by sampling $\widehat{\mathbf{r}}'_j,\widetilde{\mathbf{r}}'_j\leftarrow\mathbb{Z}_p^{d/2}$ and using the knowledge of $\mathbf{D}^*$.

Simulating $(\widehat{\mathbf{g}}_j)_{-(2\eta-1)}$ for $j\in[q]$. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j}=g_1^{\pi_{\mathrm{M}}(\mathbf{D})\widehat{\mathbf{s}}_j} \text{ and } g_1^{\mathbf{W}_i^\top\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j}=g_1^{\mathbf{W}_i^\top\pi_{\mathrm{M}}(\mathbf{D})\widehat{\mathbf{s}}_j}$$

for $i\in[n]\setminus\{2\eta-1,2\eta\}$ by sampling $\widehat{\mathbf{s}}_j\leftarrow\mathbb{Z}_p^{d/2}$ and using the knowledge of $\mathbf{D}$ and $\mathbf{W}_1,\ldots,\mathbf{W}_{2\eta-2}$, $\mathbf{W}_{2\eta+1},\ldots,\mathbf{W}_n$. Observe that

$$\mathbf{W}_{2\eta}^\top\widehat{\mathbf{A}}=\bar{\mathbf{W}}_{2\eta}^\top\widehat{\mathbf{A}}+\begin{pmatrix}\mathbf{V}^\top\\ \mathbf{0}_{d\times d}\end{pmatrix}\begin{pmatrix}\mathbf{0}_{(d/2)\times 2d}\\ \widetilde{\mathbf{B}}^\top\end{pmatrix}\widehat{\mathbf{A}}=\bar{\mathbf{W}}_{2\eta}^\top\widehat{\mathbf{A}},$$

from the fact that $\widetilde{\mathbf{B}}^\top \widehat{\mathbf{A}} = \mathbf{0}_{(d/2)\times(d/2)}$. Therefore the algorithm $\mathscr{B}$ can simulate

$$g_1^{\mathbf{W}_{2\eta}^\top \widehat{\mathbf{A}} \widehat{\mathbf{s}}_j} = g_1^{\bar{\mathbf{W}}_{2\eta}^\top \widehat{\mathbf{A}} \widehat{\mathbf{s}}_j} = g_1^{\bar{\mathbf{W}}_{2\eta}^\top \pi_{\mathrm{M}}(\mathbf{D}) \widehat{\mathbf{s}}_j}$$

using the knowledge of $\bar{\mathbf{W}}_{2\eta}$ and $\mathbf{D}$ as well as $\widehat{\mathbf{s}}_j$ we have picked. We note that algorithm $\mathscr{B}$ can not compute $g_1^{\mathbf{W}_{2\eta-1}^\top \widehat{\mathbf{A}} \widehat{\mathbf{s}}_j}$ since

$$\mathbf{W}_{2\eta-1}^\top \widehat{\mathbf{A}} = \bar{\mathbf{W}}_{2\eta-1}^\top \widehat{\mathbf{A}} + \begin{pmatrix} \mathbf{V}^\top \\ \mathbf{0}_{d\times d} \end{pmatrix} \begin{pmatrix} \widehat{\mathbf{B}}^\top \\ \mathbf{0}_{(d/2)\times 2d} \end{pmatrix} \widehat{\mathbf{A}} = \bar{\mathbf{W}}_{2\eta-1}^\top \widehat{\mathbf{A}} + \begin{pmatrix} \mathbf{V}^\top \\ \mathbf{0}_{d\times d} \end{pmatrix} \begin{pmatrix} \mathbf{I}_{(d/2)\times(d/2)} \\ \mathbf{0}_{(d/2)\times(d/2)} \end{pmatrix}$$

which contains the upper $d/2$ rows of secret matrix $\mathbf{V}$.

Simulating $(\widetilde{\mathbf{g}}_j)_{-2\eta}$ for $j \in [q]$. The simulation strategy is similar to the above. In particular, algorithm $\mathscr{B}$ can simulate

$$g_1^{\widetilde{\mathbf{A}} \widetilde{\mathbf{s}}_j} = g_1^{\pi_{\mathrm{R}}(\mathbf{D}) \widetilde{\mathbf{s}}_j} \quad \text{and} \quad g_1^{\mathbf{W}_i^\top \widetilde{\mathbf{A}} \widetilde{\mathbf{s}}_j} = g_1^{\mathbf{W}_i^\top \pi_{\mathrm{R}}(\mathbf{D}) \widetilde{\mathbf{s}}_j}$$

for $i \in [n] \setminus \{2\eta-1, 2\eta\}$ by sampling $\widetilde{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^{d/2}$ and using the knowledge of $\mathbf{D}$ and $\mathbf{W}_1, \ldots, \mathbf{W}_{2\eta-2}$, $\mathbf{W}_{2\eta+1}, \ldots, \mathbf{W}_n$. Observe that

$$\mathbf{W}_{2\eta-1}^\top \widetilde{\mathbf{A}} = \bar{\mathbf{W}}_{2\eta-1}^\top \widetilde{\mathbf{A}} + \begin{pmatrix} \mathbf{V}^\top \\ \mathbf{0}_{d\times d} \end{pmatrix} \begin{pmatrix} \widehat{\mathbf{B}}^\top \\ \mathbf{0}_{(d/2)\times 2d} \end{pmatrix} \widetilde{\mathbf{A}} = \bar{\mathbf{W}}_{2\eta-1}^\top \widetilde{\mathbf{A}},$$

from the fact that $\widehat{\mathbf{B}}^\top \widetilde{\mathbf{A}} = \mathbf{0}_{(d/2)\times(d/2)}$. Therefore the algorithm $\mathscr{B}$ can simulate

$$g_1^{\mathbf{W}_{2\eta-1}^\top \widetilde{\mathbf{A}} \widetilde{\mathbf{s}}_j} = g_1^{\bar{\mathbf{W}}_{2\eta-1}^\top \widetilde{\mathbf{A}} \widetilde{\mathbf{s}}_j} = g_1^{\bar{\mathbf{W}}_{2\eta-1}^\top \pi_{\mathrm{R}}(\mathbf{D}) \widetilde{\mathbf{s}}_j}$$

using the knowledge of $\bar{\mathbf{W}}_{2\eta-1}$ and $\mathbf{D}$ as well as $\widetilde{\mathbf{s}}_j$ we have picked. We note that algorithm $\mathscr{B}$ can not compute $g_1^{\mathbf{W}_{2\eta}^\top \widetilde{\mathbf{A}} \widetilde{\mathbf{s}}_j}$ since

$$\mathbf{W}_{2\eta}^\top \widetilde{\mathbf{A}} = \bar{\mathbf{W}}_{2\eta}^\top \widetilde{\mathbf{A}} + \begin{pmatrix} \mathbf{V}^\top \\ \mathbf{0}_{d\times d} \end{pmatrix} \begin{pmatrix} \mathbf{0}_{(d/2)\times 2d} \\ \widetilde{\mathbf{B}}^\top \end{pmatrix} \widetilde{\mathbf{A}} = \bar{\mathbf{W}}_{2\eta}^\top \widetilde{\mathbf{A}} + \begin{pmatrix} \mathbf{V}^\top \\ \mathbf{0}_{d\times d} \end{pmatrix} \begin{pmatrix} \mathbf{0}_{(d/2)\times(d/2)} \\ \mathbf{I}_{(d/2)\times(d/2)} \end{pmatrix}$$

which contains the lower $d/2$ rows of secret matrix $\mathbf{V}$.

Simulating $\mathbf{h}'_j$ for all $j \in [q']$. Let $\mathbf{T} := \underline{\mathbf{B}}\overline{\mathbf{B}}^{-1}$ where $\overline{\mathbf{B}}$ and $\underline{\mathbf{B}}$ are the upper and lower $d \times d$ sub-matrix of $\mathbf{B}$. Because $\mathbf{B} = \pi_{\mathrm{L}}(\mathbf{D}^*)$ is sampled by the simulator, it can efficiently compute the matrix $\mathbf{T}$. Since the sub-matrix $\overline{\mathbf{B}}$ is full-rank with overwhelming probability, we may implicitly sample

$$\mathbf{B}\mathbf{r}'_j = \begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix}.$$

In such a case, algorithm $\mathscr{B}$ can simulate

$$g_2^{\mathbf{B}\mathbf{r}'_j} = g_2^{\begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix}} \quad \text{and} \quad g_2^{\mathbf{W}_i \mathbf{B}\mathbf{r}'_j} = g_2^{\mathbf{W}_i \begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix}}$$

for $i \in [n] \setminus \{2\eta-1, 2\eta\}$ using $g_2^{\mathbf{t}'_j}$ and the knowledge of $\mathbf{T}, \mathbf{W}_1, \ldots, \mathbf{W}_{2\eta-2}, \mathbf{W}_{2\eta+1}, \ldots, \mathbf{W}_n$. Observe that

$$\begin{aligned}
\mathbf{W}_{2\eta-1}\mathbf{B}\mathbf{r}'_j &= \bar{\mathbf{W}}_{2\eta-1} \begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix} + \left(\widehat{\mathbf{B}} \middle| \mathbf{0}_{2d\times(d/2)}\right) \left(\mathbf{V} \middle| \mathbf{0}_{d\times d}\right) \begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix} = \bar{\mathbf{W}}_{2\eta-1} \begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix} + \left(\widehat{\mathbf{B}} \middle| \mathbf{0}_{2d\times(d/2)}\right) \mathbf{V}\mathbf{t}'_j; \\
\mathbf{W}_{2\eta}\mathbf{B}\mathbf{r}'_j &= \bar{\mathbf{W}}_{2\eta} \begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix} + \left(\mathbf{0}_{2d\times(d/2)} \middle| \widetilde{\mathbf{B}}\right) \left(\mathbf{V} \middle| \mathbf{0}_{d\times d}\right) \begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix} = \bar{\mathbf{W}}_{2\eta} \begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix} + \left(\mathbf{0}_{2d\times(d/2)} \middle| \widetilde{\mathbf{B}}\right) \mathbf{V}\mathbf{t}'_j.
\end{aligned}$$

Therefore algorithm $\mathscr{B}$ can simulate

$$g_2^{\mathbf{W}_{2\eta-1}\mathbf{B}\mathbf{r}'_j} = g_2^{\bar{\mathbf{W}}_{2\eta-1}\begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix} + \left(\widehat{\mathbf{B}} \middle| \mathbf{0}_{2d\times(d/2)}\right)\mathbf{V}\mathbf{t}'_j} \quad \text{and} \quad g_2^{\mathbf{W}_{2\eta}\mathbf{B}\mathbf{r}'_j} = g_2^{\bar{\mathbf{W}}_{2\eta}\begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix} + \left(\mathbf{0}_{2d\times(d/2)} \middle| \widetilde{\mathbf{B}}\right)\mathbf{V}\mathbf{t}'_j}$$

using $\left(g_2^{\mathbf{t}'_j}, g_2^{\mathbf{V}\mathbf{t}'_j}\right)$ and the knowledge of $\bar{\mathbf{W}}_{2\eta-1}, \bar{\mathbf{W}}_{2\eta}$ and $\mathbf{D}^*$ which is used to derive $\widehat{\mathbf{B}}, \widetilde{\mathbf{B}}$ and $\mathbf{T}$.

