Extended Nested Dual System Groups, Revisited

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

August 20, 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 setting. However only the composite-order instantiation was provided and more efficient prime-order instantiations are absent. The paper fills the blank by presenting two constructions.

- We revisit the notion of ENDSG and realize it using asymmetric 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 the multi-instance, multi-ciphertext scenario under the d-linear assumption (in the asymmetric setting). We further extended the ENDSG to capture stronger security notions, including B-weak adaptive security and full adaptive security, and show that our prime-order instantiation is B-weak adaptive secure without any additional assumption and full adaptive secure under the d-linear assumption.
- We also try to provide better solution by fine-tuning ENDSG again and realizing it following the work of Chen, Gay, and Wee [EUROCRYPT 2015]. This leads to an almost-tight fully adaptively secure IBE in the same setting with better performance than our first IBE scheme but requires a non-standard assumption, d-linear assumption with auxiliary input. However we note that, the 2-linear assumption with auxiliary input is implied by the external decisional linear assumption. Or we can realize the second instantiation using *symmetric* bilinear pairings in which case the security relies on standard decisional linear assumption.

Keywords: Dual system groups, Identity based encryptions, 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 **3**
- 1.1 Background and Problem . . . 3
- 1.2 Motivation and Observation . . . 4
- 1.3 Contributions and Techniques . . . 4
- 1.4 Comparison and Discussion . . . 7
- 1.5 Related Work . . . 8
- 1.6 Independent Work . . . 8
- 1.7 Outline . . . 8

2 Preliminaries **9**
- 2.1 Notations . . . 9
- 2.2 Identity Based Encryptions . . . 9

3 Revisiting Extended Nested Dual System Groups **11**

4 Instantiating ENDSG from d-Linear Assumption **13**
- 4.1 Prime-order Bilinear Groups and Extended d-Linear Assumption . . . 13
- 4.2 Construction . . . 15
- 4.3 Left Subgroup Indistinguishability 1 . . . 16
- 4.4 Left Subgroup Indistinguishability 2 . . . 18
- 4.5 Generalized Many-Tuple Lemma . . . 21
- 4.6 Nested-hiding Indistinguishability . . . 22

5 Concrete IBE from d-Linear Assumption **24**

6 Achieving Stronger Security Guarantee **25**
- 6.1 Warmup: Achieving B-weak Adaptive Security . . . 26
- 6.2 Computational Non-degeneracy and Full Adaptive Security . . . 26
- 6.3 Computational Non-degeneracy from d-Linear Assumption . . . 28

7 Fine-Tuning Extended Nested Dual System Groups from Section 3 **30**

8 Instantiating ENDSG from d-Linear Assumption with Auxiliary Input **33**
- 8.1 d-Linear Assumption with Auxiliary Input . . . 33
- 8.2 Construction . . . 34
- 8.3 Left subgroup indistinguishability 1 . . . 35
- 8.4 Left subgroup indistinguishability 2 . . . 37
- 8.5 Left subgroup indistinguishability 3 . . . 40
- 8.6 Nested-hiding indistinguishability . . . 41
- 8.7 Computational Non-degeneracy . . . 44

9 Concrete IBE from d-Linear Assumption with Auxiliary Input **47**

A More about Bilinear Groups and Related Assumptions **50**
- A.1 External Decision Linear Assumption [ACD$^+$12] . . . 50
- A.2 Symmetric Bilinear Groups and Decisional Linear Assumption . . . 51

B IBE from Revised ENDSG in Section 3 **51**
- B.1 Construction . . . 51
- B.2 Security Proof . . . 52

C IBE from Fine-tuned ENDSG in Section 7 **54**
- C.1 Construction . . . 54
- C.2 Security Proof . . . 55

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. 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 is now 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. The proof of Chen and Wee [CW13] 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], Chen and Wee [CW13] 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 setting where the adversary simultaneously attacks multiple challenge identities in multiple IBE instances. In Chen and Wee's paradigm [CW13], the i-th 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 i-th bit, which no longer holds in the multi-instance, multi-ciphertext 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 i-th step starts from ciphertexts with $\wedge$-semi-functional component. We then move the semi-functional components in all ciphertexts for identities whose i-th bit is 1 to the $\sim$-semi-functional space. At this moment, (1) in the $\wedge$-semi-functional space, all ciphertexts share the same i-th bit 0; (2) in the $\sim$-semi-functional space, all ciphertexts share the same i-th 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 an almost-tight IBE in the multi-instance, multi-ciphertext setting using *prime-order* bilinear groups. We emphasize that it is not just a theoretical interest to pursue a solution in the prime-order setting. 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 significantly reduce the size of base field, say 160 bits, without sacrificing the security level. Though 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* proposed by Chen and Wee [CW13], then proposed a general IBE construction from the extended NDSG (ENDSG) in the multi-instance, multi-ciphertext 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 construction proposed in [HKS15]. However we observe that the definition of ENDSG proposed in [HKS15] sets too strong requirements on the algebra structure of underlying groups, which makes it hard to be instantiated using existing technique realizing dual system technique from prime-order bilinear groups.

An ENDSG describes a set of abstract groups with a bunch of structural and computational requirements supporting the 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 highlight 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.)

On the other hand, 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 groups $\mathbb{G}$ and $\mathbb{H}$ [Lew12, LW12, OT12, CW13, CW14, CGW15]. Meanwhile subgroups of $\mathbb{G}$ and $\mathbb{H}$ are naturally simulated by subspaces of the vector space. 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 $(\textsc{pp}, \textsc{sp}) \leftarrow \mathsf{SampP}(k, n)$, the following two distributions are computationally indistinguishable.

$$\{\mathbf{g} : \mathbf{g} \leftarrow \mathsf{SampG}(\textsc{pp})\} \quad \text{and} \quad \left\{\mathbf{g} \cdot \widehat{\mathbf{g}} : \mathbf{g} \leftarrow \mathsf{SampG}(\textsc{pp}),\ \widehat{\mathbf{g}} \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})\right\}.$$

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 since the multi-instance, multi-ciphertext model is considered 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, the resulting ENDSG will still imply an IBE in the multi-instance, multi-ciphertext setting and the limitations on the underlying groups may be removed. As this happens, many existing techniques in the prime-order setting can finally be applied to realize ENDSG and it is now feasible to build an almost-tight IBE in the same setting using prime-order bilinear groups.

1.3 Contributions and Techniques

In this paper, we revisit the definition of ENDSG, and show that the revised ENDSG not only implies an IBE in the multi-instance, multi-challenge 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 the standard model. In summary, we proposed two constructions: 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 has shorter keys and ciphertexts than the first one.

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

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

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

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 we focus on in this paper. On the other hand, it still implies an IBE in the multi-instance, multi-ciphertext setting. The construction and the proof are almost the same as [HKS15] and have not essential difference.

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 the multi-instance, mulit-ciphertext setting. In contrast, an instantiation of our definition with *loose* security reduction 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-linear assumption in the *asymmetric* setting and the reduction is tight in the sense that the security loss is $\mathcal{O}(d)$ and independent of the number of samples, say q in the previous example, given to the adversary. By the generic construction [HKS15] (also c.f. Appendix B), we obtain the first almost-tight IBE in the multi-instance, multi-ciphertext setting in the prime-order setting and fill the blank left in the Hofheinz *et al.*'s work [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-dimensional subspace is the normal space, the next d-dimensional subspace is the $\wedge$-semi-functional space, and the last d-dimensional 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 computational 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-dimensional to d-dimensional which allows us to utilize the technique used in the proof of *right subgroup indistinguishability* property of Chen-Wee's prime-order instantiation of dual system groups [CW14]. So as to support this technical extension, we replace $\widehat{h}^*$ and $\widetilde{h}^*$ with two algorithms $\widehat{\mathsf{SampH}}^*$ and $\widetilde{\mathsf{SampH}}^*$ respectively in our revised ENDSG, and corresponding computational requirements are further revised following the style discussed in the previous paragraph.

Achieving Stronger Security Guarantee. Hofheinz *et al.* [HKS15] achieved weak security from their ENDS-G 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) in their composite-order instantiation.

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 security stronger than the weak one.

1. We extend 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 not hard to prove that our prime-order instantiation has already B-bounded non-degenerated and an ENDSG equipped with B-bounded non-degenerate property implies an IBE with B-weak adaptive security where the adversary allows to make at most B challenge queries for each identity in each instance (c.f. Section 2). Therefore, when building the system based on the d-linear assumption with $d > 1$, we obtain an IBE with strictly stronger security guarantee.

2. We extend the non-degeneracy property to computational version which is essentially similar to the s-BDDH assumption used in [HKS15] and states that the non-degeneracy property holds even when a single $\widehat{h}^*$ works with polynomially many $\widehat{g}_0$'s. We show that an ENDSG equipped with this computational non-degeneracy requirement achieves full security where there is no restriction on the number of challenge queries on a single identity (c.f. Section 2). Luckily, we can prove that our prime-order instantiation (with no modification) is computationally non-degenerated under the d-linear assumption, and no additional assumption is required.

Towards More Efficient Instantiation. Having obtained the first prime-order instantiation of ENDSG, 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 general d-linear assumption in the *asymmetric* setting. 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-linear 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-linear assumption 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 all two semi-functional spaces together has at least d dimension, 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 also re-organize the series of left-subgroup indistinguishability properties in our revised ENDSG. Roughly, we define three LS requirements as: (1) LS1: $\mathbf{g}$ is indistinguishable from $\mathbf{g} \cdot \widehat{\mathbf{g}} \cdot \widetilde{\mathbf{g}}$; (2) LS2: $\mathbf{g} \cdot \widehat{\mathbf{g}} \cdot \widetilde{\mathbf{g}}$ is indistinguishable from $\mathbf{g} \cdot \widetilde{\mathbf{g}}$; (3) LS3: $\mathbf{g} \cdot \widehat{\mathbf{g}} \cdot \widetilde{\mathbf{g}}$ is indistinguishable from $\mathbf{g} \cdot \widehat{\mathbf{g}}$.

In contrast, the first issue is seemingly hard to circumvent. Therefore, we decide to prove the LS2 property under an enhanced d-linear 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 (c.f. Section 8). 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 [ACD+12] (c.f. Section 9 and Appendix A.1), which has been formally introduced and used to build other cryptographic primitives.

Beside that, 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 scheme 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 security reduction from ENDSG to IBE. Though this relaxation of ENDSG has no direct motivation, we thought it will help us to reach simple, clean and efficient solution.

In summary, we have fine-tuned the ENDSG in three aspects: (1) update non-degeneracy requirement; (2) re-organize LS requirements; (3) hide parameters for SampH. We describe the fine-tuned ENDSG in Section 7 and verify in Appendix C that all these modifications will not prevent ENDSG from deriving IBE in the multi-instance, multi-ciphertext 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 and introduced a new method of randomizing the basis. Following the above discussion, we technically work with $2d \times 2d$ matrix and generate the basis using the dual pairing vector space method [OT08, OT09, LOS+10]. Assuming d is even, the first d dimensional subspace is normal space, the remaining two $d/2$ dimensional subspaces act as $\wedge$ semi-functional subspace and $\sim$ semi-functional subspace, respectively. Note that both semi-functional spaces are now smaller (each of them has $d/2$ dimension) but enough for our proof (the entire semi-functional space has d dimension). Finally, the basis is then randomized following [CGW15].

The security of this instantiation is tightly reduced to the d-linear assumption with auxiliary input in the *asymmetric* setting, which leads to an almost-tight fully adaptively secure IBE in the multi-instance, multi-

Table 1: Comparing Efficiency among existing and proposed almost-tight IBE schemes. $|\textsc{mpk}|$, $|\textsc{sk}|$, and $|\textsc{ct}|$ are the size of master public key, user's secret key and ciphertext, respectively. $|G_1|$ and $|G_2|$ indicates the size of elements in the source groups of asymmetric bilinear groups. $|G|$ indicates the size of elements in the source group of symmetric bilinear groups (of prime and composite order). For all cases, $|G_T|$ is the size of elements in the target group of bilinear groups. T_{Enc} and T_{Dec} are the running time of algorithm Enc and Dec, respectively. For simplicity, we just consider the amount of exponentiation and pairing operations. Following the similar style, we let E_1 be the running time of a single exponentiation on group G_1 in the asymmetric bilinear group, E the running time of a single exponentiation on group G in the symmetric bilinear group. For any case, E_T and P indicates the time consumed by a single exponentiation in group G_T and by a single pairing operation, respectively. "MIMC" indicates whether the construction is proven secure in the multi-instance, multi-ciphertext setting. "d-Lin" and "d-LinAI" stand for d-linear assumption and its extension with auxiliary input, respectively. "DLIN", "SXDH" and "sDLIN" stand for the decisional linear assumption in the asymmetric setting, symmetric external Diffie-Hellman assumption, and decisional linear assumption in the symmetric setting, respectively. "Static" means static assumptions in the composite-order bilinear group, such as subgroup decision assumption.

Scheme	Ord of Grp	Assump.	$\lvert\textsc{mpk}\rvert$	$\lvert\textsc{sk}\rvert$	$\lvert\textsc{ct}\rvert$	T_{Enc}	T_{Dec}	MIMC
CW [CW13]	Prime	d-Lin	$2d^2(2n+1)\lvert G_1\rvert+d\lvert G_T\rvert$	$4d\lvert G_2\rvert$	$4d\lvert G_1\rvert+\lvert G_T\rvert$	$4d^2E_1+dE_T$	$4dP$	✗
		DLIN	$(16n+8)\lvert G_1\rvert+2\lvert G_T\rvert$	$8\lvert G_2\rvert$	$8\lvert G_1\rvert+\lvert G_T\rvert$	$16E_1+2E_T$	$8P$	
		SXDH	$(4n+2)\lvert G_1\rvert+\lvert G_T\rvert$	$4\lvert G_2\rvert$	$4\lvert G_1\rvert+\lvert G_T\rvert$	$4E_1+E_T$	$4P$	
HKS [HKS15]	Comp.	Static	$(2n+1)\lvert G\rvert+\lvert G_T\rvert$	$2\lvert G\rvert$	$2\lvert G\rvert+\lvert G_T\rvert$	$2E+E_T$	$2P$	✓
Sec. 5	Prime	d-Lin	$3d^2(2n+1)\lvert G_1\rvert+d\lvert G_T\rvert$	$6d\lvert G_2\rvert$	$6d\lvert G_1\rvert+\lvert G_T\rvert$	$6d^2E_1+dE_T$	$6dP$	✓
		DLIN	$(24n+12)\lvert G_1\rvert+2\lvert G_T\rvert$	$12\lvert G_2\rvert$	$12\lvert G_1\rvert+\lvert G_T\rvert$	$24E_1+2E_T$	$12P$	
		SXDH	$(6n+3)\lvert G_1\rvert+\lvert G_T\rvert$	$6\lvert G_2\rvert$	$6\lvert G_1\rvert+\lvert G_T\rvert$	$6E_1+E_T$	$6P$	
Sec. 9	Prime	d-LinAI	$2d^2(2n+1)\lvert G_1\rvert+d\lvert G_T\rvert$	$4d\lvert G_2\rvert$	$4d\lvert G_1\rvert+\lvert G_T\rvert$	$4d^2E_1+dE_T$	$4dP$	✓
		sDLIN	$(16n+8)\lvert G\rvert+2\lvert G_T\rvert$	$8\lvert G\rvert$	$8\lvert G\rvert+\lvert G_T\rvert$	$16E_1+2E_T$	$8P$	

ciphertext setting with higher efficiency than our first construction. As we have mentioned, the concrete IBE with $d = 2$ is based on the external decision linear assumption [ACD$^+$12]. This 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-tight IBE schemes in the multi-instance,multi-ciphertext 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 constructions in Section 5 based on the d-Lin, DLIN ($d = 2$) and SXDH assumption ($d = 1$), and the prime-order construction from Section 9 based on the d-LinAI assumption in the asymmetric setting and the DLIN assumption in *symmetric* bilinear groups ($d = 2$). As a base line, we also consider the efficiency of prime-order construction by Chen and Wee [CW13] in the Table, which is *not* built for the multi-instant,multi-ciphertext setting.

Hofheinz *et al.*'s construction (see the second row in Table 1) 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 $\textsc{msk}$, $\textsc{sk}$ and $\textsc{ct}$ are smaller and Enc and Dec require less exponentiation operations and pairing operations.

When instantiating our first instantiation (see the third row in Table 1) under the decisional linear assumption, each group element in $\mathbb{G}$ and $\mathbb{H}$ is a 6-dimension vector over the underlying bilinear group 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. These of course increase the size of $\textsc{mpk}$, $\textsc{sk}$ and $\textsc{ct}$ in the derived IBE scheme compared with Chen-Wee's construction (see the first raw in Table 1) in the weak model. And so do the running time of Enc and Dec. We emphasize that in the SXDH case the cost we pay for stronger security model is quite small. In particular, the sizes of $\textsc{sk}$ and $\textsc{ct}$ increase by just 2 group elements (of G_1 or G_2), and the running time of Enc and Dec increase by the cost of just 2 exponentiations and pairings, respectively.

In our second instantiation based on the DLIN assumption in the symmetric bilinear group (see the last row in Table 1), each group element in $\mathbb{G}$ and $\mathbb{H}$ is a vector of just 4-dimension over G. The resulting IBE is of course more efficient than first instantiation under DLIN assumption in the symmetric setting. Furthermore, we remind the reader that the instantiation of Chen-Wee's almost-tight secure IBE [CW13] (*not* in the multi-instance, multi-ciphertext setting) under the DLIN assumption (see the first row in Table 1) also works with 4×4 matrix. In other word, we reach stronger security guarantee for free. However, since the latter two constructions can be realized using *asymmetric* bilinear groups, their concrete efficiency may be still better than our second one. We thought this as the first step towards better constructions and leave it as an open problem to find more efficient constructions under the d-linear assumption in the *asymmetric* setting.

1.5 Related Work

Dual System Groups and Its Variants. Chen and Wee proposed the notion of dual system group [CW14], which captures key algebra 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. Recently, Gong *et al.* [GCTC15] extended the concept of dual system group and derived an unbounded HIBE [LW11] 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 the multi-instance, multi-ciphertext setting. They developed an elegant framework for building almost-tight IBE in the multi-instance, multi-ciphertext 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 group. Their results and ours partially overlap. Their scheme with constant-size ciphertext in prime-order group (i.e., $\Phi_{\mathsf{cc}}^{\mathsf{prime}}$) is similar to our second construction based on decisional linear assumption (in symmetric case) or external decision linear assumption (in asymmetric case) 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 a non-standard assumption, d-linear assumption with auxiliary input. Furthermore, our first construction in Section 5 is full adaptively secure under the d-linear assumption in the asymmetric bilinear group, which is a more general and weaker assumption than the external decision linear assumption 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

The section defines some notations used hereinafter and reviews the definition and the security model for an identity based encryption in the multi-instance, multi-ciphertext setting.

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, both $y \leftarrow \mathsf{Alg}(x; r)$ and $\mathsf{Alg}(x; r) \rightarrow y$ mean that we run the algorithm Alg on input x (probably consisting of more than one arguments) and randomness r, and then assign the result to variable y. 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. We may omit r for brevity when it is irrelevant to our discussion and the distribution is clear from the context.

We use $\mathrm{ord}(G)$ to denote the order of group G. For a vector $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{Z}^n_{\mathrm{ord}(G)}$, we define $y\mathbf{x} := (yx_1, \ldots, yx_n) \in \mathbb{Z}^n_{\mathrm{ord}(G)}$ for any $y \in \mathbb{Z}_{\mathrm{ord}(G)}$, and $g^{\mathbf{x}} := (g^{x_1}, \ldots, g^{x_n}) \in G^n$ for any $g \in G$. Let $\mathbf{e}_i$ denote the vector with 1 on the i-th position and 0 elsewhere. For two vectors of group elements $\mathbf{g} := (g_1, \ldots, g_n) \in G^n$ 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. Furthermore, 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 i-th position is unknown (we take $\perp$ as a placeholder), and $\mathbf{x}|_i$ as its prefix of length i, i.e., $\mathbf{x}|_i := (x_1, \ldots, x_i)$. All notations described in this paragraph also apply to vector-like objects, such as lists.

2.2 Identity Based Encryptions

Algorithms. An identity-based encryption scheme in the multi-instance setting consists of five (probabilistic) polynomial time algorithms defined as follows:

- $\mathsf{Param}(1^k, \textsc{sys}) \rightarrow \textsc{gp}$. The parameter generation algorithm takes as input a security parameter $k \in \mathbb{Z}^+$ in its unary form and a system-level parameter $\textsc{sys}$, and outputs a global parameter $\textsc{gp}$.
- $\mathsf{Setup}(\textsc{gp}) \rightarrow (\textsc{mpk}, \textsc{msk})$. The setup algorithm takes as input a global parameter $\textsc{gp}$, and outputs a master public key $\textsc{mpk}$ and the corresponding master secret key $\textsc{msk}$.
- $\mathsf{KeyGen}(\textsc{mpk}, \textsc{msk}, \mathbf{y}) \rightarrow \textsc{sk}_{\mathbf{y}}$. The key generation algorithm takes as input a master public key $\textsc{mpk}$, a master secret key $\textsc{msk}$ and an identity $\mathbf{y}$, and outputs a secret key $\textsc{sk}_{\mathbf{y}}$ for the identity $\mathbf{y}$.
- $\mathsf{Enc}(\textsc{mpk}, \mathbf{x}, \textsc{m}) \rightarrow \textsc{ct}_{\mathbf{x}}$. The encryption algorithm takes as input a master public key $\textsc{mpk}$, an identity $\mathbf{x}$ and a message $\textsc{m}$, outputs a ciphertext $\textsc{ct}_{\mathbf{x}}$ for the message $\textsc{m}$ under the identity $\mathbf{x}$.
- $\mathsf{Dec}(\textsc{mpk}, \textsc{sk}, \textsc{ct}) \rightarrow \textsc{m}$. The decryption algorithm takes as input a master public key $\textsc{mpk}$, a secret key $\textsc{sk}$ and a ciphertext $\textsc{ct}$, outputs a message $\textsc{m}$ or a special symbol $\perp$ indicating a decryption failure.

