A General Framework for Lattice-Based ABE Using Evasive Inner-Product Functional Encryption

謝耀慶 (Yao-Ching Hsieh) Huijia Lin 罗辑 (Ji Luo)

Paul G. Allen School of Computer Science & Engineering,
University of Washington, Seattle
{ychsieh,rachel,luoji}@cs.washington.edu

26 May 2024

Abstract

We present a general framework for constructing attribute-based encryption (ABE) schemes for arbitrary function class based on lattices from two ingredients, *i)* a noisy linear secret sharing scheme for the class and *ii)* a new type of inner-product functional encryption (IPFE) scheme, termed *evasive* IPFE, which we introduce in this work. We propose lattice-based evasive IPFE schemes and establish their security under simple conditions based on variants of evasive learning with errors (LWE) assumption recently proposed by Wee [EUROCRYPT '22] and Tsabary [CRYPTO '22].

Our general framework is modular and conceptually simple, reducing the task of constructing ABE to that of constructing noisy linear secret sharing schemes, a more lightweight primitive. The versatility of our framework is demonstrated by three new ABE schemes based on variants of the evasive LWE assumption.

- We obtain two ciphertext-policy ABE schemes for all polynomial-size circuits with a predetermined depth bound. One of these schemes has *succinct* ciphertexts and secret keys, of size polynomial in the depth bound, rather than the circuit size. This eliminates the need for tensor LWE, another new assumption, from the previous state-of-the-art construction by Wee [EUROCRYPT '22].
- We develop ciphertext-policy and key-policy ABE schemes for deterministic finite automata (DFA) and logspace Turing machines (L). They are the first lattice-based public-key ABE schemes supporting uniform models of computation. Previous lattice-based schemes for uniform computation were limited to the secret-key setting or offered only weaker security against bounded collusion.

Lastly, the new primitive of evasive IPFE serves as the lattice-based counterpart of pairing-based IPFE, enabling the application of techniques developed in pairing-based ABE constructions to lattice-based constructions. We believe it is of independent interest and may find other applications.

Contents

Appendix

1 Introduction

Attribute-based encryption (ABE) [SW05,GPSW06] enhances public-key encryption with the capability of enforcing precise access control over encrypted data. ABE primarily comes in two flavors, key-policy (KP-) and ciphertext-policy ABE (CP-ABE). In the setting of CP-ABE, data owners encrypt their private message along with an access policy that dictates who can access and decrypt the data. User keys, on the other hand, contain attributes that describe their credentials. For a user to decrypt a ciphertext and access the encrypted data, their attributes must align with the policy embedded in the ciphertext. Conversely, KP-ABE inverts this structure by using attributes to describe the encrypted data and incorporating access policies into user keys.

Over the past decade, the area of ABE has witnessed substantial progress, thanks to insights on leveraging the mathematical structures present in integer lattices and the learning with errors (LWE) assumption. Notably, Gorbunov, Vaikuntanathan, and Wee [GVW13] achieved a significant milestone by constructing the first KP-ABE scheme for circuits of unbounded size, but with *a priori* bounded depth, relying on the LWE assumption. Subsequent work by Boneh *et al.* [BGG+14] further enhanced KP-ABE efficiency, delivering the first scheme with both short ciphertexts and short secret keys, both scaling polynomially with the depth bound and independent of the circuit size.

However, despite these remarkable advances, several challenges have persisted for over a decade in the lattice-based ABE landscape. They include *i)* the search for ABE schemes accommodating circuits of unbounded depth, *ii)* the construction of CP-ABE schemes supporting all polynomial-size circuits, even confined to predetermined depth bounds, and *iii)* extending ABE to uniform models of computation, such as deterministic finite automata (DFA), logspace Turing machines, and RAM, which have much more succinct descriptions than circuits and can process arbitrarily long inputs. Until recently, the only lattice-based solution addressing challenges *i)* and *iii)* were due to Goldwasser *et al.* [GKP+13], who presented a KP-ABE scheme for RAM. However, it relies on the heavy tools of SNARKs and extractable witness encryption, which can be instantiated using lattice, but require strong knowledge-type assumptions. Beyond the scope of lattice-based constructions, one could attain these objectives using indistinguishability obfuscation ($i\mathcal{O}$) [GGH+13, JLL23].

Recently, the landscape has evolved with the introduction of a novel class of assumptions known as *evasive LWE* [Wee22,Tsa22]. At a high level, evasive LWE captures a "generic-model view" on lattices. When presented with LWE samples $\mathbf{s}^\intercal\mathbf{B}+\mathbf{e}_1^\intercal, \mathbf{s}^\intercal\mathbf{A}+\mathbf{e}_0^\intercal$ and a short (i.e., low-norm) "trapdoor" matrix $\mathbf{K}$ sampled conditioning on $\mathbf{BK}=\mathbf{P}$, denoted as $\mathbf{K}=\mathbf{B}^{-1}(\mathbf{P})$, where all terms are relative to a randomly sampled matrix $\mathbf{B}$ and jointly sampled matrices $\mathbf{A}$ and $\mathbf{P}$, it postulates that if the LWE samples $\mathbf{s}^\intercal\mathbf{P}+\mathbf{e}_2^\intercal, \mathbf{s}^\intercal\mathbf{B}+\mathbf{e}_1^\intercal, \mathbf{s}^\intercal\mathbf{A}+\mathbf{e}_0^\intercal$ are pseudorandom when $\mathbf{e}_2^\intercal$ is freshly generated, then so are $\mathbf{s}^\intercal\mathbf{B}+\mathbf{e}_1^\intercal, \mathbf{s}^\intercal\mathbf{A}+\mathbf{e}_0^\intercal$ given $\mathbf{B}^{-1}(\mathbf{P})$. In simpler terms, adversaries are limited to utilizing the trapdoor to acquire LWE samples $(\mathbf{s}^\intercal\mathbf{P}+\mathbf{e}_2^\intercal)$ and treating them as LWE samples with fresh noises. The work by [Wee22] argued that if this type of generic attacks fail, then all known cryptanalytic techniques, including the family of zeroizing attacks (e.g., [CHL+15,CVW18,HJL21,JLLS23]), would also be ineffective.

The introduction of the evasive LWE assumption has catalyzed new progress on

multiple fronts, including *i)* and *ii)* mentioned above. Wee [Wee22] presented a CP-ABE scheme for bounded-depth circuits under evasive LWE and another new assumption called tensor LWE. More recently, Hsieh, Lin, and Luo [HLL23a] achieved the first KP-ABE scheme for circuits of unbounded depth. Two other works [VWW22, Tsa22] constructed witness encryption schemes, which can be viewed as CP-ABE where the secret key for an attribute is the attribute itself. Furthermore, Water, Wee, and Wu [WWW22] proposed a multi-authority ABE for subset policies without resorting to random oracles.

These accomplishments underscore the potential of evasive LWE. However, each work uses evasive LWE in a way tailored to each construction, relying on a customized variant of evasive LWE following the generic-model principle. This often requires intricate handling of algebraic details. In this work, we ask:

"Can we formulate a general framework for constructing ABE schemes based on evasive LWE?"

Such a framework would provide several advantages, including *i)* deepening our understanding, specifically by elucidating the role of evasive LWE, *ii)* facilitating the development of new ABE constructions, by making them more modular and conceptually simpler, and *iii)* streamlining security proofs, by encapsulating low-level algebraic intricacies within the framework and providing natural and useful tools.

Our Results. To this end, we propose a new notion called *evasive* inner-product functional encryption (IPFE), and present a general framework for constructing CP- and KP-ABE schemes for a class of functions from evasive IPFE and *noisy* linear secret sharing for the same class. Our framework can be seen as the lattice-world counterpart of the general framework for constructing ABE from pairing-based IPFE and (noiseless) linear secret sharing, as described in the prior works of [LL20a, LL20b]. Similar ideas have been employed in various previous pairing-based ABE constructions, which have been realized through different methods, including dual system encryption and hash proof systems.

Evasive Inner-Product Functional Encryption (IPFE). Let's start by introducing evasive IPFE. Similar to a standard IPFE [ABDP15], evasive IPFE allows encrypting a vector $\mathbf{u} \in \mathbb{Z}_q^Z$ into a ciphertext $\mathsf{ict}(\mathbf{u})$, and generating secret keys $\{\mathsf{isk}(\mathbf{v}_j)\}_j$ tied to vectors $\mathbf{v}_j \in \mathbb{Z}_q^Z$, except that decryption reveals *noisy* (as opposed to exact) inner products $\{\mathbf{u}^\top \mathbf{v}_j + e_j\}_j$. The more substantial difference lies in the security requirements. Standard IPFE ensures that only the inner products are revealed and no other information of the encrypted vector $\mathbf{u}$ is leaked (the key vectors $\mathbf{v}_j$ are public). Evasive IPFE tries to capture a "generic-model view" similar to evasive LWE, but at the higher abstraction level of IPFE. It postulates that if the noisy inner products $\{\mathbf{u}^\top \mathbf{v}_j + e_j\}_j$ with "idealized", i.e., freshly generated (hence independent), noises are pseudorandom, then the ciphertext is pseudorandom. More formally,

$$\begin{aligned} \text{if} \quad & \{\mathbf{v}_j,\ \mathbf{u}^\top \mathbf{v}_j + e_j\}_j \approx \{\mathbf{v}_j,\ \$\}_j, \\ \text{then} \quad & \mathsf{impk},\ \mathsf{ict}(\mathbf{u}),\ \{\mathbf{v}_j,\ \mathsf{isk}(\mathbf{v}_j)\}_j \approx \mathsf{impk},\ \$,\ \{\mathbf{v}_j,\ \mathsf{isk}(\mathbf{v}_j)\}_j, \end{aligned}$$

where e_j's are independent Gaussian noises. Evasive IPFE is a convenient tool for generating LWE samples. Consider a simple example where the encrypted vector

is an LWE secret $\mathbf{u} = \mathbf{s}$ and the key vectors correspond to an LWE public matrix $\mathbf{A}$, which may or may not be random. Security ensures that as long as the noisy outputs ($\mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal$) with *fresh* noises are pseudorandom, the ciphertext encrypting $\mathbf{s}$ is pseudorandom.

Jumping ahead, in Section 3, we present candidate lattice-based evasive IPFE schemes, and show their security for restricted distributions of plaintext vectors (observed by all our ABE constructions) based on evasive LWE.

Noisy Linear Secret Sharing Scheme (LSSS). The other ingredient of our general framework is noisy LSSS. A noisy LSSS over $\mathbb{Z}_q$ for a function f (say, of Boolean input and output), denoted as LSSS_f, consists of a non-zero vector $\mathbf{w}_{\mathrm{out}}$ and matrices $\mathbf{T}, \{\mathbf{W}_{\ell,b}\}_{\ell,b}$. Given randomness $\mathbf{s}$, the secret and the shares for an L-bit input $\mathbf{x}$ are created from those matrices.

$$\begin{aligned} &\mathsf{LSSS}_f: \quad \mathbf{w}_{\mathrm{out}}, \mathbf{T}, \{\mathbf{W}_{\ell,b}\}_{\ell\in[L],b\in\{0,1\}}; \\ &\text{Secret:} \quad \mathbf{s}^\intercal\mathbf{w}_{\mathrm{out}} + e_{\mathrm{out}}; \qquad \text{Shares:} \quad \mathbf{s}^\intercal\mathbf{T} + \mathbf{e}_{\mathrm{t}}^\intercal, \{\mathbf{s}^\intercal\mathbf{W}_{\ell,\mathbf{x}[\ell]} + \mathbf{e}_\ell^\intercal\}_{\ell\in[L]}. \end{aligned}$$

If $f(\mathbf{x}) = 1$, the secret $(\mathbf{s}^\intercal\mathbf{w}_{\mathrm{out}} + e_{\mathrm{out}})$ can be approximately recovered from the shares, whereas if $f(\mathbf{x}) = 0$, the secret and the shares must be jointly pseudorandom. In addition, we say that LSSS_f is *short* if its $\mathbf{w}_{\mathrm{out}}, \mathbf{T}, \{\mathbf{W}_{\ell,b}\}_{\ell,b}$ have small norms.

Noisy LSSS is a significantly weaker primitive than ABE in that it only considers a single function f and a single input $\mathbf{x}$, and that the shares for $\mathbf{x}$ can depend on f. In contrast, ABE supports publishing secret keys and ciphertexts for many functions and many inputs, and both must be generated independent of the data on the other side. Interestingly, almost every known LWE-based ABE scheme implicitly defines an LSSS scheme. In the literature, noiseless and information-theoretically secure LSSS are known for low-depth functions [KW93,Ber84,BDHM92,Mul87]. Based on LWE, prior works have constructed a similar primitive, called nearly linear secret sharing scheme, for arbitrary circuits [QWW21,AY20,AWY20,LLL22]. See related works for a comparison.

Our General Framework for Constructing ABE. Our general framework uses evasive IPFE to generically "lift" noisy LSSS into full-fledged ABE schemes. For technical reasons elaborated in Section 1.2, we need to start from a *short* noisy LSSS.

Main Result (Construction 5)**.** *Given a short noisy LSSS for any family of functions over $\mathbb{Z}_q$, KP- and CP-ABE schemes for the same family can be constructed from an evasive IPFE scheme over $\mathbb{Z}_q$.*

When instantiated with our evasive IPFE scheme in Section 3.4, the KP-ABE scheme has

$$|\mathsf{mpk}| = \mathrm{poly}(\log q), \qquad |\mathsf{sk}_f| = |\mathsf{LSSS}_f|\,\mathrm{poly}(\log q), \qquad |\mathsf{ct}_{\mathbf{x}}| = |\mathbf{x}|\,\mathrm{poly}(\log q),$$

and the CP-ABE scheme has

$$|\mathsf{mpk}| = \mathrm{poly}(\log q), \qquad |\mathsf{sk}_{\mathbf{x}}| = |\mathbf{x}|\,\mathrm{poly}(\log q), \qquad |\mathsf{ct}_f| = |\mathsf{LSSS}_f|\,\mathrm{poly}(\log q).$$

Here, $|\mathsf{LSSS}_f|$ is the length of its defining vectors and matrices, and polynomial factors in the security parameter λ are hidden.

Using our general framework, the task of building ABE for an arbitrary function class reduces to the simpler task of constructing noisy LSSS for it. This has led to the following new ABE constructions.

- In Section 5, we design new *short* noisy LSSS for all arithmetic circuits based on LWE. The schemes are adapted from the arithmetic garbling proposed by [AIK11]. Combining them with our evasive IPFE scheme yields a new CP-ABE scheme for bounded-depth bounded-arithmetic circuits from evasive LWE. Compared with Wee's construction [Wee22], our scheme eliminates the tensor LWE assumption, but does not have succinct ciphertext.
- For deterministic finite automata (DFA) and logspace Turing machines (L), noiseless LSSS are known, which have been used in prior works to construct pairing-based ABE for DFA and L [LL20a]. In lattice-based ABE landscape, handling uniform models of computation has been a challenge. The state-of-the-art ABE schemes for DFA only tolerate bounded collusion [AS17,Wee21], are in the secret-key setting [AMY19] (this also supports NFA), or do not come with a security proof [Wee21]. In Section 7, we apply our general framework to those LSSS with simple tweaks for uniform computation, obtaining both KP- and CP-ABE for DFA and L based on evasive LWE, the first provably secure lattice-based public-key ABE schemes tolerating unbounded collusion for these uniform models.

Corollaries 21 and 29 (CP-ABE for circuits and ABE for DFA)**.** *Assuming LWE and evasive LWE, there exists a CP-ABE scheme for arithmetic circuits with*

$$|\mathsf{mpk}| = \mathrm{poly}(d, \log M), \qquad |\mathsf{sk}_{\mathbf{x}}| = |\mathbf{x}|\,\mathrm{poly}(d, \log M), \qquad |\mathsf{ct}_C| = |C|\,\mathrm{poly}(d, \log M),$$

where $|C|$ is the circuit size, d is a bound on the depth of C, and M is a bound on the magnitude of intermediate computation values.