Simulating the challenge. The challenge is produced following the method for simulating $\mathbf{h}'_j$ but using tuples $\left\{\left(g_2^{\mathbf{t}_j}, g_2^{\mathbf{V}\mathbf{t}_j+\boldsymbol{\tau}_j}\right)\right\}_{j\in[q']}$ instead of $\left\{\left(g_2^{\mathbf{t}'_j}, g_2^{\mathbf{V}\mathbf{t}'_j}\right)\right\}_{j\in[q']}$. In particular, we implicitly set

$$\mathbf{B}\mathbf{r}_j = \begin{pmatrix}\mathbf{t}_j\\ \mathbf{T}\mathbf{t}_j\end{pmatrix}.$$

Following the above observation, algorithm $\mathscr{B}$ can simulate

$$g_2^{\mathbf{B}\mathbf{r}_j} = g_2^{\binom{\mathbf{t}_j}{\mathbf{T}\mathbf{t}_j}} \quad \text{and} \quad g_2^{\mathbf{W}_i\mathbf{B}\mathbf{r}_j} = g_2^{\mathbf{W}_i\binom{\mathbf{t}_j}{\mathbf{T}\mathbf{t}_j}}$$

for $i \in [n] \setminus \{2\eta-1, 2\eta\}$ using $g_2^{\mathbf{t}_j}$ and the knowledge of $\mathbf{T}, \mathbf{W}_1, \ldots, \mathbf{W}_{2\eta-2}, \mathbf{W}_{2\eta+1}, \ldots, \mathbf{W}_n$, and simulate

$$g_2^{\mathbf{W}_{2\eta-1}\mathbf{B}\mathbf{r}_j+\widehat{\mathbf{B}}\widehat{\mathbf{r}}_j} = g_2^{\bar{\mathbf{W}}_{2\eta-1}\binom{\mathbf{t}_j}{\mathbf{T}\mathbf{t}_j}+(\widehat{\mathbf{B}}|\mathbf{0}_{2d\times(d/2)})(\mathbf{V}\mathbf{t}_j+\boldsymbol{\tau}_j)} \text{ and } g_2^{\mathbf{W}_{2\eta}\mathbf{B}\mathbf{r}_j+\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j} = g_2^{\bar{\mathbf{W}}_{2\eta}\binom{\mathbf{t}_j}{\mathbf{T}\mathbf{t}_j}+(\mathbf{0}_{2d\times(d/2)}|\widetilde{\mathbf{B}})(\mathbf{V}\mathbf{t}_j+\boldsymbol{\tau}_j)}$$

using $\left(g_2^{\mathbf{t}_j}, g_2^{\mathbf{V}\mathbf{t}_j+\boldsymbol{\tau}_j}\right)$ and the knowledge of $\bar{\mathbf{W}}_{2\eta-1}, \bar{\mathbf{W}}_{2\eta}$ and $\mathbf{D}^*$ which is used to derive $\widehat{\mathbf{B}}, \widetilde{\mathbf{B}}$ and $\mathbf{T}$.

Analysis. Observe that, we implicitly set

$$\begin{pmatrix}\widehat{\mathbf{r}}_j\\ \widetilde{\mathbf{r}}_j\end{pmatrix} = \boldsymbol{\tau}_j$$

when simulating the challenge. Therefore, if $r_{d+i,j} = 0$, then $\boldsymbol{\tau}_j = \mathbf{0}_d$ and the output challenge has the same distribution as $\left\{\mathbf{h}_j\right\}_{j\in[q']}$; on the other hand, if $r_{d+i,j} \leftarrow \mathbb{Z}_p^*$, then $\boldsymbol{\tau}_j \leftarrow (\mathbb{Z}_p^*)^d$ and the output challenge is distributed as $\left\{\mathbf{h}_j \cdot (\widehat{h}_j^{**})^{\mathbf{e}_{2\eta-1}} \cdot (\widetilde{h}_j^{**})^{\mathbf{e}_{2\eta}}\right\}_{j\in[q']}$. We may conclude that $\mathsf{Adv}^{\mathrm{NH}(\eta)}_{\mathscr{A}}(k,q,q') \leqslant \mathsf{Adv}^{(d,d,d)\text{-LLin}}_{\mathscr{B}}(k)$. □

Corollary 8 (d-Lin ⇒ NH) *For any p.p.t. adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{NH}}_{\mathscr{A}}(k,q,q') \leqslant d \cdot \mathsf{Adv}^{d\text{-Lin}}_{\mathscr{B}}(k) + 1/(p-1),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

8.7 Computational Non-degeneracy

We may rewrite the ND advantage function as:

$$\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{A}}(k,q,q',q'') := |\Pr[\mathscr{A}(D,T_0)=1] - \Pr[\mathscr{A}(D,T_1)=1]|,$$

where

$$D := \left(\mathrm{PP}, \left\{\widehat{h}_j^* \cdot \widetilde{h}_j^*,\ \mathbf{h}_j\right\}_{j\in[q']}, \left\{\widehat{\mathbf{g}}_{j,j'} \cdot \widetilde{\mathbf{g}}_{j,j'}\right\}_{j\in[q],j'\in[q'']}\right),$$

$$T_0 := \left\{e(\widehat{g}_{0,j,j'} \cdot \widetilde{g}_{0,j,j'}, \widehat{h}_j^{**} \cdot \widetilde{h}_j^{**})\right\}_{j\in[q],j'\in[q'']},$$

$$T_1 := \left\{e(\widehat{g}_{0,j,j'} \cdot \widetilde{g}_{0,j,j'}, \widehat{h}_j^{**} \cdot \widetilde{h}_j^{**}) \cdot \widehat{R}_{j,j'}\right\}_{j\in[q],j'\in[q'']}.$$

and

$$\begin{aligned}
\mathrm{PP} &:= \left(g_1^{\mathbf{A}}, g_1^{\mathbf{W}_1^\top\mathbf{A}}, \ldots, g_1^{\mathbf{W}_n^\top\mathbf{A}}\right);\\
\widehat{h}_j^* \cdot \widetilde{h}_j^* &:= g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}'_j+\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}'_j};\\
\mathbf{h}_j &:= \left(g_2^{\mathbf{B}\mathbf{r}_j}, g_2^{\mathbf{W}_1\mathbf{B}\mathbf{r}_j}, \ldots, g_2^{\mathbf{W}_n\mathbf{B}\mathbf{r}_j}\right);\\
\widehat{\mathbf{g}}_{j,j'} \cdot \widetilde{\mathbf{g}}_{j,j'} &:= \left(g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}_{j,j'}+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_{j,j'}}, g_1^{\mathbf{W}_1^\top(\widehat{\mathbf{A}}\widehat{\mathbf{s}}_{j,j'}+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_{j,j'})}, \ldots, g_1^{\mathbf{W}_n^\top(\widehat{\mathbf{A}}\widehat{\mathbf{s}}_{j,j'}+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_{j,j'})}\right);\\
e(\widehat{g}_{0,j,j'} \cdot \widetilde{g}_{0,j,j'}, \widehat{h}_j^{**} \cdot \widetilde{h}_j^{**}) &:= e(g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}_{j,j'}+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_{j,j'}}, g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}_j+\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j}) = e(g_1,g_2)^{(\widehat{\mathbf{s}}_{j,j'}^\top, \widetilde{\mathbf{s}}_{j,j'}^\top)\binom{\widehat{\mathbf{r}}_j}{\widetilde{\mathbf{r}}_j}};\\
\widehat{R}_{j,j'} &:= e(g_1,g_2)^{\widehat{\gamma}_{j,j'}};
\end{aligned}$$

for $\widehat{\mathbf{r}}'_j, \widehat{\mathbf{r}}'_j, \widehat{\mathbf{s}}_{j,j'}, \widehat{\mathbf{r}}_j, \widetilde{\mathbf{s}}_{j,j'}, \widetilde{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^{d/2}$, $\mathbf{r}_j \leftarrow \mathbb{Z}_p^d$ and $\widehat{\gamma}_{j,j'} \leftarrow \mathbb{Z}_p$.

Lemma 16 ($(d,1,qq'')$-LLin $\Rightarrow$ ND) *For any p.p.t adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{A}}(k,q,q',q'') \leqslant \mathsf{Adv}^{(d,1,qq'')\text{-LLin}}_{\mathscr{B}}(k),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (qq''+q')d^2 \cdot \mathsf{poly}(k,n)$ *where* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Proof. The proof follows the main idea of that of Lemma 10. Given an instance of $(d,1,qq'')$-LLin problem

$$\left(g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, \left\{g_1^{b_i}\right\}_{i\in[d]}, \left\{g_1^{a_1 s_{1,j,j'}}, \ldots, g_1^{a_d s_{d,j,j'}}\right\}_{j\in[q],j'\in[q'']}, \left\{g_1^{b_1 s_{1,j,j'}+\cdots+b_d s_{d,j,j'}+s_{d+1,j,j'}}\right\}_{j\in[q],j'\in[q'']}\right)$$

as input where all $s_{d+1,j,j'}$ are either 0 or uniformly chosen from $\mathbb{Z}_p^*$, adversary $\mathscr{B}$ works as follows:

Programming $\widehat{\mathbf{s}}_{j,j'}$ and $\widetilde{\mathbf{s}}_{j,j'}$ for $j\in[q], j'\in[q'']$. Adversary $\mathscr{B}$ implicitly sets

$$\widehat{\mathbf{s}}_{j,j'} = \left(s_{1,j,j'}, \ldots, s_{d/2,j,j'}\right)^\top \text{ and } \widetilde{\mathbf{s}}_{j,j'} = \left(s_{d/2+1,j,j'}, \ldots, s_{d,j,j'}\right)^\top.$$

Programming $\mathbf{D}, \mathbf{D}^*, \mathbf{W}_1, \cdots, \mathbf{W}_n$. We define $\mathbf{U}\in\mathbb{Z}_p^{2d\times 2d}$ as

$$\mathbf{U} := \left(\begin{array}{ccc|ccc} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ \hline & & & a_1 & & \\ & & & & \ddots & \\ & & & & & a_{d/2} \\ \hline & & & a_{d/2+1} & & \\ & & & & \ddots & \\ & & & & & a_d \end{array}\right).$$

Sample $\bar{\mathbf{D}} \leftarrow \mathrm{GL}_{2d}(\mathbb{Z}_p)$ and let $\bar{\mathbf{D}}^* := (\bar{\mathbf{D}}^{-1})^\top$. Define

$$\mathbf{D} := \bar{\mathbf{D}}\mathbf{U} \text{ and } \mathbf{D}^* := \bar{\mathbf{D}}^*\mathbf{U}^*.$$

Sample $\mathbf{W}_1, \ldots, \mathbf{W}_n \leftarrow \mathbb{Z}_p^{2d\times 2d}$. Observe that $\mathbf{D}, \mathbf{D}^*$ and all $\mathbf{W}_i$ for $i\in[n]$ are distributed properly.