The so-called "multi-instance setting" indicates that we are considering a collection of IBE instances, each of which is established under the same global parameter $\textsc{gp}$. We leave the system-level parameter $\textsc{sys}$ undefined for generality. It may depend on concrete constructions or application scenarios.

Remark 1 *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* $\textsc{gp}$*. This global parameter is only fed to algorithm* Setup *to create fresh master public/secret key pairs. And all the other algorithms just take* $\textsc{mpk}$*, 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.*

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 $\textsc{sys}$, any identity $\mathbf{x}$, and any message $\textsc{m}$,

$$\Pr\left[\mathsf{Dec}(\textsc{mpk}, \textsc{sk}_{\mathbf{x}}, \textsc{ct}_{\mathbf{x}}) = \textsc{m} \;\middle|\; \begin{array}{c} \textsc{gp} \leftarrow \mathsf{Param}(1^k, \textsc{sys}) \\ (\textsc{mpk}, \textsc{msk}) \leftarrow \mathsf{Setup}(\textsc{gp}) \\ \textsc{sk}_{\mathbf{x}} \leftarrow \mathsf{KeyGen}(\textsc{mpk}, \textsc{msk}, \mathbf{x}) \\ \textsc{ct}_{\mathbf{x}} \leftarrow \mathsf{Enc}(\textsc{mpk}, \mathbf{x}, \textsc{m}) \end{array}\right] \geqslant 1 - 2^{-\Omega(k)}.$$

The probability space is defined by the random coins consumed by algorithm Param, Setup, 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. Here λ, q_K, q_C is the upper bound on the number of IBE instances, key extraction queries, and challenge *identities*, respectively. The parameter q_R is the maximum amount of challenge ciphertexts for each challenge identity in each instance, which distinguishes the weak security from the full security.

Setup. $\mathscr{C}$ gets $\textsc{gp} \leftarrow \mathsf{Param}(1^k, \textsc{sys})$ and creates λ pairs of master public key and secret key $\{(\textsc{mpk}_\iota, \textsc{msk}_\iota)\}_{\iota \in [\lambda]}$ by independently invoking Setup with $\textsc{gp}$ for λ times. We simply employ integers $1, 2, \ldots, \lambda$ as their instance handlers respectively. All master public keys and their instance handlers $\{(\iota, \textsc{mpk}_\iota)\}_{\iota \in [\lambda]}$ are sent to $\mathscr{A}$. $\mathscr{C}$ also chooses a secret random bit $\beta \in \{0, 1\}$.

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

- For each key extraction query $(\iota, \mathbf{y})$, $\mathscr{C}$ neglects it if, (1) $\iota \notin [\lambda]$, or (2) $|Q_K| > q_K$; otherwise, it outputs $\textsc{sk} \leftarrow \mathsf{KeyGen}(\textsc{mpk}_\iota, \textsc{msk}_\iota, \mathbf{y})$, and updates $Q_K := Q_K \cup \{(\iota, \mathbf{y}, \textsc{sk})\}$.
- For each challenge query $(\iota^*, \mathbf{x}^*, \textsc{m}_0^*, \textsc{m}_1^*)$, $\mathscr{C}$ neglects it if, (1) $\iota^* \notin [\lambda]$, or (2) $|\textsc{m}_0^*| \neq |\textsc{m}_1^*|$, or (3) $\left|Q_{Id}\right| > q_C$ where $Q_{Id} := \left\{(\iota, \mathbf{x}) \,\middle|\, (\iota, \mathbf{x}, \star, \star, \star) \in Q_C\right\}$, or (4) $\left|\left\{(\textsc{m}_0, \textsc{m}_1, \textsc{ct}) \,\middle|\, (\iota^*, \mathbf{x}^*, \textsc{m}_0, \textsc{m}_1, \textsc{ct}) \in Q_C\right\}\right| > q_R$; otherwise, it outputs $\textsc{ct}^* \leftarrow \mathsf{Enc}(\textsc{mpk}_{\iota^*}, \mathbf{x}^*, \textsc{m}_\beta^*)$, and updates $Q_C := Q_C \cup \left\{(\iota^*, \mathbf{x}^*, \textsc{m}_0^*, \textsc{m}_1^*, \textsc{ct}^*)\right\}$.

We emphasize that we permit the adversary to submit a single query repeatedly by letting challenger $\mathscr{C}$ keep track of each query made by adversary $\mathscr{A}$ as well as the reply.

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

We say an adversary $\mathscr{A}$ wins in the above experiment if and only if both $\beta = \beta'$ and $Q_K \cap Q_{Id} = \emptyset$ hold. We use $\mathbf{Exp}^{\mathrm{IBE}}_{\mathscr{A}}(k, \lambda, q_K, q_C, q_R) = 1$ to denote this event. The probability space is defined by the randomness consumed by both $\mathscr{C}$ and $\mathscr{A}$. We define the advantage of $\mathscr{A}$ as

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

Definition 1 (Adaptive Security of IBE) *An identity based encryption is (λ, q_K, q_C, q_R)-adaptively-secure if, for any probabilistic polynomial time 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 the weak and the full security, as (λ, q_K, q_C, B)-adaptive security where $B \geqslant 1$ is an a priori bound.

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 algebra structure which hinders the development of more instantiations beyond using composite-order bilinear groups, especially more efficient prime-order instantiations. (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) providing explicit samples in each computational assumption; (3) generalizing subgroup of $\widehat{h}^*$ and $\widetilde{h}^*$.

Syntax. Our revised extended nested dual system group consists of eight (probabilistic) polynomial time algorithms defined as follows:

- $\mathsf{SampP}(1^k, n)$: On input $(1^k, n)$, output
  - $\textsc{pp}$ containing (1) group description $(\mathbb{G}, \mathbb{H}, \mathbb{G}_T)$ and an admissible bilinear map $e : \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$; (2) an efficient linear map μ defined on $\mathbb{H}$; (3) an efficient sampler for $\mathbb{H}$ and $\mathbb{Z}_{\mathrm{ord}(\mathbb{H})}$, respectively; (4) public parameters for SampG and SampH.
  - $\textsc{sp}$ containing secret parameters for $\widehat{\mathsf{SampG}}$, $\widetilde{\mathsf{SampG}}$, $\widehat{\mathsf{SampH}}^*$ and $\widetilde{\mathsf{SampH}}^*$.
- SampGT: $\mathrm{Im}(\mu) \to \mathbb{G}_T$.
- $\mathsf{SampG}(\textsc{pp})$: Output $\mathbf{g} = (g_0, g_1, \ldots, g_n) \in \mathbb{G}^{n+1}$.
- $\mathsf{SampH}(\textsc{pp})$: Output $\mathbf{h} = (h_0, h_1, \ldots, h_n) \in \mathbb{H}^{n+1}$.
- $\widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$: Output $\widehat{\mathbf{g}} = (\widehat{g}_0, \widehat{g}_1, \ldots, \widehat{g}_n) \in \mathbb{G}^{n+1}$.
- $\widetilde{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$: Output $\widetilde{\mathbf{g}} = (\widetilde{g}_0, \widetilde{g}_1, \ldots, \widetilde{g}_n) \in \mathbb{G}^{n+1}$.
- $\widehat{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})$: Output $\widehat{h}^* \in \mathbb{H}$.
- $\widetilde{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})$: 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 $(\textsc{pp}, \textsc{sp}) \in [\mathsf{SampP}(1^k, n)]$, it is required that:

(Projective.) For all $h \in \mathbb{H}$ and all possible randomness s,

$$\mathsf{SampGT}(\mu(h); s) = e(\mathsf{SampG}_0(\textsc{pp}; s), h).$$

(Associative.) For all $(g_0, g_1, \ldots, g_n) \in [\mathsf{SampG}(\textsc{pp})]$ and all $(h_0, h_1, \ldots, h_n) \in [\mathsf{SampH}(\textsc{pp})]$,

$$e(g_0, h_i) = e(g_i, h_0), \quad \forall i \in [n].$$

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

(Orthogonality.) For all $\widehat{h}^* \in [\widehat{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})]$ and all $\widetilde{h}^* \in [\widetilde{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})]$,

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

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

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

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

(Left subgroup indistinguishability 1 (LS1).) For any probabilistic polynomial time 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

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

and

$$\mathbf{g}_j \leftarrow \mathsf{SampG}(\textsc{pp}),\ \widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp},\textsc{sp}),\quad \forall j\in[q].$$

(Left subgroup indistinguishability 2 (LS2).) For any probabilistic polynomial time 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

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

and

$$\widehat{h}^*_j \leftarrow \widehat{\mathsf{SampH}}^*(\textsc{pp},\textsc{sp}),\ \widetilde{h}^*_j \leftarrow \widetilde{\mathsf{SampH}}^*(\textsc{pp},\textsc{sp}),\quad \forall j\in[q+q'];$$
$$\mathbf{g}'_j \leftarrow \mathsf{SampG}(\textsc{pp}),\ \widehat{\mathbf{g}}'_j \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp},\textsc{sp}),\quad \forall j\in[q];$$
$$\mathbf{g}_j \leftarrow \mathsf{SampG}(\textsc{pp}),\ \widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp},\textsc{sp}),\ \widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\textsc{pp},\textsc{sp}),\quad \forall j\in[q].$$

(Nested-hiding indistinguishability (NH).) For all $\eta\in[\lfloor n/2\rfloor]$ and any probabilistic polynomial time 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

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

and

$$\widehat{h}^*_j \leftarrow \widehat{\mathsf{SampH}}^*(\textsc{pp},\textsc{sp}),\ \widetilde{h}^*_j \leftarrow \widetilde{\mathsf{SampH}}^*(\textsc{pp},\textsc{sp}),\quad \forall j\in[q+q'];$$
$$\widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp},\textsc{sp}),\ \widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\textsc{pp},\textsc{sp}),\quad \forall j\in[q];$$
$$\mathbf{h}_j \leftarrow \mathsf{SampH}(\textsc{pp}),\ \widehat{h}^{**}_j \leftarrow \widehat{\mathsf{SampH}}^*(\textsc{pp},\textsc{sp}),\ \widetilde{h}^{**}_j \leftarrow \widetilde{\mathsf{SampH}}^*(\textsc{pp},\textsc{sp}),\quad \forall j\in[q'].$$

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 2 (notations) *The ENDSG is mainly defined for building identity based encryptions and thus has strong connection to the security model for IBE. We remark that, in the description of LS1, LS2, and NH, the parameter q roughly corresponds to the maximum number of challenge ciphertexts while the parameter q' to the maximum number of key extraction queries.*

Remark 3 (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 algebra structure (say, extending the subspaces they located in 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 4 ($\mathbb{G}$-subgroup and $\mathbb{H}$-subgroup) *Since we provide adequate samples over group $\mathbb{G}^{n+1}$ 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 $NH(\eta)$ using $\mathsf{SampH}(\textsc{pp})$ 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 implies an IBE in the multi-instance, multi-ciphertext 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 further revised the definition of *non-degeneracy* to obtain an ENDSG leading to full adaptive security in Section 6.

4 Instantiating ENDSG from d-Linear Assumption

We give an instantiating 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 IBE in the multi-instance, multi-ciphertext setting using prime-order bilinear groups. We describe this IBE in Section 5.

The section begins with a brief review of prime-order bilinear groups and related computational assumptions. We define an natural extension of standard d-linear assumption in order to alleviate the complexity of the proof. An instantiation of revised ENDSG in the prime-order setting is proposed with a series of proofs showing that it indeed satisfies all correctness and security requirements.

4.1 Prime-order Bilinear Groups and Extended d-Linear Assumption

A prime-order (asymmetric) bilinear group generator $\mathsf{GrpGen}(1^k)$ takes security parameter 1^k as input and outputs $\mathcal{G} := (p, G_1, G_2, G_T, e)$, 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 3), 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 probabilistic polynomial time adversary $\mathcal{A}$, the following advantage function is negligible in k,*

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

where

$$D := \left(\mathcal{G}, g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, g_1^{a_{d+1}}, g_1^{a_1 s_1}, \ldots, g_1^{a_d s_d}\right),\ T_0 := g_1^{a_{d+1}(s_1+\cdots+s_d)},\ T_1 := g_1^{a_{d+1}(s_1+\cdots+s_d)+\boxed{s_{d+1}}}$$

and

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

"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. From now on, 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, like $\mathbf{0}_d$ (for the d-dimensional zero vector) and $\mathbf{0}_{d \times d'}$ (for the $d \times d'$ zero matrix) for some $d, d' \in \mathbb{Z}^+$.

An extended version of d-Linear Assumption. We describe an extended version of the d-linear assumption for improving the readability of our proofs, which is called (d, ℓ, q)-Linear Assumption (and (d, ℓ, q)-Lin for short, see Assumption 4). 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:

1. ℓ-extended: we first simultaneously consider ℓ independent challenges defined by $a_{d+1}, \ldots, a_{d+\ell} \leftarrow \mathbb{Z}_p^*$ and $s_{d+1}, \ldots, s_{d+\ell} \leftarrow \mathbb{Z}_p^*$, which is implicitly used in [CW13];
2. q-fold: we then consider its q-fold form [EHK+13] since the tight security reduction relies the random self-reducibility of the d-linear assumption.

We show that the (d, ℓ, q)-Linear Assumption is tightly implied by the standard d-Linear Assumption (see Lemma 1). From now on, we could prove all theorems and lemmas using the more expressive (d, ℓ, q)-linear assumption with relatively clean proofs, and finally base these results on the standard d-linear assumption. We remark that, since ℓ corresponds to a fixed and 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.

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

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

where

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

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

and

$$\begin{aligned} &\mathcal{G} := (p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k); \\ &a_1, \ldots, a_d, a_{d+i} \leftarrow \mathbb{Z}_p^*, \quad \forall i \in [\ell]; \\ &s_{1,j}, \ldots, s_{d,j} \leftarrow \mathbb{Z}_p,\ s_{d+i,j} \leftarrow \mathbb{Z}_p^*, \quad \forall i \in [\ell],\ j \in [q]. \end{aligned}$$

Lemma 1 (d-Lin $\Rightarrow$ (d, ℓ, q)-Lin) *Assume $\ell < q$. For any probabilistic polynomial time adversary $\mathcal{A}$, there exists an adversary $\mathcal{B}$ such that*

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

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

Proof. We may prove the lemma in two steps. Using the technique used in [CW13] (c.f. the proof of Lemma 8 in [CW13]), one can prove that, for any probabilistic polynomial time adversary $\mathcal{A}$, there exists an adversary $\mathcal{B}$ with $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + \ell \cdot \mathsf{poly}(k)$ such that

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

Then Lemma 1 in [EHK+13] implies that, for any probabilistic polynomial time adversary $\mathcal{A}$, there exists an adversary $\mathcal{B}$ with $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + (d+\ell)^2 q \cdot \mathsf{poly}(k)$ such that

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

Putting them together, one may deduce the lemma immediately. □

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-linear 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.

- $\mathsf{SampP}(1^k, n)$: On input $(1^k, n)$, do:
  - generate $(p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$, and let $G_1 = \langle g_1 \rangle$, $G_2 = \langle g_2 \rangle$ and $G_T = \langle g_T \rangle$;
  - 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 $\mathbf{A}_1, \ldots, \mathbf{A}_n \leftarrow \mathbb{Z}_p^{3d \times 3d}$, and define $\mathbf{B}^* := (\mathbf{B}^{-1})^\top$;
  - for all $i \in [n]$, define

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

  - for all $\mathbf{k} \in \mathbb{Z}_p^{3d}$, define $\mu(g_2^{\mathbf{k}}) := e(g_1^{\mathbf{D}}, g_2^{\mathbf{k}}) = e(g_1, g_2)^{\mathbf{D}^\top\mathbf{k}}$;

  and output

  $$\textsc{pp} := \left(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \mu; \quad \begin{array}{llll} g_1^{\mathbf{D}}, & g_1^{\mathbf{D}_1}, & \ldots, & g_1^{\mathbf{D}_n} \\ g_2^{\mathbf{D}^*}, & g_2^{\mathbf{D}_1^*}, & \ldots, & g_2^{\mathbf{D}_n^*} \end{array} \right) \text{ and } \textsc{sp} := \left(\begin{array}{llll} g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)}, g_1^{\mathbf{E}}, & g_1^{\mathbf{E}_1}, & \ldots, & g_1^{\mathbf{E}_n} \\ g_2^{\pi_{\mathrm{R}}(\mathbf{B}^*)}, g_1^{\mathbf{F}}, & g_1^{\mathbf{F}_1}, & \ldots, & g_1^{\mathbf{F}_n} \end{array} \right).$$

- $\mathsf{SampGT}(g_T^{\mathbf{p}})$: Sample $\mathbf{s} \leftarrow \mathbb{Z}_p^d$ and output $g_T^{\mathbf{s}^\top\mathbf{p}} \in G_T$.
- $\mathsf{SampG}(\textsc{pp})$: Sample $\mathbf{s} \leftarrow \mathbb{Z}_p^d$ and output $\left(g_1^{\mathbf{D}\mathbf{s}}, g_1^{\mathbf{D}_1\mathbf{s}}, \ldots, g_1^{\mathbf{D}_n\mathbf{s}}\right) \in (G_1^{3d})^{n+1}$.
- $\mathsf{SampH}(\textsc{pp})$: Sample $\mathbf{r} \leftarrow \mathbb{Z}_p^{3d}$ and output $\left(g_2^{\mathbf{D}^*\mathbf{r}}, g_2^{\mathbf{D}_1^*\mathbf{r}}, \ldots, g_2^{\mathbf{D}_n^*\mathbf{r}}\right) \in (G_2^{3d})^{n+1}$.
- $\widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$: Sample $\widehat{\mathbf{s}} \leftarrow \mathbb{Z}_p^d$ and output $\left(g_1^{\mathbf{E}\widehat{\mathbf{s}}}, g_1^{\mathbf{E}_1\widehat{\mathbf{s}}}, \ldots, g_1^{\mathbf{E}_n\widehat{\mathbf{s}}}\right) \in (G_1^{3d})^{n+1}$.
- $\widetilde{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$: Sample $\widetilde{\mathbf{s}} \leftarrow \mathbb{Z}_p^d$ and output $\left(g_1^{\mathbf{F}\widetilde{\mathbf{s}}}, g_1^{\mathbf{F}_1\widetilde{\mathbf{s}}}, \ldots, g_1^{\mathbf{F}_n\widetilde{\mathbf{s}}}\right) \in (G_1^{3d})^{n+1}$.
- $\widehat{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})$: Sample $\widehat{\mathbf{r}} \leftarrow \mathbb{Z}_p^d$ and output $g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}} \in G_2^{3d}$.
- $\widetilde{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})$: 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

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

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

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

$$e(g_1^{\mathbf{D}\mathbf{s}}, g_2^{\mathbf{D}_i^*\mathbf{r}}) = e(g_1, g_2)^{\bar{\mathbf{s}}^\top\mathbf{B}^\top(\mathbf{B}^*\mathbf{A}_i^\top\mathbf{R})\mathbf{r}} = e(g_1, g_2)^{\bar{\mathbf{s}}^\top(\mathbf{B}\mathbf{A}_i)^\top(\mathbf{B}^*\mathbf{R})\mathbf{r}} = e(g_1^{\mathbf{D}_i\mathbf{s}}, g_2^{\mathbf{D}^*\mathbf{r}}), \quad \forall i \in [n],$$

where $\bar{\mathbf{s}} := \left(\begin{smallmatrix}\mathbf{s} \\ \mathbf{0}_{2d}\end{smallmatrix}\right) \in \mathbb{Z}_p^{3d}$. The first and the last equality follow the definition of $\pi_{\mathrm{L}}(\cdot)$ while the second equality uses the fact that

$$\mathbf{B}^\top(\mathbf{B}^*\mathbf{A}_i^\top\mathbf{R}) = (\mathbf{B}^\top\mathbf{B}^*)\mathbf{A}_i^\top\mathbf{R} = \mathbf{A}_i^\top\mathbf{R} = \mathbf{A}_i^\top(\mathbf{B}^\top\mathbf{B}^*)\mathbf{R} = (\mathbf{B}\mathbf{A}_i)^\top(\mathbf{B}^*\mathbf{R}), \quad \forall i \in [n].$$

Security. We may check the following security requirements:

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

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

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 overwhelming probability $1 - 1/p^d$, sampling $\widehat{\mathbf{s}} \leftarrow \mathbb{Z}_p^d$ results in $\widehat{\mathbf{s}} \neq \mathbf{0}_d$, 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 $\mathsf{Adv}_{\mathscr{A}}^{\mathrm{LS1}}(k,q)$ as follows:

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

where

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

and

$$\begin{aligned}
\textsc{pp} &:= \left(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \mu; \begin{array}{cccc} g_1^{\pi_{\mathrm{L}}(\mathbf{B})}, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_1)}, & \dots, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_n)} \\ g_2^{\mathbf{B}^*\mathbf{R}}, & g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}}, & \dots, & g_2^{\mathbf{B}^*\mathbf{A}_n^\top\mathbf{R}} \end{array} \right); \\
\mathbf{g}_j &:= \left(g_1^{\mathbf{B}\begin{pmatrix}\mathbf{s}_j\\ \mathbf{0}_d\\ \mathbf{0}_d\end{pmatrix}}, g_1^{\mathbf{BA}_1\begin{pmatrix}\mathbf{s}_j\\ \mathbf{0}_d\\ \mathbf{0}_d\end{pmatrix}}, \dots, g_1^{\mathbf{BA}_n\begin{pmatrix}\mathbf{s}_j\\ \mathbf{0}_d\\ \mathbf{0}_d\end{pmatrix}}\right), \quad \forall j \in [q]; \\
\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j &:= \left(g_1^{\mathbf{B}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}}, g_1^{\mathbf{BA}_1\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}}, \dots, g_1^{\mathbf{BA}_n\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}}\right), \quad \forall j \in [q];
\end{aligned}$$

for $\mathbf{s}_j, \widehat{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q]$.