Under the same set of assumptions, there exist ABE schemes for DFA.

$$\begin{aligned} &\text{KP-ABE for DFA:} && |\mathsf{mpk}| = \mathrm{O}(1), && |\mathsf{sk}_{\mathbf{x}}| = \mathrm{O}(|\mathbf{x}|), && |\mathsf{ct}_{\Gamma}| = \mathrm{O}(|\Gamma|);\\ &\text{CP-ABE for DFA:} && |\mathsf{mpk}| = \mathrm{O}(1), && |\mathsf{sk}_{\Gamma}| = \mathrm{O}(|\Gamma|), && |\mathsf{ct}_{\mathbf{x}}| = \mathrm{O}(|\mathbf{x}|). \end{aligned}$$

Here, $|\mathbf{x}|$ is the input length and $|\Gamma|$ is the number of states of the DFA. Each $\mathrm{O}(\cdot)$ hides a polynomial factor in λ.

Upgrading Our General Framework. An unsatisfactory aspect of the above general framework is that it requires *short* noisy LSSS. On the other hand, there are noisy LSSS that are not *short*. For instance, the noisy LSSS underlying the ABE scheme of [BGG+14], as described below.

$$\begin{aligned} &\mathsf{LSSS}_f: \quad \mathbf{w}_{\mathrm{out}} = \mathbf{A}_f\mathbf{G}^{-1}(\mathbf{u}),\ \mathbf{T} = \mathbf{A}_0,\ \{\mathbf{W}_{\ell,b} = \mathbf{A}_\ell - b\mathbf{G}\}_{\ell\in[L],b\in\{0,1\}};\\ &\text{Secret:} \quad \mathbf{s}^\intercal\mathbf{A}_f\mathbf{G}^{-1}(\mathbf{u}); \qquad \text{Shares:} \quad \mathbf{s}^\intercal\mathbf{A}_0 + \mathbf{e}_0^\intercal, \{\mathbf{s}^\intercal(\mathbf{A}_\ell - \mathbf{x}[\ell]\mathbf{G}) + \mathbf{e}_\ell^\intercal\}_{\ell\in[L]}. \end{aligned}$$

The vector $\mathbf{u}$ and matrices $\mathbf{A}_0$, $\mathbf{A}_\ell$'s are random, hence they have large entries. (So are the entries of the gadget matrix $\mathbf{G}$.) We would like to upgrade our general framework to accommodate the BGG-based LSSS. Beyond the consideration of generality, that LSSS is the only known LSSS for circuits of succinct size, in particular, $|\mathsf{LSSS}_f| = \mathrm{poly}(d)$ grows polynomially with the maximum depth of the computation, instead of size. Once combined with our framework, we immediately obtain a CP-ABE scheme supporting bounded-depth circuits with *succinct* ciphertexts.

We achieve this by introducing a stronger variant of evasive IPFE, where the noisy inner products returned from decryption additionally contains structured noises in the form of $e'\mathbf{g}$, where $\mathbf{g}$ is the gadget vector. More precisely, in an evasive IPFE scheme with structured noises, a ciphertext $\mathsf{ict}(\mathbf{u})$ still encrypts a vector $\mathbf{u}$, but a secret key $\mathsf{isk}(\mathbf{V})$ now encodes a matrix $\mathbf{V}$ whose number of columns is the dimension of $\mathbf{g}$. Decryption produces $(\mathbf{u}^\intercal\mathbf{V} + e'\mathbf{g}^\intercal + \mathbf{e}^\intercal)$, where $\mathbf{e}$ is a small noise vector as before and $e'\mathbf{g}$ is the structured noise. Security is enhanced correspondingly. If the noisy outputs with idealized fresh noises $e', \mathbf{e}$ are pseudorandom, so is the ciphertext, i.e.,

$$\begin{aligned} \text{if} \quad & \{\mathbf{V}_j,\ \mathbf{u}^\intercal\mathbf{V}_j + e'_j\mathbf{g}^\intercal + \mathbf{e}_j^\intercal\} \approx \{\mathbf{V}_j,\ \$\}, \\ \text{then} \quad & \mathsf{impk},\ \mathsf{ict}(\mathbf{u}),\ \{\mathbf{V}_j,\ \mathsf{isk}(\mathbf{V}_j)\}_j \approx \mathsf{impk},\ \$,\ \{\mathbf{V}_j,\ \mathsf{isk}(\mathbf{V}_j)\}_j. \end{aligned}$$

Using evasive IPFE with structured noises, we can now "lift" arbitrary noisy LSSS, without any norm restriction, into full-fledged ABE schemes.

Construction 10 and Corollary 34. *Given a (not necessarily short) noisy LSSS for any family of functions over $\mathbb{Z}_q$, KP- and CP-ABE schemes for the same family can be constructed from evasive IPFE with structured noises over $\mathbb{Z}_q$.*

When instantiated with our evasive IPFE with structured noises in Section 3.5, the resultant ABE schemes have the same asymptotic sizes as those in Main Result (Construction 5) — compared with which, it is now possible to use LSSS with smaller sizes. Assuming LWE and a variant of evasive LWE, there exists a CP-ABE scheme for arithmetic circuits with

$$|\mathsf{mpk}| = \mathrm{poly}(d, \log M), \qquad |\mathsf{sk}_{\mathbf{x}}| = |\mathbf{x}|\,\mathrm{poly}(d, \log M), \qquad |\mathsf{ct}_f| = \mathrm{poly}(d, \log M).$$

Constructions of Evasive IPFE. We propose candidate lattice-based evasive IPFE schemes, both with and without structured noises in Section 3, and extensively study them. We show that when the input vectors $\mathbf{u}_i$ have certain form, namely $\mathbf{u}_i^\intercal = \mathbf{s}^\intercal\mathbf{A}_i$ for arbitrary $\mathbf{A}_i$'s sampled using public coins and independent and uniformly random $\mathbf{s}$ (e.g., $\mathbf{u}^\intercal = \mathbf{s}^\intercal = \mathbf{s}^\intercal\mathbf{I}$), then we can base the security of evasive IPFE on variants of evasive LWE. All ABE constructions in this work uses evasive IPFE to encrypt vectors of this form. For evasive IPFE without structured noises, we rely on the usual type of evasive LWE assumption, where all noises involved are small (Assumption 4). For the version with structured noises, we rely on a variant of evasive LWE with structured noises (Assumption 3).

In order to reduce to evasive LWE, the ABE constructions must use a single IPFE instance, instead of many instances with independently sampled master public key. However, we often want to impose some restriction on what inner products are produced, e.g., only computing $\mathbf{u}_1^\intercal\mathbf{v}_1$ and $\mathbf{u}_2^\intercal\mathbf{v}_2$, without revealing $\mathbf{u}_1^\intercal\mathbf{v}_2$ or $\mathbf{u}_2^\intercal\mathbf{v}_1$. To achieve this using a single instance of evasive IPFE, we extend the interface to be identity-based (see Section 3.4), so that a pair of ciphertext $\mathsf{ict}_{\mathsf{id}}(\mathbf{u})$ and secret key $\mathsf{isk}_{\mathsf{id}'}(\mathbf{v})$ only reveals the noisy inner product if $\mathsf{id} = \mathsf{id}'$.

A nice addition is that the study of evasive IPFE already employs several pairing-based techniques. Nevertheless, the constructions and security proofs are intricate and require ironing out many details. We hope our evasive IPFE schemes serve as useful tools in future works.

1.1 Related Works

Lattice-Based Linear Secret Sharing Schemes for Circuits. The works of [AY20, AWY20] constructed CP-ABE with succinct ciphertext for NC^1 relying on LWE and the generic (pairing) group model (GGM). The work of [LLL22] constructed CP-ABE for circuits with constant-size keys (i.e., of size $\mathrm{poly}(\lambda)$) and succinct ciphertext (of size $|\mathbf{x}|\,\mathrm{poly}(\lambda)$, independent of depth). Both works combine a nearly linear secret sharing scheme with a pairing-based IPFE.

Their nearly LSSS is very similar to our noisy LSSS. Both are noisy and if $f(\mathbf{x}) = 1$, evaluation reveals a random secret s perturbed by a small noise e. The major differences are *i)* nearly LSSS requires linear reconstruction and the noise e to be small, which are not required by noisy LSSS, and *ii)* noisy LSSS shares must have the specific form $\mathbf{s}^\intercal\mathbf{T} + \mathbf{e}_t^\intercal, \mathbf{s}^\intercal\mathbf{W}_{\ell,\mathbf{x}[\ell]} + \mathbf{e}_\ell^\intercal$, where the noises are small. These requirements are tailored for combination with pairing-based IPFE and evasive IPFE, respectively. We compare their secret sharing scheme and techniques with ours.

Based on the laconic function evaluation (LFE) protocol of [QWW18], which in turn is based on the circuit KP-ABE scheme of [BGG+14], the works of [AY20,AWY20] proposed a nearly linear secret sharing scheme for NC^1 and that of [LLL22] one for circuits. They then "lift" their nearly LSSS schemes to ABE schemes using pairing-based IPFE — their ABE schemes use pairing-based IPFE to compute rerandomized nearly LSSS shares. As a result, the computed shares reside in the exponent of the pairing group. By the fact that reconstruction is linear, the perturbed secret $(s + e)$ can be recovered in the exponent if $f(\mathbf{x}) = 1$. However, to recover the secret s in the clear, the schemes rely on exhaustive search of the noise e. Therefore, e must be polynomially bounded, which is particularly difficult to achieve when evaluating high-depth circuits [LLL22]. The fact that e must be small also makes the security proof more challenging, as one cannot rely on noise smudging.

Differently, in our general framework and hence CP-ABE schemes, the secret shares are computed in the clear and the evaluation noise e just need to be sufficiently small compared to the modulus, and evaluation can be non-linear. However, we require randomization of the shares in the form of $\mathbf{s}^\intercal\mathbf{T} + \mathbf{e}_t^\intercal, \{\mathbf{s}^\intercal\mathbf{W}_{\ell,\mathbf{x}[\ell]} + \mathbf{e}_\ell^\intercal\}_{\ell\in[L]}$ with small noises, in order to match the precondition of evasive LWE, where these small noises are "idealized" to be fresh.

A technical relation is that their ABE schemes rely on GGM or knowledge-type assumption (the Knowledge of OrthogonALity Assumption, or "KOALA") on the pairing group to argue that correlated instances of secret sharing for different computation $\{(f_i, \mathbf{x}_j)\}_{i,j}$ where $f_i(\mathbf{x}_j) = 0$ for all i, j (resulting from decrypting all pairs of secret keys and ciphertexts) are jointly secure. In our ABE scheme, we rely on evasive IPFE (and by reduction, evasive LWE) to assert the security of the many correlated instances of secret sharing. In folklore, evasive LWE is regarded as a lattice counterpart of GGM, although there has been no formal known connections between them.

The works of [AY20,AWY20,LLL22] *combine* pairing techniques with lattice ones, relying on pairing-based IPFE as well as LWE. We *translate* pairing techniques to the lattice world, creating a lattice implementation of IPFE, suitable for ABE when combined with LWE. Evasive IPFE is reminiscent to the very strong simulation security of IPFE in GGM proven in [LLL22], which can be seen as a step towards understanding the connection between GGM and evasive LWE.

Noisy IPFE. The work of [Agr19], followed by [AP20], proposed a primitive called noisy IPFE, which evaluates noisy inner products, i.e., $(\mathbf{u}^\intercal \mathbf{v} + e)$. It is defined with strong security property. For any two plaintext vectors $\mathbf{u}, \mathbf{u}'$ and key vectors $\mathbf{v}_1, \ldots, \mathbf{v}_J$ such that their inner products are approximately equal, i.e., $(\mathbf{u} - \mathbf{u}')^\intercal \mathbf{v}_j$ is small for all j, the ciphertext encrypting $\mathbf{u}$ and that encrypting $\mathbf{u}'$ must be indistinguishable. The rationale is that the noises resulting from decryption is sufficient to "smudge" the difference between the inner products. This primitive is strong enough to imply indistinguishability obfuscation when combined with other well-studied assumptions.

Our evasive IPFE has a different security requirement. It states that if the inner products plus fresh noises are pseudorandom, i.e., $\{\mathbf{u}_i^\intercal \mathbf{v}_j + e_{ij}\} \approx \{\$\}$, then the ciphertexts encrypting $\mathbf{u}_i$'s are pseudorandom given the keys. We achieve this under evasive LWE for a general class of distributions of inputs.

ABE for DFA from Lattices. The work of [Wee21] proposed a candidate public-key KP-ABE for DFA based on lattices. In terms of methods, it relies on the DFA garbling implicit in [Wat12] (made explicit in [LL20a]), which we also use for our scheme. While many known ABE for DFA [Wat12,GWW19,LL20a,GW20,Wee21] (and this work) are roughly based on the idea of *securely* computing a pseudorandom garbling for DFA, to our knowledge, only this work and [LL20a] explicitly use *functional encryption* (concretely, IPFE) to do so. In terms of results, in [Wee21], there was no reductionist proof of security but some preliminary cryptanalysis, in which an idea in the line of evasive LWE is used (replacing a trapdoor preimage by the intended usage result, with fresh noises). The development of evasive LWE is later to [Wee21]. While we do not immediately see a proof from the version of evasive LWE in our work or [Wee22, Tsa22], it is plausible that another variant of evasive LWE could be formulated to prove its security (or a slightly modified variant thereof).

1.2 Technical Overview

We give an overview of our techniques. Our starting point is a well-established method of constructing ABE using linear secret sharing schemes (LSSS) and pairing.

Recap of ABE from LSSS and Pairing-Based IPFE. A traditional LSSS is noiseless and information-theoretic. An LSSS for an arithmetic function $f : \mathbb{Z}^L \to \{0,1\}$ is specified by vector $\mathbf{w}_{\mathsf{out}} \neq \mathbf{0}$ and matrices $\mathbf{T}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell \in [L]}$ over $\mathbb{Z}_q$, each of which has n_f rows. Given randomness $\mathbf{s} \in \mathbb{Z}_q^{n_f}$, the secret is $\mathbf{s}^\intercal \mathbf{w}_{\mathsf{out}}$ and the shares are

$$\mathbf{s}^\intercal \mathbf{T} \quad \text{and} \quad \{\mathbf{s}^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times})\}_{\ell \in [L]}.$$

For correctness, if $f(\mathbf{x}) = 1$, the shares $\mathbf{s}^\intercal \mathbf{T}, \{\mathbf{s}^\intercal \mathbf{W}_{\ell,\mathbf{x}[\ell]}\}_{\ell \in [L]}$ must determine the secret $\mathbf{s}^\intercal \mathbf{w}_{\mathsf{out}}$, where $\mathbf{W}_{\ell,\mathbf{x}[\ell]} = \mathbf{W}_{\ell,0} + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times}$. For security, if $f(\mathbf{x}) = 0$ and $\mathbf{s}$ is uniformly random, the shares must be independent of $\mathbf{s}^\intercal \mathbf{w}_{\mathsf{out}}$.

To obtain (KP- or CP-) ABE from LSSS, pairing is used so that when decrypting a ciphertext using a key, tied to $f, \mathbf{x}$, it will first compute the LSSS_f shares of $\mathbf{x}$ encoded in the exponent of the target group, which can then be used to recover the secret (again, in the exponent) if $f(\mathbf{x}) = 1$. For the construction to be secure, the computed shares should use good randomness, which must be kept hidden. The mechanism of computing the shares vary across known constructions, and oftentimes it can be

regarded as inner-product functional encryption (IPFE) [ABDP15]. Since the shares are computed in the exponent, pseudorandomness can often be obtained from DDH-style rerandomization, e.g., in CP-ABE, keys contain random r, ciphertexts contain $\mathbf{s}^\top\mathbf{T}, \{\mathbf{s}^\top\mathbf{W}_{\ell,\mathbf{x}[\ell]}\}_{\ell\in[L]}$ with random $\mathbf{s}$, and the computed shares are $r\mathbf{s}^\top\mathbf{T}, \{r\mathbf{s}^\top\mathbf{W}_{\ell,\mathbf{x}[\ell]}\}_{\ell\in[L]}$ in the exponent of the target group. Intuitively, by the decisional Diffie–Hellman assumption, the computed shares, using $r\mathbf{s}$ as its randomness, is indistinguishable from being created with fresh randomness. This intuition has been implemented in multiple ways, e.g., dual system encryption, hash proof systems, or using slotted IPFE.