Simulating PP. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\mathbf{A}} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{U})} = g_1^{\bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{U})} \text{ and } g_1^{\mathbf{W}_i^\top\mathbf{A}} = g_1^{\mathbf{W}_i^\top\pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{U})} = g_1^{\mathbf{W}_i^\top\bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{U})}$$

for $i\in[n]$ using the knowledge of $\pi_{\mathrm{L}}(\mathbf{U})$ and $\bar{\mathbf{D}}, \mathbf{W}_1, \ldots, \mathbf{W}_n$.

Simulating $\widehat{h}_j^* \cdot \widetilde{h}_j^*$ for $j\in[q']$. It is not hard to compute $\mathbf{U}^*\in\mathbb{Z}_p^{2d\times 2d}$ as

$$\mathbf{U}^* := \left(\begin{array}{ccc|ccc} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ \hline & & & a_1^{-1} & & \\ & & & & \ddots & \\ & & & & & a_{d/2}^{-1} \\ \hline & & & a_{d/2+1}^{-1} & & \\ & & & & \ddots & \\ & & & & & a_d^{-1} \end{array}\right).$$

For all $j\in[q']$, we sample $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d$ and implicitly set

$$\begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}'_j \\ \widetilde{\mathbf{r}}'_j \end{pmatrix} = (\mathbf{U}^*)^{-1}\begin{pmatrix} \mathbf{0}_d \\ \bar{\mathbf{r}}_j \end{pmatrix}.$$

Since the right-most $d \times d$ sub-matrix of $\mathbf{U}^*$ is full-rank with overwhelming probability, both $\widehat{\mathbf{r}}'_j$ and $\widetilde{\mathbf{r}}'_j$ are distributed properly and $\mathscr{B}$ can simulate

$$\widehat{h}^*_j \cdot \widetilde{h}^*_j = g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}'_j + \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}'_j} = g_2^{\bar{\mathbf{D}}^*\mathbf{U}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}'_j \\ \widetilde{\mathbf{r}}'_j \end{pmatrix}} = g_2^{\bar{\mathbf{D}}^* \begin{pmatrix} \mathbf{0}_d \\ \bar{\mathbf{r}}_j \end{pmatrix}}$$

using the knowledge of $\bar{\mathbf{D}}^*$ and $\bar{\mathbf{r}}_j$.

Simulating $\mathbf{h}_j$ for all $j \in [q']$. Algorithm $\mathscr{B}$ may compute $\mathrm{HP} := \left(g_2^{\mathbf{B}}, g_2^{\mathbf{W}_1\mathbf{B}}, \ldots, g_2^{\mathbf{W}_n\mathbf{B}}\right)$ where

$$g_2^{\mathbf{B}} = g_2^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}^*\mathbf{U}^*)} = g_2^{\bar{\mathbf{D}}^*\pi_{\mathrm{L}}(\mathbf{U}^*)} \text{ and } g_2^{\mathbf{W}_i\mathbf{B}} = g_2^{\mathbf{W}_i\pi_{\mathrm{L}}(\bar{\mathbf{D}}^*\mathbf{U}^*)} = g_2^{\mathbf{W}_i\bar{\mathbf{D}}^*\pi_{\mathrm{L}}(\mathbf{U}^*)}$$

for $i \in [n]$ using the knowledge of $\bar{\mathbf{D}}^*$, $\pi_{\mathrm{L}}(\mathbf{U}^*)$ and $\mathbf{W}_1, \ldots, \mathbf{W}_n$. This allows it to simulate $\mathbf{h}_j$ by running $\mathsf{SampH}(\mathrm{PP}, \mathrm{HP})$.

Simulating $\widehat{\mathbf{g}}_{j,j'} \cdot \widetilde{\mathbf{g}}_{j,j'}$ for $j \in [q], j' \in [q']$. Algorithm $\mathscr{B}$ can simulate

$$g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}_{j,j'} + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_{j,j'}} = g_1^{\bar{\mathbf{D}}\mathbf{U} \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{s}}_{j,j'} \\ \widetilde{\mathbf{s}}_{j,j'} \end{pmatrix}} \text{ and } g_1^{\mathbf{W}_i^\top\left(\widehat{\mathbf{A}}\widehat{\mathbf{s}}_{j,j'} + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_{j,j'}\right)} = g_1^{\mathbf{W}_i^\top\bar{\mathbf{D}}\mathbf{U} \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{s}}_{j,j'} \\ \widetilde{\mathbf{s}}_{j,j'} \end{pmatrix}}$$

using the knowledge of $\bar{\mathbf{D}}, \mathbf{W}_1, \ldots, \mathbf{W}_n$ and

$$g_1^{\mathbf{U} \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{s}}_{j,j'} \\ \widetilde{\mathbf{s}}_{j,j'} \end{pmatrix}} = g_1^{\begin{pmatrix} \mathbf{0}_d \\ a_1 s_{1,j,j'} \\ \vdots \\ a_d s_{d,j,j'} \end{pmatrix}}.$$

Simulating the challenge. Define an additional matrix $\mathbf{V}$ of rank d as

$$\mathbf{V} := \begin{pmatrix} a_1 & & \\ & \ddots & \\ & & a_d \\ b_1 & \cdots & b_d \end{pmatrix} \in \mathbb{Z}_p^{(d+1)\times d}.$$

For all $j \in [q]$, algorithm $\mathscr{B}$ samples $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^{d+1}$ and implicitly set $\left(\widehat{\mathbf{r}}_j^\top, \widetilde{\mathbf{r}}_j^\top\right) := \bar{\mathbf{r}}_j^\top \mathbf{V}$. Then $\mathscr{B}$ computes

$$g_1^{\left(\widehat{\mathbf{r}}_j^\top, \widetilde{\mathbf{r}}_j^\top\right)\begin{pmatrix} \widehat{\mathbf{s}}_{j,j'} \\ \widetilde{\mathbf{s}}_{j,j'} \end{pmatrix} + \widehat{\gamma}_{j,j'}} = g_1^{\bar{\mathbf{r}}_j^\top \begin{pmatrix} a_1 s_{1,j,j'} \\ \vdots \\ a_d s_{d,j,j'} \\ b_1 s_{1,j,j'} + \cdots + b_d s_{d,j,j'} + s_{d+1,j,j'} \end{pmatrix}}$$

using the knowledge of $\left\{g_1^{a_1 s_{1,j,j'}}, \ldots, g_1^{a_d s_{d,j,j'}}, g_1^{b_1 s_{1,j,j'} + \cdots + b_d s_{d,j,j'} + s_{d+1,j,j'}}\right\}$ and outputs $e(g_1^{\left(\widehat{\mathbf{r}}_j^\top, \widetilde{\mathbf{r}}_j^\top\right)\begin{pmatrix} \widehat{\mathbf{s}}_{j,j'} \\ \widetilde{\mathbf{s}}_{j,j'} \end{pmatrix} + \widehat{\gamma}_{j,j'}}, g_2)$.

Analysis. Observe that, if $s_{d+1,j,j'} = 0$, then the output challenge is distributed as

$$e(g_1^{\bar{\mathbf{r}}_j^\top \mathbf{V}\begin{pmatrix} \widehat{\mathbf{s}}_{j,j'} \\ \widetilde{\mathbf{s}}_{j,j'} \end{pmatrix}}, g_2) = e(g_1, g_2)^{\left(\widehat{\mathbf{s}}_{j,j'}^\top, \widetilde{\mathbf{s}}_{j,j'}^\top\right)\begin{pmatrix} \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j \end{pmatrix}},$$

which is identical to T_0 where $\widehat{\gamma}_{j,j'} := 0$; if $s_{d+1,j,j'} \leftarrow \mathbb{Z}_p^*$, then the output challenge is distributed as

$$e(g_1^{\bar{\mathbf{r}}_j^\top\left(\mathbf{V}\begin{pmatrix} \widehat{\mathbf{s}}_{j,j'} \\ \widetilde{\mathbf{s}}_{j,j'} \end{pmatrix} + \mathbf{e}_{d+1} s_{d+1,j,j'}\right)}, g_2) = e(g_1, g_2)^{\left(\widehat{\mathbf{s}}_{j,j'}^\top, \widetilde{\mathbf{s}}_{j,j'}^\top\right)\begin{pmatrix} \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j \end{pmatrix}} \cdot \boxed{e(g_1, g_2)^{s_{d+1,j,j'}\mathbf{e}_{d+1}^\top \bar{\mathbf{r}}_j}},$$

which is identical to T_1 where $\widehat{\gamma}_{j,j'} := s_{d+1,j,j'}\mathbf{e}_{d+1}^\top \bar{\mathbf{r}}_j$ (in the box) is uniformly distributed over $\mathbb{Z}_p$. Therefore we may conclude that $\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{A}}(k, q, q', q'') \leqslant \mathsf{Adv}^{(d,1,qq'')\text{-LLin}}_{\mathscr{B}}(k)$. □

Corollary 9 (d-Lin ⇒ ND) *For any p.p.t adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

$$\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{A}}(k, q, q', q'') \leqslant \mathsf{Adv}^{d\text{-Lin}}_{\mathscr{B}}(k) + 1/(p-1),$$

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (qq'' + q')d^2 \cdot \mathsf{poly}(k, n)$ *where* $\mathsf{poly}(k, n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

9 Concrete IBE from d-Linear Assumption with Auxiliary Input

This section present an concrete IBE scheme derived from our prime-order instantiation in Section 8 and the generic construction in Appendix C which is an adaptation of Hofheinz *et al.*'s (c.f. Section B). Let GrpGen be the bilinear group generator described in Section 4.1 and $\pi_{\mathrm{L}}(\cdot)$ be the function mapping from a $2d \times 2d$ matrix to its left-most d columns.