Lemma 2 ((d,d,q)-Lin $\Rightarrow$ LS1) *For any probabilistic polynomial time adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

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

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

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

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

as input where all $s_{d+i,j}$ with $i \in [d]$ and $j \in [q]$ 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

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

Programming $\mathbf{B}, \mathbf{B}^*, \mathbf{A}_1, \ldots, \mathbf{A}_n, \mathbf{R}$. Define $\mathbf{W}$ as

$$\mathbf{W} := \left(\begin{array}{ccc|ccc|ccc} a_1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & a_d & & & & & & \\ \hline a_{d+1} & \cdots & a_{d+1} & 1 & & & & & \\ \vdots & & \vdots & & \ddots & & & & \\ a_{2d} & \cdots & a_{2d} & & & 1 & & & \\ \hline & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{array}\right) \in \mathbb{Z}_p^{3d \times 3d}$$

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

$$(\mathbf{B}, \mathbf{B}^*) := (\bar{\mathbf{B}}\mathbf{W}, \bar{\mathbf{B}}^*\mathbf{W}^*), \quad \mathbf{R} := \mathbf{W}^\top \bar{\mathbf{R}}, \quad \mathbf{A}_i := \mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W}, \quad \forall i \in [n].$$

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

$$\begin{aligned} \mathbf{B}\mathbf{A}_i &= (\bar{\mathbf{B}}\mathbf{W})(\mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W}) = \bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}, \quad \forall i \in [n]; \\ \mathbf{B}^*\mathbf{R} &= (\bar{\mathbf{B}}^*\mathbf{W}^*)(\mathbf{W}^\top\bar{\mathbf{R}}) = \bar{\mathbf{B}}^*\bar{\mathbf{R}}; \\ \mathbf{B}^*\mathbf{A}_i^\top\mathbf{R} &= (\bar{\mathbf{B}}^*\mathbf{W}^*)(\mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W})^\top(\mathbf{W}^\top\bar{\mathbf{R}}) = \bar{\mathbf{B}}^*\bar{\mathbf{A}}_i^\top\bar{\mathbf{R}}, \quad \forall i \in [n]. \end{aligned}$$

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

$$\begin{array}{lcl} g_1^{\pi_{\mathrm{L}}(\mathbf{B})} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{B}}\mathbf{W})} = g_1^{\bar{\mathbf{B}}\pi_{\mathrm{L}}(\mathbf{W})} & \text{and} & g_1^{\pi_{\mathrm{L}}(\mathbf{B}\mathbf{A}_i)} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W})} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\pi_{\mathrm{L}}(\mathbf{W})}, \quad \forall i \in [n], \\ g_2^{\mathbf{B}^*\mathbf{R}} = g_2^{\bar{\mathbf{B}}^*\bar{\mathbf{R}}} & \text{and} & g_2^{\mathbf{B}^*\mathbf{A}_i^\top\mathbf{R}} = g_2^{\bar{\mathbf{B}}^*\bar{\mathbf{A}}_i^\top\bar{\mathbf{R}}}, \quad \forall i \in [n], \end{array}$$

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 $\mathscr{B}$ computes

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

and

$$g_1^{\mathbf{B}\mathbf{A}_i\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\begin{pmatrix} a_1 s_{1,j} \\ \vdots \\ a_d s_{d,j} \\ a_{d+1}(s_{1,j}+\cdots+s_{d,j})+s_{d+1,j} \\ \vdots \\ a_{2d}(s_{1,j}+\cdots+s_{d,j})+s_{2d,j} \\ \mathbf{0}_d \end{pmatrix}}, \quad \forall i \in [n], j \in [q].$$

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

We immediately have the following corollary from Lemma 1.

Corollary 1 (d-Lin $\Rightarrow$ LS1) *For any probabilistic polynomial time 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 $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + d^2 \cdot q \cdot \mathsf{poly}(k,n)$ *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

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

and

$$\begin{aligned}
\textsc{pp} &:= \left(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \mu; \begin{array}{llll} g_1^{\pi_{\mathrm{L}}(\mathbf{B})}, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_1)}, & \ldots, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_n)} \\ g_2^{\mathbf{B}^*\mathbf{R}}, & g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}}, & \ldots, & g_2^{\mathbf{B}^*\mathbf{A}_n^\top\mathbf{R}} \end{array} \right); \\
\widehat{h}^*_j \cdot \widetilde{h}^*_j &:= g_2^{\mathbf{B}^*\begin{pmatrix}\mathbf{0}_d\\ \widehat{\mathbf{r}}_j\\ \widetilde{\mathbf{r}}_j\end{pmatrix}}, \quad \forall j \in [q+q']; \\
\mathbf{g}'_j \cdot \widehat{\mathbf{g}}'_j &:= \left(g_1^{\mathbf{B}\begin{pmatrix}\mathbf{s}'_j\\ \widehat{\mathbf{s}}'_j\\ \mathbf{0}_d\end{pmatrix}}, g_1^{\mathbf{BA}_1\begin{pmatrix}\mathbf{s}'_j\\ \widehat{\mathbf{s}}'_j\\ \mathbf{0}_d\end{pmatrix}}, \ldots, g_1^{\mathbf{BA}_n\begin{pmatrix}\mathbf{s}'_j\\ \widehat{\mathbf{s}}'_j\\ \mathbf{0}_d\end{pmatrix}}\right), \quad \forall j \in [q]; \\
\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j &:= \left(g_1^{\mathbf{B}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}}, g_1^{\mathbf{BA}_1\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}}, \ldots, g_1^{\mathbf{BA}_n\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \mathbf{0}_d\end{pmatrix}}\right), \quad \forall j \in [q]; \\
\mathbf{g}_j \cdot \widetilde{\mathbf{g}}_j &:= \left(g_1^{\mathbf{B}\begin{pmatrix}\mathbf{s}_j\\ \mathbf{0}_d\\ \widetilde{\mathbf{s}}_j\end{pmatrix}}, g_1^{\mathbf{BA}_1\begin{pmatrix}\mathbf{s}_j\\ \mathbf{0}_d\\ \widetilde{\mathbf{s}}_j\end{pmatrix}}, \ldots, g_1^{\mathbf{BA}_n\begin{pmatrix}\mathbf{s}_j\\ \mathbf{0}_d\\ \widetilde{\mathbf{s}}_j\end{pmatrix}}\right), \quad \forall j \in [q];
\end{aligned}$$

where $\widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q+q']$, $\mathbf{s}'_j, \widehat{\mathbf{s}}'_j, \mathbf{s}_j, \widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q]$.

Lemma 3 ((d,d,q)-Lin $\Rightarrow$ LS2) *For any probabilistic polynomial time adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

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

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + d^2 \cdot (q+q') \cdot \mathsf{poly}(k,n)$ *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 $T_{1/2} = \{\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j\}_{j\in[q]}$ where

$$\mathbf{g}_j \cdot \widehat{\mathbf{g}}_j \cdot \widetilde{\mathbf{g}}_j := \left(g_1^{\mathbf{B}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \widetilde{\mathbf{s}}_j\end{pmatrix}}, g_1^{\mathbf{BA}_1\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \widetilde{\mathbf{s}}_j\end{pmatrix}}, \ldots, g_1^{\mathbf{BA}_n\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \widetilde{\mathbf{s}}_j\end{pmatrix}}\right), \quad \forall j \in [q],$$

and $\mathbf{s}_j, \widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q]$. In particular, we prove that, given D, distribution T_0 and $T_{1/2}$ are computational indistinguishable under the (d,d,q)-linear assumption (see Lemma 4), and so do $T_{1/2}$ and T_1 (see Lemma 5). Because the proofs of them are quite similar, we completely describe the proof of Lemma 4 and sketch the proof of Lemma 5 by pointing out the differences between them.

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

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

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

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

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

as input where all $s_{d+i,j}$ with $i \in [d]$ and $j \in [q]$ 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

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

Programming $\mathbf{B}, \mathbf{B}^*, \mathbf{A}_1, \ldots, \mathbf{A}_n, \mathbf{R}$. We define $\mathbf{W}$ as

$$\mathbf{W} := \left(\begin{array}{ccc|ccc|ccc} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ \hline & & & a_1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & a_d & & & \\ \hline & & & a_{d+1} & \cdots & a_{d+1} & 1 & & \\ & & & \vdots & & \vdots & & \ddots & \\ & & & a_{2d} & \cdots & a_{2d} & & & 1 \end{array}\right) \in \mathbb{Z}_p^{3d \times 3d}$$

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

$$(\mathbf{B}, \mathbf{B}^*) := (\bar{\mathbf{B}}\mathbf{W}, \bar{\mathbf{B}}^*\mathbf{W}^*), \quad \mathbf{R} := \mathbf{W}^\top \bar{\mathbf{R}}, \quad \mathbf{A}_i := \mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W}, \quad \forall i \in [n].$$

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

$$\begin{aligned} \mathbf{B}\mathbf{A}_i &= (\bar{\mathbf{B}}\mathbf{W})(\mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W}) = \bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}, \quad \forall i \in [n]; \\ \mathbf{B}^*\mathbf{R} &= (\bar{\mathbf{B}}^*\mathbf{W}^*)(\mathbf{W}^\top\bar{\mathbf{R}}) = \bar{\mathbf{B}}^*\bar{\mathbf{R}}; \\ \mathbf{B}^*\mathbf{A}_i^\top\mathbf{R} &= (\bar{\mathbf{B}}^*\mathbf{W}^*)(\mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W})^\top(\mathbf{W}^\top\bar{\mathbf{R}}) = \bar{\mathbf{B}}^*\bar{\mathbf{A}}_i^\top\bar{\mathbf{R}}, \quad \forall i \in [n]. \end{aligned}$$

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

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

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 $j \in [q+q']$. By a simple calculation, we have

$$\mathbf{W}^* := \left(\begin{array}{ccc|ccc|ccc} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ \hline & & & a_1^{-1} & & & -a_1^{-1}a_{d+1} & \cdots & -a_1^{-1}a_{2d} \\ & & & & \ddots & & \vdots & & \vdots \\ & & & & & a_d^{-1} & -a_d^{-1}a_{d+1} & \cdots & -a_d^{-1}a_{2d} \\ \hline & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{array}\right) \in \mathbb{Z}_p^{3d \times 3d}.$$

Observe that the right-bottom $2d \times 2d$ sub-matrix of $\mathbf{W}^*$ is full-rank with overwhelming probability and

$$\left\{ \mathbf{W}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j \end{pmatrix} : \widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d \right\} = \left\{ \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}'_j \\ \widetilde{\mathbf{r}}'_j \end{pmatrix} : \widehat{\mathbf{r}}'_j, \widetilde{\mathbf{r}}'_j \leftarrow \mathbb{Z}_p^d \right\},$$

which means that $\mathscr{B}$ may properly produce

$$\widehat{h}_j^* \cdot \widetilde{h}_j^* = g_2^{\mathbf{B}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j \end{pmatrix}} = g_2^{\bar{\mathbf{B}}^*\mathbf{W}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}_j \\ \widetilde{\mathbf{r}}_j \end{pmatrix}}, \quad \forall j \in [q+q']$$

by sampling $\widehat{\mathbf{r}}'_j, \widetilde{\mathbf{r}}'_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q+q']$ and setting

$$\widehat{h}^*_j \cdot \widetilde{h}^*_j = g_2^{\bar{\mathbf{B}}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}'_j \\ \widetilde{\mathbf{r}}'_j \end{pmatrix}}, \quad \forall j \in [q+q'].$$

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

$$g_1^{\mathbf{B}\begin{pmatrix} \mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \mathbf{0}_d \end{pmatrix}} = g_1^{\bar{\mathbf{B}}\mathbf{W}\begin{pmatrix} \mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \mathbf{0}_d \end{pmatrix}} \quad \text{and} \quad g_1^{\mathbf{B}\mathbf{A}_i\begin{pmatrix} \mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \mathbf{0}_d \end{pmatrix}} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}\begin{pmatrix} \mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \mathbf{0}_d \end{pmatrix}}, \quad \forall i \in [n],\ j \in [q],$$

by sampling $\mathbf{s}'_j, \widehat{\mathbf{s}}'_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q]$ 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 $\mathscr{B}$ samples $\mathbf{s}_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q]$ and computes

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

and

$$g_1^{\mathbf{B}\mathbf{A}_i\begin{pmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j \end{pmatrix}} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}\begin{pmatrix} \mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j \end{pmatrix}} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\begin{pmatrix} \mathbf{s}_j \\ a_1 s_{1,j} \\ \vdots \\ a_d s_{d,j} \\ a_{d+1}(s_{1,j}+\cdots+s_{d,j})+s_{d+1,j} \\ \vdots \\ a_{2d}(s_{1,j}+\cdots+s_{d,j})+s_{2d,j} \end{pmatrix}}, \quad i \in [n],\ j \in [q].$$

Analysis. Observe that if $s_{d+i,j} = 0$ for all $i \in [d]$ and $j \in [q]$, then $\widetilde{\mathbf{s}}_j = \mathbf{0}$ for all $j \in [q]$ 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 $s_{d+i,j} \leftarrow \mathbb{Z}_p^*$ for all $i \in [d]$ and $j \in [q]$, then $\widetilde{\mathbf{s}}_j \leftarrow (\mathbb{Z}_p^*)^d$ for all $j \in [q]$ 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 $\left|\Pr[\mathscr{A}(D,T_0)=1] - \Pr[\mathscr{A}(D,T_{1/2})=1]\right| \leqslant \mathsf{Adv}_{\mathscr{B}}^{(d,d,q)\text{-Lin}}(k)$. □

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

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

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + d^2 \cdot (q+q') \cdot \mathsf{poly}(k,n)$ *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)-linear problem

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

as input where all $s_{d+i,j}$ with $i \in [d]$ and $j \in [q]$ 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

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

Defining W. Adversary $\mathscr{B}$ defines $\mathbf{W}$ as

$$\mathbf{W} := \left(\begin{array}{ccc|ccc|ccc} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ \hline & & & 1 & & & a_{2d} & \cdots & a_{2d} \\ & & & & \ddots & & \vdots & & \vdots \\ & & & & & 1 & a_{d+1} & \cdots & a_{d+1} \\ \hline & & & & & & a_d & & \\ & & & & & & & \ddots & \\ & & & & & & & & a_1 \end{array}\right) \in \mathbb{Z}_p^{3d\times 3d}.$$

Then algorithm $\mathscr{B}$ may 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 by the strategies used in the proof of Lemma 4. □

Following hybrid argument, combining Lemma 4 and Lemma 5 proves Lemma 3. We further obtain the following corollary from Lemma 1.

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

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

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + d^2\cdot(q+q')\cdot \mathsf{poly}(k,n)$ *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-linear assumption, this subsection is going to establish a generalized version from the (d,d,d)-linear assumption, i.e., (d,ℓ,q)-linear assumption with $\ell = q = d$.

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*

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

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 $\boldsymbol{\tau}_j$ for $j\in[q]$ are either $\mathbf{0}_d$ or uniformly distributed over $\mathbb{Z}_p^d$ depending on whether all $r_{d+i,j}$ for $i,j\in[d]$ are 0 or uniformly distributed over $\mathbb{Z}_p$.

Proof. The algorithm works as follows:

Programming V and Z. We implicitly define

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

and

$$\mathbf{P} := \left(\begin{array}{ccc|ccc} a_1 & & & a_{d+1} & \cdots & a_{2d} \\ & \ddots & & \vdots & & \vdots \\ & & a_d & a_{d+1} & \cdots & a_{2d} \end{array}\right) \in \mathbb{Z}_p^{d\times 2d}.$$

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

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

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

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

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

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

and thus implicitly set $\boldsymbol{\tau}_j = \mathbf{T}\widetilde{\mathbf{t}}_j$ for all $j \in [q]$. Clearly, we have

$$\begin{pmatrix} \mathbf{t}_j \\ \boldsymbol{\tau}_j \end{pmatrix} = \begin{pmatrix} \mathbf{P} \\ \mathbf{T} \end{pmatrix} \widetilde{\mathbf{t}}_j, \quad \forall j \in [q].$$

Since the left half of $\mathbf{P}$ (i.e., matrix $\mathbf{Z}$) and the right half of $\mathbf{T}$ are full-rank (with overwhelming probability), the matrix on the right side is also full-rank, and therefore all $\boldsymbol{\tau}_j$ for $j \in [q]$ are distributed uniformly and independent of all $\mathbf{t}_j$ with $j \in [q]$. □

4.6 Nested-hiding Indistinguishability

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

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

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{array}{rcl}
\mathrm{PP} & := & \left(\begin{array}{ccccc} p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \mu; & 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{array} \right); \\
\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}, \quad \forall j \in [q+q']; \\
\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), \quad \forall j \in [q]; \\
\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), \quad \forall j \in [q]; \\
\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), \quad \forall j \in [q']; \\
\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), \; \forall j \in [q'];
\end{array}$$

where $\widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q+q']$, $\widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q]$, $\mathbf{r}_j \leftarrow \mathbb{Z}_p^{3d}$ and $\widehat{\boldsymbol{\gamma}}_j, \widetilde{\boldsymbol{\gamma}}_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q']$.

Lemma 7 ((d, d, d)-Lin ⇒ NH) *For all $\eta \in [\lfloor n/2 \rfloor]$ and for any probabilistic polynomial time adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

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

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

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

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

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

Generating $2q'$ tuples. Algorithm $\mathcal{B}$ runs the algorithm described in Lemma 6 on the input $2q'$, group G_2, and the (d,d,d)-linear 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 $\mathbf{B},\mathbf{B}^*,\mathbf{R},\mathbf{A}_1,\dots,\mathbf{A}_n$. 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 $\mathcal{B}$ can:

- simulate
$$g_1^{\pi_{\mathrm{L}}(\mathbf{B})} \quad \text{and} \quad g_1^{\pi_{\mathrm{L}}(\mathbf{B}\mathbf{A}_i)} = g_1^{\mathbf{B}\pi_{\mathrm{L}}(\mathbf{A}_i)}, \quad \forall i\in[n]$$
using the knowledge of $\mathbf{B}, \pi_{\mathrm{L}}(\mathbf{A}_1),\dots,\pi_{\mathrm{L}}(\mathbf{A}_n)$. Especially, we note that
$$\pi_{\mathrm{L}}(\mathbf{A}_{2\eta-1}) = \pi_{\mathrm{L}}(\bar{\mathbf{A}}_{2\eta-1}) \text{ and } \pi_{\mathrm{L}}(\mathbf{A}_{2\eta}) = \pi_{\mathrm{L}}(\bar{\mathbf{A}}_{2\eta}),$$
which are known to $\mathcal{B}$.
- 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 $\mathbf{A}_i$ for $i\in[n]\setminus\{2\eta-1,2\eta\}$.
- simulate
$$g_2^{\mathbf{B}^*\mathbf{A}_{2\eta-1}^\top\mathbf{R}} = g_2^{\mathbf{B}^*\bar{\mathbf{A}}_{2\eta-1}^\top\left(\begin{smallmatrix}\mathbf{I}&\mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{Z}&\mathbf{0}\\ \mathbf{0}&\mathbf{0}&\mathbf{Z}\end{smallmatrix}\right)\bar{\mathbf{R}}+\mathbf{B}^*\left(\begin{smallmatrix}\mathbf{0}&\mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{VZ}&\mathbf{0}\\ \mathbf{0}&\mathbf{0}&\mathbf{0}\end{smallmatrix}\right)\bar{\mathbf{R}}} \quad \text{and} \quad g_2^{\mathbf{B}^*\mathbf{A}_{2\eta}^\top\mathbf{R}} = g_2^{\mathbf{B}^*\bar{\mathbf{A}}_{2\eta}^\top\left(\begin{smallmatrix}\mathbf{I}&\mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{Z}&\mathbf{0}\\ \mathbf{0}&\mathbf{0}&\mathbf{Z}\end{smallmatrix}\right)\bar{\mathbf{R}}+\mathbf{B}^*\left(\begin{smallmatrix}\mathbf{0}&\mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{0}&\mathbf{0}\\ \mathbf{0}&\mathbf{0}&\mathbf{VZ}\end{smallmatrix}\right)\bar{\mathbf{R}}}$$
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 $\mathcal{B}$ can simulate

$$\widehat{h}_j^* = g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}_j} \quad \text{and} \quad \widetilde{h}_j^* = g_2^{\pi_{\mathrm{R}}(\mathbf{B}^*)\widetilde{\mathbf{r}}_j}, \quad \forall j\in[q+q'],$$

by sampling $\widehat{\mathbf{r}}_j,\widetilde{\mathbf{r}}_j\leftarrow\mathbb{Z}_p^d$ for $j\in[q+q']$ 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 $\mathcal{B}$ can simulate

$$g_1^{\pi_{\mathrm{M}}(\mathbf{B})\widehat{\mathbf{s}}_j} \quad \text{and} \quad g_1^{\pi_{\mathrm{M}}(\mathbf{B}\mathbf{A}_i)\widehat{\mathbf{s}}_j} = g_1^{\mathbf{B}\pi_{\mathrm{M}}(\mathbf{A}_i)\widehat{\mathbf{s}}_j}, \quad \forall i\in[n]\setminus\{2\eta-1\},\ j\in[q],$$

by sampling $\widehat{\mathbf{s}}_j\leftarrow\mathbb{Z}_p^d$ and using the knowledge of $\mathbf{B}$ and $\mathbf{A}_i$ 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 $\mathcal{B}$, but $\pi_{\mathrm{M}}(\mathbf{A}_{2\eta-1})$ containing secret matrix $\mathbf{V}$ is not. In a similar manner, algorithm $\mathcal{B}$ can also simulate

$$g_1^{\pi_{\mathrm{R}}(\mathbf{B})\widetilde{\mathbf{s}}_j} \quad \text{and} \quad g_1^{\pi_{\mathrm{R}}(\mathbf{B}\mathbf{A}_i)\widetilde{\mathbf{s}}_j} = g_1^{\mathbf{B}\pi_{\mathrm{R}}(\mathbf{A}_i)\widetilde{\mathbf{s}}_j}, \quad \forall i\in[n]\setminus\{2\eta\},\ j\in[q],$$

by sampling $\widetilde{\mathbf{s}}_j\leftarrow\mathbb{Z}_p^d$ and using the knowledge of $\mathbf{B}$ and $\mathbf{A}_i$ 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 $\mathcal{B}$, but $\pi_{\mathrm{R}}(\mathbf{A}_{2\eta})$ containing secret matrix $\mathbf{V}$ is not.