ABE from LSSS and Lattices. Pairing-based ABE only supports low-depth computations, because the expressive power of polynomial-size LSSS is limited in NC [KW93, Ber84,BDHM92,Mul87]. To circumvent this lower bound, we must relax the notion of LSSS to support richer classes of functions. We consider LSSS with noises, termed *noisy linear secret sharing* in this work. Such a scheme for f is again specified by $\mathbf{w}_{\text{out}}, \mathbf{T}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}$, but the secret and the shares now contain noises.

$$\begin{aligned} &\text{Secret:} && \mathbf{s}^\top\mathbf{w}_{\text{out}} + e_{\text{out}}; \\ &\text{Shares:} && \mathbf{s}^\top\mathbf{T} + \mathbf{e}_{\text{t}}^\top \quad \text{and} \quad \{\mathbf{s}^\top(\mathbf{W}_{\ell,0} + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_\ell^\top\}_{\ell\in[L]}. \end{aligned}$$

We additionally allow the secret sharing to have a reusable part $\mathbf{R}$ (always given and not randomized). If $f(\mathbf{x}) = 1$, correctness only guarantees recovering the secret approximately. For security, if $f(\mathbf{x}) = 0$, the secret and the shares should be jointly pseudorandom even given $\mathbf{R}, \mathbf{w}_{\text{out}}, \mathbf{T}, \mathbf{W}$'s. Following the previous pairing-based general framework, we now need an appropriate IPFE that can compute and rerandomize the noisy secret shares.

Insufficiency of Existing IPFE. When using pairing-based IPFE, since the shares are computed in the exponent, one can only reconstruct the secret plus noise in the exponent. To be able to recover the secret in the clear, the noise must be polynomially small to allow for exhaustive search. However, keeping the noise small is difficult when evaluating arbitrary depth circuits.

To get around this limitation, we ask whether lattice-based IPFE schemes such as [ABDP15,ALS16] can help. They yield decryption results in the clear, dispensing the limitation on the magnitude of noise. However, it is unclear how to use these scheme to generate the noises needed in each rerandomized secret shares. In fact, Agrawal [Agr19] together with Pellet-Mary [AP20] showed that a noisy IPFE scheme capable of computing inner products added with (computationally) good Gaussian noises implies $i\mathcal{O}$ when combined with other standard assumption.

Evasive IPFE. We solve the aforementioned challenges with a new primitive called *evasive IPFE*. For convenience, we directly define the identity-based variant. Each key is for a vector $\mathbf{v}$ under some identity id', and each ciphertext, $\mathbf{u}$ and id. Decryption yields an approximation of $\mathbf{u}^\top\mathbf{v}$ if and only if $\mathsf{id} = \mathsf{id}'$.

$$\begin{bmatrix} \mathsf{isk}_{\mathsf{id}'}(\mathbf{v}) \\ \mathsf{ict}_{\mathsf{id}}(\mathbf{u}^\top) \end{bmatrix} \xrightarrow{\mathsf{Dec}} \begin{cases} \mathbf{u}^\top\mathbf{v} + e, & \text{if } \mathsf{id} = \mathsf{id}'; \\ \bot, & \text{if } \mathsf{id} \neq \mathsf{id}'. \end{cases}$$

We also denote batches of keys and ciphertexts as $\mathsf{isk}_{\mathsf{id}}(\mathbf{V})$ and $\mathsf{ict}_{\mathsf{id}}(\mathbf{U}^\top)$ so that their pairwise decryptions approximate $\mathbf{U}^\top\mathbf{V}$.

For security, given arbitrarily many keys, the ciphertexts hide $\mathbf{u}$'s if all the inner products that can be computed are jointly pseudorandom, when they are added with fresh Gaussian noises. More precisely, given secret keys generated for $\mathbf{v}_j$ under identity id'_j and ciphertexts for $\mathbf{u}_i$ under id_i, the security property stipulates that

$$\begin{aligned} \text{if} \quad & \mathsf{aux}, \{\mathbf{u}_i^\intercal \mathbf{v}_j + e_{i,j}\}_{\mathsf{id}_i = \mathsf{id}'_j} \approx \mathsf{aux}, \{\$\}_{\mathsf{id}_i = \mathsf{id}'_j}, \\ \text{then} \quad & \mathsf{aux}, \{\mathsf{isk}_{\mathsf{id}'_j}(\mathbf{v}_j)\}_j, \{\mathsf{ict}_{\mathsf{id}_i}(\mathbf{u}_i)\}_i \approx \mathsf{aux}, \{\mathsf{isk}_{\mathsf{id}'_j}(\mathbf{v}_j)\}_j, \{\mathsf{ict}_{\mathsf{id}_i}(\$)\}_i. \end{aligned}$$

Here, $e_{i,j}$'s are fresh (Gaussian) noises, and aux is auxiliary information about $\mathbf{v}, \mathbf{u}$'s — think of it as the public coins for sampling $\mathbf{v}$'s.

ABE from Noisy Linear Secret Sharing and Evasive IPFE. Given evasive IPFE and noisy LSSS, our general framework for constructing KP- and CP-ABE is modular and simple. For example, to construct CP-ABE, each ciphertext for a function f consists of a series of IPFE ciphertexts encoding $\mathbf{S}^\intercal \mathbf{W}$'s, where $\mathbf{S}$ is an LWE secret matrix, and each key consists of IPFE keys encoding LWE public vector $\mathbf{r}$ and input $\mathbf{x}$. Their decryption produces rerandomized secret shares of the form $(\mathbf{r}^\intercal \mathbf{S}^\intercal \mathbf{W} + \mathbf{e}^\intercal)$. Below is an example.

$$\begin{array}{rlll} \mathsf{sk}_{\mathbf{x}}: & \mathsf{isk}_0(\ \lfloor q/2 \rceil, & \mathbf{r}\ ), & \{\mathsf{isk}_\ell(\ \mathbf{r},\ \mathbf{x}[\ell]\mathbf{r}\ )\}_{\ell \in [L]}; \\ \mathsf{ct}_f: & \mathsf{ict}_0(\ \mathbf{0}, & \mathbf{T}^\intercal \mathbf{S}\ ), & \{\mathsf{ict}_\ell(\ \mathbf{W}_{\ell,0}^\intercal \mathbf{S}, \mathbf{W}_{\ell,\times}^\intercal \mathbf{S}\ )\}_{\ell \in [L]}, \\ & \mathsf{ict}_0(\ \mu, & \mathbf{w}_{\mathrm{out}}^\intercal \mathbf{S}\ ). & \end{array}$$

The decryption yields a noisy version of $\mathbf{W}^\intercal \mathbf{S}\mathbf{r}$'s as desired.

The security proof of ABE involves invoking the security of evasive IPFE to argue that all IPFE ciphertexts are hiding, which only applies if all noisy inner products are pseudorandom. Recall that in the ABE security game, there are multiple keys, and the inner products are

$$\begin{aligned} \text{for } \mathbf{x}_j: \quad & (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{T} + \mathbf{e}_{\mathrm{t},j}^\intercal, \quad \{(\mathbf{S}\mathbf{r}_j)^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_{j,\ell}^\intercal\}_{\ell \in [L]}, \\ & (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{w}_{\mathrm{out}} + e_{\mathrm{out},j} + \mu \cdot \lfloor q/2 \rceil. \end{aligned}$$

Those are exactly the shares and secrets generated using randomness $\mathbf{S}\mathbf{r}_j$, but the components as-is are not pseudorandom, because $\mathbf{r}_j$'s are public, and the same secret matrix $\mathbf{S}$ and matrices $\mathbf{T}$, $\mathbf{W}$'s are reused in all shares for different $\mathbf{x}_j$'s. Hopefully, if we could somehow change the randomness from $\mathbf{S}\mathbf{r}_j$ to $(\mathbf{S}\mathbf{r}_j + \mathbf{e}_{\mathrm{r},j}^\intercal)$ with independent Gaussian $\mathbf{e}_{\mathrm{r},j}$'s, then by LWE (for secret $\mathbf{S}$), the rerandomization generates pseudorandomness, i.e., $\mathbf{S}\mathbf{r}_j + \mathbf{e}_{\mathrm{r},j}^\intercal \approx \$$, and then so are the shares. We show that how to achieve this in two ways.

Short Secret Sharing. If we require that the secret sharing be short, i.e., the matrices $\mathbf{w}_{\mathrm{out}}$, $\mathbf{T}$, $\mathbf{W}_{\ell,\mathbf{x}[\ell]}$'s have sufficiently small norm, then thanks to flooding by $\mathbf{e}_{\mathrm{t},j}$'s,

$$\{(\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{T} + \mathbf{e}_{\mathrm{t},j}^\intercal\}_{j \in [J]} \approx_{\mathrm{s}} \{(\mathbf{S}\mathbf{r}_j + \mathbf{e}'_{\mathrm{t},j})^\intercal \mathbf{T} + \mathbf{e}_{\mathrm{t},j}^\intercal\}_{j \in [J]}.$$

The same holds for all the shares and secrets, achieving rerandomization.

Evasive IPFE with Structured Noises. Alternatively, we can augment the notion of evasive IPFE to include structured noises to work with arbitrary noisy LSSS. Instead of having only small noises, the noisy inner products now look like $(\mathbf{u}^\intercal \mathbf{V} + \mathbf{e}^\intercal \mathbf{G} + (\mathbf{e}')^\intercal)$,

where $\mathbf{V}$ is associated with the secret key. Correspondingly, security holds if the noisy inner products with fresh noises $\mathbf{e}'$ and $\mathbf{e}$ are pseudorandom.

To use such evasive IPFE with structured noises to rerandomize secret shares, we need an additional trick. The problem is that the structured noises are of the form $\mathbf{e}^\intercal\mathbf{G}$ instead of $\mathbf{e}^\intercal\mathbf{W}_{\ell,\mathbf{x}_j[\ell]}$, and the $\mathbf{e}$ here is different for each decryption (e.g., for computing the shares corresponding to $\mathbf{x}_j[1]$ and $\mathbf{x}_j[2]$ for the *same* j). To get consistent $\mathbf{e}_j^\intercal\mathbf{W}_{\ell,\mathbf{x}[\ell]}$ for each $\mathbf{x}_j$ (key) independent of ℓ (input bit index), we perform a noisy secret sharing

$$\underbrace{(\mathbf{r}_j^\intercal\mathbf{S}+\mathbf{e}_j^\intercal)\mathbf{W}_{\ell,\mathbf{x}_j[\ell]}}=\underbrace{\mathbf{r}_j^\intercal\mathbf{S}\mathbf{W}_{\ell,\mathbf{x}_j[\ell]}-(\mathbf{r}'_j)^\intercal\mathbf{Q}\mathbf{G}^{-1}(\mathbf{W}_{\ell,\mathbf{x}_j[\ell]})}+(\underbrace{(\mathbf{r}'_j)^\intercal\mathbf{Q}+\mathbf{e}_j^\intercal\mathbf{G}}\mathbf{G}^{-1}(\mathbf{W}_{\ell,\mathbf{x}_j[\ell]}),$$

where the wavy underlines indicate small noises (without $\mathbf{G}$ structure) and each noisy term on the right-hand side can be computed using evasive IPFE, without and with structured noises respectively. Importantly, each ABE key only uses one structured noise $\mathbf{e}_j$, and the shares for the L input bits use the same $\mathbf{e}_j$ in $\mathbf{e}_j^\intercal\mathbf{W}_{\ell,\mathbf{x}_j[\ell]}$.

Short Noisy Linear Secret Sharing for Polynomial-Size Circuits. Our construction of short noisy linear secret sharing scheme for circuits is inspired by the arithmetic garbling scheme of [AIK11] and deals with the circuit gate by gate. In this paragraph, we will use "labels" for "shares" as it is customary for circuits (noisy linear secret sharing is a partially hiding garbling).

To begin, each gate i is associated with two random low-norm matrices $\mathbf{W}_{i,0},\mathbf{W}_{i,\times}$, which we call the label function of gate i. The label (before randomization) encoding some value x is $\mathbf{W}_{i,x}=\mathbf{W}_{i,0}+x\mathbf{W}_{i,\times}$, and its randomized version is $(\mathbf{s}^\intercal\mathbf{W}_{i,x}+\mathbf{e}_{i,x}^\intercal)$. Suppose gate i has its input wires connected to gates i_1,i_2 and their output values are x,x_1,x_2, we can sample a random low-norm matrix $\mathbf{U}_i$ and rewrite

$$\begin{aligned}\mathbf{W}_{i,0}+(x_1+x_2)\mathbf{W}_{i,\times}&=(\mathbf{U}_i+x_1\mathbf{W}_{i,\times})+(\mathbf{W}_{i,0}-\mathbf{U}_i+x_2\mathbf{W}_{i,\times}),\\ \mathbf{W}_{i,0}+x_1x_2\mathbf{W}_{i,\times}&=x_2(\mathbf{U}_i+x_1\mathbf{W}_{i,\times})+(\mathbf{W}_{i,0}-x_2\mathbf{U}_i),\end{aligned}$$

depending on whether gate i is an addition or multiplication gate. We denote the decomposed terms that are affine in x_1 as $\widetilde{\mathbf{W}}_{i_1,x_1}$, and those affine in x_2 as $\widetilde{\mathbf{W}}_{i_2,x_2}$ — note that $\widetilde{\mathbf{W}}_{i,x}$ is contributed by all the fan-outs of gate i. The randomized, noisy $\mathbf{s}^\intercal\mathbf{W}_{i,x}$ can be obtained from the noisy versions of $\mathbf{s}^\intercal\widetilde{\mathbf{W}}_{i_1,x_1}$ and $\mathbf{s}^\intercal\widetilde{\mathbf{W}}_{i_2,x_2}$. We call $\widetilde{\mathbf{W}}$'s and $\mathbf{s}^\intercal\widetilde{\mathbf{W}}$'s the expanded labels, and consequently, $\mathbf{W}$'s and $\mathbf{s}^\intercal\mathbf{W}$'s the shrunken labels.

For each gate i, we also sample and publish a low-norm matrix $\mathbf{R}_i$, and publish in $\mathbf{s}^\intercal\mathbf{T}$ (the "garbled table entries" for gate i)

$$\mathbf{s}^\intercal(\mathbf{W}_{i,0}\mathbf{R}_i+\widetilde{\mathbf{W}}_{i,0})+(\mathbf{e}'_{i,0})^\intercal,\qquad \mathbf{s}^\intercal(\mathbf{W}_{i,\times}\mathbf{R}_i+\widetilde{\mathbf{W}}_{i,\times})+(\mathbf{e}'_{i,\times})^\intercal.$$

The values of $\mathbf{R}_i$'s and noisy $\mathbf{s}^\intercal\mathbf{T}$ are given regardless of the input values. Note that these values enable computing the expanded label $\mathbf{s}^\intercal\widetilde{\mathbf{W}}_{i,x}$ from the shrunken label $\mathbf{s}^\intercal\mathbf{W}_{i,x}$ for any value x that gate i happens to take. We also include $\mathbf{s}^\intercal(\mathbf{W}_{|C|,0}\mathbf{R}_{|C|}+\mathbf{w}_{\text{out}})$ to facilitate the recovery of secret when $C(\mathbf{x})=0$. The shares consist of the input labels as well as the garbled table entries for all gates.