– $\mathsf{Param}(1^k, n)$: Run $(p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$. Sample $\mathbf{D} \leftarrow \mathrm{GL}_{2d}(\mathbb{Z}_p)$ and $\mathbf{W}_1, \ldots, \mathbf{W}_{2n} \leftarrow \mathbb{Z}_p^{2d \times 2d}$, and set $\mathbf{D}^* := (\mathbf{D}^{-1})^\top$. Output

$$\mathrm{GP} := \left(p, G_1^{2d}, G_2^{2d}, G_T, e; \quad \begin{matrix} g_1^{\pi_{\mathrm{L}}(\mathbf{D})}, & g_1^{\mathbf{W}_1^\top \pi_{\mathrm{L}}(\mathbf{D})}, & \cdots, & g_1^{\mathbf{W}_{2n}^\top \pi_{\mathrm{L}}(\mathbf{D})} \\ g_2^{\pi_{\mathrm{L}}(\mathbf{D}^*)}, & g_2^{\mathbf{W}_1 \pi_{\mathrm{L}}(\mathbf{D}^*)}, & \cdots, & g_2^{\mathbf{W}_{2n} \pi_{\mathrm{L}}(\mathbf{D}^*)} \end{matrix} \right).$$

– $\mathsf{Setup}(\mathrm{GP})$: Sample $\mathbf{k} \leftarrow \mathbb{Z}_p^{2d}$ and output

$$\begin{aligned} \mathrm{MPK} &:= \left(p, G_1^{2d}, G_2^{2d}, G_T, e; e(g_1, g_2)^{\pi_{\mathrm{L}}(\mathbf{D})^\top \mathbf{k}}, g_1^{\pi_{\mathrm{L}}(\mathbf{D})}, g_1^{\mathbf{W}_1^\top \pi_{\mathrm{L}}(\mathbf{D})}, \ldots, g_1^{\mathbf{W}_{2n}^\top \pi_{\mathrm{L}}(\mathbf{D})} \right); \\ \mathrm{MSK} &:= \left(g_2^{\mathbf{k}}, g_2^{\pi_{\mathrm{L}}(\mathbf{D}^*)}, g_2^{\mathbf{W}_1 \pi_{\mathrm{L}}(\mathbf{D}^*)}, \ldots, g_2^{\mathbf{W}_{2n} \pi_{\mathrm{L}}(\mathbf{D}^*)} \right). \end{aligned}$$

– $\mathsf{KeyGen}(\mathrm{MPK}, \mathrm{MSK}, \mathbf{y})$: Let $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$. Sample $\mathbf{r} \leftarrow \mathbb{Z}_p^d$ and output

$$\mathrm{SK}_{\mathbf{y}} := \left(K_0 := g_2^{\pi_{\mathrm{L}}(\mathbf{D}^*)\mathbf{r}}, \; K_1 := g_2^{\mathbf{k} + (\mathbf{W}_{2-y_1} + \cdots + \mathbf{W}_{2n-y_n}) \pi_{\mathrm{L}}(\mathbf{D}^*)\mathbf{r}} \right).$$

– $\mathsf{Enc}(\mathrm{MPK}, \mathbf{x}, \mathrm{M})$: Let $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$ and $\mathrm{M} \in \mathbb{G}_T$. Sample $\mathbf{s} \leftarrow \mathbb{Z}_p^d$ and output

$$\mathrm{CT}_{\mathbf{x}} := \left(C_0 := g_1^{\pi_{\mathrm{L}}(\mathbf{D})\mathbf{s}}, \; C_1 := g_1^{(\mathbf{W}_{2-x_1} + \cdots + \mathbf{W}_{2n-x_n})^\top \pi_{\mathrm{L}}(\mathbf{D})\mathbf{s}}, \; C_2 := e(g_1, g_2)^{\mathbf{s}^\top \pi_{\mathrm{L}}(\mathbf{D})^\top \mathbf{k}} \cdot \mathrm{M} \right).$$

– $\mathsf{Dec}(\mathrm{MPK}, \mathrm{SK}, \mathrm{CT})$. Let $\mathrm{SK} = (K_0, K_1)$ and $\mathrm{CT} = (C_0, C_1, C_2)$. Output $\mathrm{M} := C_2 \cdot e(C_1, K_0)/e(C_0, K_1)$.

One may argue that the d-LinAI assumption is not standard and quite complex. We show that, when setting $d = 2$, we obtain the following concrete assumption.

Assumption 5 (2-LinAI) *For any p.p.t. adversary $\mathscr{A}$, the following advantage function is negligible in k,*

$$\mathsf{Adv}_{\mathscr{A}}^{\text{2-LinAI}}(k) := \left| \Pr[\mathscr{A}(D, \mathrm{AUX}, T_0) = 1] - \Pr[\mathscr{A}(D, \mathrm{AUX}, T_1) = 1] \right|,$$

where

$$D := \left(\mathscr{G}, g_1, g_2, g_1^{a_1}, g_1^{a_2}, g_1^{a_3}, g_1^{a_1 s_1}, g_1^{a_2 s_2} \right), \; \mathrm{AUX} := \left(g_2^{a_3}, g_2^{a_1} \right),$$

$$T_0 := g_1^{a_3(s_1+s_2)}, \; T_1 := g_1^{a_3(s_1+s_2)+s_3}$$

and $\mathscr{G} := (p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$ and $s_1, s_2 \leftarrow \mathbb{Z}_p$, $a_1, a_2, a_3, s_3 \leftarrow \mathbb{Z}_p^*$.

It is easy to verify that this special instantiation is implied by the External Decision Linear Assumption [ACD+12] (c.f. Appendix A.2). Motivated by this observation, we remark that we may build the above IBE system using *symmetric* bilinear pairings and base the security on the well-known and *standard* Decisional Linear Assumption (c.f. Appendix A.3), where $G_1 = G_2$ and auxiliary input AUX in G_2 is automatically revealed to the adversary.

Acknowledgement. We want to thank Hoeteck Wee for helpful discussions and the anonymous reviewers of AsiaCrypt 2015 for helpful comments on an earlier draft of this paper. This work is supported by the National Natural Science Foundation of China (Grant Nos. 61472142, 61411146001, 61321064, 61371083, 61373154, 61172085, 61170080, U1135004), 973 Program (No. 2014CB360501), Science and Technology Commission of Shanghai Municipality (Grant Nos. 14YF1404200, 13JC1403500), the Specialized Research Fund for the Doctoral Program of Higher Education of China through the Prioritized Development Projects under Grant 20130073130004, Guangdong Provincial Natural Science Foundation (No. 2014A030308006), Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (2011).

References

[ABB10a] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In *Advances in Cryptology - EUROCRYPT 2010*, pages 553–572, 2010. 7

[ABB10b] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In *Advances in Cryptology - CRYPTO 2010*, pages 98–115, 2010. 7

[ACD+12] Masayuki Abe, Melissa Chase, Bernardo David, Markulf Kohlweiss, Ryo Nishimaki, and Miyako Ohkubo. Constant-size structure-preserving signatures: Generic constructions and simple assumptions. In *Advances in Cryptology–ASIACRYPT 2012*, pages 4–24. Springer, 2012. 2, 6, 43, 46

[AHY15] Nuttapong Attrapadung, Goichiro Hanaoka, and Shota Yamada. A framework for identity-based encryption with almost tight security. *IACR Cryptology ePrint Archive*, 2015. 7

[Att14] Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In *Advances in Cryptology - EUROCRYPT 2014*, pages 557–577, 2014. 3, 7

[AY15] Nuttapong Attrapadung and Shota Yamada. Duality in abe: Converting attribute based encryption for dual predicate and dual policy via computational encodings. In *Topics in Cryptology—CT-RSA 2015*, pages 87–105. Springer, 2015. 3

[BB04a] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2004*, pages 223–238, 2004. 3, 7

[BB04b] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology - CRYPTO 2004*, pages 443–459, 2004. 7

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *Advances in Cryptology–CRYPTO 2004*, pages 41–55. Springer, 2004. 46

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology - CRYPTO 2001*, pages 213–229, 2001. 3, 7, 9

[BKP14] Olivier Blazy, Eike Kiltz, and Jiaxin Pan. (hierarchical) identity-based encryption from affine message authentication. In *Advances in Cryptology–CRYPTO 2014*, pages 408–425. Springer, 2014. 3, 6, 7

[BWY11] Mihir Bellare, Brent Waters, and Scott Yilek. Identity-based encryption secure against selective opening attack. In *TCC 2011*, pages 235–252, 2011. 3

[CGW15] Jie Chen, Romain Gay, and Hoeteck Wee. Improved dual system abe in prime-order groups via predicate encodings. In *Advances in Cryptology-EUROCRYPT 2015*, pages 595–624. Springer, 2015. 3, 4, 6, 7, 30

[CLL+12] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Hoeteck Wee. Shorter IBE and signatures via asymmetric pairings. In *Pairing-Based Cryptography - Pairing 2012*, pages 122–140, 2012. 3

[Coc01] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In *Cryptography and Coding, 8th IMA International Conference, Cirencester, UK, December 17-19, 2001, Proceedings*, pages 360–363, 2001. 7

[CW13] Jie Chen and Hoeteck Wee. Fully, (almost) tightly secure IBE and dual system groups. In *Advances in Cryptology - CRYPTO 2013 - Part II*, pages 435–460, 2013. 3, 4, 5, 6, 7, 10, 11, 12, 13, 19

[CW14] Jie Chen and Hoeteck Wee. Dual system groups and its applications - compact HIBE and more. *IACR Cryptology ePrint Archive*, 2014:265, 2014. 3, 4, 5, 7

[EHK+13] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Rafols, and Jorge Villar. An algebraic framework for diffie-hellman assumptions. In *Advances in Cryptology–CRYPTO 2013*, pages 129–147. Springer, 2013. 12, 29

[GCTC15] Junqing Gong, Zhenfu Cao, Shaohua Tang, and Jie Chen. Extended dual system group and shorter unbounded hierarchical identity based encryption. *Designs, Codes and Cryptography*, 2015. DOI 10.1007/s10623-015-0117-z. 7

[Gen06] Craig Gentry. Practical identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2006*, pages 445–464, 2006. 7

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 197–206, 2008. 7

[HKS15] Dennis Hofheinz, Jessica Koch, and Christoph Striecks. Identity-based encryption with (almost) tight security in the multi-instance, multi-ciphertext setting. In *Public-Key Cryptography - PKC 2015*, 2015. 3, 4, 5, 6, 7, 8, 9, 11, 23, 48, 49, 51, 52

[JR13] Charanjit S. Jutla and Arnab Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In *Advances in Cryptology - ASIACRYPT 2013 -Part I*, pages 1–20, 2013. 3

[JR14] Charanjit S Jutla and Arnab Roy. Switching lemma for bilinear tests and constant-size nizk proofs for linear subspaces. In *Advances in Cryptology–CRYPTO 2014*, pages 295–312. Springer, 2014. 2, 12, 46

[Lew12] Allison B. Lewko. Tools for simulating features of composite order bilinear groups in the prime order setting. In *Advances in Cryptology - EUROCRYPT 2012*, pages 318–335, 2012. 3, 4, 7

[LOS+10] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *Advances in Cryptology - EUROCRYPT 2010*, pages 62–91, 2010. 3, 6, 30

[LW11] Allison Lewko and Brent Waters. Unbounded hibe and attribute-based encryption. In *Advances in Cryptology–EUROCRYPT 2011*, pages 547–567. Springer, 2011. 7

[LW12] Allison B. Lewko and Brent Waters. New proof methods for attribute-based encryption: Achieving full security through selective techniques. In *Advances in Cryptology - CRYPTO 2012*, pages 180–198, 2012. 3, 4

[NR04] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004. 3

[OT08] Tatsuaki Okamoto and Katsuyuki Takashima. Homomorphic encryption and signatures from vector decomposition. In *Pairing-Based Cryptography–Pairing 2008*, pages 57–74. Springer, 2008. 6, 30

[OT09] Tatsuaki Okamoto and Katsuyuki Takashima. Hierarchical predicate encryption for inner-products. In *Advances in Cryptology–ASIACRYPT 2009*, pages 214–231. Springer, 2009. 6, 30

[OT12] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure unbounded inner-product and attribute-based encryption. In *Advances in Cryptology - ASIACRYPT 2012*, pages 349–366, 2012. 3, 4

[RCS12] Somindu C. Ramanna, Sanjit Chatterjee, and Palash Sarkar. Variants of waters' dual system primitives using asymmetric pairings - (extended abstract). In *Public Key Cryptography - PKC 2012*, pages 298–315, 2012. 3

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology, Proceedings of CRYPTO '84*, pages 47–53, 1984. 7

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology–EUROCRYPT 97*, pages 256–266. Springer, 1997. 29, 46

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2005*, pages 114–127, 2005. 3, 7

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In *Advances in Cryptology - CRYPTO 2009*, pages 619–636, 2009. 3, 7

[Wee14] Hoeteck Wee. Dual system encryption via predicate encodings. In *Theory of Cryptography 2014*, pages 616–637, 2014. 3

A More about Bilinear Groups and Related Assumptions

A.1 d-Lifted Linear Assumption [JR14]

We assume a prime-order (asymmetric) bilinear group generator $\mathsf{GrpGen}(1^k)$ taking security parameter 1^k as input and outputting $\mathcal{G} := (p, G_1, G_2, G_T, e)$. We state the d-Lifted Linear Assumption in G_1 as follows, the analogous assumption in G_2 can be defined by exchanging the role of G_1 and G_2.