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

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

and compute

$$g_2^{\mathbf{B}^*\mathbf{R}\mathbf{r}_j} = g_2^{\mathbf{B}^*\left(\begin{smallmatrix}\bar{\mathbf{r}}_j\\ \mathbf{t}_{2j-1}\\ \mathbf{t}_{2j}\end{smallmatrix}\right)} \quad \text{and} \quad g_2^{\mathbf{B}^*\mathbf{A}_i^\top\mathbf{R}\mathbf{r}_j} = g_2^{\mathbf{B}^*\mathbf{A}_i^\top\left(\begin{smallmatrix}\bar{\mathbf{r}}_j\\ \mathbf{t}_{2j-1}\\ \mathbf{t}_{2j}\end{smallmatrix}\right)}, \quad \forall i\in[n]\setminus\{2\eta-1,2\eta\},$$

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

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

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

$$\widehat{\boldsymbol{\gamma}}_j = \boldsymbol{\tau}_{2j-1} \text{ and } \widetilde{\boldsymbol{\gamma}}_j = \boldsymbol{\tau}_{2j}.$$

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

We immediately have the following corollary from Lemma 1.

Corollary 3 (d-Lin $\Rightarrow$ NH) *For any probabilistic polynomial time adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

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

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

5 Concrete IBE from d-Linear Assumption

We now show 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, i.e., a $3d \times d$ sub-matrix.

– $\mathsf{Param}(1^k, n)$: 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

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

– $\mathsf{Setup}(\textsc{gp})$: Sample $\mathbf{k} \leftarrow \mathbb{Z}_p^{3d}$ and output

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

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

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

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

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

– $\mathsf{Dec}(\textsc{mpk}, \textsc{sk}, \textsc{ct})$: Let $\textsc{sk} = (K_0, K_1)$ and $\textsc{ct} = (C_0, C_1, C_2)$. Output

$$\textsc{m} := C_2 \frac{e(C_1, K_0)}{e(C_0, K_1)}.$$

6 Achieving Stronger Security Guarantee

Hofheinz *et al.* [HKS15] pointed out that their extension of NDSG is seemingly not sufficient for achieving IBE with full security. As a solution, they introduced a subgroup variant of the BDDH assumption (s-BDDH) to enhance their ENDSG (in particular, the *non-degeneracy* property) and proved that the full security is now available. We remark that our revised ENDSG in Section 3 is seemingly not sufficient for that purpose either.

This section will investigate two flavors of stronger adaptive security: *B-weak* and *full* adaptive security (see Section 2). Following Hofheinz *et al.*'s methodology, we only need to revise the *non-degeneracy* property in our ENDSG shown in Section 3 and upgrade the proof for the indistinguishability between Game$_3$ and Game$_4$ (c.f. Appendix B) accordingly. In detail, we extend the *non-degeneracy* requirement to *B-bounded non-degeneracy* and *computational non-degeneracy*, which implies the B-weak and the full adaptive security, respectively. Surprisingly, we can prove that our prime-order instantiation proposed in Section 4 has already satisfied these two extended versions of non-degeneracy: it is proven to be d-bounded non-degenerated unconditionally and computational non-degenerated under the d-linear assumption. This means that the concete IBE scheme shown in Section 5 which is derived from the construction in Section 4 achieves both the d-weak and the full adaptive security.

A Brief Review. Before we begin to work, we first recall Game$_3$ and Game$_4$ (c.f. Appendix B). 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 $(\text{PP},\text{SP}) \leftarrow \mathsf{SampP}(1^k, 2n)$ and $\text{MSK}_\iota \leftarrow \mathbb{H}$ (using PP) for all $\iota \in [\lambda]$, and returns

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

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

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

which are defined in an on-the-fly manner. When we refer to $\widehat{\mathsf{R}}_n(\iota, \mathbf{x})$ (resp. $\widetilde{\mathsf{R}}_n(\iota,\mathbf{x})$) for some pair $(\iota,\mathbf{x}) \in [\lambda]\times\{0,1\}^n$ which has never been used, we sample $\widehat{h} \leftarrow \widehat{\mathsf{SampH}}^*(\text{PP},\text{SP})$ (resp. $\widetilde{h} \leftarrow \widetilde{\mathsf{SampH}}^*(\text{PP},\text{SP})$) freshly and define $\widehat{\mathsf{R}}_n(\iota,\mathbf{x}) := \widehat{h}$ (resp. $\widetilde{\mathsf{R}}_n(\iota,\mathbf{x}) := \widetilde{h}$).

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

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

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

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

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

and updates $Q_C := Q_C \cup \left\{(\iota^*, \mathbf{x}^*, \text{M}_0^*, \text{M}_1^*, \text{CT}^*)\right\}$.

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.

We also review that the unique difference between weak, B-weak, and full adaptive security is how many challenge ciphertexts are permitted for each challenge identity in each instance, which is 1, at most B, and polynomially-many, respectively. Therefore, all proofs for moving from Game$_0$ to Game$_3$ (c.f. Appendix B) still work well for all three models.

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 $\widehat{g}_0 \leftarrow \widehat{\mathsf{SampG}_0}(\textsc{pp}, \textsc{sp})$, with overwhelming probability $1-2^{-\Omega(k)}$, $e(\widehat{g}_0, \widehat{h}^*)$ is distributed uniformly over $\mathbb{G}_T$ when sampling $\widehat{h}^* \leftarrow \widehat{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})$.

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 dimensional 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 there may be leftover entropy in $\widehat{h}^*$ given $e(\widehat{g}_0, \widehat{h}^*)$, and our prime-order instantiation may achieve higher security level even relying on no additional 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 $(\widehat{g}_{0,1}, \ldots, \widehat{g}_{0,B}) \leftarrow \widehat{\mathsf{SampG}_0}^B(\textsc{pp}, \textsc{sp})$, 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 $\widehat{h}^* \leftarrow \widehat{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})$.

It is a straightforward extension of the original non-degeneracy property. We now prove that our prime-order instantiation in Section 4 indeed reaches this stronger version of non-degeneracy property.

Lemma 8 *Our prime-order instantiation of ENDSG in Section 4 based on the d-linear 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 all $\widehat{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^d$ ($j \in [d]$) 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 $\widehat{\mathbf{r}}^\top(\widehat{\mathbf{s}}_1, \ldots, \widehat{\mathbf{s}}_d)^\top$ is distributed uniformly over $\mathbb{Z}_p^d$ when picking $\widehat{\mathbf{r}} \leftarrow \mathbb{Z}_p^d$. $\square$

On the other hand, one may readily show that the ENDSG with B-bounded non-degeneracy implies a B-weak adaptively secure IBE in the multi-instance, multi-ciphertext setting. 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 a single identity. As a special case, if we set $d = 1$ (i.e., the SXDH assumption), the resulting IBE is still weak adaptive secure as the basic scheme in [HKS15].

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 used in [HKS15]. Following the style of our revised ENDSG (in Section 3), our definition is more general than the s-BDDH assumption which involves some special algebra structure as their ENDSG.

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

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

where

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

and

$$\begin{aligned}
&\widehat{h}_j^* \leftarrow \widehat{\mathsf{SampH}}^*(\text{PP},\text{SP}),\ \widetilde{h}_j^* \leftarrow \widetilde{\mathsf{SampH}}^*(\text{PP},\text{SP}), \quad \forall j \in [q'];\\
&\widehat{h}_j^{**} \leftarrow \widehat{\mathsf{SampH}}^*(\text{PP},\text{SP}), \quad \forall j \in [q];\\
&\widehat{\mathbf{g}}_{j,j'} = \left(\widehat{g}_{0,j,j'}, \widehat{g}_{1,j,j'}, \ldots, \widehat{g}_{n,j,j'}\right) \leftarrow \widehat{\mathsf{SampG}}(\text{PP},\text{SP}), \quad R_{j,j'} \leftarrow \mathbb{G}_T, \quad \forall j \in [q],\ j' \in [q''].
\end{aligned}$$

Let $\mathsf{Adv}_{\mathscr{A}}^{\mathsf{Game}_3}(k,\lambda,q_K,q_C,q_R)$ and $\mathsf{Adv}_{\mathscr{A}}^{\mathsf{Game}_4}(k,\lambda,q_K,q_C,q_R)$ be the advantage function of adversary $\mathscr{A}$ in Game_3 and Game_4, respectively, in the full adaptive security model. As defined in Section 2, parameters λ, q_K, q_C, q_R indicate the upper bound of the number of IBE instances, key extraction queries, challenge identities, and challenge ciphertexts for each identity in each instance, respectively. We prove (in Lemma 9) that the computational non-degeneracy property implies that Game_3 and Game_4 are indistinguishable in the full adaptive security model. As we have discussed, this suffice to conclude that an ENDSG with computational non-degeneracy property (instead of the original one) implies a full adaptively secure IBE in the multi-instance, multi-ciphertext setting.

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

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

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

Proof. Given

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

where $\widehat{\mathbf{g}}_{j,j'} = \left(\widehat{g}_{0,j,j'}, \widehat{g}_{1,j,j'}, \ldots, \widehat{g}_{n,j,j'}\right)$ 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 $\text{MSK}_\iota \leftarrow \mathbb{H}$ (using PP) for all $\iota \in [\lambda]$, and output

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

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

Key extraction queries. On the j-th query $(\iota,\mathbf{y})$, sample $\mathbf{h} \leftarrow \mathsf{SampH}(\text{PP})$. If query $(\iota,\mathbf{y})$ has been made before, say the j'-th query $(j' < j)$, set $\overline{\text{MSK}} = \text{MSK}_\iota \cdot \widehat{h}_{j'}^* \cdot \widetilde{h}_{j'}^*$; otherwise, set $\overline{\text{MSK}} = \text{MSK}_\iota \cdot \widehat{h}_j^* \cdot \widetilde{h}_j^*$. Finally, $\mathscr{B}$ outputs

$$\text{SK} \leftarrow \overline{\mathsf{KeyGen}}(\text{PP}, \overline{\text{MSK}}, \mathbf{y}; \mathbf{h}).$$

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 $(\iota^*, \mathbf{x}^*, \text{M}_0^*, \text{M}_1^*)$, we let the query be the j'-th occurrence of j-th query of the form $(\iota^*, \mathbf{x}^*, *, *)$ and $\mathbf{x}^* = (x_1^*, \ldots, x_n^*) \in \{0,1\}^n$. $\mathscr{B}$ samples $(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\text{PP})$ and outputs

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

Here we implicitly set $\widehat{\mathsf{R}}_n(\iota^*, \mathbf{x}^*) := \widehat{h}_j^{**}$. Due to the restriction of the security model, the assignment here won't conflict with the assignment when answering key extraction queries.

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 $e(g_0 \cdot \widehat{g}_{0,j,j'}, \text{MSK}_{\iota^*}) \cdot e(\widehat{g}_{0,j,j'}, \widehat{h}_j^{**}) \cdot \text{M}_\beta^*$, 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 $\left|\mathsf{Adv}_{\mathscr{A}}^{\mathsf{Game}_3}(k,\lambda,q_K,q_C,q_R) - \mathsf{Adv}_{\mathscr{A}}^{\mathsf{Game}_4}(k,\lambda,q_K,q_C,q_R)\right| \leqslant \mathsf{Adv}_{\mathscr{B}}^{\mathrm{ND}}(k,q_K,q_C,q_R)$. □

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 defined by the previous subsection. And this immediately implies that the concrete IBE scheme shown in Section 5 is fully adaptively secure in the multi-instance, and multi-ciphertext setting.

We first 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

$$\begin{aligned}
\mathrm{PP} &:= \left(\begin{array}{llllll} p,\mathbb{G},\mathbb{H},\mathbb{G}_T,e,\mu; & g_1^{\pi_{\mathrm{L}}(\mathbf{B})}, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_1)}, & \dots, & g_1^{\pi_{\mathrm{L}}(\mathbf{BA}_n)} \\ & g_2^{\mathbf{B}^*\mathbf{R}}, & g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}}, & \dots, & g_2^{\mathbf{B}^*\mathbf{A}_n^\top\mathbf{R}} \end{array}\right); \\
\widehat{h}^*_j\cdot\widetilde{h}^*_j &:= g_2^{\mathbf{B}^*\begin{pmatrix}\mathbf{0}_d\\ \widehat{\mathbf{r}}'_j\\ \widetilde{\mathbf{r}}'_j\end{pmatrix}}, \quad \forall j\in[q']; \\
\widehat{\mathbf{g}}_{j,j'} &:= \left(g_1^{\pi_{\mathrm{M}}(\mathbf{B})\widehat{\mathbf{s}}_{j,j'}}, g_1^{\pi_{\mathrm{M}}(\mathbf{BA}_1)\widehat{\mathbf{s}}_{j,j'}},\dots,g_1^{\pi_{\mathrm{M}}(\mathbf{BA}_n)\widehat{\mathbf{s}}_{j,j'}}\right), \quad \forall j\in[q], j'\in[q'']; \\
e(\widehat{g}_{0,j,j'},\widehat{h}^{**}_j) &:= e(g_1^{\pi_{\mathrm{M}}(\mathbf{B})\widehat{\mathbf{s}}_{j,j'}}, g_2^{\pi_{\mathrm{M}}(\mathbf{B}^*)\widehat{\mathbf{r}}_j}) = e(g_1,g_2)^{\widehat{\mathbf{s}}^\top_{j,j'}\widehat{\mathbf{r}}_j}, \quad \forall j\in[q], j'\in[q'']; \\
\widehat{R}_{j,j'} &:= e(g_1,g_2)^{\widehat{\gamma}_{j,j'}}, \quad \forall j\in[q], j'\in[q''];
\end{aligned}$$

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

Lemma 10 ($(d,1,qq'')$-Lin $\Rightarrow$ ND) *For any probabilistic polynomial time adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

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

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + d^2\cdot(qq''+q')\cdot\mathsf{poly}(k,n)$ *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 security result:

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

The proof is quite direct in the case of $q=1$, i.e., there exists only one $\widehat{\mathbf{r}}_j$ (or challenge identity in the context of IBE). However more than one $\widehat{\mathbf{r}}_j$ (or multiple challenge identity in the context of IBE) may arise $\mathscr{O}(q)$ security loss if we continue to adopt this trivial proof strategy. In order to obtain a tight reduction in such a case, we further rewrite the challenge term as

$$g_1^{\widehat{\mathbf{s}}^\top_{j,j'}\widehat{\mathbf{r}}_j} = g_1^{\widehat{\mathbf{s}}^\top_{j,j'}\mathbf{V}^\top\bar{\mathbf{r}}_j} = g_1^{\bar{\mathbf{r}}^\top_j\mathbf{V}\widehat{\mathbf{s}}_{j,j'}}, \quad \forall j\in[q], j'\in[q'']$$

where $\mathbf{V}$ may be any $(d+1)\times d$ matrix over $\mathbb{Z}_p$ of rank d and $\bar{\mathbf{r}}_j\leftarrow\mathbb{Z}_p^{d+1}$. Clearly, we implicitly define $\widehat{\mathbf{r}}_j := \mathbf{V}^\top\bar{\mathbf{r}}_j$ for all $j\in[q]$. 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 $(d,1,qq'')$-linear problem (i.e., set $\ell=1$ and $q=qq''$)

$$\left(g_1,g_2,g_1^{a_1},\dots,g_1^{a_d},g_1^{a_{d+1}},\left\{g_1^{a_1s_{1,j,j'}},\dots,g_1^{a_ds_{d,j,j'}}\right\}_{j\in[q],j'\in[q'']},\left\{g_1^{a_{d+1}(s_{1,j,j'}+\dots+s_{d,j,j'})+s_{d+1,j,j'}}\right\}_{j\in[q],j'\in[q'']}\right)$$

as input where all $s_{d+1,j,j'}$ for $j\in[q]$ and $j'\in[q'']$ 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

$$\widehat{\mathbf{s}}_{j,j'} = (s_{1,j,j'},\dots,s_{d,j,j'})^\top, \quad \forall j\in[q], j'\in[q'].$$

Programming $\mathbf{B}, \mathbf{B}^*, \mathbf{A}_1, \ldots, \mathbf{A}_n, \mathbf{R}$. Define $\mathbf{W}$ as

$$\mathbf{W} := \left(\begin{array}{ccc|ccc|ccc} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ \hline & & & a_1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & a_d & & & \\ \hline & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{array}\right) \in \mathbb{Z}_p^{3d \times 3d}$$

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

$$(\mathbf{B}, \mathbf{B}^*) := (\bar{\mathbf{B}}\mathbf{W}, \bar{\mathbf{B}}^*\mathbf{W}^*), \quad \mathbf{R} := \mathbf{W}^\top \bar{\mathbf{R}}, \quad \mathbf{A}_i := \mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W}, \quad \forall i \in [n].$$

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

$$\begin{array}{rcl} \mathbf{B}\mathbf{A}_i & = & (\bar{\mathbf{B}}\mathbf{W})(\mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W}) = \bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}, \quad \forall i \in [n]; \\ \mathbf{B}^*\mathbf{R} & = & (\bar{\mathbf{B}}^*\mathbf{W}^*)(\mathbf{W}^\top\bar{\mathbf{R}}) = \bar{\mathbf{B}}^*\bar{\mathbf{R}}; \\ \mathbf{B}^*\mathbf{A}_i^\top\mathbf{R} & = & (\bar{\mathbf{B}}^*\mathbf{W}^*)(\mathbf{W}^{-1}\bar{\mathbf{A}}_i\mathbf{W})^\top(\mathbf{W}^\top\bar{\mathbf{R}}) = \bar{\mathbf{B}}^*\bar{\mathbf{A}}_i^\top\bar{\mathbf{R}}, \quad \forall i \in [n]. \end{array}$$

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

$$\begin{array}{lcl} g_1^{\pi_{\mathrm{L}}(\mathbf{B})} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{B}}\mathbf{W})} = g_1^{\bar{\mathbf{B}}\pi_{\mathrm{L}}(\mathbf{W})} & \text{and} & g_1^{\pi_{\mathrm{L}}(\mathbf{B}\mathbf{A}_i)} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W})} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\pi_{\mathrm{L}}(\mathbf{W})}, \quad \forall i \in [n], \\ g_2^{\mathbf{B}^*\mathbf{R}} = g_2^{\bar{\mathbf{B}}^*\bar{\mathbf{R}}} & \text{and} & g_2^{\mathbf{B}^*\mathbf{A}_1^\top\mathbf{R}} = g_2^{\bar{\mathbf{B}}^*\bar{\mathbf{A}}_i^\top\bar{\mathbf{R}}}, \quad \forall i \in [n], \end{array}$$

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 $j \in [q']$. By a simple calculation, we have

$$\mathbf{W}^* := \left(\begin{array}{ccc|ccc|ccc} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ \hline & & & a_1^{-1} & & & & & \\ & & & & \ddots & & & & \\ & & & & & a_d^{-1} & & & \\ \hline & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{array}\right) \in \mathbb{Z}_p^{3d \times 3d}.$$

Observe that the right-most $2d \times 2d$ sub-matrix of $\mathbf{W}^*$ is full-rank with overwhelming probability and

$$\left\{ \mathbf{W}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}'_j \\ \widetilde{\mathbf{r}}'_j \end{pmatrix} : \widehat{\mathbf{r}}'_j, \widetilde{\mathbf{r}}'_j \leftarrow \mathbb{Z}_p^d \right\} = \left\{ \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}''_j \\ \widetilde{\mathbf{r}}''_j \end{pmatrix} : \widehat{\mathbf{r}}''_j, \widetilde{\mathbf{r}}''_j \leftarrow \mathbb{Z}_p^d \right\}.$$

which means that $\mathscr{B}$ may properly produce

$$\widehat{h}_j^* \cdot \widetilde{h}_j^* = g_2^{\mathbf{B}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}'_j \\ \widetilde{\mathbf{r}}'_j \end{pmatrix}} = g_2^{\bar{\mathbf{B}}^*\mathbf{W}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}'_j \\ \widetilde{\mathbf{r}}'_j \end{pmatrix}}, \quad \forall j \in [q']$$

by sampling $\widehat{\mathbf{r}}''_j, \widetilde{\mathbf{r}}''_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q']$ and setting

$$\widehat{h}_j^* \cdot \widetilde{h}_j^* = g_2^{\bar{\mathbf{B}}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}''_j \\ \widetilde{\mathbf{r}}''_j \end{pmatrix}}, \quad \forall j \in [q'].$$

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

$$g_1^{\mathbf{B}\begin{pmatrix}\mathbf{0}_d\\ \widehat{\mathbf{s}}_{j,j'}\\ \mathbf{0}_d\end{pmatrix}} = g_1^{\bar{\mathbf{B}}\mathbf{W}\begin{pmatrix}\mathbf{0}_d\\ \widehat{\mathbf{s}}_{j,j'}\\ \mathbf{0}_d\end{pmatrix}} = g_1^{\bar{\mathbf{B}}\begin{pmatrix}\mathbf{0}_d\\ a_1 s_{1,j,j'}\\ \vdots\\ a_d s_{d,j,j'}\\ \mathbf{0}_d\end{pmatrix}} \quad \text{and} \quad g_1^{\mathbf{B}\mathbf{A}_i\begin{pmatrix}\mathbf{0}_d\\ \widehat{\mathbf{s}}_{j,j'}\\ \mathbf{0}_d\end{pmatrix}} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\mathbf{W}\begin{pmatrix}\mathbf{0}_d\\ \widehat{\mathbf{s}}_{j,j'}\\ \mathbf{0}_d\end{pmatrix}} = g_1^{\bar{\mathbf{B}}\bar{\mathbf{A}}_i\begin{pmatrix}\mathbf{0}_d\\ a_1 s_{1,j,j'}\\ \vdots\\ a_d s_{d,j,j'}\\ \mathbf{0}_d\end{pmatrix}}, \quad \forall i \in [n]$$

using the knowledge of $\left\{g_1^{a_1 s_{1,j,j'}}, \ldots, g_1^{a_d s_{d,j,j'}}\right\}$ and $\bar{\mathbf{B}}, \bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_n$.