Evaluation also proceeds gate by gate, starting from the input gates. For each gate i with value x, the evaluation procedure recovers the shrunken label, then use $\mathbf{R}_i$ and the garbled table to recover the expanded label, which is later used to recover the shrunken labels for gates taking gate i as input. Eventually, if the output is 0, we approximately recover $\mathbf{s}^\intercal\mathbf{w}_{\text{out}}$ from $\mathbf{s}^\intercal\mathbf{W}_{|C|,0}$ and $\mathbf{s}^\intercal(\mathbf{W}_{|C|,0}\mathbf{R}_{|C|}+\mathbf{w}_{\text{out}})$ and $\mathbf{R}_{|C|}$.

We sketch the security proof. In the first step, we replace all the noisy $\mathbf{s}^\intercal \mathbf{W}_{i,0}$'s and $\mathbf{s}^\intercal \mathbf{W}_{i,\times}$'s by random $\boldsymbol{\delta}_{i,0}^\intercal$ and $\boldsymbol{\delta}_{i,\times}^\intercal$, using the flipped LWE assumption. Then, for a particular input $\mathbf{x}$, we simulate $\boldsymbol{\delta}_{i,0}$ and $\boldsymbol{\delta}_{i,\times}$ using $\boldsymbol{\delta}_{i,x_i}$ (call this "active") and $\boldsymbol{\delta}_{i,\times}$, where x_i is the value of gate i when evaluated on input $\mathbf{x}$. The key observation here is that in such hybrids, the active expanded labels and garbled table entries no longer depend on the "×" ones. To argue the input labels and the garbled table are jointly pseudorandom, we deal with active labels/table and "×" ones in two passes. In the first pass, we work on the "×" labels/table bottom-up (output gate to input gate). This step is computational and involves flipped LWE for public matrix $\mathbf{R}$ — for each "×" table $(\boldsymbol{\delta}_{i,\times}^\intercal \mathbf{R}_i + \widetilde{\boldsymbol{\delta}}_{i,\times}^\intercal)$, the secret $\boldsymbol{\delta}_{i,\times}^\intercal$ will be absent everywhere else when the proof comes to this table, so flipped LWE ensures that the first term is pseudorandom, hiding $\widetilde{\boldsymbol{\delta}}_{i,\times}$, thus removing information about the "×" labels of its fan-outs. In the second pass, we work on the active labels/table top-down. This step is statistical using the information-theoretic garbling security (so technically security is already in place when we finish the first pass), and the top-down order is just a pedagogy device for understanding the proof.

Succinct CP-ABE for Circuits. A (non-short) noisy linear secret sharing scheme can be distilled from the celebrated KP-ABE for circuits due to [BGG+14], as observed by [AWY20]. The scheme is succinct — the sizes of $\mathbf{R}$, $\mathbf{T}$, $\mathbf{W}$'s only depend on the depth, not the size of the circuit. Combined with evasive IPFE with structured noises, we immediately obtain succinct CP-ABE.

Secret Sharing and ABE for DFA. The existing linear secret sharing scheme for DFA [Wat12,LL20a] can be cast as a noisy linear secret sharing and is short. The definition of noisy linear secret sharing inherently considers fixed input length, while in (say, CP-) ABE for uniform computation like DFA, we would like a ciphertext (tied to DFA) to potentially authorize a key (tied to input) of arbitrary length, and conversely a key to be accepted by a ciphertext of arbitrarily many states. In the previous scheme, ct_f determines the input length it accepts, so the same construction cannot be directly used.

We follow the paradigm in [Wat12,LL20a] and exploit the locality of DFA computation. Namely, each DFA step is simply reading an input bit from a fixed location and transitioning the previous state to the next. At a very high level, the most important shares (related to state transitions) are of the form $(\mathbf{s}_\ell^\intercal \boldsymbol{\Gamma}_{\mathbf{x}[\ell]} - \mathbf{s}_{\ell-1}^\intercal)$, where $\boldsymbol{\Gamma}_0, \boldsymbol{\Gamma}_1$ are determined by the DFA state transition function, and $\mathbf{s}_0, \ldots, \mathbf{s}_L$, each of dimension $\mathfrak{Q}$ (number of states of DFA), together constitutes the secret sharing randomness $\mathbf{s}$ of length $L\mathfrak{Q}$. The locality of shares in $\mathbf{s}$ is inherited from the locality of DFA computation, and the subtraction expresses the state transition. Evaluation is a telescoping sum that chains the state transitions.

In (again, say CP-) ABE, we set $\mathbf{s}$ to be LWE samples jointly generated by the keys (size dependent on L) and the ciphertexts (size dependent on $\mathfrak{Q}$), $\mathbf{s}_\ell = \underset{\sim}{\mathbf{S}}\underset{\sim}{\mathbf{r}}_\ell$ for $\mathbf{r}_\ell$'s in each key and $\mathbf{S}$ in each ciphertext. Continuing with the high level form of the most important shares, we set

$$\begin{aligned} \mathsf{sk}_{\mathbf{x}} &: \quad \{\mathsf{isk}(\ (1-\mathbf{x}[\ell])\mathbf{r}_\ell,\ \mathbf{x}[\ell]\mathbf{r}_\ell,\ \mathbf{r}_{\ell-1}\ )\}_{\ell\in[L]}, \\ \mathsf{ct}_{\mathrm{DFA}} &: \quad \mathsf{ict}(\quad \boldsymbol{\Gamma}_0^\intercal \mathbf{S},\quad \boldsymbol{\Gamma}_1^\intercal \mathbf{S},\quad -\mathbf{S}\ ). \end{aligned}$$

Upon decryption, they compute all the shares, which reveal the secret approximately if the computation is accepting. For security, we first argue that $\underset{\sim}{\mathbf{S}}\underset{\sim}{\mathbf{r}}$'s are pseudoran-

dom, hence the shares are indistinguishable to created using fresh randomness, then we invoke the (information-theoretic) security of DFA secret sharing, and conclude ABE security from evasive IPFE security and pseudorandomness of the shares.

Instantiation of Evasive IPFE. Our candidate evasive IPFE schemes draw inspiration heavily from known IPFE constructions, and we proceed in three steps.

Basic Scheme. The basic scheme without identities nor structured noises is reminiscent to the schemes due to [ABDP15,ALS16]. The master public key consists of two matrices $\mathbf{B}, \mathbf{A}$, the master secret key is a trapdoor of $\mathbf{B}$, and

$$\mathsf{isk}(\mathbf{v}) = \mathbf{B}^{-1}(\mathbf{A}\mathbf{G}^{-1}(\mathbf{v})), \qquad \mathsf{ict}(\mathbf{u}) = (\underbrace{\mathbf{d}^\intercal\mathbf{B}}, \underbrace{\mathbf{d}^\intercal\mathbf{A} + \mathbf{u}^\intercal\mathbf{G}}) = (\mathbf{c}_1^\intercal, \mathbf{c}_2^\intercal),$$

where $\mathbf{B}^{-1}(\mathbf{p})$ is a low-norm vector $\mathbf{k}$ satisfying $\mathbf{B}\mathbf{k} = \mathbf{p}$, which can be sampled using the master secret key. For correctness, observe that $\mathbf{c}_2^\intercal\mathbf{G}^{-1}(\mathbf{v}) - \mathbf{c}_1^\intercal\mathbf{k}$ approximates $\mathbf{u}^\intercal\mathbf{v}$. We study the security of this simple scheme extensively in Section 3. Most importantly, its security can be reduced to a variant of evasive LWE if the plaintext vectors $\mathbf{u}$ are uniformly random over a publicly known subspace.

Identity-Based Scheme. We borrow techniques from pairing to generically achieve identity-based schemes. In pairing-based IPFE, identity binding can be achieved using a single-identity scheme (denoted with primes) by adding a blinding factor in the decryption result. Let φ, ψ be random, then

$$\begin{aligned} \mathsf{isk}_{\mathsf{id}'}(\mathbf{v}): &\quad \mathsf{isk}'(\ \mathbf{v}, \quad \psi, \mathsf{id}' \cdot \psi\ ), \\ \mathsf{ict}_{\mathsf{id}}(\mathbf{u}): &\quad \mathsf{ict}'(\ \mathbf{u}, \mathsf{id} \cdot \varphi, \quad -\varphi\ ). \end{aligned}$$

The decryption outcome is $(\mathbf{u}^\intercal\mathbf{v} + (\mathsf{id} - \mathsf{id}')\varphi\psi)$. When the identities match, the result is the desired. When they do not match, the result is blinded by $\varphi\psi$, which is pseudorandom in the exponent (e.g., by DDH). The similar idea can be applied to evasive IPFE, except the blinding factor is $\boldsymbol{\varphi}^\intercal\boldsymbol{\psi}$, which is pseudorandom by LWE. To see security, observe that non-matching identities, due to $\underbrace{\boldsymbol{\varphi}^\intercal\boldsymbol{\psi}}$, yield pseudorandom inner products concerned by the underlying scheme, whereas matching inner products are already pseudorandom by premise.

Scheme with Structured Noises. The structured noise $\mathbf{g}^\intercal \otimes \mathbf{e}^\intercal$ is simply $\mathbf{e}$ multiplied by powers of two. To obtain them, we change the format of $\underbrace{\mathbf{d}^\intercal\mathbf{B}}$ in ict into

$$\mathbf{d}^\intercal\mathbf{B}_0 + 2^0\mathbf{e}^\intercal + (\mathbf{e}'_0)^\intercal, \ \ldots, \ \mathbf{d}^\intercal\mathbf{B}_{K-1} + 2^{K-1}\mathbf{e}^\intercal + (\mathbf{e}'_{K-1})^\intercal,$$

where $\mathbf{e}, \mathbf{e}'_0, \ldots, \mathbf{e}'_{K-1}$ are Gaussian noises. For $\underbrace{\mathbf{u}^\intercal\mathbf{V} + \mathbf{g}^\intercal \otimes e''}$, we would like to transform the previous blocks into

$$\mathbf{d}^\intercal\mathbf{A}\mathbf{G}^{-1}(\mathbf{v}_0) + 2^0 e'' + e'''_0, \ \ldots, \ \mathbf{d}^\intercal\mathbf{A}\mathbf{G}^{-1}(\mathbf{v}_{K-1}) + 2^{K-1}e'' + e'''_{K-1},$$

so that they will cancel $\mathbf{d}^\intercal\mathbf{A}\mathbf{G}^{-1}(\mathbf{V})$ from $(\underbrace{\mathbf{d}^\intercal\mathbf{A} + \mathbf{u}^\intercal\mathbf{G}})\mathbf{G}^{-1}(\mathbf{V})$ while attaching $\mathbf{g}^\intercal \otimes e''$ to it. This can be done by finding low-norm $\mathbf{K}$ such that $\mathbf{B}_k\mathbf{K} = \mathbf{A}\mathbf{G}^{-1}(\mathbf{v}_k)$ for all k, or equivalently $\widetilde{\mathbf{B}}\mathbf{K} = \widetilde{\mathbf{P}}$ for

$$\widetilde{\mathbf{B}} = \begin{pmatrix} \mathbf{B}_0 \\ \vdots \\ \mathbf{B}_{K-1} \end{pmatrix}, \qquad \widetilde{\mathbf{P}} = \begin{pmatrix} \mathbf{A}\mathbf{G}^{-1}(\mathbf{v}_1) \\ \vdots \\ \mathbf{A}\mathbf{G}^{-1}(\mathbf{v}_k) \end{pmatrix}.$$

We sample $\widetilde{\mathbf{B}}$ with trapdoor to be able to sample $\mathbf{K}$. Some more care is required to support decryption with either structured noises or not. Relying on a modified version of evasive LWE, termed *evasive learning with structured errors* assumption, our candidate is again secure for $\mathbf{u}$ random over public subspace, sufficient for our application of succinct ABE.

2 Preliminaries

We denote the (computational) security parameter by λ, omitted for brevity except in definitions. Efficient algorithms are probabilistic polynomial-time Turing machines. So are efficient adversaries, but in addition they might also be given $\mathrm{poly}(\lambda)$-bit advice dependent on λ.[1] The statistical security parameter is $\kappa \in \omega(\log \lambda) \cap \lambda^{O(1)}$.

We use boldfaced lower-case letters for vectors, and boldfaced upper-case letters for matrices. They are always indexed using brackets and never using subscripts, so $\mathbf{v}_i$ is the i^{th} vector among a series of related vectors, and $\mathbf{U}[i,j]$ is the (i,j)-entry of $\mathbf{U}$. The $n \times n$ identity matrix is $\mathbf{I}_n$ (or simply $\mathbf{I}$ when the shape is not of importance). When the dimension is clear from context, $\boldsymbol{\iota}_i$ denotes the i^{th} standard basis vector, e.g., $\mathbf{I} = (\boldsymbol{\iota}_1, \dots)$. We write $\mathbf{0}_{n\times m}$ for the $n \times m$ zero matrix and $\mathbf{0}_n = \mathbf{0}_{n\times 1}$, with dimensions possibly omitted. We consider the infinity norm and its operator norm:

$$\|\mathbf{v}\| = \max_i |\mathbf{v}[i]|, \qquad \|\mathbf{U}\| = \max_i \sum_j |\mathbf{U}[i,j]|.$$

We strictly follow the convention of vectors being columns. If $\mathbf{v} \in \mathbb{Z}^z$ is a vector, $\|\mathbf{v}^\intercal\|$ is an operator norm and $\|\mathbf{v}^\intercal\| \le z\|\mathbf{v}\|$. For two matrices $\mathbf{A}, \mathbf{B}$ of shapes $n_1 \times m_1$ and $n_2 \times m_2$, their Kronecker product is an $n_1 n_2 \times m_1 m_2$ matrix,

$$\mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} \mathbf{A}[1,1]\mathbf{B} & \cdots & \mathbf{A}[1,m_1]\mathbf{B} \\ \vdots & \ddots & \vdots \\ \mathbf{A}[n_1,1]\mathbf{B} & \cdots & \mathbf{A}[n_1,m_1]\mathbf{B} \end{pmatrix}.$$

A useful property is $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}$ whenever all multiplications are compatible. The by-column flattening operator $\mathrm{col}(\cdot)$ concatenates all the columns of its input matrix. A convenient fact is $\mathrm{col}(\mathbf{ABC}) = (\mathbf{C}^\intercal \otimes \mathbf{A})\mathrm{col}(\mathbf{B})$ whenever $\mathbf{ABC}$ is compatible. We write $\star$ for the Kleene star, extend its usage to rows and columns of matrices, and overload it also for unnamed suitable value. For example, $\mathbf{E} \in \mathbb{Z}^{2\times\star}$ could mean that $\mathbf{E}$ is a 2-row integer matrix of some suitable number of columns so that its operations with other vectors and matrices are compatible, which should be clear from the context.

For integers L, B, we write $[L,B]$ for $\{\, z \in \mathbb{Z} \mid L \le z \le B \,\}$. For $n \ge 0$, we write $[n]$ for $[1,n]$. For natural number $q \ge 2$, we denote by $\mathbb{Z}_q$ the integers modulo q. Matrices over $\mathbb{Z}$ are naturally mapped to those over $\mathbb{Z}_q$, and this mapping is implicit so that they can be freely mixed for various operations. When $z \in \mathbb{Z}_q$, saying $z \in [L,B]$ means that a representative of z is in $[L,B]$.

Symbols. Table 1 explains select single-letter symbols used in this work.

[1] The reductions in this work do not "increase non-uniformity". Samplers, considered in many security definitions in this work, are a part of the adversary.