Assumption 6 (d-Lifted Linear Assumption in G_1) *For any p.p.t. adversary $\mathcal{A}$, the following advantage function is negligible in k,*

$$\mathsf{Adv}^{d\text{-Lin}}_{\mathcal{A}}(k) := \left|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\right|,$$

where

$$D := \left(\mathcal{G}, g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, g_1^{b_1}, \cdots, g_1^{b_d}, g_1^{a_1 s_1}, \ldots, g_1^{a_d s_d}\right),$$

$$T_0 := g_1^{b_1 s_1 + \cdots + b_d s_d}, \; T_1 := g_1^{b_1 s_1 + \cdots + b_d s_d + \boxed{s_{d+1}}}$$

and $\mathcal{G} := (p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$, $a_1, \ldots, a_d, b_1, \cdots, b_d \leftarrow \mathbb{Z}_p^*$, $s_1, \ldots, s_d \leftarrow \mathbb{Z}_p$, $s_{d+1} \leftarrow \mathbb{Z}_p^*$.

A.2 External Decision Linear Assumption [ACD+12]

We assume a prime-order (asymmetric) bilinear group generator $\mathsf{GrpGen}(1^k)$ taking security parameter 1^k as input and outputting $\mathcal{G} := (p, G_1, G_2, G_T, e)$. We state the external decisional linear assumption in G_1 as follows, the analogous assumption in G_2 can be defined by exchanging the role of G_1 and G_2.

Assumption 7 (External Decision Linear Assumption in G_1) *For any p.p.t. adversary $\mathcal{A}$, the following advantage function is negligible in k,*

$$\mathsf{Adv}^{\text{XDLIN}}_{\mathcal{A}}(k) := \left|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\right|,$$

where

$$D := \left(\mathcal{G}, g_1, g_2, \begin{array}{c} g_1^{a_1}, g_1^{a_2}, g_1^{a_3}, g_1^{a_1 s_1}, g_1^{a_2 s_2} \\ g_2^{a_1}, g_2^{a_2}, g_2^{a_3}, g_2^{a_1 s_1}, g_2^{a_2 s_2} \end{array}\right), \; T_0 := g_1^{a_3(s_1+s_2)}, \; T_1 := g_1^{a_3(s_1+s_2)+\boxed{s_3}}$$

and $\mathcal{G} := (p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$, $s_1, s_2 \leftarrow \mathbb{Z}_p$ *and* $a_1, a_2, a_3, s_3 \leftarrow \mathbb{Z}_p^*$.

A.3 Symmetric Bilinear Groups and Decisional Linear Assumption

A prime-order symmetric bilinear group generator $\mathsf{sGrpGen}(1^k)$ takes security parameter 1^k as input and outputs $\mathcal{G} := (p, G, G_T, e)$, where G and G_T are finite cyclic groups of prime order p, and $e : G \times G \to G_T$ is a non-degenerated and efficiently computable bilinear map. We let g and $g_T := e(g, g)$ be a generator of G and G_T, respectively. We state the decisional linear assumption as follows.

Assumption 8 (Decisional Linear Assumption) *For any p.p.t. adversary $\mathcal{A}$, the following advantage function is negligible in k,*

$$\mathsf{Adv}^{\text{DLIN}}_{\mathcal{A}}(k) := \left|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\right|,$$

where

$$D := \left(\mathcal{G}, g, g^{a_1}, g^{a_2}, g^{a_3}, g^{a_1 s_1}, g^{a_2 s_2}\right), \; T_0 := g^{a_3(s_1+s_2)}, \; T_1 := g^{a_3(s_1+s_2)+\boxed{s_3}}$$

and $\mathcal{G} := (p, G, G_T, e) \leftarrow \mathsf{sGrpGen}(1^k)$, $s_1, s_2 \leftarrow \mathbb{Z}_p$ *and* $a_1, a_2, a_3, s_3 \leftarrow \mathbb{Z}_p^*$.

A.4 Generic Security for d-Linear Assumption with Auxiliary Input

This section shows that the d-linear assumption with auxiliary input defined in Section 8 holds in the generic group model [Sho97]. In the model, each group element (in G_1, G_2, or G_T) is associated with an arbitrary string with no structure, therefore equality of two elements can be detected by adversary $\mathcal{A}$. However group operations and pairing operations are done through respective oracles. We prove the following lemma based on the proof for the DLIN assumption [BBS04].

Lemma 17 (Generic Security for d-LinAI) *Let $\mathscr{A}$ be an algorithm that solves the d-linear problem with auxiliary input in the generic group model. Assume that $\xi_1,\ \xi_2,\ \xi_T$ are random encoding functions for $G_1,\ G_2,\ G_T$. If $\mathscr{A}$ makes a total of at most q queries to the oracles computing the group action in $G_1,\ G_2,\ G_T$ and the bilinear pairing e, then*

$$\left|\Pr\left[\mathscr{A}\left(\begin{array}{l} p,\ \xi_1(1),\ \xi_1(a_1),\ldots,\xi_1(a_d),\ \xi_1(a_{d+1}),\\ \xi_2(1),\ \xi_2(a),\ \xi_2(aa_1^{-1}a_{d+1}),\ldots,\xi_2(aa_{d/2}^{-1}a_{d+1}),\\ \xi_1(a_1s_1),\ldots,\xi_1(a_ds_d),\ \xi_1(t_0),\ \xi_1(t_1)\end{array}\right)=b\right]-\frac{1}{2}\right|\leqslant\frac{(d/4+1)(q+5d/2+6)^2}{2p},$$

where $a_1,\ldots,a_d,a_{d+1},s_1,\ldots,s_d,s_{d+1}\leftarrow\mathbb{Z}_p^*$, $a:=a_1\cdots a_{d/2}$, $b\leftarrow\{0,1\}$, $t_b:=a_{d+1}(s_1+\cdots+s_d)$, and $t_{1-b}:=a_{d+1}(s_1+\cdots+s_d)+s_{d+1}$. (Note that $aa_i^{-1}a_{d+1}=a_1\cdots a_{i-1}a_{i+1}\cdots a_{d/2}a_{d+1}$ for $i\in[d/2]$ actually contain no inverse of element at all.)

Proof. We first describe an algorithm $\mathscr{B}$ answering $\mathscr{A}$'s oracle queries. Algorithm $\mathscr{B}$ maintains three lists:

$$\mathscr{L}_1:=\left\{(F_{1,i},\xi_{1,i}):i\in[\tau_1]\right\},\ \mathscr{L}_2:=\left\{(F_{2,i},\xi_{2,i}):i\in[\tau_2]\right\},\ \mathscr{L}_T:=\left\{(F_{T,i},\xi_{T,i}):i\in[\tau_T]\right\},$$

where $F_{*,*}\in\mathbb{Z}_p[A_1,\ldots,A_d,A_{d+1},S_1,\ldots,S_d,S_{d+1},T_0,T_1]$ and $\xi_{*,*}\in\{0,1\}^*$. Algorithm $\mathscr{B}$ proceeds as follows.

Initialize. Algorithm $\mathscr{B}$ sets

$$\begin{array}{l}F_{1,1}:=1,\ F_{1,2}:=A_1,\ \ldots,\ F_{1,d+1}:=A_d,\ F_{1,d+2}:=A_{d+1},\\ F_{2,1}=1,\ F_{2,2}:=A_1\cdots A_{d/2},\ F_{2,3}:=A_2\cdots A_{d/2}A_{d+1},\ldots,F_{2,d/2+2}:=A_1\cdots A_{d/2-1}A_{d+1},\\ F_{1,d+3}:=A_1S_1,\ldots,F_{1,2d+2}:=A_dS_d,\ F_{1,2d+3}:=T_0,\ F_{1,2d+4}:=T_1.\end{array}$$

Set $\xi_{1,i}$ for $i\in[2d+4]$ and $\xi_{2,i}$ for $i\in[d/2+2]$ to be distinct strings from $\{0,1\}^*$. Send $\xi_{1,i}$ and $\xi_{2,i}$ to adversary $\mathscr{A}$. We let $\tau_1=2d+4$, $\tau_2=d/2+2$, and $\tau_T=0$.

Group Operation. On input $(\xi_{1,i},\xi_{1,j},\mathtt{Type})$ where $i,j\in[\tau_1]$ and $\mathtt{Type}\in\{\mathtt{Mult},\mathtt{Div}\}$, compute $F_{1,\tau_1+1}:=F_{1,i}\pm F_{1,j}$ according to parameter $\mathtt{Type}$. If there exists $k\in[\tau_1]$ such that $F_{1,\tau_1+1}=F_{1,k}$, set $\xi_{1,\tau_1+1}:=\xi_{1,k}$; otherwise, pick a new string (different from all existing $\xi_{1,i}$) for ξ_{1,τ_1+1}. Return ξ_{1,τ_1+1} to adversary $\mathscr{A}$, and update $\mathscr{L}_1:=\mathscr{L}_1\cup\{(F_{1,\tau_1+1},\xi_{1,\tau_1+1})\}$ and $\tau_1:=\tau_1+1$. We also answer query of form $(\xi_{2,i},\xi_{2,j},\mathtt{Type})$ for G_2 and $(\xi_{T,i},\xi_{T,j},\mathtt{Type})$ for G_T in the same way as above.

Pairing Operation. On input $(\xi_{1,i},\xi_{2,j})$ where $i\in[\tau_1]$ and $j\in[\tau_2]$, compute $F_{T,\tau_T+1}:=F_{1,i}\cdot F_{2,j}$. If there exists $k\in[\tau_T]$ such that $F_{T,\tau_T+1}=F_{T,k}$, set $\xi_{T,\tau_T+1}:=\xi_{T,k}$; otherwise, pick a new string (different from all existing $\xi_{T,i}$) for ξ_{T,τ_T+1}. Return ξ_{T,τ_T+1} to adversary $\mathscr{A}$, and update $\mathscr{L}_T:=\mathscr{L}_T\cup\{(F_{T,\tau_T+1},\xi_{T,\tau_T+1})\}$ and $\tau_T:=\tau_T+1$.