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

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

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

$$g_1^{\widehat{\mathbf{r}}_j^\top \widehat{\mathbf{s}}_{j,j'} + \widehat{\gamma}_{j,j'}} = g_1^{\bar{\mathbf{r}}_j^\top \begin{pmatrix} a_1 s_{1,j,j'} \\ \vdots \\ a_d s_{d,j,j'} \\ a_{d+1}(s_{1,j,j'} + \cdots + s_{d,j,j'}) + s_{d+1,j,j'} \end{pmatrix}}, \quad \forall j \in [q],\ j' \in [q'']$$

using the knowledge of $\left\{g_1^{a_1 s_{1,j,j'}}, \ldots, g_1^{a_d s_{d,j,j'}}, g_1^{a_{d+1}(s_{1,j,j'} + \cdots + s_{d,j,j'}) + s_{d+1,j,j'}}\right\}$, and outputs the challenge as

$$e(g_1^{\widehat{\mathbf{r}}_j^\top \widehat{\mathbf{s}}_{j,j'} + \widehat{\gamma}_{j,j'}}, g_2).$$

Analysis. Observe that, if $s_{d+1,j,j'} = 0$ for all $j \in [q]$ and $j' \in [q']$, then the output challenge is distributed as

$$e(g_1^{\bar{\mathbf{r}}_j^\top \mathbf{V} \widehat{\mathbf{s}}_{j,j'}}, g_2) = e(g_1, g_2)^{\widehat{\mathbf{s}}_{j,j'}^\top \widehat{\mathbf{r}}_j}, \quad \forall j \in [q],\ j' \in [q'],$$

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

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

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

We immediately have the following corollary from Lemma 1.

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

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

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + d^2 \cdot (qq'' + q') \cdot \mathsf{poly}(k, n)$ *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 the ENDSG shown in Section 3 equipped with computational non-degeneracy defined in Section 6. Namely, instead of investigating the weak adaptive security as a stepping stone, we are going to consider the full adaptive security directly. As discussed in Section 1, key points are (1) updating computational non-degeneracy; (2) re-organizing the LS requirements; and (3) hiding parameters for SampH from the adversary. We show in Appendix C that the ENDSG after fine-tuning still implies IBE in the multi-instance, multi-ciphertext setting by showing the construction and the sketch of the proof.

Syntax. The fine-tuned extended nested dual system group consists of eight (probabilistic) polynomial time algorithms defined as follows:

- $\mathsf{SampP}(1^k, n)$: On input $(1^k, n)$, output
  - $\textsc{pp}$ containing (1) group description $(\mathbb{G}, \mathbb{H}, \mathbb{G}_T)$ and an admissible bilinear map $e : \mathbb{G} \times \mathbb{H} \to \mathbb{G}_T$; (2) an efficient linear map μ defined on $\mathbb{H}$; (3) an efficient sampler for $\mathbb{H}$ and $\mathbb{Z}_{\mathrm{ord}(\mathbb{H})}$, respectively; (4) public parameters for SampG.
  - $\textsc{hp}$ containing parameters for SampH.
  - $\textsc{sp}$ containing secret parameters for $\widehat{\mathsf{SampG}}$, $\widetilde{\mathsf{SampG}}$, $\widehat{\mathsf{SampH}}^*$ and $\widetilde{\mathsf{SampH}}^*$.
- SampGT: $\mathrm{Im}(\mu) \to \mathbb{G}_T$.
- $\mathsf{SampG}(\textsc{pp})$: Output $\mathbf{g} = (g_0, g_1, \ldots, g_n) \in \mathbb{G}^{n+1}$.
- $\mathsf{SampH}(\textsc{pp}, \textsc{hp})$: Output $\mathbf{h} = (h_0, h_1, \ldots, h_n) \in \mathbb{H}^{n+1}$.
- $\widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$: Output $\widehat{\mathbf{g}} = (\widehat{g}_0, \widehat{g}_1, \ldots, \widehat{g}_n) \in \mathbb{G}^{n+1}$.
- $\widetilde{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$: Output $\widetilde{\mathbf{g}} = (\widetilde{g}_0, \widetilde{g}_1, \ldots, \widetilde{g}_n) \in \mathbb{G}^{n+1}$.
- $\widehat{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})$: Output $\widehat{h}^* \in \mathbb{H}$.
- $\widetilde{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})$: 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 and Security. The *projective*, *associative*, *orthogonality*, and $\mathbb{H}$-*subgroup* requirement are identical to those defined in Section 3.

(Left subgroup indistinguishability 1 (LS1).) For any probabilistic polynomial time 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

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

and

$$\mathbf{g}_j \leftarrow \mathsf{SampG}(\textsc{pp}),\ \widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp}),\ \widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\textsc{pp}, \textsc{sp}), \quad \forall j \in [q];$$
$$\mathbf{h}_j \leftarrow \mathsf{SampH}(\textsc{pp}, \textsc{hp}), \quad \forall j \in [q'].$$

(Left subgroup indistinguishability 2 (LS2).) For any probabilistic polynomial time 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

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

and

$$\widehat{h}^*_j \leftarrow \widehat{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp}),\ \widetilde{h}^*_j \leftarrow \widetilde{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp}), \quad \forall j \in [q+q'];$$
$$\mathbf{g}'_j \leftarrow \mathsf{SampG}(\textsc{pp}),\ \widehat{\mathbf{g}}'_j \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp}),\ \widetilde{\mathbf{g}}'_j \leftarrow \widetilde{\mathsf{SampG}}(\textsc{pp}, \textsc{sp}), \quad \forall j \in [q];$$
$$\mathbf{g}_j \leftarrow \mathsf{SampG}(\textsc{pp}),\ \widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp}),\ \widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\textsc{pp}, \textsc{sp}), \quad \forall j \in [q];$$
$$\mathbf{h}_j \leftarrow \mathsf{SampH}(\textsc{pp}, \textsc{hp}), \quad \forall j \in [q'].$$

(Left subgroup indistinguishability 3 (LS3).) For any probabilistic polynomial time 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

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

and

$$\begin{aligned}
&\widehat{h}^*_j \leftarrow \widehat{\mathsf{SampH}}^*(\text{PP},\text{SP}),\ \widetilde{h}^*_j \leftarrow \widetilde{\mathsf{SampH}}^*(\text{PP},\text{SP}), \quad \forall j\in[q+q'];\\
&\mathbf{g}'_j \leftarrow \mathsf{SampG}(\text{PP}),\ \widetilde{\mathbf{g}}'_j \leftarrow \widetilde{\mathsf{SampG}}(\text{PP},\text{SP}), \quad \forall j\in[q];\\
&\mathbf{g}_j \leftarrow \mathsf{SampG}(\text{PP}),\ \widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\text{PP},\text{SP}),\ \widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\text{PP},\text{SP}), \quad \forall j\in[q];\\
&\mathbf{h}_j \leftarrow \mathsf{SampH}(\text{PP},\text{HP}), \quad \forall j\in[q'].
\end{aligned}$$

(Nested-hiding indistinguishability (NH).) For all $\eta \in [\lfloor n/2\rfloor]$ and any probabilistic polynomial time 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

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

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

and

$$\begin{aligned}
&\widehat{h}^*_j \leftarrow \widehat{\mathsf{SampH}}^*(\text{PP},\text{SP}),\ \widetilde{h}^*_j \leftarrow \widetilde{\mathsf{SampH}}^*(\text{PP},\text{SP}), \quad \forall j\in[q+q'];\\
&\widehat{\mathbf{g}}_j \leftarrow \widehat{\mathsf{SampG}}(\text{PP},\text{SP}),\ \widetilde{\mathbf{g}}_j \leftarrow \widetilde{\mathsf{SampG}}(\text{PP},\text{SP}), \quad \forall j\in[q];\\
&\mathbf{h}_j \leftarrow \mathsf{SampH}(\text{PP},\text{HP}),\ \widehat{h}^{**}_j \leftarrow \widehat{\mathsf{SampH}}^*(\text{PP},\text{SP}),\ \widetilde{h}^{**}_j \leftarrow \widetilde{\mathsf{SampH}}^*(\text{PP},\text{SP}), \quad \forall j\in[q'];\\
&\mathbf{h}'_j \leftarrow \mathsf{SampH}(\text{PP},\text{HP}), \quad \forall j\in[q'].
\end{aligned}$$

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 probabilistic polynomial time 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

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

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

and

$$\begin{aligned}
&\widehat{h}^*_j \leftarrow \widehat{\mathsf{SampH}}^*(\text{PP},\text{SP}),\ \widetilde{h}^*_j \leftarrow \widetilde{\mathsf{SampH}}^*(\text{PP},\text{SP}),\ \mathbf{h}_j \leftarrow \mathsf{SampH}(\text{PP},\text{HP}), \quad \forall j\in[q'];\\
&\widehat{h}^{**}_j \leftarrow \widehat{\mathsf{SampH}}^*(\text{PP},\text{SP}),\ \widetilde{h}^{**}_j \leftarrow \widetilde{\mathsf{SampH}}^*(\text{PP},\text{SP}), \quad \forall j\in[q];\\
&\widehat{\mathbf{g}}_{j,j'} = \left(\widehat{g}_{0,j,j'}, \widehat{g}_{1,j,j'}, \ldots, \widehat{g}_{n,j,j'}\right) \leftarrow \widehat{\mathsf{SampG}}(\text{PP},\text{SP}), \quad \forall j\in[q],\ j'\in[q''];\\
&\widetilde{\mathbf{g}}_{j,j'} = \left(\widetilde{g}_{0,j,j'}, \widetilde{g}_{1,j,j'}, \ldots, \widetilde{g}_{n,j,j'}\right) \leftarrow \widetilde{\mathsf{SampG}}(\text{PP},\text{SP}), \quad \forall j\in[q],\ j'\in[q''];\\
&R_{j,j'} \leftarrow G_T, \quad \forall j\in[q],\ j'\in[q''].
\end{aligned}$$

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 IBE in the multi-instance, multi-ciphertext setting using prime-order bilinear groups following the generic construction shown in Appendix C. We describe this IBE and its variants in Section 9.

The section begins with a definition of d-linear assumption with auxiliary input, which is an generalization of d-linear assumption in Section 4. An instantiation of fine-tuned ENDSG in the prime-order setting is proposed with a series of proofs showing that it indeed satisfies all correctness and security requirements.

8.1 d-Linear Assumption with Auxiliary Input

We assume a prime-order (asymmetric) bilinear group generator $\mathsf{GrpGen}(1^k)$ as defined in Section 4, which takes security parameter 1^k as input and outputs group description $\mathcal{G} := (p, G_1, G_2, G_T, e)$. We also assume that d is an even positive integer greater than 1. The d-linear assumption in G_1 with auxiliary input in G_2 is defined as follows, the analogous assumption in G_2 can be defined by exchanging the role of G_1 and G_2.

Assumption 3 (d-Linear Assumption in G_1 with Auxiliary Input in G_2) *For any probabilistic polynomial time adversary $\mathcal{A}$, the following advantage function is negligible in k,*

$$\mathsf{Adv}_{\mathcal{A}}^{d\text{-LinAI}}(k) := \Big|\Pr[\mathcal{A}(D, \textsc{Aux}, T_0) = 1] - \Pr[\mathcal{A}(D, \textsc{Aux}, T_1) = 1]\Big|,$$

where

$$D := \left(\mathcal{G}, g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, g_1^{a_{d+1}}, g_1^{a_1 s_1}, \ldots, g_1^{a_d s_d}\right), \ \textsc{Aux} := \left(g_2^{a a_1^{-1} a_{d+1}}, \ldots, g_2^{a a_{d/2}^{-1} a_{d+1}}, g_2^{a}\right)$$

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

and

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

As we have done in Section 4, we also define an natural extension of d-linear assumption with auxiliary input in order to alleviate the complexity of the proof. The relation between them is shown in Lemma 11.

Assumption 4 ((d,ℓ,q)-Linear Assumption in G_1 with Auxiliary Input in G_2) *For any probabilistic polynomial time adversary $\mathcal{A}$, the following advantage function is negligible in k,*

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

where

$$D := \left(\mathcal{G}, g_1, g_2, g_1^{a_1}, \ldots, g_1^{a_d}, \left\{g_1^{a_{d+i}}\right\}_{i\in[\ell]}, \left\{g_1^{a_1 s_{1,j}}, \ldots, g_1^{a_d s_{d,j}}\right\}_{j\in[q]}\right), \ \textsc{Aux} := \left(\left\{g_2^{a a_1^{-1} a_{d+i}}, \ldots, g_2^{a a_{d/2}^{-1} a_{d+i}}\right\}_{i\in[\ell]}, g_2^{a}\right)$$

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

and

$$\mathcal{G} := (p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k);$$
$$a_1, \ldots, a_d, a_{d+i} \leftarrow \mathbb{Z}_p^*, \quad \forall i \in [\ell], \quad a := a_1 \cdots a_{d/2};$$
$$s_{1,j}, \ldots, s_{d,j} \leftarrow \mathbb{Z}_p, \ s_{d+i,j} \leftarrow \mathbb{Z}_p^*, \quad \forall i \in [\ell], \ j \in [q].$$

Lemma 11 (d-LinAI $\Rightarrow$ (d,ℓ,q)-LinAI) *Assume $\ell < q$. For any probabilistic polynomial time adversary $\mathcal{A}$, there exists an adversary $\mathcal{B}$ such that*

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

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

8.2 Construction

The construction is based on prime-order instantiation of DSG by Chen, Gay and Wee [CGW15] and works with $2d \times 2d$ matrix. Technically, we sample the basis by the dual pairing vector space technique introduced by Okamoto and Takashima [OT08, OT09, LOS⁺10] instead of sampling from some specific matrix distribution. This allows us to create one normal space and two semi-functional space with specific orthogonal relations. The randomization of the basis follows the technique proposed by Chen, Gay and Wee [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. Algorithms of the fine-tuned ENDSG are shown as follows.

- $\mathsf{SampP}(1^k, n)$: On input $(1^k, n)$, the algorithm does
  - generate $(p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$, and let $G_1 = \langle g_1 \rangle$, $G_2 = \langle g_2 \rangle$ and $G_T = \langle g_T \rangle$;
  - define $(\mathbb{G}, \mathbb{H}, \mathbb{G}_T, e) := (G_1^{2d}, G_2^{2d}, G_T, e)$;
  - sample $\mathbf{D} \leftarrow \mathrm{GL}_{2d}(\mathbb{Z}_p)$, define $\mathbf{D}^* := (\mathbf{D}^{-1})^\top$ and

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

  - for all $\mathbf{k} \in \mathbb{Z}_p^{2d}$, define $\mu(g_2^{\mathbf{k}}) := e(g_1^{\mathbf{A}}, g_2^{\mathbf{k}}) = e(g_1, g_2)^{\mathbf{A}^\top \mathbf{k}}$;
  - sample $\mathbf{W}_1, \ldots, \mathbf{W}_n \leftarrow \mathbb{Z}_p^{2d \times 2d}$;

  and output

$$\textsc{pp} := \left(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \mu; g_1^{\mathbf{A}}, g_1^{\mathbf{W}_1^\top \mathbf{A}}, \ldots, g_1^{\mathbf{W}_n^\top \mathbf{A}}\right), \quad \textsc{hp} := \left(g_2^{\mathbf{B}}, g_2^{\mathbf{W}_1 \mathbf{B}}, \ldots, g_2^{\mathbf{W}_n \mathbf{B}}\right)$$

$$\textsc{sp} := \begin{pmatrix} g_2^{\widehat{\mathbf{B}}}, & g_1^{\widehat{\mathbf{A}}}, & g_1^{\mathbf{W}_1^\top \widehat{\mathbf{A}}}, & \ldots, & g_1^{\mathbf{W}_n^\top \widehat{\mathbf{A}}} \\ g_2^{\widetilde{\mathbf{B}}}, & g_1^{\widetilde{\mathbf{A}}}, & g_1^{\mathbf{W}_1^\top \widetilde{\mathbf{A}}}, & \ldots, & g_1^{\mathbf{W}_n^\top \widetilde{\mathbf{A}}} \end{pmatrix}.$$

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

- $\mathsf{SampG}(\textsc{pp})$: Sample $\mathbf{s} \leftarrow \mathbb{Z}_p^d$ and output $\left(g_1^{\mathbf{As}}, g_1^{\mathbf{W}_1^\top \mathbf{As}}, \ldots, g_1^{\mathbf{W}_n^\top \mathbf{As}}\right) \in (G_1^{2d})^{n+1}$.

- $\mathsf{SampH}(\textsc{pp}, \textsc{hp})$: Sample $\mathbf{r} \leftarrow \mathbb{Z}_p^d$ and output $\left(g_2^{\mathbf{Br}}, g_2^{\mathbf{W}_1 \mathbf{Br}}, \ldots, g_2^{\mathbf{W}_n \mathbf{Br}}\right) \in (G_2^{2d})^{n+1}$.

- $\widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$: Sample $\widehat{\mathbf{s}} \leftarrow \mathbb{Z}_p^{d/2}$ and output $\left(g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}}, g_1^{\mathbf{W}_1^\top \widehat{\mathbf{A}}\widehat{\mathbf{s}}}, \ldots, g_1^{\mathbf{W}_n^\top \widehat{\mathbf{A}}\widehat{\mathbf{s}}}\right) \in (G_1^{2d})^{n+1}$.

- $\widetilde{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$: Sample $\widetilde{\mathbf{s}} \leftarrow \mathbb{Z}_p^{d/2}$ and output $\left(g_1^{\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}}, g_1^{\mathbf{W}_1^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}}, \ldots, g_1^{\mathbf{W}_n^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}}\right) \in (G_1^{2d})^{n+1}$.

- $\widehat{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})$: Sample $\widehat{\mathbf{r}} \leftarrow \mathbb{Z}_p^{d/2}$ and output $g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}} \in G_2^{2d}$.

- $\widetilde{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})$: 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

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

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

($\mathbb{H}$-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.

8.3 Left subgroup indistinguishability 1

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

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

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

$$\begin{aligned}
\mathrm{PP} &:= \left(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \mu; g_1^{\mathbf{A}}, g_1^{\mathbf{W}_1^\top\mathbf{A}}, \dots, g_1^{\mathbf{W}_n^\top\mathbf{A}}\right);\\
\mathbf{h}_j &:= \left(g_2^{\mathbf{B}\mathbf{r}_j}, g_2^{\mathbf{W}_1\mathbf{B}\mathbf{r}_j}, \dots, g_2^{\mathbf{W}_n\mathbf{B}\mathbf{r}_j}\right), \quad \forall j\in[q'];\\
\mathbf{g}_j &:= \left(g_1^{\mathbf{A}\mathbf{s}_j}, g_1^{\mathbf{W}_1^\top\mathbf{A}\mathbf{s}_j}, \dots, g_1^{\mathbf{W}_n^\top\mathbf{A}\mathbf{s}_j}\right), \quad \forall j\in[q];\\
\mathbf{g}_j\cdot\widehat{\mathbf{g}}_j\cdot\widetilde{\mathbf{g}}_j &:= \left(g_1^{\mathbf{A}\mathbf{s}_j+\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j}, g_1^{\mathbf{W}_1^\top(\mathbf{A}\mathbf{s}_j+\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j)}, \dots, g_1^{\mathbf{W}_n^\top(\mathbf{A}\mathbf{s}_j+\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j)}\right), \quad \forall j\in[q];
\end{aligned}$$

for $\mathbf{r}_j \leftarrow \mathbb{Z}_p^d$ for all $j\in[q']$, $\mathbf{s}_j \leftarrow \mathbb{Z}_p^d, \widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^{d/2}$ for all $j\in[q]$.