Table 1. Select single-letter symbols.

context	symbol	meaning
generic	λ, κ	computational/statistical security parameter
	$\beta, \delta, \boldsymbol{\delta}, \boldsymbol{\Delta}$	challenge bit, random scalar/vector/matrix
ABE	P, X, Y	predicate (family), key-tied set, ciphertext-tied set
	x, y, μ	key-tied value, ciphertext-tied value, message
	J, j	key count, index
	$\mathbf{r}, \mathbf{s}$	rerandomizer, garbling randomness
lattices	n, m, q, ρ	dimension, dimension, modulus, hardness exponent
	$\mathbf{g}, \mathbf{G}$	gadget vector/matrix
	$\mathbf{p}, \mathbf{P}, \mathbf{k}, \mathbf{K}$	image vector/matrix, preimage vector/matrix
	$\mathbf{A}, \mathbf{B}, \tau$	matrix, matrix with trapdoor, trapdoor
	$\mathcal{D}, \sigma$	discrete Gaussian distribution, width
	$e, \mathbf{e}, \mathbf{E}, B$	noise scalar/vector/matrix, noise bound
	$\mathbf{d}, r$	(evasive) LWE secret, sampler randomness
	K, k	structured noise length/index
IPFE	$\mathcal{I}, q, Z, z$	identity space (family), modulus, dimension, index
	B, σ	error bound, noise width
	$\mathbf{v}, \mathbf{u}^\intercal$	key/plaintext vector
	J, I, j, i	key/ciphertext count, key/ciphertext index
	$\boldsymbol{\psi}, \boldsymbol{\varphi}^\intercal$	key/ciphertext randomness for identity binding
garbling	$F, f, \mathbf{x}, L, \ell$	function family, function, input, length, index
	$q, n, \mathbf{s}$	modulus, randomness dimension, garbling randomness
	$w, \mathbf{w}, \mathbf{w}, \mathbf{W}$	secret scalar/vector, label (function) vector/matrix
	$\mathbf{t}, \mathbf{T}$	garbled table vector/matrix
	$\mathbf{R}$	reusable information
	$B, \mathbf{e}$	noise bound, garbling noise
circuits	C, d	circuit, depth (bound)
	x, M	wire value, wire value bound
	$\mathbf{x}, L, \ell$	input, length, index
DFA	$\Gamma, \mathfrak{Q}, \mathfrak{q}, \boldsymbol{\Gamma}$	DFA, number of states, state, transition matrix
	$\boldsymbol{\iota}_1, \boldsymbol{\xi}, \overline{L}$	initial/rejection state vector, input length bound
	$\mathbf{x}, L, \ell$	input, length, index

2.1 Attribute-Based Encryption

We consider *promise* ABE with respect to *partial* predicates. Correctness holds when the predicate outputs 1; security holds when it outputs 0; neither is guaranteed if the output is $\perp$. This feature is used to exclude arithmetic computations with out-of-bound wire values, and it enables accurate specification of bounded ABE. Promise variants of ABE are already considered in [JLL23].

Definition 1 (ABE [GPSW06])**.** Let $P = \{P_{\lambda,\mathsf{param}}\}_{\lambda\in\mathbb{N},\mathsf{param}\in\mathsf{Params}_\lambda}$ be a predicate family, where each $P_{\lambda,\mathsf{param}}$ is a function $X_{\lambda,\mathsf{param}}\times Y_{\lambda,\mathsf{param}}\to\{0,1,\perp\}$ and $\mathsf{Params}=\{\mathsf{Params}_\lambda\}_{\lambda\in\mathbb{N}}$ is a sequence of predicate description sets. An *attribute-based encryption scheme* for P consists of four efficient algorithms.

- $\mathsf{Setup}(1^\lambda,\mathsf{param})$ takes the predicate description $\mathsf{param}\in\mathsf{Params}_\lambda$ as input, and outputs a pair of master public/secret keys $(\mathsf{mpk},\mathsf{msk})$.
- $\mathsf{KeyGen}(1^\lambda,\mathsf{msk},x)$ takes as input msk and some $x\in X_{\lambda,\mathsf{param}}$. It outputs a secret key sk for x.
- $\mathsf{Enc}(1^\lambda,\mathsf{mpk},y,\mu)$ takes as input mpk, some $y\in Y_{\lambda,\mathsf{param}}$, and a single-bit message $\mu\in\{0,1\}$. It outputs a ciphertext ct of μ tied to y.
- $\mathsf{Dec}(1^\lambda,\mathsf{mpk},x,\mathsf{sk},y,\mathsf{ct})$ takes as input $\mathsf{mpk},x,\mathsf{sk},y,\mathsf{ct}$. It is supposed to output μ if $P_{\lambda,\mathsf{param}}(x,y)=1$.

The scheme must be *correct*, i.e., for all $\lambda\in\mathbb{N}$, $\mathsf{param}\in\mathsf{Params}_\lambda$, $x\in X_{\lambda,\mathsf{param}}$, $y\in Y_{\lambda,\mathsf{param}}$, $\mu\in\{0,1\}$ such that $P_{\lambda,\mathsf{param}}(x,y)=1$, it holds that

$$\Pr\left[\begin{array}{l}(\mathsf{mpk},\mathsf{msk})\xleftarrow{\$}\mathsf{Setup}(1^\lambda,\mathsf{param})\\ \qquad\mathsf{sk}\xleftarrow{\$}\mathsf{KeyGen}(1^\lambda,\mathsf{msk},x)\ :\ \mathsf{Dec}(1^\lambda,\mathsf{mpk},x,\mathsf{sk},y,\mathsf{ct})=\mu\\ \qquad\mathsf{ct}\xleftarrow{\$}\mathsf{Enc}(1^\lambda,\mathsf{mpk},y,\mu)\end{array}\right]=1.$$

Security. We consider *very selective* [AWY20] (also known as *static* [RW15]) security in this work. Before the ABE is set up, the adversary chooses all the x's and the y for which the secret keys and the challenge ciphertext are to be generated.

Definition 2 (ABE security [AWY20])**.** An ABE scheme (Definition 1) is *very selectively secure* if $\mathsf{Exp}^0_{\mathrm{ABE}}\approx\mathsf{Exp}^1_{\mathrm{ABE}}$, where $\mathsf{Exp}^\beta_{\mathrm{ABE}}(1^\lambda,\mathcal{A})$ proceeds as follows.

- **Challenge.** Launch $\mathcal{A}(1^\lambda)$ and receive from it

$$\mathsf{param}\in\mathsf{Params}_\lambda,\quad \{x_j\}_{j\in[J]}\ (x_j\in X_{\lambda,\mathsf{param}}\text{ for all }j\in[J]),\quad y^*\in Y_{\lambda,\mathsf{param}}.$$

- **Setup.** Run

$$\begin{aligned}(\mathsf{mpk},\mathsf{msk})&\xleftarrow{\$}\mathsf{Setup}(1^\lambda,\mathsf{param}),\\ \mathsf{sk}_j&\xleftarrow{\$}\mathsf{KeyGen}(1^\lambda,\mathsf{msk},x_j)\text{ for all }j\in[J],\\ \mathsf{ct}^*&\xleftarrow{\$}\mathsf{Enc}(1^\lambda,\mathsf{mpk},y^*,\beta),\end{aligned}$$

  and send $\mathsf{mpk},\{\mathsf{sk}_j\}_{j\in[J]},\mathsf{ct}^*$ to $\mathcal{A}$.

- **Guess.** The adversary $\mathcal{A}$ outputs $\beta'\in\{0,1\}$. The output of the experiment is β' if $P_{\lambda,\mathsf{param}}(x_j,y^*)=0$ for all $j\in[J]$. Otherwise, the output is set to 0.

2.2 Lattices

Let $n, m \geq 1$ and $q \geq 2$ be integers such that $\log_2 q \leq m/n \in \mathbb{Z}$. Let

$$\mathbf{g} = (2^0, \ldots, 2^{m/n-1})^\intercal \quad \text{and} \quad \mathbf{G} = \mathbf{I}_n \otimes \mathbf{g}^\intercal$$

be the gadget vector and the gadget matrix. Given $\mathbf{p} \in \mathbb{Z}_q^n$, we denote by $\mathbf{G}^{-1}(\mathbf{p})$ the vector $\mathbf{k} \in \{0,1\}^m$, where $\mathbf{k}[(i-1)\cdot m/n+1], \ldots, \mathbf{k}[i \cdot m/n]$ are the bits of (the non-negative representative less than q of) $\mathbf{p}[i]$, low to high. It holds that $\mathbf{G}\mathbf{G}^{-1}(\mathbf{p}) = \mathbf{p}$. Given $K \in \mathbb{N}$, we let (note the different order in the Kronecker product)

$$\widetilde{\mathbf{g}} = (2^0, \ldots, 2^{K-1})^\intercal \quad \text{and} \quad \widetilde{\mathbf{G}} = \widetilde{\mathbf{g}}^\intercal \otimes \mathbf{I}.$$

When $2^K \geq q$, the vector $\widetilde{\mathbf{G}}^{-1}(\mathbf{p})$ is again the bit decomposition of $\mathbf{p} \in \mathbb{Z}_q^\star$, arranged so that $\widetilde{\mathbf{G}}\widetilde{\mathbf{G}}^{-1}(\mathbf{p}) = \mathbf{p}$. The notations $\mathbf{G}^{-1}(\cdot), \widetilde{\mathbf{G}}^{-1}(\cdot)$ extend to matrices column-wise.

Given $\mathbf{B} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{p} \in \mathbb{Z}_q^n$ such that $\mathbf{B}\mathbf{k} = \mathbf{p}$ has a solution $\mathbf{k}^* \in \mathbb{Z}^m$, we write the lattice coset

$$\Lambda_{\mathbf{p}}^\perp(\mathbf{B}) = \{\, \mathbf{k} \in \mathbb{Z}^m \mid \mathbf{B}\mathbf{k} = \mathbf{p} \,\} = \mathbf{k}^* + \Lambda_{\mathbf{0}}^\perp(\mathbf{B}).$$

Let Λ' be any lattice coset and $\sigma \geq 0$, we denote by $\mathcal{D}_{\Lambda',\sigma}$ the discrete Gaussian distribution [MP11] over Λ' with width σ. Since we insist on perfect correctness, it is necessary to truncate unbounded noises in algorithms:

Definition 3 (truncated $\mathcal{D}_{\mathbb{Z},\sigma}$)**.** Let $B \geq 0$, the distribution $\mathcal{D}_{\mathbb{Z},\sigma,\leq B}$ is sampled by first sampling $x \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma}$, then returning x if $|x| \leq B$, and 0 otherwise.

Lemma 1 (truncation of $\mathcal{D}_{\mathbb{Z},\sigma}$)**.** *For all $\sigma \geq 1$ and $\kappa \geq 2$,*

$$\Pr\left[\, x \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma} \;:\; |x| > \sigma\sqrt{\kappa} \,\right] \leq 2^{-\kappa},$$

so $\mathcal{D}_{\mathbb{Z},\sigma,\leq\sigma\sqrt{\kappa}}$ is $2^{-\kappa}$-close to $\mathcal{D}_{\mathbb{Z},\sigma}$.

We will also need noise flooding with discrete Gaussian:

Lemma 2 (noise flooding using $\mathcal{D}_{\mathbb{Z},\sigma}$)**.** *For all $y \in \mathbb{Z}$ and $\sigma \geq 2^{\kappa+6}y$, it holds that $(y + \mathcal{D}_{\mathbb{Z},\sigma})$ is $2^{-\kappa}$-close to $\mathcal{D}_{\mathbb{Z},\sigma}$.*

Trapdoor Generation and Gaussian Sampling. We need the following lemma:

Lemma 3 ([MP12; Theorem 2])**.** *There are efficient algorithms* TrapGen *and* SampleD*, and functions $m_0 \in \Theta(n \log q)$ and $\sigma_0 \in \omega(\sqrt{m \log m}) \cap \mathrm{O}(m)$ satisfying these conditions.*

- $\mathsf{TrapGen}(1^n, 1^m, q)$ *takes as input $n \geq 1$, $q \geq 2$, and $m \geq m_0(n,q)$. It outputs $(\mathbf{B}, \tau)$ such that $\mathbf{B} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{B}$ is $\mathrm{negl}(n)$-close to uniform over $\mathbb{Z}_q^{n\times m}$.*
- $\mathsf{SampleD}(\mathbf{B}, \tau, \mathbf{p}, \sigma)$ *takes as input $\mathbf{B}, \tau$ from* TrapGen*, some $\mathbf{p} \in \mathbb{Z}_q^n$, and $\sigma \geq \sigma_0(n,m)$. It outputs $\mathbf{k} \in \mathbb{Z}^m$ such that $\mathbf{B}\mathbf{k} = \mathbf{p}$, $\|\mathbf{k}\| \leq \sigma\sqrt{m}$, and $\mathbf{k}$ is $\mathrm{negl}(n)$-close to $\mathcal{D}_{\Lambda_{\mathbf{p}}^\perp(\mathbf{B}),\sigma}$.*

Batch Notation. It is convenient to extend SampleD to process multiple $\mathbf{p}$'s in one shot. Let $\mathbf{P} = (\mathbf{p}_1, \ldots, \mathbf{p}_{m'})$ be a matrix or a batch of vectors, then $\mathsf{SampleD}(\mathbf{B}, \tau, \mathbf{P}, \sigma)$ is

$$\mathbf{K} \xleftarrow{\$} \big(\mathsf{SampleD}(\mathbf{B}, \tau, \mathbf{p}_1, \sigma),\ \ldots,\ \mathsf{SampleD}(\mathbf{B}, \tau, \mathbf{p}_{m'}, \sigma)\big),$$

with fresh randomness for each call to SampleD on the right-hand side. We also write $\mathbf{B}^{-1}(\mathbf{P})$ for an output of $\mathsf{SampleD}(\mathbf{B}, \tau, \mathbf{P}, \sigma)$.

Learning with Errors Assumption. We rely on the LWE assumption.

Assumption 1 (LWE [Reg05])**.** Let $n, m \le \mathrm{poly}(\lambda)$, $q = q(\lambda)$, $\sigma = \sigma(\lambda)$, and

$$\mathbf{A} \xleftarrow{\$} \mathbb{Z}_{q(\lambda)}^{n(\lambda)\times m(\lambda)}, \quad \mathbf{s} \xleftarrow{\$} \mathbb{Z}_{q(\lambda)}^{n(\lambda)}, \quad \mathbf{e} \xleftarrow{\$} \mathbb{Z}_{q(\lambda)}^{m(\lambda)}, \quad \boldsymbol{\delta} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma(\lambda)}.$$

The *LWE assumption* $\mathsf{LWE}_{n,m,q,\sigma}$ states that

$$\{(1^\lambda, \mathbf{A}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal)\}_{\lambda\in\mathbb{N}} \approx \{(1^\lambda, \mathbf{A}, \boldsymbol{\delta}^\intercal)\}_{\lambda\in\mathbb{N}}.$$

Assumption 2 (flipped LWE)**.** Let $N, m \le \mathrm{poly}(\lambda)$, $q = q(\lambda)$, $\sigma = \sigma(\lambda)$, and

$$\mathbf{A} \xleftarrow{\$} \{0,1\}^{N(\lambda)\times m(\lambda)}, \quad \mathbf{s} \xleftarrow{\$} \mathbb{Z}_{q(\lambda)}^{N(\lambda)}, \quad \mathbf{e} \xleftarrow{\$} \mathbb{Z}_{q(\lambda)}^{m(\lambda)}, \quad \boldsymbol{\delta} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma(\lambda)}.$$

The *flipped LWE assumption* $\mathsf{FlipLWE}_{N,m,q,\sigma}$ states that

$$\{(1^\lambda, \mathbf{A}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal)\}_{\lambda\in\mathbb{N}} \approx \{(1^\lambda, \mathbf{A}, \boldsymbol{\delta}^\intercal)\}_{\lambda\in\mathbb{N}}.$$

Lemma 4 (flipped LWE)**.** *Suppose* $\mathsf{LWE}_{n,m,q,\sigma}$ *holds and* $N \le \mathrm{poly}(\lambda)$ *satisfies*

$$0 \le N/n - \log_2 q \in \omega(\log\lambda),$$

then $\mathsf{FlipLWE}_{N,m,q,\sigma}$ *holds.*

Parameters. In this work, we rely on the LWE assumption with subexponential modulus-to-noise ratio, i.e., $n = \lambda^{\Theta(1)}$, $m \le \mathrm{poly}(n)$, $q \le 2^{\mathrm{poly}(n)}$, $\frac{q}{\sigma\sqrt{n}} = \Theta(2^{n^\rho})$, where $0 < \rho < 1$ is a constant. Strictly speaking, those parameters should be chosen by the adversary subject to those constraints — most algorithms require a prime q, it is currently unknown [TCH12] how to pick $\mathrm{poly}(\lambda)$-bit primes in deterministic polynomial time, and the adversary's choice of scheme parameters affect the ranges of LWE parameters.