Finalize. When adversary $\mathscr{A}$ returns a bit b', algorithm $\mathscr{B}$ samples $a_1,\ldots,a_d,\ a_{d+1},\ s_1,\ldots,s_d,\ s_{d+1}\leftarrow\mathbb{Z}_p^*$, $b\leftarrow\{0,1\}$ and assigns $A_1:=a_1,\ldots,A_d:=a_d,\ A_{d+1}:=a_{d+1},\ S_1:=s_1,\ldots,S_d:=s_d,\ S_{d+1}:=s_{d+1},\ T_b:=a_{d+1}(s_1+\cdots+s_d),\ T_{1-b}:=a_{d+1}(s_1+\cdots+s_d)+s_{d+1}$.

We now argue that the last step of the simulation, i.e., assigning all arguments in polynomials, did not violate the equality relation in our simulation. At first, we see that the assignments for T_0 and T_1 have deviated from our simulation where both of them should be independent of other arguments. We may view, from the polynomial-level, that $T_b:=A_{d+1}(S_1+\cdots+S_d)$ and $T_{1-b}:=A_{d+1}(S_1+\cdots+S_d)+S_{d+1}$. Since S_{d+1} appears nowhere else, the argument T_{1-b} is still independent and won't break the equality relation we have established. It remains to investigate the other case.

We claim that T_b remains independent (from polynomial-level) even if it is now algebraically connected to other arguments. By the (bi-)linearity, we just need to consider the *initial* elements in $\mathscr{L}_1$ and all combination (pairing) of *initial* elements in $\mathscr{L}_1$ and $\mathscr{L}_2$. Firstly, since $A_{d+1}S_i$ never appears in $\mathscr{L}_1$, adversary $\mathscr{A}$ can not derive T_b from initial elements of $\mathscr{L}_1$ (not including T_b itself) by group operations. In other word, it preserves the equality relation in $\mathscr{L}_1$. Secondly, we observe that pairing T_b with initial elements in $\mathscr{L}_2$ results polynomials with one of the following terms

$$A_{d+1}S_d,\ A_1\cdots A_{d/2}A_{d+1}S_d,\ A_1\cdots A_{d/2}A_{d+1}^2S_d/A_i,\ i\in[d/2].$$

To obtain these terms without T_b, $\mathscr{A}$ must ask pairing operation oracle with first parameter being $\xi_{1,2d+2}$ (i.e., A_dS_d) since this is the only terms with S_d. However it will introduce an additional term A_d into the result because no element in $\mathscr{L}_2$ can cancel it out through pairing. Therefore we can say the equality relation in $\mathscr{L}_T$ is also preserved.

Next, we see that our assignments for all polynomials may also violate the equality relation. In particular, we consider the following events, denoted by Coll, there exist i,j such that

– $F_{1,i}(a_1,\ldots,a_d,a_{d+1},s_1,\ldots,s_d,s_{d+1}) = F_{1,j}(a_1,\ldots,a_d,a_{d+1},s_1,\ldots,s_d,s_{d+1})$ but $F_{1,i} \neq F_{1,j}$; or

– $F_{2,i}(a_1,\ldots,a_d,a_{d+1},s_1,\ldots,s_d,s_{d+1}) = F_{2,j}(a_1,\ldots,a_d,a_{d+1},s_1,\ldots,s_d,s_{d+1})$ but $F_{2,i} \neq F_{2,j}$; or

– $F_{T,i}(a_1,\ldots,a_d,a_{d+1},s_1,\ldots,s_d,s_{d+1}) = F_{T,j}(a_1,\ldots,a_d,a_{d+1},s_1,\ldots,s_d,s_{d+1})$ but $F_{T,i} \neq F_{T,j}$.

It is not hard to see that if the event occurs two different polynomials should be recognized as identical after assignment. Since total degree of polynomials in $\mathscr{L}_1$ is at most 2, and this parameter is at most $d/2$ and $d/2+2$ in $\mathscr{L}_2$ and $\mathscr{L}_T$, respectively, the probability of the event is bounded as

$$\Pr[\mathsf{Coll}] \leqslant \binom{\tau_1}{2}\frac{2}{p} + \binom{\tau_2}{2}\frac{d/2}{p} + \binom{\tau_T}{2}\frac{d/2+2}{p} \leqslant \frac{(d/4+1)(q+5d/2+6)^2}{p},$$

following the Schwartz-Zippel Lemma.

We observe that if Coll did not occur, the simulation is perfect and the adversary $\mathscr{A}$ can guess b correctly with probability $1/2$, since b is picked after $\mathscr{A}$ have guessed it. Therefore we have

$$\begin{aligned} |\Pr[\mathscr{A}=b]-1/2| &\leqslant |\Pr[\mathscr{A}=b|\mathsf{Coll}]\Pr[\mathsf{Coll}] + \Pr[\mathscr{A}=b|\neg\mathsf{Coll}](1-\Pr[\mathsf{Coll}]) - 1/2| \\ &\leqslant |\Pr[\mathscr{A}=b|\mathsf{Coll}]\Pr[\mathsf{Coll}] - 1/2\Pr[\mathsf{Coll}]| \\ &\leqslant |1/2\Pr[\mathsf{Coll}]| \leqslant (d/4+1)(q+5d/2+6)^2/2p. \end{aligned}$$

This proved the lemma. □

B IBE from Revised ENDSG in Section 3

We have claimed that our revised ENDSG (in Section 3) implies an almost-tight IBE in the multi-instance, multi-ciphertext setting. Both construction and its security proof are nearly the same as those described in [HKS15]. For completeness and reference, we present both the construction and the organization of its proof in this section.

B.1 Construction

We assume the identity space is $\{0,1\}^n$ for some $n \in \mathbb{Z}^+$ and let n be system-level parameter SYS.

– $\mathsf{Param}(1^k, n)$. Sample $(\mathrm{PP}, \mathrm{SP}) \leftarrow \mathsf{SampP}(1^k, 2n)$ and output

$$\mathrm{GP} := \mathrm{PP}.$$

We assume that GP also contains k and n.

– $\mathsf{Setup}(\mathrm{GP})$. Sample $\mathrm{MSK} \leftarrow \mathbb{H}$ and output

$$\mathrm{MPK} := \big(\mathrm{PP}, \mu(\mathrm{MSK})\big) \quad \text{and} \quad \mathrm{MSK}.$$

– $\mathsf{KeyGen}(\mathrm{MPK}, \mathrm{MSK}, \mathbf{y})$. Let $\mathbf{y} = (y_1,\ldots,y_n) \in \{0,1\}^n$. Sample

$$(h_0, h_1, \ldots, h_{2n}) \leftarrow \mathsf{SampH}(\mathrm{PP})$$

and output

$$\mathrm{SK}_{\mathbf{y}} := \Big(K_0 := h_0,\ K_1 := \mathrm{MSK}\cdot h_{2-y_1}\cdots h_{2n-y_n}\Big).$$

– $\mathsf{Enc}(\mathrm{MPK}, \mathbf{x}, \mathrm{M})$. Let $\mathbf{x} = (x_1,\ldots,x_n) \in \{0,1\}^n$ and $\mathrm{M} \in \mathbb{G}_T$. Sample random coin s and compute

$$(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\mathrm{PP}; s) \quad \text{and} \quad g'_T \leftarrow \mathsf{SampGT}(\mu(\mathrm{MSK}); s).$$

Output

$$\mathrm{CT}_{\mathbf{x}} := \Big(C_0 := g_0,\ C_1 := g_{2-x_1}\cdots g_{2n-x_n},\ C_2 := g'_T \cdot \mathrm{M}\Big).$$

– $\mathsf{Dec}(\mathrm{MPK}, \mathrm{SK}, \mathrm{CT})$. Let $\mathrm{SK} = (K_0, K_1)$ and $\mathrm{CT} = (C_0, C_1, C_2)$. Output

$$\mathrm{M} := C_2 \cdot \frac{e(C_1, K_0)}{e(C_0, K_1)}.$$

Correctness. For any $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$, one may check that

$$\frac{e(C_1, K_0)}{e(C_0, K_1)} = \frac{e(g_{2-x_1} \cdots g_{2n-x_n}, h_0)}{e(g_0, \mathrm{MSK} \cdot h_{2-y_1} \cdots h_{2n-y_n})} = \big(e(g_0, \mathrm{MSK})\big)^{-1} = \big(g'_T\big)^{-1},$$

where the second equality follows the *associative* property, and the last one follows the *projective* property.

B.2 Security Proof

We just present here the main theorem and the sequence of games with definitions for various auxiliary algorithms and distributions. One may easily derive the detailed proofs according to Hofheinz *et al.*'s proof [HKS15].

Theorem 1 *Assuming an extended nested dual system group defined as Section 3, the IBE scheme shown above is weak adaptively secure in the multi-instance, multi-ciphertext setting. More concretely, for any adversary $\mathscr{A}$ making at most q_K key extraction queries and at most q_C challenge queries for pairwise distinct challenge identity against at most λ instances, there exist adversaries $\mathscr{B}_1$, $\mathscr{B}_2$, and $\mathscr{B}_3$ such that*

$$\mathsf{Adv}^{\mathrm{IBE}}_{\mathscr{A}}(k, \lambda, q_K, q_C, 1) \leqslant \mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{B}_1}(k, q_C) + 2n \cdot \mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{B}_2}(k, q_C, q_K) + n \cdot \mathsf{Adv}^{\mathrm{NH}}_{\mathscr{B}_3}(k, q_C, q_K) + 2^{-\Omega(k)},$$

where $\max_{i \in [3]} \mathsf{Time}(\mathscr{B}_i) \approx \mathsf{Time}(\mathscr{A}) + (\lambda + q_C + q_K) \cdot \mathsf{poly}(k, n)$ *and* $\mathsf{poly}(k, n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Auxiliary Algorithms. We describe two auxiliary algorithms:

- $\overline{\mathsf{KeyGen}}(\mathrm{PP}, \overline{\mathrm{MSK}}, \mathbf{y}; \mathbf{t})$. Let $\overline{\mathrm{MSK}} \in \mathbb{H}$, $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$, and $\mathbf{t} = (T_0, T_1, \ldots, T_{2n}) \in \mathbb{H}^{2n+1}$, output
$$\mathrm{SK}_{\mathbf{y}} := \big(K_0 := T_0,\ K_1 := \overline{\mathrm{MSK}} \cdot T_{2-y_1} \cdots T_{2n-y_n}\big).$$
- $\overline{\mathsf{Enc}}(\mathrm{PP}, \mathbf{x}, \mathrm{M}; \overline{\mathrm{MSK}}, \mathbf{t})$. Let $\overline{\mathrm{MSK}} \in \mathbb{H}$, $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$, $\mathrm{M} \in \mathbb{G}_T$, and $\mathbf{t} = (T_0, T_1, \ldots, T_{2n}) \in \mathbb{G}^{2n+1}$, output
$$\mathrm{CT}_{\mathbf{x}} := \big(C_0 := T_0,\ C_1 := T_{2-x_1} \cdots T_{2n-x_n},\ C_2 := e(T_0, \overline{\mathrm{MSK}}) \cdot \mathrm{M}\big).$$

Auxiliary Distributions. We first define two families of random functions $\{\widehat{\mathsf{R}}_i\}_{i \in [0,n]}$ and $\{\widetilde{\mathsf{R}}_i\}_{i \in [0,n]}$ where

$$\widehat{\mathsf{R}}_i : [\lambda] \times \{0,1\}^i \to [\widehat{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})] \quad \text{and} \quad \widetilde{\mathsf{R}}_i : [\lambda] \times \{0,1\}^i \to [\widetilde{\mathsf{SampH}}^*(\mathrm{PP}, \mathrm{SP})]$$

for all $i \in [0, n]$. For simplicity, we may feed a n-bit string into $\widehat{\mathsf{R}}_i(\iota, \cdot)$ and $\widetilde{\mathsf{R}}_i(\iota, \cdot)$. In such a case, we view the i-bit prefix of the input as actual input and simply neglect the remaining bits.