Lemma 12 ((d,d,q)-Lin $\Rightarrow$ LS1) *For any probabilistic polynomial time adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

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

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

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

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

as input where all $s_{d+i,j}$ with $i\in[d]$ and $j\in[q]$ 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

$$\mathbf{s}_j = \left(s_{1,j}, \dots, s_{d,j}\right)^\top \text{ and } \widehat{\mathbf{s}}_j = \left(s_{d+1,j}, \dots, s_{3d/2,j}\right)^\top \text{ and } \widetilde{\mathbf{s}}_j = \left(s_{(3d/2+1),j}, \dots, s_{2d,j}\right)^\top, \quad \forall j\in[q].$$

Programming $\mathbf{D}, \mathbf{D}^*, \mathbf{W}_1, \cdots, \mathbf{W}_n$. Define

$$\mathbf{V} := \left(\begin{array}{ccc|ccc|ccc}
a_1 & & & & & & & & \\
 & \ddots & & & & & & & \\
 & & a_d & & & & & & \\
\hline
a_{d+1} & \cdots & a_{d+1} & 1 & & & & & \\
\vdots & & \vdots & & \ddots & & & & \\
a_{3d/2} & \cdots & a_{3d/2} & & & 1 & & & \\
\hline
a_{3d/2+1} & \cdots & a_{3d/2+1} & & & & 1 & & \\
\vdots & & \vdots & & & & & \ddots & \\
a_{2d} & \cdots & a_{2d} & & & & & & 1
\end{array}\right) \in \mathbb{Z}_p^{2d\times 2d}.$$

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

$$g_1^{\mathbf{A}} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{V})} = g_1^{\bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{V})} \quad\text{and}\quad g_1^{\mathbf{W}_i^\top\mathbf{A}} = g_1^{\mathbf{W}_i^\top\pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{V})} = g_1^{\mathbf{W}_i^\top\bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{V})}, \quad \forall i \in [n]$$

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 $j \in [q']$. By a simple calculation, we have

$$\mathbf{V}^* := \left(\begin{array}{ccc|ccc|ccc}
a_1^{-1} & & & -a_1^{-1}a_{d+1} & \cdots & -a_1^{-1}a_{3d/2} & -a_1^{-1}a_{3d/2+1} & \cdots & -a_1^{-1}a_{2d} \\
 & \ddots & & \vdots & & \vdots & \vdots & & \vdots \\
 & & a_d^{-1} & -a_d^{-1}a_{d+1} & \cdots & -a_d^{-1}a_{3d/2} & -a_d^{-1}a_{3d/2+1} & \cdots & -a_d^{-1}a_{2d} \\
\hline
 & & & 1 & & & & & \\
 & & & & \ddots & & & & \\
 & & & & & 1 & & & \\
\hline
 & & & & & & 1 & & \\
 & & & & & & & \ddots & \\
 & & & & & & & & 1
\end{array}\right) \in \mathbb{Z}_p^{2d\times 2d}.$$

Observe that the upper-left $d \times d$ sub-matrix of $\pi_{\mathrm{L}}(\mathbf{V}^*)$ is full-rank with overwhelming probability and therefore we have

$$\left\{\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j : \mathbf{r}_j \leftarrow \mathbb{Z}_p^d\right\} = \left\{\begin{pmatrix}\bar{\mathbf{r}}_j \\ \mathbf{0}_d\end{pmatrix} : \bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d\right\},$$

which means that $\mathscr{B}$ may simulate

$$g_2^{\mathbf{B}\mathbf{r}_j} = g_2^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}^*\mathbf{V}^*)\mathbf{r}_j} = g_2^{\bar{\mathbf{D}}^*\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j} \quad\text{and}\quad g_2^{\mathbf{W}_i\mathbf{B}\mathbf{r}_j} = g_2^{\mathbf{W}_i\pi_{\mathrm{L}}(\bar{\mathbf{D}}^*\mathbf{V}^*)\mathbf{r}_j} = g_2^{\mathbf{W}_i\bar{\mathbf{D}}^*\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j}, \quad \forall i \in [n],$$

by sampling $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q']$ and setting

$$g_2^{\mathbf{B}\mathbf{r}_j} = g_2^{\bar{\mathbf{D}}^*\begin{pmatrix}\bar{\mathbf{r}}_j \\ \mathbf{0}_d\end{pmatrix}} \quad\text{and}\quad g_2^{\mathbf{W}_i\mathbf{B}\mathbf{r}_j} = g_2^{\mathbf{W}_i\bar{\mathbf{D}}^*\begin{pmatrix}\bar{\mathbf{r}}_j \\ \mathbf{0}_d\end{pmatrix}}, \quad \forall i \in [n],\ j \in [q'].$$

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

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

and

$$g_1^{\mathbf{W}_i^\top(\mathbf{A}\mathbf{s}_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j)} = g_1^{\mathbf{W}_i^\top\bar{\mathbf{D}}\mathbf{V}\begin{pmatrix}\mathbf{s}_j \\ \widehat{\mathbf{s}}_j \\ \widetilde{\mathbf{s}}_j\end{pmatrix}} = g_1^{\mathbf{W}_i^\top\bar{\mathbf{D}}\begin{pmatrix} a_1 s_{1,j} \\ \vdots \\ a_d s_{d,j} \\ a_{d+1}(s_{1,j}+\cdots+s_{d,j})+s_{d+1,j} \\ \vdots \\ a_{2d}(s_{1,j}+\cdots+s_{d,j})+s_{2d,j}\end{pmatrix}}, \quad \forall i \in [n],\ j \in [q].$$

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

We immediately have the following corollary from Lemma 1.

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

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

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q') \cdot \mathsf{poly}(k,n)$ *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 $\mathsf{Adv}_{\mathscr{A}}^{\mathrm{LS2}}(k,q,q')$ as follows:

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

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(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);\\
\widehat{h}_j^* \cdot \widetilde{h}_j^* &:= g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}_j + \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j}, \quad \forall j \in [q+q'];\\
\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), \quad \forall j \in [q];\\
\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), \quad \forall j \in [q'];\\
\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), \quad \forall j \in [q];\\
\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), \quad \forall j \in [q];
\end{aligned}$$

for $\widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^{d/2}$ for all $j \in [q+q']$, $\mathbf{r}_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q']$, $\mathbf{s}_j, \mathbf{s}'_j \leftarrow \mathbb{Z}_p^d$, $\widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j, \widehat{\mathbf{s}}'_j, \widetilde{\mathbf{s}}'_j \leftarrow \mathbb{Z}_p^{d/2}$ for all $j \in [q]$.

Lemma 13 ($(d,d/2,q)$-LinAI ⇒ LS2) *For any probabilistic polynomial time adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

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

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

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

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

along with auxiliary input

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

as input where all $s_{d+i,j}$ with $i \in [d/2]$ and $j \in [q]$ 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 $\bar{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^{d/2}$ for all $j \in [q]$. Adversary $\mathscr{B}$ implicitly sets

$$\mathbf{s}_j = \left(\bar{\mathbf{s}}_j, s_{1,j}, \ldots, s_{(d/2),j}\right)^\top \text{ and } \widehat{\mathbf{s}}_j = \left(s_{(d+1),j}, \ldots, s_{(3d/2),j}\right)^\top \text{ and } \widetilde{\mathbf{s}}_j = \left(s_{(d/2+1),j}, \ldots, s_{d,j}\right)^\top, \quad \forall j \in [q].$$

Programming $\mathbf{D}, \mathbf{D}^*, \mathbf{W}_1, \cdots, \mathbf{W}_n$. Define

$$\mathbf{V} := \left(\begin{array}{ccc|ccc|ccc|ccc}
1 & & & & & & & & & & & \\
& \ddots & & & & & & & & & & \\
& & 1 & & & & & & & & & \\
\hline
& & & a_1 & & & & & & & & \\
& & & & \ddots & & & & & & & \\
& & & & & a_{d/2} & & & & & & \\
\hline
& & & a_{d+1} & \cdots & a_{d+1} & 1 & & & a_{d+1} & \cdots & a_{d+1} \\
& & & \vdots & & \vdots & & \ddots & & \vdots & & \vdots \\
& & & a_{3d/2} & \cdots & a_{3d/2} & & & 1 & a_{3d/2} & \cdots & a_{3d/2} \\
\hline
& & & & & & & & & a_{d/2+1} & & \\
& & & & & & & & & & \ddots & \\
& & & & & & & & & & & a_d
\end{array}\right) \in \mathbb{Z}_p^{2d \times 2d}.$$

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

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

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

$$g_1^{\mathbf{A}} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{V})} = g_1^{\bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{V})} \quad \text{and} \quad g_1^{\mathbf{W}_i^\top \mathbf{A}} = g_1^{\mathbf{W}_i^\top \pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{V})} = g_1^{\mathbf{W}_i^\top \bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{V})}, \quad \forall i \in [n]$$

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 $j \in [q+q']$. By a simple calculation, we have

$$\mathbf{V}^* := \left(\begin{array}{ccc|ccc|ccc|ccc}
1 & & & & & & & & & & & \\
& \ddots & & & & & & & & & & \\
& & 1 & & & & & & & & & \\
\hline
& & & a_1^{-1} & & & -a_1^{-1}a_{d+1} & \cdots & -a_1^{-1}a_{3d/2} & & & \\
& & & & \ddots & & \vdots & & \vdots & & & \\
& & & & & a_{d/2}^{-1} & -a_{d/2}^{-1}a_{d+1} & \cdots & -a_{d/2}^{-1}a_{3d/2} & & & \\
\hline
& & & & & & 1 & & & & & \\
& & & & & & & \ddots & & & & \\
& & & & & & & & 1 & & & \\
\hline
& & & & & & -a_{d/2+1}^{-1}a_{d+1} & \cdots & -a_{d/2+1}^{-1}a_{3d/2} & a_{d/2+1}^{-1} & & \\
& & & & & & \vdots & & \vdots & & \ddots & \\
& & & & & & -a_d^{-1}a_{d+1} & \cdots & -a_d^{-1}a_{3d/2} & & & a_d^{-1}
\end{array}\right) \in \mathbb{Z}_p^{2d \times 2d}.$$

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 the following one

$$\left\{ \left(\begin{array}{ccc|c} & & & \\ & & & \\ \hline -aa_1^{-1}a_{d+1} & \cdots & -aa_1^{-1}a_{3d/2} & \\ \vdots & & \vdots & \\ -aa_{d/2}^{-1}a_{d+1} & \cdots & -aa_{d/2}^{-1}a_{3d/2} & \\ \hline a & & & \\ & \ddots & & \\ & & a & \\ \hline & & & \begin{matrix} 1 & & \\ & \ddots & \\ & & 1 \end{matrix} \end{array} \right) \begin{pmatrix} \widehat{\mathbf{r}}'_j \\ \widetilde{\mathbf{r}}'_j \end{pmatrix} : \widehat{\mathbf{r}}'_j, \widetilde{\mathbf{r}}'_j \leftarrow \mathbb{Z}_p^{d/2} \right\} \tag{1}$$

which means that algorithm $\mathscr{B}$ can simulate

$$\widehat{h}_j^* \cdot \widetilde{h}_j^* = g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}_j + \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j} = g_2^{\bar{\mathbf{D}}^*\left(\pi_{\mathrm{M}}(\mathbf{V}^*)\widehat{\mathbf{r}}_j + \pi_{\mathrm{R}}(\mathbf{V}^*)\widetilde{\mathbf{r}}_j\right)}, \quad j \in [q+q'];$$

by sampling $\widehat{\mathbf{r}}'_j, \widetilde{\mathbf{r}}'_j \leftarrow \mathbb{Z}_p^{d/2}$ and setting

$$\widehat{h}_j^* \cdot \widetilde{h}_j^* = g_2^{\bar{\mathbf{D}}^*\bar{\mathbf{V}}^*\begin{pmatrix} \widehat{\mathbf{r}}'_j \\ \widetilde{\mathbf{r}}'_j \end{pmatrix}}, \quad j \in [q+q'];$$

where matrix $\bar{\mathbf{V}}^*$ refers to the matrix in Eq. 1 using the knowledge of Aux (i.e., $g_2^{\bar{\mathbf{V}}^*}$) and $\bar{\mathbf{D}}^*$.

Simulating $\mathbf{h}_j$ for $j \in [q']$. Observe that the upper-left $d \times d$ sub-matrix of $\pi_{\mathrm{L}}(\mathbf{V}^*)$ is full-rank with overwhelming probability and therefore we have

$$\left\{\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j : \mathbf{r}_j \leftarrow \mathbb{Z}_p^d\right\} = \left\{ \begin{pmatrix} \bar{\mathbf{r}}_j \\ \mathbf{0}_d \end{pmatrix} : \bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d \right\},$$

which means that $\mathscr{B}$ may simulate

$$g_2^{\mathbf{B}\mathbf{r}_j} = g_2^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}^*\mathbf{V}^*)\mathbf{r}_j} = g_2^{\bar{\mathbf{D}}^*\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j} \quad \text{and} \quad g_2^{\mathbf{W}_i\mathbf{B}\mathbf{r}_j} = g_2^{\mathbf{W}_i\pi_{\mathrm{L}}(\bar{\mathbf{D}}^*\mathbf{V}^*)\mathbf{r}_j} = g_2^{\mathbf{W}_i\bar{\mathbf{D}}^*\pi_{\mathrm{L}}(\mathbf{V}^*)\mathbf{r}_j}, \quad \forall i \in [n],$$

by sampling $\bar{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^d$ for all $j \in [q']$ and setting

$$g_2^{\mathbf{B}\mathbf{r}_j} = g_2^{\bar{\mathbf{D}}^*\begin{pmatrix} \bar{\mathbf{r}}_j \\ \mathbf{0}_d \end{pmatrix}} \quad \text{and} \quad g_2^{\mathbf{W}_i\mathbf{B}\mathbf{r}_j} = g_2^{\mathbf{W}_i\bar{\mathbf{D}}^*\begin{pmatrix} \bar{\mathbf{r}}_j \\ \mathbf{0}_d \end{pmatrix}}, \quad \forall i \in [n],\ j \in [q'].$$

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

$$g_1^{\mathbf{A}\mathbf{s}'_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}'_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j} = g_1^{\bar{\mathbf{D}}\mathbf{V}\begin{pmatrix} \mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \widetilde{\mathbf{s}}'_j \end{pmatrix}} \quad \text{and} \quad g_1^{\mathbf{W}_i^\top(\mathbf{A}\mathbf{s}'_j + \widehat{\mathbf{A}}\widehat{\mathbf{s}}'_j + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j)} = g_1^{\mathbf{W}_i^\top\bar{\mathbf{D}}\mathbf{V}\begin{pmatrix} \mathbf{s}'_j \\ \widehat{\mathbf{s}}'_j \\ \widetilde{\mathbf{s}}'_j \end{pmatrix}}, \quad \forall i \in [n], j \in [q],$$

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

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

and

$$g_1^{\mathbf{W}_i^\top(\mathbf{A}\mathbf{s}_j+\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j)} = g_1^{\mathbf{W}_i^\top \bar{\mathbf{D}}\mathbf{V}\begin{pmatrix}\mathbf{s}_j\\ \widehat{\mathbf{s}}_j\\ \widetilde{\mathbf{s}}_j\end{pmatrix}} = g_1^{\mathbf{W}_i^\top \bar{\mathbf{D}}\begin{pmatrix}\bar{\mathbf{s}}_j\\ a_1 s_{1,j}\\ \vdots\\ a_{d/2}s_{d/2,j}\\ a_{d+1}(s_{1,j}+\cdots+s_{d,j})+s_{d+1,j}\\ \vdots\\ a_{3d/2}(s_{1,j}+\cdots+s_{d,j})+s_{3d/2,j}\\ a_{d/2+1}s_{(d/2+1),j}\\ \vdots\\ a_d s_{d,j}\end{pmatrix}}, \quad \forall i\in[n], j\in[q].$$

Analysis. Observe that if $s_{d+i,j}=0$ for all $i\in[d/2]$ and $j\in[q]$, then $\widehat{\mathbf{s}}_j=\mathbf{0}_{d/2}$ for all $j\in[q]$ 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 $s_{d+i,j}\leftarrow\mathbb{Z}_p^*$ for all $i\in[d/2]$ and $j\in[q]$, then $\widehat{\mathbf{s}}_j\leftarrow(\mathbb{Z}_p^*)^{d/2}$ for all $j\in[q]$ 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 $\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{A}}(k,q,q')\leqslant\mathsf{Adv}^{(d,d/2,q)\text{-LinAI}}_{\mathscr{B}}(k)$. $\square$

We immediately have the following corollary from Lemma 11.

Corollary 6 (d-LinAI $\Rightarrow$ LS2) *For any probabilistic polynomial time 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 $\mathsf{Time}(\mathscr{B})\approx\mathsf{Time}(\mathscr{A})+(q+q')\cdot\mathsf{poly}(k,n)$ *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:

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

where

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

and

$$\begin{aligned}
\text{PP} &:= \left(p,\mathbb{G},\mathbb{H},\mathbb{G}_T,e,\mu;g_1^{\mathbf{A}},g_1^{\mathbf{W}_1^\top\mathbf{A}},\ldots,g_1^{\mathbf{W}_n^\top\mathbf{A}}\right);\\
\widehat{h}_j^*\cdot\widetilde{h}_j^* &:= g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}_j+\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j},\quad\forall j\in[q+q'];\\
\mathbf{g}'_j\cdot\widetilde{\mathbf{g}}'_j &:= \left(g_1^{\mathbf{A}\mathbf{s}'_j+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j},g_1^{\mathbf{W}_1^\top(\mathbf{A}\mathbf{s}'_j+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j)},\ldots,g_1^{\mathbf{W}_n^\top(\mathbf{A}\mathbf{s}'_j+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}'_j)}\right),\quad\forall j\in[q];\\
\mathbf{h}_j &:= \left(g_2^{\mathbf{B}\mathbf{r}_j},g_2^{\mathbf{W}_1\mathbf{B}\mathbf{r}_j},\ldots,g_2^{\mathbf{W}_n\mathbf{B}\mathbf{r}_j}\right),\quad\forall j\in[q'];\\
\mathbf{g}_j\cdot\widehat{\mathbf{g}}_j\cdot\widetilde{\mathbf{g}}_j &:= \left(g_1^{\mathbf{A}\mathbf{s}_j+\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j},g_1^{\mathbf{W}_1^\top(\mathbf{A}\mathbf{s}_j+\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j)},\ldots,g_1^{\mathbf{W}_n^\top(\mathbf{A}\mathbf{s}_j+\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j+\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j)}\right),\quad\forall j\in[q];\\
\mathbf{g}_j\cdot\widehat{\mathbf{g}}_j &:= \left(g_1^{\mathbf{A}\mathbf{s}_j+\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j},g_1^{\mathbf{W}_1^\top(\mathbf{A}\mathbf{s}_j+\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j)},\ldots,g_1^{\mathbf{W}_n^\top(\mathbf{A}\mathbf{s}_j+\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j)}\right),\quad\forall j\in[q];
\end{aligned}$$

for $\widehat{\mathbf{r}}_j,\widetilde{\mathbf{r}}_j\leftarrow\mathbb{Z}_p^{d/2}$ for all $j\in[q+q']$, $\mathbf{r}_j\leftarrow\mathbb{Z}_p^d$ for all $j\in[q']$, $\mathbf{s}_j,\mathbf{s}'_j\leftarrow\mathbb{Z}_p^d$, $\widehat{\mathbf{s}}_j,\widetilde{\mathbf{s}}_j,\widetilde{\mathbf{s}}'_j\leftarrow\mathbb{Z}_p^{d/2}$ for all $j\in[q]$.