2.3 Evasive LWE Assumption

Conceptually, we consider two knowledge assumptions — evasive learning with errors (evasive LWE) and evasive learning with *structured* errors (not abbreviated). The former is a special case of the latter, and we formulate the definitions as such.

Assumption 3 (evasive learning with structured errors)**.** Let $\mathcal{S}(1^\lambda; r)$ be an algorithm that, given randomness r, outputs

$$1^n, 1^m, 1^K, q, 1^{n'}, 1^J, 1^I, 1^{m'}, \qquad \sigma_{-1}, \quad \sigma_{\mathrm{post}} \ge \sigma_{\mathrm{pre}} \ge 0,$$
$$\widetilde{\mathbf{P}} \in \mathbb{Z}_q^{n(K+1)\times J}, \qquad \{\mathbf{A}_{1,i}, \mathbf{A}_{0,i}\}_{i\in[I]} \quad (\mathbf{A}_{1,i} \in \mathbb{Z}_q^{n\times m'},\ \mathbf{A}_{0,i} \in \mathbb{Z}_q^{n'\times m'}),$$

where $m \ge m_0(n(K+1), q)$ and $\sigma_{-1} \ge \sigma_0(n(K+1), m)$ are constrained by Lemma 3. Suppose

$$(\widetilde{\mathbf{B}}, \tau) \xleftarrow{\$} \mathsf{TrapGen}(1^{n(K+1)}, 1^m, q), \qquad \mathbf{K} \xleftarrow{\$} \mathsf{SampleD}(\widetilde{\mathbf{B}}, \tau, \widetilde{\mathbf{P}}, \sigma_{-1}),$$

$$\begin{aligned}
&\mathbf{d}_0 \xleftarrow{\$} \mathbb{Z}_q^{n'}; && \mathbf{e}_{\mathrm{pre},i,0} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{pre}}}^{m'}, && \mathbf{e}_{\mathrm{post},i,0} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{post}}}^{m'}, && \boldsymbol{\delta}_{i,0} \xleftarrow{\$} \mathbb{Z}_q^{m'},\\
&\mathbf{d}_i \xleftarrow{\$} \mathbb{Z}_q^{n}, && \mathbf{e}_{\mathrm{pre},i,1} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{pre}}}^{m(K+1)}, && \mathbf{e}_{\mathrm{post},i,1} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{post}}}^{m(K+1)}, && \boldsymbol{\delta}_{i,1} \xleftarrow{\$} \mathbb{Z}_q^{m(K+1)},\\
& && \mathbf{e}_{\mathrm{pre},i,2} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{pre}}}^{J(K+1)}, && \mathbf{e}_{\mathrm{pre},i,4} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{pre}}}^{J}, && \boldsymbol{\delta}_{i,2} \xleftarrow{\$} \mathbb{Z}_q^{J(K+1)},\\
& && \mathbf{e}_{\mathrm{pre},i,3} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{pre}}}^{m}, && \mathbf{e}_{\mathrm{post},i,3} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{post}}}^{m}, && \text{for all } i \in [I].
\end{aligned}$$

Let $\widetilde{\mathbf{g}}^\intercal = (2^0, 2^1, \ldots, 2^{K-1})^\intercal$ and

$$\begin{aligned}
\widetilde{\mathbf{B}} &= (\mathbf{B}_{-1}^\intercal, \mathbf{B}_0^\intercal, \ldots, \mathbf{B}_{K-1}^\intercal)^\intercal, & \mathbf{B} &= (\mathbf{B}_{-1}, \mathbf{B}_0, \ldots, \mathbf{B}_{K-1}),\\
\widetilde{\mathbf{P}} &= (\mathbf{P}_{-1}^\intercal, \mathbf{P}_0^\intercal, \ldots, \mathbf{P}_{K-1}^\intercal)^\intercal, & \mathbf{P} &= (\mathbf{P}_{-1}, \mathbf{P}_0, \ldots, \mathbf{P}_{K-1}),
\end{aligned}$$

where $\mathbf{B}_k \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{P}_k \in \mathbb{Z}_q^{n\times J}$. The precondition $\mathsf{evLWsE}_{\mathrm{pre}}^{\mathcal{S}}$ is

$$\left\{\begin{pmatrix} 1^\lambda, \\ r, \\ \mathbf{B}, \end{pmatrix}, \begin{Bmatrix} \mathbf{d}_i^\intercal \mathbf{P} + (\mathbf{0}_{1\times J}, \widetilde{\mathbf{g}}^\intercal \otimes \mathbf{e}_{\mathrm{pre},i,4}^\intercal) + \mathbf{e}_{\mathrm{pre},i,2}^\intercal, \\ \mathbf{d}_i^\intercal \mathbf{B} + (\mathbf{0}_{1\times m}, \widetilde{\mathbf{g}}^\intercal \otimes \mathbf{e}_{\mathrm{pre},i,3}^\intercal) + \mathbf{e}_{\mathrm{pre},i,1}^\intercal, \\ \mathbf{d}_i^\intercal \mathbf{A}_{1,i} + \mathbf{d}_0^\intercal \mathbf{A}_{0,i} + \mathbf{e}_{\mathrm{pre},i,0}^\intercal \end{Bmatrix}_{i\in[I]}\right\}_{\lambda\in\mathbb{N}} \approx \left\{\begin{pmatrix} 1^\lambda, \\ r, \\ \mathbf{B}, \end{pmatrix}, \begin{Bmatrix} \boldsymbol{\delta}_{i,2}^\intercal, \\ \boldsymbol{\delta}_{i,1}^\intercal, \\ \boldsymbol{\delta}_{i,0}^\intercal \end{Bmatrix}_{i\in[I]}\right\}_{\lambda\in\mathbb{N}}.$$

The postcondition $\mathsf{evLWsE}_{\mathrm{post}}^{\mathcal{S}}$ is

$$\left\{\begin{pmatrix} 1^\lambda, r, \\ \mathbf{B}, \mathbf{K}, \end{pmatrix}, \begin{Bmatrix} \mathbf{d}_i^\intercal \mathbf{B} + (\mathbf{0}_{1\times m}, \widetilde{\mathbf{g}}^\intercal \otimes \mathbf{e}_{\mathrm{post},i,3}^\intercal) + \mathbf{e}_{\mathrm{post},i,1}^\intercal, \\ \mathbf{d}_i^\intercal \mathbf{A}_{1,i} + \mathbf{d}_0^\intercal \mathbf{A}_{0,i} + \mathbf{e}_{\mathrm{post},i,0}^\intercal \end{Bmatrix}_{i\in[I]}\right\}_{\lambda\in\mathbb{N}} \approx \left\{\begin{pmatrix} 1^\lambda, r, \\ \mathbf{B}, \mathbf{K}, \end{pmatrix}, \begin{Bmatrix} \boldsymbol{\delta}_{i,1}^\intercal, \\ \boldsymbol{\delta}_{i,0}^\intercal \end{Bmatrix}_{i\in[I]}\right\}_{\lambda\in\mathbb{N}}.$$

The *evasive learning with structured errors assumption* states that $\mathsf{evLWsE}_{\mathrm{pre}}^{\mathcal{S}}$ implies $\mathsf{evLWsE}_{\mathrm{post}}^{\mathcal{S}}$ for all efficient $\mathcal{S}$.

Assumption 4 (evasive LWE)**.** A sampler $\mathcal{S}$ (Assumption 3) does not use structured noises if $\Pr[K = 0] = 1$, for which $\mathsf{evLWE}_{\mathrm{pre}}^{\mathcal{S}}$ (resp. $\mathsf{evLWE}_{\mathrm{post}}^{\mathcal{S}}$) is defined to be $\mathsf{evLWsE}_{\mathrm{pre}}^{\mathcal{S}}$ (resp. $\mathsf{evLWsE}_{\mathrm{post}}^{\mathcal{S}}$). The *evasive LWE assumption* states that $\mathsf{evLWE}_{\mathrm{pre}}^{\mathcal{S}}$ implies $\mathsf{evLWE}_{\mathrm{post}}^{\mathcal{S}}$ for all efficient $\mathcal{S}$ not using structured noises.

Note that for evasive LWE, $\widetilde{\mathbf{B}} = \mathbf{B} = \mathbf{B}_{-1}$ and $\widetilde{\mathbf{P}} = \mathbf{P} = \mathbf{P}_{-1}$.

Remark 1 (public-coin samplers). As in [Wee22,WWW22], our evasive assumptions are only for public-coin samplers, which we enforce by providing the sampler randomness to the distinguisher. It is weaker than the versions [Tsa22,VWW22] used for witness encryption, which considers private-coin samplers and the distinguisher, instead of the sampler randomness, gets the auxiliary information, an additional output produced by the sampler. The public-coin assumption avoids obfuscation-based counterexamples, where $\mathbf{P}$ is sampled with a trapdoor and the auxiliary information contains an obfuscated program with the trapdoor hardwired.

Remark 2 (comparison with [Wee22,WWW22]). As suggested in [Wee22], the noise magnitude in the postcondition can be made larger than that in the precondition for a more conservative assumption. We can implement it by requiring the assumptions to hold only for samplers with a gap between σ_{post} and σ_{pre}. We additionally allow a worse Gaussian preimage $\mathbf{K}$. Other than the magnitudes of samples, our version of evasive LWE is stronger than the version in [Wee22], and is similar to but incomparable with that in [WWW22].

While appearing much more complex, our evasive learning with structured errors assumption follows the same rationale as all existing versions of public-coin evasive LWE. In evasive LWE (without structured noises), the basic idea is that given

$$\mathbf{K} \xleftarrow{\$} \mathbf{B}^{-1}(\mathbf{P}), \qquad \mathbf{d}^\intercal\mathbf{B} + \mathbf{e}_1^\intercal, \qquad \mathbf{d}^\intercal\mathbf{A} + \mathbf{e}_0^\intercal,$$

the trapdoor preimages can only be meaningfully used to multiply the corresponding LWE samples, i.e., $(\mathbf{d}^\intercal\mathbf{B} + \mathbf{e}_1^\intercal)\mathbf{K} = \mathbf{d}^\intercal\mathbf{P} + \mathbf{e}_1^\intercal\mathbf{K}$, and that moreover, the correlation between $\mathbf{e}_1$ and $\mathbf{e}_1^\intercal\mathbf{K}$ is not useful to the attacker if the samples have no apparent relations when $\mathbf{e}_1^\intercal\mathbf{K}$ is replaced by fresh $\mathbf{e}_2^\intercal$. Therefore, the precondition requires that

$$\mathbf{d}^\intercal\mathbf{P} + \mathbf{e}_2^\intercal, \qquad \mathbf{d}^\intercal\mathbf{B} + \mathbf{e}_1^\intercal, \qquad \mathbf{d}^\intercal\mathbf{A} + \mathbf{e}_0^\intercal$$

be pseudorandom, and evasive LWE asserts that $(\mathbf{d}^\intercal\mathbf{B} + \mathbf{e}_1^\intercal, \mathbf{d}^\intercal\mathbf{A} + \mathbf{e}_0^\intercal)$ is pseudorandom in the presence of $\mathbf{K}$.

For evasive learning with structured errors, suppose we are given

$$\mathbf{K}, \qquad \{\mathbf{d}^\intercal\mathbf{B}_k + 2^k\mathbf{e}_3^\intercal + \mathbf{e}_{1,k}^\intercal\}_{0\le k\le K-1}, \qquad \mathbf{d}^\intercal\mathbf{A} + \mathbf{e}_0^\intercal,$$

where $\mathbf{B}_k\mathbf{K} = \mathbf{P}_k$ for all k. Again, the trapdoor preimages can only be meaningfully used to multiply the corresponding LWE samples to obtain

$$(\mathbf{d}^\intercal\mathbf{B}_k + 2^k\mathbf{e}_3^\intercal + \mathbf{e}_{1,k}^\intercal)\mathbf{K} = \mathbf{d}^\intercal\mathbf{P}_k + 2^k\mathbf{e}_3^\intercal\mathbf{K} + \mathbf{e}_{1,k}^\intercal\mathbf{K}.$$

Following the same rationale of "correlation among noises not helpful if samples with fresh noises are pseudorandom", we replace $\{\mathbf{e}_{1,k}^\intercal\mathbf{K}\}_k, \mathbf{e}_3^\intercal\mathbf{K}$ by independent $\{\mathbf{e}_{2,k}^\intercal\}_k, \mathbf{e}_4^\intercal$. This gives rise to the conditions in our assumption. It remains to figure out how to sample $\mathbf{K}$ simultaneously satisfying $\mathbf{B}_k\mathbf{K} = \mathbf{P}_k$ for all k. The equations can be rewritten as

$$\begin{pmatrix}\mathbf{B}_1\\ \vdots\\ \mathbf{B}_k\end{pmatrix}\mathbf{K} = \begin{pmatrix}\mathbf{P}_1\\ \vdots\\ \mathbf{P}_k\end{pmatrix},$$

so it suffices to sample the vertically blocked $\mathbf{B}_k$'s with trapdoor. We further incorporate a block ($\mathbf{B}_{-1}$ and $\mathbf{P}_{-1}$) without structured noises and arrive at Assumption 3.

3 Evasive Inner-Product Functional Encryption

We define identity-based IPFE with structured noises and its security. A scheme might not support every combination of moduli, identities, and noise structures, which can be signaled by the algorithms aborting.

Definition 4 ((IB-)evIPFE(-w/sn))**.** Let $\mathcal{I} = \{\mathcal{I}_{\lambda,q,K,Z}\}_{\lambda,q,K,Z\in\mathbb{N}}$ be a family of sets. An *(identity-based) evasive IPFE (with structured noises) scheme* for $\mathcal{I}$ consists of four efficient algorithms.

- $\mathsf{Setup}(1^\lambda, q, 1^K, 1^Z)$ takes as input the modulus q, the noise parameter K, and the dimension Z. It outputs a pair $(\mathsf{impk}, \mathsf{imsk})$ of master public/secret keys.

- $\mathsf{KeyGen}(1^\lambda, \mathsf{imsk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g})$ takes as input imsk, an identity $\mathsf{id} \in \mathcal{I}_{\lambda,q,K,Z}$, a key vector $\mathbf{v}_\mathfrak{o} \in \mathbb{Z}_q^Z$, and a key matrix $\mathbf{V}_\mathfrak{g} \in \mathbb{Z}_q^{Z\times K}$. It outputs a secret key isk for $(\mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g})$ under identity id.
- $\mathsf{Enc}(1^\lambda, \mathsf{impk}, \mathsf{id}, \mathfrak{s}, \mathbf{u}^\intercal)$ takes as input impk, id, a noise structure $\mathfrak{s} \in \{\mathfrak{o}, \mathfrak{g}\}$, and a plaintext vector $\mathbf{u} \in \mathbb{Z}_q^Z$. It outputs a ciphertext ict of $\mathbf{u}$ under id for noise structure $\mathfrak{s}$.
- $\mathsf{Dec}(1^\lambda, \mathsf{impk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g}, \mathsf{isk}, \mathfrak{s}, \mathsf{ict})$ takes as input $\mathsf{impk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g}, \mathsf{isk}, \mathfrak{s}, \mathsf{ict}$. If ict is for noise structure $\mathfrak{s}$ and both isk and ict are under id, the decryption result is in $\mathbb{Z}_q$ and is supposed to approximate $\mathbf{u}^\intercal\mathbf{v}_\mathfrak{o}$ (when $\mathfrak{s} = \mathfrak{o}$) or $\mathbf{u}^\intercal\mathbf{V}_\mathfrak{g}$ (when $\mathfrak{s} = \mathfrak{g}$).