Secondly, for all $(\mathrm{PP}, \mathrm{SP}) \in [\mathsf{SampP}(1^k, 2n)]$, all $\mathrm{MSK} \in \mathbb{H}$, all $\iota \in [\lambda]$, all $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$, and all $\mathrm{M} \in \mathbb{G}_T$, we define four forms of ciphertext $\mathsf{Enc}(\mathrm{MPK}, \mathbf{x}, \mathrm{M})$ in the ιth instance with $\mathrm{MPK} := (\mathrm{PP}, \mu(\mathrm{MSK}))$:

(Normal ciphertext.)

$$\overline{\mathsf{Enc}}(\mathrm{PP}, \mathbf{x}, \mathrm{M}; \mathrm{MSK}, \mathbf{g}),$$

where $\mathbf{g} \leftarrow \mathsf{SampG}(\mathrm{PP})$; more explicitly, the distribution is

$$\left(g_0,\ \prod_{i=1}^{n} g_{2i-x_i},\ e(g_0, \mathrm{MSK}) \cdot \mathrm{M}\right),$$

where $(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\mathrm{PP})$. By the *projective* property, the distribution is indeed identical to the output of real encryption algorithm Enc.

(Pseudo-normal ciphertext.)

$$\overline{\mathsf{Enc}}(\mathrm{PP}, \mathbf{x}, \mathrm{M}; \mathrm{MSK}, \mathbf{g} \cdot \widehat{\mathbf{g}}),$$

where $\mathbf{g} \leftarrow \mathsf{SampG}(\mathrm{PP})$ and $\widehat{\mathbf{g}} \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$; more explicitly, the distribution is

$$\left(g_0 \cdot \widehat{g}_0,\ \prod_{i=1}^{n} \left(g_{2i-x_i} \cdot \widehat{g}_{2i-x_i}\right),\ e(g_0 \cdot \widehat{g}_0, \mathrm{MSK}) \cdot \mathrm{M}\right),$$

where $(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\mathrm{PP})$ and $(\widehat{g}_0, \widehat{g}_1, \ldots, \widehat{g}_{2n}) \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$.

(Semi-functional type-$(\wedge, i)$ ciphertexts for $i \in [0, n]$.)

$$\overline{\mathsf{Enc}}(\mathrm{PP}, \mathbf{x}, \mathrm{M}; \mathrm{MSK} \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{x}) \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{x}), \mathbf{g} \cdot \widehat{\mathbf{g}}),$$

where $\mathbf{g} \leftarrow \mathsf{SampG}(\mathrm{PP})$ and $\widehat{\mathbf{g}} \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$; more explicitly, the distribution is

$$\left(g_0 \cdot \widehat{g}_0,\ \prod_{j=1}^{n} \left(g_{2j-x_j} \cdot \widehat{g}_{2j-x_j} \right),\ e(g_0 \cdot \widehat{g}_0, \mathrm{MSK} \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{x})) \cdot \mathrm{M} \right),$$

where $(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\mathrm{PP})$ and $(\widehat{g}_0, \widehat{g}_1, \ldots, \widehat{g}_{2n}) \leftarrow \widehat{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$. We note that $\widetilde{\mathsf{R}}_i(\iota, \mathbf{x})$ vanishes due to the *orthogonality* property.

(Semi-functional type-$(\sim, i)$ ciphertexts for $i \in [0, n]$.)

$$\overline{\mathsf{Enc}}(\mathrm{PP}, \mathbf{x}, \mathrm{M}; \mathrm{MSK} \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{x}) \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{x}), \mathbf{g} \cdot \widetilde{\mathbf{g}}),$$

where $\mathbf{g} \leftarrow \mathsf{SampG}(\mathrm{PP})$ and $\widetilde{\mathbf{g}} \leftarrow \widetilde{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$; more explicitly, the distribution is

$$\left(g_0 \cdot \widetilde{g}_0,\ \prod_{j=1}^{n} \left(g_{2j-x_j} \cdot \widetilde{g}_{2j-x_j} \right),\ e(g_0 \cdot \widetilde{g}_0, \mathrm{MSK} \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{x})) \cdot \mathrm{M} \right),$$

where $(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\mathrm{PP})$ and $(\widetilde{g}_0, \widetilde{g}_1, \ldots, \widetilde{g}_{2n}) \leftarrow \widetilde{\mathsf{SampG}}(\mathrm{PP}, \mathrm{SP})$. We note that $\widehat{\mathsf{R}}_i(\iota, \mathbf{x})$ vanishes due to the *orthogonality* property.

Finally, for all $(\mathrm{PP}, \mathrm{SP}) \in [\mathsf{SampP}(1^k, 2n)]$, all $\mathrm{MSK} \in \mathbb{H}$, all $\iota \in [\lambda]$, and all $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$, we define two forms of secret key $\mathsf{KeyGen}(\mathrm{MPK}, \mathrm{MSK}, \mathbf{y})$ in the ιth instance with $\mathrm{MPK} := (\mathrm{PP}, \mu(\mathrm{MSK}))$:

(Normal secret key.)

$$\overline{\mathsf{KeyGen}}(\mathrm{PP}, \mathrm{MSK}, \mathbf{y}; \mathbf{h}),$$

where $\mathbf{h} \leftarrow \mathsf{SampH}(\mathrm{PP})$; more explicitly, the distribution is

$$\left(h_0,\ \mathrm{MSK} \cdot \prod_{i=1}^{n} h_{2i-y_i} \right),$$

where $(h_0, h_1, \ldots, h_{2n}) \leftarrow \mathsf{SampH}(\mathrm{PP})$.

(Semi-functional type-i secret key for $i \in [0, n]$.)

$$\overline{\mathsf{KeyGen}}(\mathrm{PP}, \mathrm{MSK} \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{y}) \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{y}), \mathbf{y}; \mathbf{h}),$$

where $\mathbf{h} \leftarrow \mathsf{SampH}(\mathrm{PP})$; more explicitly, the distribution is

$$\left(h_0,\ \mathrm{MSK} \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{y}) \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{y}) \cdot \prod_{i=1}^{n} h_{2i-y_i} \right),$$

where $(h_0, h_1, \ldots, h_{2n}) \leftarrow \mathsf{SampH}(\mathrm{PP})$.

Game Sequence. The proof requires a sequence of games defined as follows.

- Game_0 is identical to the original experiment in Section 2.
- Game_1 is identical to Game_0 except that all challenge ciphertexts are pseudo-normal.
- $\mathsf{Game}_{2.i.0}$ ($i \in [n+1]$) is identical to Game_1 except that all secret keys are type-$(i-1)$ semi-functional and all challenge ciphertexts are type-$(\wedge, i-1)$ semi-functional.
- $\mathsf{Game}_{2.i.1}$ ($i \in [n]$) is identical to $\mathsf{Game}_{2.i.0}$ except that
  - all challenge ciphertexts for identities whose ith bit is 1 are type-$(\sim, i-1)$ semi-functional.
- $\mathsf{Game}_{2.i.2}$ ($i \in [n]$) is identical to $\mathsf{Game}_{2.i.1}$ except that

- all secret keys are type-i semi-functional;
- all challenge ciphertexts for identities whose ith bit is 0 are type-$(\wedge, i)$ semi-functional;
- all challenge ciphertexts for identities whose ith bit is 1 are type-$(\sim, i)$ semi-functional.

– Game_3 is identical to $\mathsf{Game}_{2.(n+1).0}$.

– Game_4 is identical to Game_3 except that all challenge ciphertexts are for random messages.

We sketch the proof. We first move from Game_0 to Game_1 using the LS1 property. We note that $\mathsf{Game}_{2.1.0}$ is the same as Game_1 just with conceptual difference. For $i \in [n]$, we move from $\mathsf{Game}_{2.i.0}$ to $\mathsf{Game}_{2.i.1}$ using the LS2 property, and move from $\mathsf{Game}_{2.i.1}$ to $\mathsf{Game}_{2.i.2}$ using the NH property, and move from $\mathsf{Game}_{2.i.2}$ to $\mathsf{Game}_{2.(i+1).0}$ again using the LS2 property. Then we stop the loop at $\mathsf{Game}_{2.(n+1).0}$ which is defined as Game_3. We finally prove that Game_3 and Game_4 are statistically indistinguishable from the *non-degeneracy* property. It is clear that all challenge ciphertexts in the last game are irrelevant to challenge messages and the adversary's advantage is exactly 0. The main theorem is now proved by combining all above results together.

C IBE from Fine-tuned ENDSG in Section 7

We fine-tuned our revised ENDSG in Section 7. This section is devoted to showing that it also implies an almost-tight IBE in the multi-instance, multi-ciphertext setting. In particular, the construction is almost the same as those shown in Appendix B and thus similar to that in [HKS15], but the proof is slightly different.

C.1 Construction

We assume the identity space is $\{0,1\}^n$ for some $n \in \mathbb{Z}^+$ and let n be system-level parameter SYS.

– $\mathsf{Param}(1^k, n)$. Sample $(\mathrm{PP}, \mathrm{HP}, \mathrm{SP}) \leftarrow \mathsf{SampP}(1^k, 2n)$ and output

$$\mathrm{GP} := (\mathrm{PP}, \mathrm{HP}).$$

We assume that GP also contains k and n.