Lemma 14 ($(d,d/2,q)$-LinAI $\Rightarrow$ LS3) *For any probabilistic polynomial time adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

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

and $\mathsf{Time}(\mathscr{B})\approx\mathsf{Time}(\mathscr{A})+(q+q')\cdot\mathsf{poly}(k,n)$ *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 $(d,d/2,q)$-linear problem (i.e., set $\ell=d$)

$$\left(g_1,g_2,g_1^{a_1},\ldots,g_1^{a_d},\left\{g_1^{a_{d+i}}\right\}_{i\in[d/2]},\left\{g_1^{a_1s_{1,j}},\ldots,g_1^{a_ds_{d,j}}\right\}_{j\in[q]},\left\{g_1^{a_{d+i}(s_{1,j}+\cdots+s_{d,j})+s_{d+i,j}}\right\}_{i\in[d/2],j\in[q]}\right)$$

along with auxiliary input

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

as input where all $s_{d+i,j}$ with $i \in [d/2]$ and $j \in [q]$ 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 $\bar{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^{d/2}$ for all $j \in [q]$. Adversary $\mathscr{B}$ implicitly sets

$$\mathbf{s}_j = \left(\bar{\mathbf{s}}_j, s_{1,j}, \ldots, s_{(d/2),j} \right)^\top \text{ and } \widehat{\mathbf{s}}_j = \left(s_{(d/2+1),j}, \ldots, s_{d,j} \right)^\top \text{ and } \widetilde{\mathbf{s}}_j = \left(s_{(d+1),j}, \ldots, s_{(3d/2),j} \right)^\top, \quad \forall j \in [q].$$

Programming V. Define

$$\mathbf{V} := \left(\begin{array}{ccc|ccc|ccc|ccc}
1 & & & & & & & & & & & \\
 & \ddots & & & & & & & & & & \\
 & & 1 & & & & & & & & & \\
\hline
 & & & a_1 & & & & & & & & \\
 & & & & \ddots & & & & & & & \\
 & & & & & a_{d/2} & & & & & & \\
\hline
 & & & & & & a_{d/2+1} & & & & & \\
 & & & & & & & \ddots & & & & \\
 & & & & & & & & a_d & & & \\
\hline
 & & & a_{d+1} & \cdots & a_{d+1} & a_{d+1} & \cdots & a_{d+1} & 1 & & \\
 & & & \vdots & & \vdots & \vdots & & \vdots & & \ddots & \\
 & & & a_{3d/2} & \cdots & a_{3d/2} & a_{3d/2} & \cdots & a_{3d/2} & & & 1
\end{array} \right) \in \mathbb{Z}_p^{2d \times 2d}.$$

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

We immediately have the following corollary from Lemma 11.

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

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

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

8.6 Nested-hiding indistinguishability

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

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

where

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

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

and

$$\begin{aligned}
\textsc{pp} &:= \left(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \mu; g_1^{\mathbf{A}}, g_1^{\mathbf{W}_1^\top \mathbf{A}}, \ldots, g_1^{\mathbf{W}_n^\top \mathbf{A}} \right); \\
\widehat{h}_j^* &:= g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}_j'}, \quad \widetilde{h}_j^* := g_2^{\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j'}, \quad \forall j \in [q+q']; \\
\widehat{\mathbf{g}}_j &:= \left(g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j}, g_1^{\mathbf{W}_1^\top \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j}, \ldots, g_1^{\mathbf{W}_n^\top \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j} \right), \quad \forall j \in [q]; \\
\widetilde{\mathbf{g}}_j &:= \left(g_1^{\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j}, g_1^{\mathbf{W}_1^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j}, \ldots, g_1^{\mathbf{W}_n^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j} \right), \quad \forall j \in [q]; \\
\mathbf{h}_j' &:= \left(g_2^{\mathbf{B}\mathbf{r}_j'}, g_2^{\mathbf{W}_1 \mathbf{B}\mathbf{r}_j'}, \ldots, g_2^{\mathbf{W}_n \mathbf{B}\mathbf{r}_j'} \right), \quad \forall j \in [q']; \\
\mathbf{h}_j \cdot (\widehat{h}_j^{**})^{\mathbf{e}_{2\eta-1}} \cdot (\widetilde{h}_j^{**})^{\mathbf{e}_{2\eta}} &:= \left(g_2^{\mathbf{B}\mathbf{r}_j}, g_2^{\mathbf{W}_1 \mathbf{B}\mathbf{r}_j}, \ldots, g_2^{\mathbf{W}_{2\eta-1}\mathbf{B}\mathbf{r}_j + \widehat{\mathbf{B}}\widehat{\mathbf{r}}_j}, g_2^{\mathbf{W}_{2\eta}\mathbf{B}\mathbf{r}_j + \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j}, \ldots, g_2^{\mathbf{W}_n \mathbf{B}\mathbf{r}_j} \right), \quad \forall j \in [q'];
\end{aligned}$$

for $\widehat{\mathbf{r}}'_j, \widetilde{\mathbf{r}}'_j \leftarrow \mathbb{Z}_p^{d/2}$ for all $j \in [q+q']$, $\widehat{\mathbf{s}}_j, \widetilde{\mathbf{s}}_j \leftarrow \mathbb{Z}_p^{d/2}$ for all $j \in [q]$, and $\mathbf{r}_j, \mathbf{r}'_j \leftarrow \mathbb{Z}_p^d$, $\widehat{\mathbf{r}}_j, \widetilde{\mathbf{r}}_j \leftarrow \mathbb{Z}_p^{d/2}$ for all $j \in [q']$.

Lemma 15 ((d,d,d)-Lin $\Rightarrow$ NH) *For all $\eta \in [\lfloor n/2 \rfloor]$ and for any probabilistic polynomial time adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

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

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

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

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

where all $r_{d+i,j}$ for $i,j \in [d]$ 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)-linear 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' tuples

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

where $\mathbf{t}'_j \in \mathbb{Z}_p^d$ using $\left(g_2^{\mathbf{VZ}}, g_2^{\mathbf{Z}}\right)$. In the simulation, we will not use $\left(g_2^{\mathbf{VZ}}, g_2^{\mathbf{Z}}\right)$ anymore.

Programming $\mathbf{D}, \mathbf{D}^*$ and $\mathbf{W}_1, \dots, \mathbf{W}_n$. 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 $\mathbf{W}_1, \dots, \mathbf{W}_{2(\eta-1)}, \bar{\mathbf{W}}_{2\eta-1}, \bar{\mathbf{W}}_{2\eta}, \mathbf{W}_{2(\eta+1)-1} \dots, \mathbf{W}_n \leftarrow \mathbb{Z}_p^{2d\times 2d}$ 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

$$g_1^{\mathbf{A}} = g_1^{\pi_{\mathrm{L}}(\mathbf{D})} \quad \text{and} \quad g_1^{\mathbf{W}_i^\top \mathbf{A}} = g_1^{\mathbf{W}_i^\top \pi_{\mathrm{L}}(\mathbf{D})}, \quad \forall i \in [n] \setminus \{2\eta-1, 2\eta\}$$

using the knowledge of $\mathbf{D}$ and $\mathbf{W}_1, \dots, \mathbf{W}_{2\eta-2}, \mathbf{W}_{2\eta+1}, \dots, \mathbf{W}_n$. Observe that

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

and

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

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

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

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

$$\widehat{h}_j^* = g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}'_j} = g_2^{\pi_{\mathrm{M}}(\mathbf{D}^*)\widehat{\mathbf{r}}'_j} \quad \text{and} \quad \widetilde{h}_j^* = g_2^{\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}'_j} = g_2^{\pi_{\mathrm{R}}(\mathbf{D}^*)\widetilde{\mathbf{r}}'_j}, \quad \forall j \in [q+q'],$$

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

$$g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}_j} = g_1^{\pi_{\mathrm{M}}(\mathbf{D})\widehat{\mathbf{s}}_j} \quad \text{and} \quad g_1^{\mathbf{W}_i^\top \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j} = g_1^{\mathbf{W}_i^\top \pi_{\mathrm{M}}(\mathbf{D})\widehat{\mathbf{s}}_j}, \quad \forall i \in [n] \setminus \{2\eta-1, 2\eta\}, j \in [q]$$

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

$$g_1^{\mathbf{W}_{2\eta}^\top \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j} = g_1^{\bar{\mathbf{W}}_{2\eta}^\top \widehat{\mathbf{A}}\widehat{\mathbf{s}}_j} = g_1^{\bar{\mathbf{W}}_{2\eta}^\top \pi_{\mathrm{M}}(\mathbf{D})\widehat{\mathbf{s}}_j}, \quad \forall j \in [q]$$

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

$$g_1^{\widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j} = g_1^{\pi_{\mathrm{R}}(\mathbf{D})\widetilde{\mathbf{s}}_j} \quad \text{and} \quad g_1^{\mathbf{W}_i^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j} = g_1^{\mathbf{W}_i^\top \pi_{\mathrm{R}}(\mathbf{D})\widetilde{\mathbf{s}}_j}, \quad \forall i \in [n] \setminus \{2\eta-1, 2\eta\}, j \in [q]$$

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

$$g_1^{\mathbf{W}_{2\eta-1}^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j} = g_1^{\bar{\mathbf{W}}_{2\eta-1}^\top \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_j} = g_1^{\bar{\mathbf{W}}_{2\eta-1}^\top \pi_{\mathrm{R}}(\mathbf{D})\widetilde{\mathbf{s}}_j}, \quad \forall j \in [q]$$

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

$$\mathbf{B}\mathbf{r}'_j = \begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix}, \quad \forall j \in [q'].$$

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

$$g_2^{\mathbf{B}\mathbf{r}'_j} = g_2^{\begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix}} \quad \text{and} \quad g_2^{\mathbf{W}_i\mathbf{B}\mathbf{r}'_j} = g_2^{\mathbf{W}_i\begin{pmatrix} \mathbf{t}'_j \\ \mathbf{T}\mathbf{t}'_j \end{pmatrix}}, \quad \forall i \in [n] \setminus \{2\eta-1, 2\eta\},\ j \in [q']$$

using $g_2^{\mathbf{t}'_j}$ and the knowledge of $\mathbf{T}, \mathbf{W}_1, \ldots, \mathbf{W}_{2\eta-2}, \mathbf{W}_{2\eta+1}, \ldots, \mathbf{W}_n$. For all $j \in [q']$, 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

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

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

$$\mathbf{B}\mathbf{r}_j = \begin{pmatrix}\mathbf{t}_j\\ \mathbf{T}\mathbf{t}_j\end{pmatrix}, \quad \forall j \in [q'].$$

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

$$g_2^{\mathbf{B}\mathbf{r}_j} = g_2^{\begin{pmatrix}\mathbf{t}_j\\ \mathbf{T}\mathbf{t}_j\end{pmatrix}} \quad\text{and}\quad g_2^{\mathbf{W}_i\mathbf{B}\mathbf{r}_j} = g_2^{\mathbf{W}_i\begin{pmatrix}\mathbf{t}_j\\ \mathbf{T}\mathbf{t}_j\end{pmatrix}}, \quad \forall i \in [n]\setminus\{2\eta-1, 2\eta\},\ j\in[q']$$

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

$$g_2^{\mathbf{W}_{2\eta-1}\mathbf{B}\mathbf{r}_j+\widehat{\mathbf{B}}\widehat{\mathbf{r}}_j} = g_2^{\bar{\mathbf{W}}_{2\eta-1}\begin{pmatrix}\mathbf{t}_j\\ \mathbf{T}\mathbf{t}_j\end{pmatrix}+(\widehat{\mathbf{B}}|\mathbf{0}_{2d\times(d/2)})(\mathbf{V}\mathbf{t}_j+\boldsymbol{\tau}_j)} \quad\text{and}\quad g_2^{\mathbf{W}_{2\eta}\mathbf{B}\mathbf{r}_j+\widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j} = g_2^{\bar{\mathbf{W}}_{2\eta}\begin{pmatrix}\mathbf{t}_j\\ \mathbf{T}\mathbf{t}_j\end{pmatrix}+(\mathbf{0}_{2d\times(d/2)}|\widetilde{\mathbf{B}})(\mathbf{V}\mathbf{t}_j+\boldsymbol{\tau}_j)}, \ \forall j\in[q']$$

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

$$\begin{pmatrix}\widehat{\mathbf{r}}_j\\ \widetilde{\mathbf{r}}_j\end{pmatrix} = \boldsymbol{\tau}_j, \quad \forall j\in[q'],$$

when producing the challenge. Therefore, if $r_{d+i,j} = 0$ for all $i, j \in [d]$, then $\boldsymbol{\tau}_j = \mathbf{0}_d$ for all $j \in [q']$ and the output challenge has the same distribution as $\left\{\mathbf{h}_j\right\}_{j\in[q']}$; on the other hand, if $r_{d+i,j} \leftarrow \mathbb{Z}_p^*$ for all $i, j \in [d]$, then $\boldsymbol{\tau}_j \leftarrow (\mathbb{Z}_p^*)^d$ for all $j \in [q']$ 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 $\mathsf{Adv}_{\mathscr{A}}^{\mathrm{NH}(\eta)}(k,q,q') \leqslant \mathsf{Adv}_{\mathscr{B}}^{(d,d,d)\text{-Lin}}(k)$. □

We immediately have the following corollary from Lemma 1.

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

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

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (q+q')\cdot\mathsf{poly}(k,n)$ *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:

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

where

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

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

and

$$\begin{aligned}
\textsc{pp} &:= \left(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \mu; g_1^{\mathbf{A}}, g_1^{\mathbf{W}_1^\top \mathbf{A}}, \ldots, g_1^{\mathbf{W}_n^\top \mathbf{A}}\right);\\
\widehat{h}_j^* \cdot \widetilde{h}_j^* &:= g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}'_j + \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}'_j}, \quad \forall j \in [q'];\\
\mathbf{h}_j &:= \left(g_2^{\mathbf{B}\mathbf{r}_j}, g_2^{\mathbf{W}_1\mathbf{B}\mathbf{r}_j}, \ldots, g_2^{\mathbf{W}_n\mathbf{B}\mathbf{r}_j}\right), \quad \forall j \in [q'];\\
\widehat{\mathbf{g}}_{j,j'} \cdot \widetilde{\mathbf{g}}_{j,j'} &:= \left(g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}_{j,j'} + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_{j,j'}}, g_1^{\mathbf{W}_1^\top(\widehat{\mathbf{A}}\widehat{\mathbf{s}}_{j,j'} + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_{j,j'})}, \ldots, g_1^{\mathbf{W}_n^\top(\widehat{\mathbf{A}}\widehat{\mathbf{s}}_{j,j'} + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_{j,j'})}\right), \quad \forall j \in [q], j' \in [q''];\\
e(\widehat{g}_{0,j,j'} \cdot \widetilde{g}_{0,j,j'}, \widehat{h}_j^{**} \cdot \widetilde{h}_j^{**}) &:= e(g_1^{\widehat{\mathbf{A}}\widehat{\mathbf{s}}_{j,j'} + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_{j,j'}}, g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}_j + \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}_j}) = e(g_1, g_2)^{(\widehat{\mathbf{s}}_{j,j'}^\top, \widetilde{\mathbf{s}}_{j,j'}^\top)\binom{\widehat{\mathbf{r}}_j}{\widetilde{\mathbf{r}}_j}}, \quad \forall j \in [q], j' \in [q''];\\
\widehat{R}_{j,j'} &:= e(g_1, g_2)^{\widehat{\gamma}_{j,j'}}, \quad \forall j \in [q], j' \in [q''];
\end{aligned}$$

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

Lemma 16 ($(d, 1, qq'')$-Lin $\Rightarrow$ ND) *For any probabilistic polynomial time adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

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

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (qq'' + q') \cdot \mathsf{poly}(k, n)$ *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 $(d, 1, qq'')$-linear problem (i.e., set $\ell = 1$ and $q = qq''$)

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

as input where all $s_{d+1,j,j'}$ for $j \in [q]$ and $j' \in [q'']$ 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

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

Programming $\mathbf{D}, \mathbf{D}^*, \mathbf{W}_1, \cdots, \mathbf{W}_n$. We define $\mathbf{U}$ as

$$\mathbf{U} := \left(\begin{array}{c|c|c}
\begin{matrix}1 & & \\ & \ddots & \\ & & 1\end{matrix} & & \\
\hline
 & \begin{matrix}a_1 & & \\ & \ddots & \\ & & a_{d/2}\end{matrix} & \\
\hline
 & & \begin{matrix}a_{d/2+1} & & \\ & \ddots & \\ & & a_d\end{matrix}
\end{array}\right) \in \mathbb{Z}_p^{2d \times 2d}.$$

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

$$g_1^{\mathbf{A}} = g_1^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{U})} = g_1^{\bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{U})} \quad \text{and} \quad g_1^{\mathbf{W}_i^\top \mathbf{A}} = g_1^{\mathbf{W}_i^\top \pi_{\mathrm{L}}(\bar{\mathbf{D}}\mathbf{U})} = g_1^{\mathbf{W}_i^\top \bar{\mathbf{D}}\pi_{\mathrm{L}}(\mathbf{U})}, \quad \forall i \in [n]$$

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 $j \in [q']$. By a simple calculation, we have

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

Observe that the right-most $d \times d$ sub-matrix of $\mathbf{U}^*$ is full-rank with overwhelming probability and

$$\left\{ \mathbf{U}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}'_j \\ \widetilde{\mathbf{r}}'_j \end{pmatrix} : \widehat{\mathbf{r}}'_j, \widetilde{\mathbf{r}}'_j \leftarrow \mathbb{Z}_p^{d/2} \right\} = \left\{ \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}''_j \\ \widetilde{\mathbf{r}}''_j \end{pmatrix} : \widehat{\mathbf{r}}''_j, \widetilde{\mathbf{r}}''_j \leftarrow \mathbb{Z}_p^{d/2} \right\}.$$

which means that $\mathscr{B}$ may properly produce

$$\widehat{h}_j^* \cdot \widetilde{h}_j^* = g_2^{\widehat{\mathbf{B}}\widehat{\mathbf{r}}'_j + \widetilde{\mathbf{B}}\widetilde{\mathbf{r}}'_j} = g_2^{\bar{\mathbf{D}}^* \mathbf{U}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}'_j \\ \widetilde{\mathbf{r}}'_j \end{pmatrix}}, \quad \forall j \in [q']$$

by sampling $\widehat{\mathbf{r}}''_j, \widetilde{\mathbf{r}}''_j \leftarrow \mathbb{Z}_p^{d/2}$ for all $j \in [q']$ and setting

$$\widehat{h}_j^* \cdot \widetilde{h}_j^* = g_2^{\bar{\mathbf{D}}^* \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{r}}''_j \\ \widetilde{\mathbf{r}}''_j \end{pmatrix}}, \quad \forall j \in [q'].$$

Simulating $\mathbf{h}_j$ for all $j \in [q']$. Algorithm $\mathscr{B}$ may compute

$$\textsc{hp} := \left(g_2^{\mathbf{B}}, g_2^{\mathbf{W}_1 \mathbf{B}}, \ldots, g_2^{\mathbf{W}_n \mathbf{B}} \right)$$

where

$$g_2^{\mathbf{B}} = g_2^{\pi_{\mathrm{L}}(\bar{\mathbf{D}}^* \mathbf{U}^*)} = g_2^{\bar{\mathbf{D}}^* \pi_{\mathrm{L}}(\mathbf{U}^*)} \quad \text{and} \quad g_2^{\mathbf{W}_i \mathbf{B}} = g_2^{\mathbf{W}_i \pi_{\mathrm{L}}(\bar{\mathbf{D}}^* \mathbf{U}^*)} = g_2^{\mathbf{W}_i \bar{\mathbf{D}}^* \pi_{\mathrm{L}}(\mathbf{U}^*)}, \ \forall i \in [n]$$

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

$$\left\{ \mathbf{h}_j \right\}_{j \in [q']} \leftarrow \mathsf{SampH}^{q'}(\textsc{pp}, \textsc{hp}).$$

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

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

and

$$g_1^{\mathbf{W}_i^\top \left(\widehat{\mathbf{A}}\widehat{\mathbf{s}}_{j,j'} + \widetilde{\mathbf{A}}\widetilde{\mathbf{s}}_{j,j'} \right)} = g_1^{\mathbf{W}_i^\top \bar{\mathbf{D}}\mathbf{U} \begin{pmatrix} \mathbf{0}_d \\ \widehat{\mathbf{s}}_{j,j'} \\ \widetilde{\mathbf{s}}_{j,j'} \end{pmatrix}} = g_1^{\mathbf{W}_i^\top \bar{\mathbf{D}} \begin{pmatrix} \mathbf{0}_d \\ a_1 s_{1,j,j'} \\ \vdots \\ a_d s_{d,j,j'} \end{pmatrix}} , \forall i \in [n], j \in [q], j \in [q'],$$

using the knowledge of $\left\{ g_1^{a_1 s_{1,j,j'}}, \ldots, g_1^{a_d s_{d,j,j'}} \right\}$ and $\bar{\mathbf{D}}, \mathbf{W}_1, \ldots, \mathbf{W}_n$.

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

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

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

$$g_1^{\left(\widehat{\mathbf{r}}_j^\top, \widetilde{\mathbf{r}}_j^\top\right)\begin{pmatrix}\widehat{\mathbf{s}}_{j,j'}\\ \widetilde{\mathbf{s}}_{j,j'}\end{pmatrix} + \widehat{\gamma}_{j,j'}} = g_1^{\bar{\mathbf{r}}_j^\top \begin{pmatrix} a_1 s_{1,j,j'} \\ \vdots \\ a_d s_{d,j,j'} \\ a_{d+1}(s_{1,j,j'} + \cdots + s_{d,j,j'}) + s_{d+1,j,j'} \end{pmatrix}}, \quad \forall j \in [q],\ j' \in [q'']$$

using the knowledge of $\left\{ g_1^{a_1 s_{1,j,j'}}, \ldots, g_1^{a_d s_{d,j,j'}}, g_1^{a_{d+1}(s_{1,j,j'} + \cdots + s_{d,j,j'}) + s_{d+1,j,j'}} \right\}$, and outputs the challenge as

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

Analysis. Observe that, if $s_{d+1,j,j'} = 0$ for all $j \in [q]$ and $j' \in [q']$, then the output challenge is distributed as

$$e(g_1^{\bar{\mathbf{r}}_j^\top \mathbf{V} \begin{pmatrix}\widehat{\mathbf{s}}_{j,j'}\\ \widetilde{\mathbf{s}}_{j,j'}\end{pmatrix}}, g_2) = e(g_1, g_2)^{(\widehat{\mathbf{s}}_{j,j'}^\top, \widetilde{\mathbf{s}}_{j,j'}^\top)\begin{pmatrix}\widehat{\mathbf{r}}_j\\ \widetilde{\mathbf{r}}_j\end{pmatrix}}, \quad \forall j \in [q],\ j' \in [q'],$$

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

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

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

We immediately have the following corollary from Lemma 1.