Let B be a function mapping (λ, q, K, Z) to natural numbers. The scheme is *B-correct for* $\mathfrak{o}$ if for all $\lambda, q, K, Z \in \mathbb{N}$, $\mathsf{id} \in \mathcal{I}_{\lambda,q,K,Z}$, $\mathbf{V}_\mathfrak{g} \in \mathbb{Z}_q^{Z\times K}$, $\mathbf{v}_\mathfrak{o}, \mathbf{u} \in \mathbb{Z}_q^Z$, it holds that

$$\Pr\left[\begin{array}{ll} (\mathsf{impk}, \mathsf{imsk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, q, 1^K, 1^Z) & \mathsf{Dec}(1^\lambda, \mathsf{impk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g}, \mathsf{isk}, \mathfrak{g}, \mathsf{ict}) \\ \mathsf{isk} \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda, \mathsf{imsk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g}) : & \quad = \mathbf{u}^\intercal\mathbf{v}_\mathfrak{o} + e_\mathfrak{o} \quad \text{for some} \\ \mathsf{ict} \xleftarrow{\$} \mathsf{Enc}(1^\lambda, \mathsf{impk}, \mathsf{id}, \mathfrak{o}, \mathbf{u}^\intercal) & e_\mathfrak{o} \in [-B(\lambda, q, K, Z), B(\lambda, q, K, Z)] \end{array}\right] = 1.$$

For *B-correctness for* $\mathfrak{g}$, the requirement is

$$\Pr\left[\begin{array}{ll} \text{—}''\text{—} \quad \text{for impk, imsk, isk} & \mathsf{Dec}(1^\lambda, \mathsf{impk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g}, \mathsf{isk}, \mathfrak{g}, \mathsf{ict}) \\ \mathsf{ict} \xleftarrow{\$} \mathsf{Enc}(1^\lambda, \mathsf{impk}, \mathsf{id}, \mathfrak{g}, \mathbf{u}^\intercal) & : \quad = \mathbf{u}^\intercal\mathbf{V}_\mathfrak{g} + \widetilde{\mathbf{g}}^\intercal \otimes e_{\mathfrak{g}\mathfrak{g}} + \mathbf{e}_{\mathfrak{g}\mathfrak{o}}^\intercal \quad \text{for some} \\ & e_{\mathfrak{g}\mathfrak{g}}, \mathbf{e}_{\mathfrak{g}\mathfrak{o}} \in [-B(\lambda, q, K, Z), B(\lambda, q, K, Z)]^\star \end{array}\right] = 1.$$

The scheme is *B-correct* if both are satisfied.

The correctness property is trivial when $\mathcal{I} = \varnothing$ or $B \geq q/2$. We are interested in large enough $\mathcal{I}$ and reasonably small B, for useful choices of q, Z. Jumping ahead, Constructions 2 and 3 support exponentially large $|\mathcal{I}|$ and q/B, which suffice for our ABE schemes.

When referring to specific schemes, if $|\mathcal{I}|$ is small, we remove "identity-based" (IB), and if they only support $K = 0$, we remove "with structured noises" (w/sn).

Batch Notations. It is convenient to consider KeyGen, Enc, Dec in batches. Let

$$\mathbf{V}_\mathfrak{o} = (\mathbf{v}_{1,\mathfrak{o}}, \ldots, \mathbf{v}_{J,\mathfrak{o}}) \in \mathbb{Z}_q^{Z\times J}, \quad \mathbf{V}_\mathfrak{g} = (\mathbf{V}_{1,\mathfrak{g}}, \ldots, \mathbf{v}_{J,\mathfrak{g}}) \in \mathbb{Z}_q^{Z\times KJ}, \quad \mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_I) \in \mathbb{Z}_q^{Z\times I}$$

be matrices (batches of vectors/matrices) and $\mathsf{id} \in \mathcal{I}$, $\mathfrak{s} \in \{\mathfrak{o}, \mathfrak{g}\}$. We write

$$\mathsf{isk} \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{imsk}, \mathsf{id}, \mathbf{V}_\mathfrak{o}, \mathbf{V}_\mathfrak{g}) \quad \text{and} \quad \mathsf{ict} \xleftarrow{\$} \mathsf{Enc}(\mathsf{impk}, \mathsf{id}, \mathfrak{s}, \mathbf{U}^\intercal)$$

for $\mathsf{isk} \leftarrow \{\mathsf{isk}_j\}_{j\in[J]}$ and $\mathsf{ict} \leftarrow \{\mathsf{ict}_i\}_{i\in[I]}$, where

$$\begin{aligned} \mathsf{isk}_j &\xleftarrow{\$} \mathsf{KeyGen}(\mathsf{imsk}, \mathsf{id}, \mathbf{v}_{j,\mathfrak{o}}, \mathbf{V}_{j,\mathfrak{g}}) && \text{for all } j \in [J], \\ \mathsf{ict}_i &\xleftarrow{\$} \mathsf{Enc}(\mathsf{impk}, \mathsf{id}, \mathfrak{s}, \mathbf{u}_i^\intercal) && \text{for all } i \in [I], \end{aligned}$$

and $\mathsf{Dec}(\mathsf{impk}, \mathsf{id}, \mathbf{V}_\mathfrak{o}, \mathbf{V}_\mathfrak{g}, \mathsf{isk}, \mathfrak{s}, \mathsf{ict})$ means

$$\begin{pmatrix} \mathsf{Dec}(\mathsf{impk}, \mathsf{id}, \mathbf{v}_{1,\mathfrak{o}}, \mathbf{V}_{1,\mathfrak{g}}, \mathsf{isk}_1, \mathfrak{s}, \mathsf{ict}_1) & \cdots & \mathsf{Dec}(\mathsf{impk}, \mathsf{id}, \mathbf{v}_{J,\mathfrak{o}}, \mathbf{V}_{J,\mathfrak{g}}, \mathsf{isk}_J, \mathfrak{s}, \mathsf{ict}_1) \\ \vdots & \ddots & \vdots \\ \mathsf{Dec}(\mathsf{impk}, \mathsf{id}, \mathbf{v}_{1,\mathfrak{o}}, \mathbf{V}_{1,\mathfrak{g}}, \mathsf{isk}_1, \mathfrak{s}, \mathsf{ict}_I) & \cdots & \mathsf{Dec}(\mathsf{impk}, \mathsf{id}, \mathbf{v}_{J,\mathfrak{o}}, \mathbf{V}_{J,\mathfrak{g}}, \mathsf{isk}_J, \mathfrak{s}, \mathsf{ict}_I) \end{pmatrix},$$

which is approximately $\mathbf{U}^\intercal\mathbf{V}_\mathfrak{s}$. The mnemonic is that the decryption result is the plaintext matrix $\mathbf{U}^\intercal$ multiplied by the key matrix $\mathbf{V}_\mathfrak{s}$.

Security. We consider *very selective* (also known as *static*) security with respect to a sampler producing pseudorandom structurally noisy outputs.

Definition 5 (IB-evIPFE-w/sn security)**.** Let $\mathcal{S} = (\mathcal{S}_{\mathrm{V}}, \mathcal{S}_{\mathrm{U}})$ be two algorithms with the following syntax.

- $\mathcal{S}_{\mathrm{V}}(1^\lambda; r_{\mathrm{pub}})$, given randomness r_{pub}, outputs

$$q, 1^K, 1^Z, \mathsf{ID} \subseteq \mathcal{I}_{\lambda,q,K,Z}, \{1^{J_{\mathsf{id}}}, 1^{I_{\mathsf{id},\mathfrak{o}}}, 1^{I_{\mathsf{id},\mathfrak{g}}}\}_{\mathsf{id}\in\mathsf{ID}}, \qquad \sigma_{\mathrm{pre}} \geq 0,$$
$$\{\mathbf{V}_{\mathsf{id},\mathfrak{o}}, \mathbf{V}_{\mathsf{id},\mathfrak{g}}\}_{\mathsf{id}\in\mathsf{ID}} \qquad (\mathbf{V}_{\mathsf{id},\mathfrak{o}} \in \mathbb{Z}_q^{Z\times J_{\mathsf{id}}}, \mathbf{V}_{\mathsf{id},\mathfrak{g}} \in \mathbb{Z}_q^{Z\times KJ_{\mathsf{id}}}).$$

- $\mathcal{S}_{\mathrm{U}}(1^\lambda, r_{\mathrm{pub}}; r_{\mathrm{priv}})$, given r_{pub} and additional randomness r_{priv}, outputs

$$\{\mathbf{U}_{\mathsf{id},\mathfrak{s}}^\intercal\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}} \qquad (\mathbf{U}_{\mathsf{id},\mathfrak{s}} \in \mathbb{Z}_q^{Z\times I_{\mathsf{id},\mathfrak{s}}}),$$

  where $q, Z, \mathsf{ID}, \{I_{\mathsf{id},\mathfrak{s}}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}$ agree with the output of $\mathcal{S}_{\mathrm{V}}(1^\lambda; r_{\mathrm{pub}})$.

Suppose $(\mathsf{impk}, \mathsf{imsk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, q, 1^K, 1^Z)$ and for all $\mathsf{id} \in \mathsf{ID}$, $\mathfrak{s} \in \{\mathfrak{o}, \mathfrak{g}\}$,

$$\mathbf{E}_{\mathsf{id},\mathfrak{o}} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{pre}}}^{J_{\mathsf{id}}\times I_{\mathsf{id},\mathfrak{o}}}, \qquad \mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{g}} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{pre}}}^{J_{\mathsf{id}}\times I_{\mathsf{id},\mathfrak{g}}}, \qquad \mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{o}} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{pre}}}^{KJ_{\mathsf{id}}\times I_{\mathsf{id},\mathfrak{g}}},$$
$$\mathbf{\Delta}_{\mathrm{pre},\mathsf{id},\mathfrak{o}} \xleftarrow{\$} \mathbb{Z}_q^{J_{\mathsf{id}}\times I_{\mathsf{id},\mathfrak{o}}}, \qquad \mathbf{\Delta}_{\mathrm{pre},\mathsf{id},\mathfrak{g}} \xleftarrow{\$} \mathbb{Z}_q^{KJ_{\mathsf{id}}\times I_{\mathsf{id},\mathfrak{g}}},$$
$$\mathsf{isk}_{\mathsf{id}} \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda, \mathsf{imsk}, \mathsf{id}, \mathbf{V}_{\mathsf{id},\mathfrak{o}}, \mathbf{V}_{\mathsf{id},\mathfrak{g}}),$$
$$\mathsf{ict}_{\mathsf{id},\mathfrak{s}}^* \xleftarrow{\$} \mathsf{Enc}(1^\lambda, \mathsf{impk}, \mathsf{id}, \mathfrak{s}, \mathbf{U}_{\mathsf{id},\mathfrak{s}}^\intercal),$$
$$\mathbf{U}_{\$,\mathsf{id},\mathfrak{s}} \xleftarrow{\$} \mathbb{Z}_q^{Z\times I_{\mathsf{id},\mathfrak{s}}}, \qquad \mathsf{ict}_{\mathsf{id},\mathfrak{s}}^\$ \xleftarrow{\$} \mathsf{Enc}(1^\lambda, \mathsf{impk}, \mathsf{id}, \mathfrak{s}, \mathbf{U}_{\$,\mathsf{id},\mathfrak{s}}^\intercal).$$

$\mathcal{S}$ has *pseudorandom structurally noisy inner products*, denoted $\mathsf{IPFEsec}_{\mathrm{pre}}^{\mathcal{S}}$, if

$$\left\{\begin{pmatrix} 1^\lambda, \\ r_{\mathrm{pub}}, \end{pmatrix}\begin{Bmatrix} \mathbf{U}_{\mathsf{id},\mathfrak{o}}^\intercal\mathbf{V}_{\mathsf{id},\mathfrak{o}} & + \mathbf{E}_{\mathsf{id},\mathfrak{o}}^\intercal, \\ \mathbf{U}_{\mathsf{id},\mathfrak{g}}^\intercal\mathbf{V}_{\mathsf{id},\mathfrak{g}} + \widetilde{\mathbf{g}}^\intercal \otimes \mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{g}}^\intercal & + \mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{o}}^\intercal, \end{Bmatrix}_{\mathsf{id}\in\mathsf{ID}}\right\}_{\lambda\in\mathbb{N}} \approx \left\{\begin{pmatrix} 1^\lambda, \\ r_{\mathrm{pub}}, \end{pmatrix}\begin{Bmatrix} \mathbf{\Delta}_{\mathrm{pre},\mathsf{id},\mathfrak{o}}^\intercal, \\ \mathbf{\Delta}_{\mathrm{pre},\mathsf{id},\mathfrak{g}}^\intercal, \end{Bmatrix}_{\mathsf{id}\in\mathsf{ID}}\right\}_{\lambda\in\mathbb{N}}.$$

The scheme is $\mathcal{S}$*-secure*, denoted $\mathsf{IPFEsec}_{\mathrm{post}}^{\mathcal{S}}$, if

$$\left\{\begin{pmatrix} 1^\lambda, \\ r_{\mathrm{pub}}, \\ \mathsf{impk}, \end{pmatrix}\begin{Bmatrix} \mathsf{ict}_{\mathsf{id},\mathfrak{o}}^*, \\ \mathsf{ict}_{\mathsf{id},\mathfrak{g}}^*, \\ \mathsf{isk}_{\mathsf{id}} \end{Bmatrix}_{\mathsf{id}\in\mathsf{ID}}\right\}_{\lambda\in\mathbb{N}} \approx \left\{\begin{pmatrix} 1^\lambda, \\ r_{\mathrm{pub}}, \\ \mathsf{impk}, \end{pmatrix}\begin{Bmatrix} \mathsf{ict}_{\mathsf{id},\mathfrak{o}}^\$, \\ \mathsf{ict}_{\mathsf{id},\mathfrak{g}}^\$, \\ \mathsf{isk}_{\mathsf{id}} \end{Bmatrix}_{\mathsf{id}\in\mathsf{ID}}\right\}_{\lambda\in\mathbb{N}}.$$

The scheme is $\sigma_{\mathrm{pre},0}$*-secure* if it is $\mathcal{S}$-secure for all efficient $\mathcal{S}$ such that $\sigma_{\mathrm{pre}} \leq \sigma_{\mathrm{pre},0}$ and $\mathsf{IPFEsec}_{\mathrm{pre}}^{\mathcal{S}}$ holds.

Remark 3 (public and private coins). It is necessary to make the sampling of $\mathbf{U}$ private-coin. Since IPFE does not protect $\mathbf{V}$ (the key vectors/matrices), the adversary is assumed to have full knowledge of $\mathbf{V}$. If $\mathbf{U}$ were sampled with public coins, the distinguisher against the ciphertexts (in $\mathsf{IPFEsec}_{\mathrm{post}}$) would know both $\mathbf{V}$ and $\mathbf{U}$, thus $\mathbf{U}^\intercal\mathbf{V}$. In the first distribution (containing ciphertexts of $\mathbf{U}$), decryption will yield the correct output, whereas it is not the case for the second distribution (containing ciphertexts of random vectors).

We formulate $\mathcal{S}_{\mathrm{V}}$ as public-coin and $\mathcal{S}_{\mathrm{U}}$ as fully private-coin, to avoid obvious counterexamples (e.g., obfuscation-based).

3.1 Basic Construction — Single-Identity, No Structured Noises

We first present a simple candidate for one identity without structured noises. The scheme is reminiscent to many existing IPFE constructions [ABDP15,ALS16,Wee17, Agr19,AP20,LL20a,LL20b].

Ingredients of Construction 1. We rely on Lemma 3 for correctness. The security of the scheme is elaborated in Sections 3.2 and 3.3.