– $\mathsf{Setup}(\mathrm{GP})$. Sample $\mathrm{MSK}_0 \leftarrow \mathbb{H}$ and output

$$\mathrm{MPK} := \left(\mathrm{PP}, \mu(\mathrm{MSK}_0)\right) \quad \text{and} \quad \mathrm{MSK} := \left(\mathrm{HP}, \mathrm{MSK}_0\right).$$

– $\mathsf{KeyGen}(\mathrm{MPK}, \mathrm{MSK}, \mathbf{y})$. Let $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$. Sample

$$(h_0, h_1, \ldots, h_{2n}) \leftarrow \mathsf{SampH}(\mathrm{PP}, \mathrm{HP})$$

and output

$$\mathrm{SK}_{\mathbf{y}} := \left(K_0 := h_0,\ K_1 := \mathrm{MSK}_0 \cdot h_{2-y_1} \cdots h_{2n-y_n}\right).$$

– $\mathsf{Enc}(\mathrm{MPK}, \mathbf{x}, \mathrm{M})$. Let $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$ and $\mathrm{M} \in \mathbb{G}_T$. Sample random coin s and compute

$$(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\mathrm{PP}; s) \quad \text{and} \quad g'_T \leftarrow \mathsf{SampGT}(\mu(\mathrm{MSK}_0); s).$$

Output

$$\mathrm{CT}_{\mathbf{x}} := \left(C_0 := g_0,\ C_1 := g_{2-x_1} \cdots g_{2n-x_n},\ C_2 := g'_T \cdot \mathrm{M}\right).$$

– $\mathsf{Dec}(\mathrm{MPK}, \mathrm{SK}, \mathrm{CT})$. Let $\mathrm{SK} = (K_0, K_1)$ and $\mathrm{CT} = (C_0, C_1, C_2)$. Output

$$\mathrm{M} := C_2 \frac{e(C_1, K_0)}{e(C_0, K_1)}.$$

Correctness. For any $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$, one may check that

$$\frac{e(C_1, K_0)}{e(C_0, K_1)} = \frac{e(g_{2-x_1} \cdots g_{2n-x_n}, h_0)}{e(g_0, \mathrm{MSK}_0 \cdot h_{2-y_1} \cdots h_{2n-y_n})} = \left(e(g_0, \mathrm{MSK}_0)\right)^{-1} = \left(g'_T\right)^{-1},$$

where the second equality follows the *associative* property, and the last one follows the *projective* property.

C.2 Security Proof

As before, we just present here the main theorem and the sequence of games. One may easily derive the detailed proofs according to Hofheinz *et al.*'s proof [HKS15]. Due to the similarity, we will borrow a lot of definitions from Appendix B.

Theorem 2 *Assume an extended nested dual system group defined as Section 7, the IBE scheme shown above is full-adaptively secure in the multi-instance, multi-ciphertext setting. More concretely, for any adversary $\mathscr{A}$ making at most q_K key extraction queries and at most q_R challenge queries for each of q_C distinct challenge identity against at most λ instances, there exist adversaries $\mathscr{B}_1$, $\mathscr{B}_2$, $\mathscr{B}_3$, $\mathscr{B}_4$ and $\mathscr{B}_5$ such that*

$$\begin{aligned}\mathsf{Adv}^{\mathrm{IBE}}_{\mathscr{A}}(k,\lambda,q_K,q_C,q_R) \leqslant\ & \mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{B}_1}(k,q_Cq_R,q_K)+2n\cdot\left(\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{B}_2}(k,q_Cq_R,q_K)+\mathsf{Adv}^{\mathrm{LS3}}_{\mathscr{B}_3}(k,q_Cq_R,q_K)\right)\\ & +n\cdot\mathsf{Adv}^{\mathrm{NH}}_{\mathscr{B}_4}(k,q_Cq_R,q_K)+\mathsf{Adv}^{\mathrm{ND}}_{\mathscr{B}_5}(k,q_C,q_K,q_R)+2^{-\Omega(k)},\end{aligned}$$

where $\max_{i\in[5]}\mathsf{Time}(\mathscr{B}_i)\approx\mathsf{Time}(\mathscr{A})+(\lambda+q_Cq_R+q_K)\cdot\mathsf{poly}(k,n)$ *and* $\mathsf{poly}(k,n)$ *is independent of* $\mathsf{Time}(\mathscr{A})$.

Auxiliary Algorithms and Distributions. The auxiliary algorithms $\overline{\mathsf{KeyGen}}$ and $\overline{\mathsf{Enc}}$ and the truly random functions $\{\widehat{\mathsf{R}}_i\}_{i\in[0,n]}$ and $\{\widetilde{\mathsf{R}}_i\}_{i\in[0,n]}$ we needed here are identical to those defined in Appendix B.

For all $(\mathrm{PP},\mathrm{HP},\mathrm{SP})\in[\mathsf{SampP}(1^k,2n)]$, all $\mathrm{MSK}_0\in\mathbb{H}$, all $\iota\in[\lambda]$, all $\mathbf{x}=(x_1,\ldots,x_n)\in\{0,1\}^n$, and all $\mathrm{M}\in\mathbb{G}_T$, we define four forms of ciphertext $\mathsf{Enc}(\mathrm{MPK},\mathbf{x},\mathrm{M})$ in the ιth instance with $\mathrm{MPK}:=(\mathrm{PP},\mu(\mathrm{MSK}_0))$. The normal ciphertext, semi-functional type-$(\wedge,i)$ ciphertexts (for $i\in[0,n]$) and semi-functional type-$(\sim,i)$ ciphertexts (for $i\in[0,n]$) are defined as in Appendix B and the last form is defined as follows:

(Semi-functional type-i ciphertexts for $i\in[0,n]$.)

$$\overline{\mathsf{Enc}}(\mathrm{PP},\mathbf{x},\mathrm{M};\mathrm{MSK}_0\cdot\widehat{\mathsf{R}}_i(\iota,\mathbf{x})\cdot\widetilde{\mathsf{R}}_i(\iota,\mathbf{x}),\mathbf{g}\cdot\widehat{\mathbf{g}}\cdot\widetilde{\mathbf{g}}),$$

where $\mathbf{g}\leftarrow\mathsf{SampG}(\mathrm{PP})$, $\widehat{\mathbf{g}}\leftarrow\widehat{\overline{\mathsf{SampG}}}(\mathrm{PP},\mathrm{SP})$ and $\widetilde{\mathbf{g}}\leftarrow\widetilde{\overline{\mathsf{SampG}}}(\mathrm{PP},\mathrm{SP})$; more explicitly, the distribution is

$$\left(g_0\cdot\widehat{g}_0\cdot\widetilde{g}_0,\ \prod_{j=1}^{n}\left(g_{2j-x_j}\cdot\widehat{g}_{2j-x_j}\cdot\widetilde{g}_{2j-x_j}\right),\ e(g_0\cdot\widehat{g}_0\cdot\widetilde{g}_0,\mathrm{MSK}_0\cdot\widehat{\mathsf{R}}_i(\iota,\mathbf{x})\cdot\widetilde{\mathsf{R}}_i(\iota,\mathbf{x}))\cdot\mathrm{M}\right),$$

where $(g_0,g_1,\ldots,g_{2n})\leftarrow\mathsf{SampG}(\mathrm{PP})$, $(\widehat{g}_0,\widehat{g}_1,\ldots,\widehat{g}_{2n})\leftarrow\widehat{\overline{\mathsf{SampG}}}(\mathrm{PP},\mathrm{SP})$ and $(\widetilde{g}_0,\widetilde{g}_1,\ldots,\widetilde{g}_{2n})\leftarrow\widetilde{\overline{\mathsf{SampG}}}(\mathrm{PP},\mathrm{SP})$.

For all $(\mathrm{PP},\mathrm{HP},\mathrm{SP})\in[\mathsf{SampP}(1^k,2n)]$, all $\mathrm{MSK}_0\in\mathbb{H}$, all $\iota\in[\lambda]$, and all $\mathbf{y}=(y_1,\ldots,y_n)\in\{0,1\}^n$, we define two forms of secret key $\mathsf{KeyGen}(\mathrm{MPK},\mathrm{MSK},\mathbf{y})$ in the ιth instance with $\mathrm{MPK}:=(\mathrm{PP},\mu(\mathrm{MSK}_0))$, the normal secret key and the semi-functional type-i secret key for $i\in[0,n]$, in a similar fashion as Appendix B.

Game Sequence. The proof requires a sequence of games defined as follows.

- Game_0 is identical to the original experiment in Section 2.
- Game_1 is identical to Game_0 except that all challenge ciphertexts and secret keys are type-0 semi-functional.
- $\mathsf{Game}_{2.i.0}$ ($i\in[n+1]$) is identical to Game_1 except that all challenge ciphertexts and secret keys are type-$(i-1)$ semi-functional.
- $\mathsf{Game}_{2.i.1}$ ($i\in[n]$) is identical to $\mathsf{Game}_{2.i.0}$ except that
  - all challenge ciphertexts for identities whose ith bit is 1 are type-$(\sim,i-1)$ semi-functional.
- $\mathsf{Game}_{2.i.2}$ ($i\in[n]$) is identical to $\mathsf{Game}_{2.i.1}$ except that
  - all challenge ciphertexts for identities whose ith bit is 0 are type-$(\wedge,i-1)$ semi-functional.
- $\mathsf{Game}_{2.i.3}$ ($i\in[n]$) is identical to $\mathsf{Game}_{2.i.2}$ except that
  - all secret keys are type-i semi-functional;
  - all challenge ciphertexts for identities whose ith bit is 0 are type-$(\wedge,i)$ semi-functional;
  - all challenge ciphertexts for identities whose ith bit is 1 are type-$(\sim,i)$ semi-functional.

- $\mathsf{Game}_{2.i.4}$ ($i \in [n]$) is identical to $\mathsf{Game}_{2.i.3}$ except that
  - all challenge ciphertexts for identities whose ith bit is 0 are type-i semi-functional.
- $\mathsf{Game}_{2.i.5}$ ($i \in [n]$) is identical to $\mathsf{Game}_{2.i.4}$ except that
  - all challenge ciphertexts for identities whose ith bit is 1 are type-i semi-functional.
- Game_3 is identical to $\mathsf{Game}_{2.n+1.0}$.
- Game_4 is identical to Game_3 except that all challenge ciphertexts are for random messages.

We sketch the proof. We first move from Game_0 to Game_1 using the LS1 property and an conceptual transformation. We note that $\mathsf{Game}_{2.1.0}$ is the same as Game_1. For $i \in [n]$, we move from $\mathsf{Game}_{2.i.0}$ to $\mathsf{Game}_{2.i.1}$ using the LS2 property, and move from $\mathsf{Game}_{2.i.1}$ to $\mathsf{Game}_{2.i.2}$ using the LS3 property, the indistinguishability of $\mathsf{Game}_{2.i.2}$ and $\mathsf{Game}_{2.i.3}$ relies on the NH propoerty, then we move from $\mathsf{Game}_{2.i.3}$ to $\mathsf{Game}_{2.i.5}$ again using the LS3 and LS2 property. Note that $\mathsf{Game}_{2.i.5}$ is the same as $\mathsf{Game}_{2.i+1.0}$. Then we stop the loop at $\mathsf{Game}_{2.n+1,0}$ which is defined as Game_3. We finally prove that Game_3 and Game_4 are indistinguishable using the ND property. It is clear that all challenge ciphertexts in the last game are irrelevant to challenge messages and the adversary's advantage is exactly 0. The main theorem is now proved by combining all above results together.