Corollary 9 (d-Lin $\Rightarrow$ ND) *For any probabilistic polynomial time adversary $\mathscr{A}$, there exists an adversary $\mathscr{B}$ such that*

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

and $\mathsf{Time}(\mathscr{B}) \approx \mathsf{Time}(\mathscr{A}) + (qq'' + q') \cdot \mathsf{poly}(k, n)$ *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, i.e., a $2d \times d$ sub-matrix.

- $\mathsf{Param}(1^k, n)$: 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

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

- $\mathsf{Setup}(\mathrm{GP})$: 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

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

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

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

– $\mathsf{Dec}(\text{MPK}, \text{SK}, \text{CT})$. Let $\text{SK} = (K_0, K_1)$ and $\text{CT} = (C_0, C_1, C_2)$. Output

$$\text{M} := C_2 \frac{e(C_1, K_0)}{e(C_0, K_1)}.$$

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

Assumption 5 (2-Linear Assumption in G_1 with Auxiliary Input in G_2) *For any probabilistic polynomial time adversary $\mathcal{A}$, the following advantage function is negligible in k,*

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

where

$$D := \left(\mathcal{G}, g_1, g_2, g_1^{a_1}, g_1^{a_2}, g_1^{a_3}, g_1^{a_1 s_1}, g_1^{a_2 s_2} \right),\ \text{AUX} := \left(g_2^{a_3}, g_2^{a_1} \right),\ T_0 := g_1^{a_3(s_1+s_2)},\ T_1 := g_1^{a_3(s_1+s_2) + \boxed{s_3}}$$

and $\mathcal{G} := (p, G_1, G_2, G_T, e) \leftarrow \mathsf{GrpGen}(1^k)$ 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.1). 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.2), 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. 8

[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. 8

[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, 7, 48, 50

[AHY15] Nuttapong Attrapadung, Goichiro Hanaoka, and Shota Yamada. A framework for identity-based encryption with almost tight security. *IACR Cryptology ePrint Archive*, 2015. 8

[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, 8

[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, 8

[BB04b] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology - CRYPTO 2004*, pages 443–459, 2004. 8

[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, 8, 10

[BKP14] Olivier Blazy, Eike Kiltz, and Jiaxin Pan. (hierarchical) identity-based encryption from affine message authentication. In *Advances in Cryptology - CRYPTO 2014 Part I*, pages 408–425, 2014. 3, 8

[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, 34

[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. 8

[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, 7, 8, 11, 13, 14, 15, 21

[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, 6, 8

[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. 14

[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. 8

[Gen06] Craig Gentry. Practical identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2006*, pages 445–464, 2006. 8

[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. 8

[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, 9, 10, 11, 13, 25, 26, 51, 52, 54, 55

[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

[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

[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, 34

[LW11] Allison Lewko and Brent Waters. Unbounded hibe and attribute-based encryption. In *Advances in Cryptology–EUROCRYPT 2011*, pages 547–567. Springer, 2011. 8

[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, 34

[OT09] Tatsuaki Okamoto and Katsuyuki Takashima. Hierarchical predicate encryption for inner-products. In *Advances in Cryptology–ASIACRYPT 2009*, pages 214–231. Springer, 2009. 6, 34

[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. 8

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2005*, pages 114–127, 2005. 3, 8

[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, 8

[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 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 6 (External Decision Linear Assumption in G_1) *For any probabilistic polynomial time adversary $\mathcal{A}$, the following advantage function is negligible in k,*

$$\mathsf{Adv}^{\mathrm{EDLIN}}_{\mathcal{A}}(k) := \Big|\Pr[\mathcal{A}(D, T_0) = 1] - \Pr[\mathcal{A}(D, T_1) = 1]\Big|,$$

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

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

A.2 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 7 (Decisional Linear Assumption) *For any probabilistic polynomial time adversary $\mathscr{A}$, the following advantage function is negligible in k,*

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

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

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

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 future 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 $\textsc{sys}$.

– $\mathsf{Param}(1^k, n) \to \textsc{gp}$. Sample $(\textsc{pp}, \textsc{sp}) \leftarrow \mathsf{SampP}(1^k, 2n)$ and output

$$\textsc{gp} := \textsc{pp}.$$

We assume that $\textsc{gp}$ also contains k and n.

– $\mathsf{Setup}(\textsc{gp}) \to (\textsc{mpk}, \textsc{msk})$. Sample $\textsc{msk} \leftarrow \mathbb{H}$ and output

$$\textsc{mpk} := (\textsc{pp}, \mu(\textsc{msk})) \quad \text{and} \quad \textsc{msk}.$$

– $\mathsf{KeyGen}(\textsc{mpk}, \textsc{msk}, \mathbf{y}) \to \textsc{sk}_{\mathbf{y}}$. Let $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$. Sample

$$(h_0, h_1, \ldots, h_{2n}) \leftarrow \mathsf{SampH}(\textsc{pp})$$

and output

$$\textsc{sk}_{\mathbf{y}} := \left(K_0 := h_0, \ K_1 := \textsc{msk} \cdot h_{2-y_1} \cdots h_{2n-y_n}\right).$$

– $\mathsf{Enc}(\textsc{mpk}, \mathbf{x}, \textsc{m}) \to \textsc{ct}_{\mathbf{x}}$. Let $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$ and $\textsc{m} \in \mathbb{G}_T$. Sample random coin s and compute

$$(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\textsc{pp}; s) \ \text{ and } \ g'_T \leftarrow \mathsf{SampGT}(\mu(\textsc{msk}); s).$$

Output

$$\textsc{ct}_{\mathbf{x}} := \left(C_0 := g_0, \ C_1 := g_{2-x_1} \cdots g_{2n-x_n}, \ C_2 := g'_T \cdot \textsc{m}\right).$$

– $\mathsf{Dec}(\textsc{mpk}, \textsc{sk}, \textsc{ct}) \to \textsc{m}$. Let $\textsc{sk} = (K_0, K_1)$ and $\textsc{ct} = (C_0, C_1, C_2)$. Output

$$\textsc{m} := C_2 \cdot \frac{e(C_1, K_0)}{e(C_0, K_1)}.$$

Correctness. For any $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$, one may check that

$$\frac{e(C_1, K_0)}{e(C_0, K_1)} = \frac{e(g_{2-x_1} \cdots g_{2n-x_n}, h_0)}{e(g_0, \textsc{msk} \cdot h_{2-y_1} \cdots h_{2n-y_n})} = (e(g_0, \textsc{msk}))^{-1} = \left(g'_T\right)^{-1},$$

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:

- $\overline{\mathsf{KeyGen}}(\textsc{pp}, \overline{\textsc{msk}}, \mathbf{y}; \mathbf{t})$. Let $\overline{\textsc{msk}} \in \mathbb{H}$, $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$, and $\mathbf{t} = (T_0, T_1, \ldots, T_{2n})$, output

$$\textsc{sk}_{\mathbf{y}} := \left(K_0 := T_0,\ K_1 := \overline{\textsc{msk}} \cdot T_{2-y_1} \cdots T_{2n-y_n}\right).$$

- $\overline{\mathsf{Enc}}(\textsc{pp}, \mathbf{x}, \textsc{m}; \overline{\textsc{msk}}, \mathbf{t})$. Let $\overline{\textsc{msk}} \in \mathbb{H}$, $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$, $\textsc{m} \in \mathbb{G}_T$, and $\mathbf{t} = (T_0, T_1, \ldots, T_{2n})$, output

$$\textsc{ct}_{\mathbf{x}} := \left(C_0 := T_0,\ C_1 := T_{2-x_1} \cdots T_{2n-x_n},\ C_2 := e(T_0, \overline{\textsc{msk}}) \cdot \textsc{m}\right).$$

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

$$\widehat{\mathsf{R}}_i : [\lambda] \times \{0,1\}^i \to [\widehat{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})] \quad \text{and} \quad \widetilde{\mathsf{R}}_i : [\lambda] \times \{0,1\}^i \to [\widetilde{\mathsf{SampH}}^*(\textsc{pp}, \textsc{sp})]$$

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 $(\textsc{pp}, \textsc{sp}) \in [\mathsf{SampP}(1^k, 2n)]$, all $\textsc{msk} \in \mathbb{H}$, all $\iota \in [\lambda]$, all $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$, and all $\textsc{m} \in \mathbb{G}_T$, we define four forms of ciphertext $\mathsf{Enc}(\textsc{mpk}, \mathbf{x}, \textsc{m})$ in the ι-th instance with $\textsc{mpk} := (\textsc{pp}, \mu(\textsc{msk}))$:

(Normal ciphertext.)

$$\overline{\mathsf{Enc}}(\textsc{pp}, \mathbf{x}, \textsc{m}; \textsc{msk}, \mathbf{g}),$$

where $\mathbf{g} \leftarrow \mathsf{SampG}(\textsc{pp})$; more explicitly, the distribution is

$$\left(g_0,\ \prod_{i=1}^{n} g_{2i-x_i},\ e(g_0, \textsc{msk}) \cdot \textsc{m}\right),$$

where $(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\textsc{pp})$. By the *projective* property, the distribution is indeed identical to the output of real encryption algorithm Enc.

(Pseudo-normal ciphertext.)

$$\overline{\mathsf{Enc}}(\textsc{pp}, \mathbf{x}, \textsc{m}; \textsc{msk}, \mathbf{g} \cdot \widehat{\mathbf{g}}),$$

where $\mathbf{g} \leftarrow \mathsf{SampG}(\textsc{pp})$ and $\widehat{\mathbf{g}} \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$; more explicitly, the distribution is

$$\left(g_0 \cdot \widehat{g}_0,\ \prod_{i=1}^{n} \left(g_{2i-x_i} \cdot \widehat{g}_{2i-x_i}\right),\ e(g_0 \cdot \widehat{g}_0, \textsc{msk}) \cdot \textsc{m}\right),$$

where $(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\textsc{pp})$ and $(\widehat{g}_0, \widehat{g}_1, \ldots, \widehat{g}_{2n}) \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$.

(Semi-functional type-$(\wedge, i)$ ciphertexts for $i \in [0, n]$.)

$$\overline{\mathsf{Enc}}(\textsc{pp}, \mathbf{x}, \textsc{m}; \textsc{msk} \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{x}) \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{x}), \mathbf{g} \cdot \widehat{\mathbf{g}}),$$

where $\mathbf{g} \leftarrow \mathsf{SampG}(\textsc{pp})$ and $\widehat{\mathbf{g}} \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$; more explicitly, the distribution is

$$\left(g_0 \cdot \widehat{g}_0, \prod_{i=1}^{n} \left(g_{2i-x_i} \cdot \widehat{g}_{2i-x_i} \right), \ e(g_0 \cdot \widehat{g}_0, \textsc{msk} \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{x})) \cdot \textsc{m} \right),$$

where $(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\textsc{pp})$ and $(\widehat{g}_0, \widehat{g}_1, \ldots, \widehat{g}_{2n}) \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$. We note that $\widetilde{\mathsf{R}}_i(\iota, \mathbf{x})$ vanishes due to the *orthogonality* property.

(Semi-functional type-$(\sim, i)$ ciphertexts for $i \in [0, n]$.)

$$\overline{\mathsf{Enc}}(\textsc{pp}, \mathbf{x}, \textsc{m}; \textsc{msk} \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{x}) \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{x}), \mathbf{g} \cdot \widetilde{\mathbf{g}}),$$

where $\mathbf{g} \leftarrow \mathsf{SampG}(\textsc{pp})$ and $\widetilde{\mathbf{g}} \leftarrow \widetilde{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$; more explicitly, the distribution is

$$\left(g_0 \cdot \widetilde{g}_0, \prod_{i=1}^{n} \left(g_{2i-x_i} \cdot \widetilde{g}_{2i-x_i} \right), \ e(g_0 \cdot \widetilde{g}_0, \textsc{msk} \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{x})) \cdot \textsc{m} \right),$$

where $(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\textsc{pp})$ and $(\widetilde{g}_0, \widetilde{g}_1, \ldots, \widetilde{g}_{2n}) \leftarrow \widetilde{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$. We note that $\widehat{\mathsf{R}}_i(\iota, \mathbf{x})$ vanishes due to the *orthogonality* property.

Finally, for all $(\textsc{pp}, \textsc{sp}) \in [\mathsf{SampP}(1^k, 2n)]$, all $\textsc{msk} \in \mathbb{H}$, all $\iota \in [\lambda]$, and all $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$, we define two forms of secret key $\mathsf{KeyGen}(\textsc{mpk}, \textsc{msk}, \mathbf{y})$ in the ι-th instance with $\textsc{mpk} := (\textsc{pp}, \mu(\textsc{msk}))$:

(Normal secret key.)

$$\overline{\mathsf{KeyGen}}(\textsc{pp}, \textsc{msk}, \mathbf{y}; \mathbf{h}),$$

where $\mathbf{h} \leftarrow \mathsf{SampH}(\textsc{pp})$; more explicitly, the distribution is

$$\left(h_0, \ \textsc{msk} \cdot \prod_{i=1}^{n} h_{2i-y_i} \right),$$

where $(h_0, h_1, \ldots, h_{2n}) \leftarrow \mathsf{SampH}(\textsc{pp})$.

(Semi-functional type-i secret key for $i \in [0, n]$.)

$$\overline{\mathsf{KeyGen}}(\textsc{pp}, \textsc{msk} \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{y}) \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{y}), \mathbf{y}; \mathbf{h}),$$

where $\mathbf{h} \leftarrow \mathsf{SampH}(\textsc{pp})$; more explicitly, the distribution is

$$\left(h_0, \ \textsc{msk} \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{y}) \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{y}) \cdot \prod_{i=1}^{n} h_{2i-y_i} \right),$$

where $(h_0, h_1, \ldots, h_{2n}) \leftarrow \mathsf{SampH}(\textsc{pp})$.

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 i-th 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 i-th bit is 0 are type-$(\wedge, i)$ semi-functional;
  - all challenge ciphertexts for identities whose i-th 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 $\textsc{sys}$. Compared with the construction in Appendix B, the improvement here is that we explicitly put parameters only used to generate secret keys into the master secret key and only parameters for the encryption algorithm are published.

- $\mathsf{Param}(1^k, n) \to \textsc{gp}$. Sample $(\textsc{pp}, \textsc{hp}, \textsc{sp}) \leftarrow \mathsf{SampP}(1^k, 2n)$ and output

$$\textsc{gp} := (\textsc{pp}, \textsc{hp}).$$

  We assume that $\textsc{gp}$ also contains k and n.

- $\mathsf{Setup}(\textsc{gp}) \to (\textsc{mpk}, \textsc{msk})$. Sample $\textsc{msk}_0 \leftarrow \mathbb{H}$ and output

$$\textsc{mpk} := \big(\textsc{pp}, \mu(\textsc{msk}_0)\big) \quad \text{and} \quad \textsc{msk} = \big(\textsc{hp}, \textsc{msk}_0\big).$$

- $\mathsf{KeyGen}(\textsc{mpk}, \textsc{msk}, \mathbf{y}) \to \textsc{sk}_{\mathbf{y}}$. Let $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$. Sample

$$(h_0, h_1, \ldots, h_{2n}) \leftarrow \mathsf{SampH}(\textsc{pp}, \textsc{hp})$$

  and output

$$\textsc{sk}_{\mathbf{y}} := \Big(K_0 := h_0,\ K_1 := \textsc{msk}_0 \cdot h_{2-y_1} \cdots h_{2n-y_n}\Big).$$

- $\mathsf{Enc}(\textsc{mpk}, \mathbf{x}, \textsc{m}) \to \textsc{ct}_{\mathbf{x}}$. Let $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$ and $\textsc{m} \in \mathbb{G}_T$. Sample random coin s and compute

$$(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\textsc{pp}; s) \quad \text{and} \quad g'_T \leftarrow \mathsf{SampGT}(\mu(\textsc{msk}_0); s).$$

  Output

$$\textsc{ct}_{\mathbf{x}} := \Big(C_0 := g_0,\ C_1 := g_{2-x_1} \cdots g_{2n-x_n},\ C_2 := g'_T \cdot \textsc{m}\Big).$$

- $\mathsf{Dec}(\textsc{mpk}, \textsc{sk}, \textsc{ct}) \to \textsc{m}$. Let $\textsc{sk} = (K_0, K_1)$ and $\textsc{ct} = (C_0, C_1, C_2)$. Output

$$\textsc{m} := C_2 \frac{e(C_1, K_0)}{e(C_0, K_1)}.$$

Correctness. For any $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$, one may check that

$$\frac{e(C_1, K_0)}{e(C_0, K_1)} = \frac{e(g_{2-x_1} \cdots g_{2n-x_n}, h_0)}{e(g_0, \textsc{msk}_0 \cdot h_{2-y_1} \cdots h_{2n-y_n})} = \left(e(g_0, \textsc{msk}_0)\right)^{-1} = \left(g_T'\right)^{-1},$$

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*

$$\begin{aligned}\mathsf{Adv}^{\mathrm{IBE}}_{\mathscr{A}}(k, \lambda, q_K, q_C, q_R) \quad \leqslant \quad & \mathsf{Adv}^{\mathrm{LS1}}_{\mathscr{B}_1}(k, q_C q_R, q_K) + 2n \cdot \left(\mathsf{Adv}^{\mathrm{LS2}}_{\mathscr{B}_2}(k, q_C q_R, q_K) + \mathsf{Adv}^{\mathrm{LS3}}_{\mathscr{B}_3}(k, q_C q_R, q_K)\right) \\ & + n \cdot \mathsf{Adv}^{\mathrm{NH}}_{\mathscr{B}_4}(k, q_C q_R, q_K) + \mathsf{Adv}^{\mathrm{ND}}_{\mathscr{B}_5}(k, q_C, q_K, q_R) + 2^{-\Omega(k)},\end{aligned}$$

where $\max_{i \in [5]} \mathsf{Time}(\mathscr{B}_i) \approx \mathsf{Time}(\mathscr{A}) + (\lambda + q_C q_R + q_K) \cdot \mathsf{poly}(k, n)$ *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 $(\textsc{pp}, \textsc{hp}, \textsc{sp}) \in [\mathsf{SampP}(1^k, 2n)]$, all $\textsc{msk}_0 \in \mathbb{H}$, all $\iota \in [\lambda]$, all $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$, and all $\textsc{m} \in \mathbb{G}_T$, we define four forms of ciphertext $\mathsf{Enc}(\textsc{mpk}, \mathbf{x}, \textsc{m})$ in the ι-th instance with $\textsc{mpk} := (\textsc{pp}, \mu(\textsc{msk}_0))$. 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 $i \in [0,n]$.)

$$\overline{\mathsf{Enc}}(\textsc{pp}, \mathbf{x}, \textsc{m}; \textsc{msk}_0 \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{x}) \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{x}), \mathbf{g} \cdot \widehat{\mathbf{g}} \cdot \widetilde{\mathbf{g}}),$$

where $\mathbf{g} \leftarrow \mathsf{SampG}(\textsc{pp})$, $\widehat{\mathbf{g}} \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$ and $\widetilde{\mathbf{g}} \leftarrow \widetilde{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$; more explicitly, the distribution is

$$\left(g_0 \cdot \widehat{g}_0 \cdot \widetilde{g}_0, \ \prod_{i=1}^{n} \left(g_{2i-x_i} \cdot \widehat{g}_{2i-x_i} \cdot \widetilde{g}_{2i-x_i}\right), \ e(g_0 \cdot \widehat{g}_0 \cdot \widetilde{g}_0, \textsc{msk}_0 \cdot \widehat{\mathsf{R}}_i(\iota, \mathbf{x}) \cdot \widetilde{\mathsf{R}}_i(\iota, \mathbf{x})) \cdot \textsc{m} \right),$$

where $(g_0, g_1, \ldots, g_{2n}) \leftarrow \mathsf{SampG}(\textsc{pp})$, $(\widehat{g}_0, \widehat{g}_1, \ldots, \widehat{g}_{2n}) \leftarrow \widehat{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$ and $(\widetilde{g}_0, \widetilde{g}_1, \ldots, \widetilde{g}_{2n}) \leftarrow \widetilde{\mathsf{SampG}}(\textsc{pp}, \textsc{sp})$.

For all $(\textsc{pp}, \textsc{hp}, \textsc{sp}) \in [\mathsf{SampP}(1^k, 2n)]$, all $\textsc{msk}_0 \in \mathbb{H}$, all $\iota \in [\lambda]$, and all $\mathbf{y} = (y_1, \ldots, y_n) \in \{0,1\}^n$, we define two forms of secret key $\mathsf{KeyGen}(\textsc{mpk}, \textsc{msk}, \mathbf{y})$ in the ι-th instance with $\textsc{mpk} := (\textsc{pp}, \mu(\textsc{msk}_0))$, the normal secret key and the semi-functional type-i secret key for $i \in [0,n]$, as in 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 i-th 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 i-th 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 i-th bit is 0 are type-$(\wedge, i)$ semi-functional;
  - all challenge ciphertexts for identities whose i-th 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 i-th 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 i-th 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 $\mathsf{Game}_{2.i+1,0}$ 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.