Construction 1 (evIPFE)**.** Our scheme works as follows.

- $\mathsf{Setup}(q, 1^K, 1^Z)$ takes as input q, K, Z. If $K \neq 0$, it sets $\mathcal{I}_{q,K,Z} = \emptyset$ and terminates (aborting case). Otherwise, the algorithm sets $\mathcal{I}_{q,K,Z} = \{1\}$, and *deterministically* picks n, m and $\sigma_{-1} \geq \sigma_0(n, m)$ appropriate for Lemma 3, as well as $\sigma_{\mathrm{post}}, \kappa$. It samples and sets

$$(\mathbf{B}, \tau) \xleftarrow{\$} \mathsf{TrapGen}(1^n, 1^m, q), \qquad \mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times Z\lceil \log_2 q \rceil},$$
$$\mathsf{impk} = (1^n, 1^m, q, 1^Z, \sigma_{-1}, \sigma_{\mathrm{post}}, 1^\kappa, \mathbf{A}, \mathbf{B}), \qquad \mathsf{imsk} = (\mathsf{impk}, \tau).$$

  The algorithm outputs $(\mathsf{impk}, \mathsf{imsk})$.

- $\mathsf{KeyGen}(\mathsf{imsk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g})$ takes as input imsk, $\mathsf{id} = 1$, $\mathbf{v}_\mathfrak{o} \in \mathbb{Z}_q^Z$, and $\mathbf{V}_\mathfrak{g} = \perp$. It runs

$$\mathbf{k} \xleftarrow{\$} \mathsf{SampleD}(\mathbf{B}, \tau, \mathbf{A}\mathbf{G}^{-1}(\mathbf{v}_\mathfrak{o}), \sigma_{-1}).$$

  Here, $\mathbf{G}$ is of shape $Z \times Z\lceil \log_2 q \rceil$. The algorithm outputs $\mathsf{isk} = \mathbf{k}$.

- $\mathsf{Enc}(\mathsf{impk}, \mathsf{id}, \mathfrak{s}, \mathbf{u}^\intercal)$ takes as input impk, $\mathsf{id} = 1$, $\mathfrak{s} \in \{\mathfrak{o}, \mathfrak{g}\}$, and $\mathbf{u} \in \mathbb{Z}_q^Z$. If $\mathfrak{s} = \mathfrak{g}$, the algorithm outputs $\perp$ and terminates (the scheme does not produce any key capable of decrypting such a ciphertext). Otherwise, it samples

$$\mathbf{d} \xleftarrow{\$} \mathbb{Z}_q^n, \quad \mathbf{e}_1 \xleftarrow{\$} \mathcal{D}^m_{\mathbb{Z}, \sigma_{\mathrm{post}}, \leq \sigma_{\mathrm{post}}\sqrt{\kappa}}, \quad \mathbf{e}_0 \xleftarrow{\$} \mathcal{D}^{Z\lceil \log_2 q \rceil}_{\mathbb{Z}, \sigma_{\mathrm{post}}, \leq \sigma_{\mathrm{post}}\sqrt{\kappa}}.$$

  The algorithm outputs $\mathsf{ict} = (\mathbf{d}^\intercal\mathbf{B} + \mathbf{e}_1^\intercal,\ \mathbf{d}^\intercal\mathbf{A} + \mathbf{e}_0^\intercal + \mathbf{u}^\intercal\mathbf{G})$.

- $\mathsf{Dec}(\mathsf{impk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g}, \mathsf{isk}, \mathfrak{s}, \mathsf{ict})$ computes and outputs

$$\mathsf{ict} \cdot \begin{pmatrix} -\mathsf{isk} \\ \mathbf{G}^{-1}(\mathbf{v}_\mathfrak{o}) \end{pmatrix}.$$

Correctness. By Lemma 3, we have $\mathbf{B}\mathbf{k} = \mathbf{A}\mathbf{G}^{-1}(\mathbf{v}_\mathfrak{o})$ and $\|\mathbf{k}\| \leq \sigma_{-1}\sqrt{m}$, so

$$\begin{aligned}
\mathsf{ict} \cdot \begin{pmatrix} -\mathsf{isk} \\ \mathbf{G}^{-1}(\mathbf{v}_\mathfrak{o}) \end{pmatrix} - \mathbf{u}^\intercal\mathbf{v}_\mathfrak{o} &= \left(\mathbf{d}^\intercal\mathbf{B} + \mathbf{e}_1^\intercal,\ \mathbf{d}^\intercal\mathbf{A} + \mathbf{e}_0^\intercal + \mathbf{u}^\intercal\mathbf{G}\right) \begin{pmatrix} -\mathbf{k} \\ \mathbf{G}^{-1}(\mathbf{v}_\mathfrak{o}) \end{pmatrix} - \mathbf{u}^\intercal\mathbf{v}_\mathfrak{o} \\
&= -\mathbf{e}_1^\intercal\mathbf{k} + \mathbf{e}_0^\intercal\mathbf{G}^{-1}(\mathbf{v}_\mathfrak{o}), \\
|-\mathbf{e}_1^\intercal\mathbf{k} + \mathbf{e}_0^\intercal\mathbf{G}^{-1}(\mathbf{v}_\mathfrak{o})| &\leq m \cdot \|\mathbf{e}_1\| \cdot \|\mathbf{k}\| + Z\lceil \log_2 q \rceil \cdot \|\mathbf{e}_0\| \cdot \|\mathbf{G}^{-1}(\mathbf{v}_\mathfrak{o})\| \\
&\leq \kappa^{1/2}\sigma_{\mathrm{post}}(m^{3/2}\sigma_{-1} + Z\lceil \log_2 q \rceil).
\end{aligned}$$

Let B be the right-hand side, then Construction 1 is B-correct.

3.2 Security as an Assumption

Inspired by the evasive LWE assumption, we propose a new assumption that is closely related and translates to the security of Construction 1.

The precondition of evasive LWE rules out a combination of two attacking strategies [Wee22], attacks against LWE and zeroizing attacks. Alternatively, the evasive LWE assumption can be seen as a belief that *i)* the only meaningful way of using $\mathbf{B}^{-1}(\mathbf{P})$ is to multiply it to a noisy version of $\mathbf{d}^\intercal\mathbf{B}$, and that *ii)* the skewed noise attached to $\mathbf{d}^\intercal\mathbf{P}$ in $(\mathbf{d}^\intercal\mathbf{B}+\mathbf{e}^\intercal)\cdot\mathbf{B}^{-1}(\mathbf{P})$ does not give adversaries more advantage as long as there is *no apparent relation* among quantities related to $\mathbf{d}$ when the noises are not skewed. The pseudorandomness in the precondition of evasive LWE exactly captures the absence of apparent relation.

Similar to the belief captured by evasive LWE, we propose the following:

Assumption 5 (evasive IPFE)**.** Let $\mathcal{S}=(\mathcal{S}_{\mathrm{V}},\mathcal{S}_{\mathrm{U}})$ be two algorithms with the following syntax.

- $\mathcal{S}_{\mathrm{V}}(1^\lambda;r_{\mathrm{pub}})$, given randomness r_{pub}, outputs

$$1^n,1^m,q,1^Z,1^J,1^I,\qquad \sigma_{-1},\qquad \sigma_{\mathrm{post}}\geq\sigma_{\mathrm{pre}}\geq 0,\qquad \mathbf{V}\in\mathbb{Z}_q^{Z\times J},$$

  where $m\geq m_0(n,q)$ and $\sigma_{-1}\geq\sigma_0(n,m)$ are constrained by Lemma 3.

- $\mathcal{S}_{\mathrm{U}}(1^\lambda,r_{\mathrm{pub}};r_{\mathrm{priv}})$, given r_{pub} and additional randomness r_{priv}, outputs $\mathbf{U}^\intercal$, where $\mathbf{U}\in\mathbb{Z}_q^{Z\times I}$ and q,Z,I agree with the output of $\mathcal{S}_{\mathrm{V}}(1^\lambda;r_{\mathrm{pub}})$.

Suppose

$$\begin{aligned}
(\mathbf{B},\tau)&\xleftarrow{\$}\mathsf{TrapGen}(1^n,1^m,q), & \mathbf{A}&\xleftarrow{\$}\mathbb{Z}_q^{n\times Z\lceil\log_2 q\rceil}, & \mathbf{D}&\xleftarrow{\$}\mathbb{Z}_q^{n\times I},\\
\boldsymbol{\Delta}_0&\xleftarrow{\$}\mathbb{Z}_q^{Z\lceil\log_2 q\rceil\times I}, & \boldsymbol{\Delta}_1&\xleftarrow{\$}\mathbb{Z}_q^{m\times I}, & \boldsymbol{\Delta}_2&\xleftarrow{\$}\mathbb{Z}_q^{J\times I},\\
\mathbf{E}_{\mathrm{pre},0}&\xleftarrow{\$}\mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{pre}}}^{Z\lceil\log_2 q\rceil\times I}, & \mathbf{E}_{\mathrm{pre},1}&\xleftarrow{\$}\mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{pre}}}^{m\times I}, & \mathbf{E}_{\mathrm{pre},2}&\xleftarrow{\$}\mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{pre}}}^{J\times I},\\
\mathbf{E}_{\mathrm{post},0}&\xleftarrow{\$}\mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{post}}}^{Z\lceil\log_2 q\rceil\times I}, & \mathbf{E}_{\mathrm{post},1}&\xleftarrow{\$}\mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{post}}}^{m\times I},\\
\mathbf{K}&\xleftarrow{\$}\mathsf{SampleD}(\mathbf{B},\tau,\mathbf{A}\mathbf{G}^{-1}(\mathbf{V}),\sigma_{-1}).
\end{aligned}$$

The *inner-product* precondition $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{ipre}}$ is

$$\{(1^\lambda,r_{\mathrm{pub}},\mathbf{U}^\intercal\mathbf{V}+\mathbf{E}_{\mathrm{pre},2}^\intercal)\}_{\lambda\in\mathbb{N}}\approx\{(1^\lambda,r_{\mathrm{pub}},\boldsymbol{\Delta}_2^\intercal)\}_{\lambda\in\mathbb{N}}.$$

The *stricter* precondition $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{spre}}$ is

$$\begin{aligned}
&\{(1^\lambda,r_{\mathrm{pub}},\mathbf{A},\mathbf{B},\mathbf{D}^\intercal\mathbf{A}\mathbf{G}^{-1}(\mathbf{V})+\mathbf{E}_{\mathrm{pre},2}^\intercal,\mathbf{D}^\intercal\mathbf{B}+\mathbf{E}_{\mathrm{pre},1}^\intercal,\mathbf{D}^\intercal\mathbf{A}+\mathbf{E}_{\mathrm{pre},0}^\intercal+\mathbf{U}^\intercal\mathbf{G})\}_{\lambda\in\mathbb{N}}\\
\approx\{(1^\lambda,r_{\mathrm{pub}},\mathbf{A},\mathbf{B},&\quad\boldsymbol{\Delta}_2^\intercal\quad,\quad\boldsymbol{\Delta}_1^\intercal\quad,\quad\boldsymbol{\Delta}_0^\intercal\quad)\}_{\lambda\in\mathbb{N}}.
\end{aligned}$$

The postcondition $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{post}}$ is

$$\begin{aligned}
&\{(1^\lambda,r_{\mathrm{pub}},\mathbf{A},\mathbf{B},\mathbf{K},\mathbf{D}^\intercal\mathbf{B}+\mathbf{E}_{\mathrm{post},1}^\intercal,\mathbf{D}^\intercal\mathbf{A}+\mathbf{E}_{\mathrm{post},0}^\intercal+\mathbf{U}^\intercal\mathbf{G})\}_{\lambda\in\mathbb{N}}\\
\approx\{(1^\lambda,r_{\mathrm{pub}},\mathbf{A},\mathbf{B},\mathbf{K},&\quad\boldsymbol{\Delta}_1^\intercal\quad,\quad\boldsymbol{\Delta}_0^\intercal\quad)\}_{\lambda\in\mathbb{N}}.
\end{aligned}$$

The $(n_0, \sigma_{\mathrm{post},0})$-*evasive IPFE assumption* states that $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{ipre}}$ implies $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{post}}$ for all efficient $\mathcal{S}$ such that $n \geq n_0$ and $\sigma_{\mathrm{post}} \geq \sigma_{\mathrm{post},0}$. The *conservative evasive IPFE assumption* states that $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{spre}}$ implies $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{post}}$ for all efficient $\mathcal{S}$.[2]

The rationale of public and private coins in the above formulation is similar to that (Remark 3) of evasive IPFE security definition.

Evasive IPFE Security from Evasive IPFE Assumption. There are only minor syntactical differences between the samplers of evasive IPFE (Assumption 5) and IPFEsec (Definition 5) instantiated for Construction 1. The evasive IPFE assumption implies the security of our basic scheme:

Theorem 5 (¶)**.** *Construction 1 is σ_{post}-secure (Definition 5) under $(n, \sigma_{\mathrm{post}})$-evasive IPFE (Assumption 5), where $n, \sigma_{\mathrm{post}}$ are those picked by* Setup.

Proof (Theorem 5). Let $\mathcal{S} = (\mathcal{S}_{\mathrm{V}}, \mathcal{S}_{\mathrm{U}})$ be an efficient sampler (Definition 5) satisfying $\sigma_{\mathrm{pre}} \leq \sigma_{\mathrm{post}}$ and $\mathsf{IPFEsec}^{\mathcal{S}}_{\mathrm{pre}}$. Consider the following $\mathcal{S}' = (\mathcal{S}'_{\mathrm{V}}, \mathcal{S}'_{\mathrm{U}})$ for Assumption 5.

- $\mathcal{S}'_{\mathrm{V}}(r_{\mathrm{pub}})$ runs

  $$(q, 1^0, 1^Z, \{1\}, 1^{J_1}, 1^{I_{1,\mathfrak{o}}}, 1^0, \sigma_{\mathrm{pre}}, \mathbf{V}_{1,\mathfrak{o}}, \bot) \leftarrow \mathcal{S}_{\mathrm{V}}(r_{\mathrm{pub}}),$$

  picks $n, m, \sigma_{-1}, \sigma_{\mathrm{post}}$ according to Construction 1, sets $J, I, \mathbf{V}$ to $J_1, I_{1,\mathfrak{o}}, \mathbf{V}_{1,\mathfrak{o}}$, and outputs[3]

  $$1^n, 1^m, q, 1^Z, 1^J, 1^I, \qquad \sigma_{-1}, \qquad \sigma_{\mathrm{post}}, \sigma_{\mathrm{pre}}, \qquad \mathbf{V}.$$

- $\mathcal{S}'_{\mathrm{U}}(r_{\mathrm{pub}}; r_{\mathrm{priv}})$ runs $(\mathbf{U}^{\intercal}_{1,\mathfrak{o}}, \bot) \leftarrow \mathcal{S}_{\mathrm{U}}(r_{\mathrm{pub}}; r_{\mathrm{priv}})$ and outputs $\mathbf{U}^{\intercal} = \mathbf{U}^{\intercal}_{1,\mathfrak{o}}$.

Clearly, $\mathsf{evIPFE}^{\mathcal{S}'}_{\mathrm{ipre}}$ is simply $\mathsf{IPFEsec}^{\mathcal{S}}_{\mathrm{pre}}$, which is assumed to hold. Therefore, by the $(n, \sigma_{\mathrm{post}})$-evasive IPFE assumption, $\mathsf{evIPFE}^{\mathcal{S}'}_{\mathrm{post}}$ holds, implying

$$\begin{aligned}
&(r_{\mathrm{pub}}, \mathbf{A}, \mathbf{B}, \mathbf{K}, \mathbf{D}^{\intercal}\mathbf{B} + \mathbf{E}^{\intercal}_{\mathrm{post},1}, \mathbf{D}^{\intercal}\mathbf{A} + \mathbf{E}^{\intercal}_{\mathrm{post},0} + \mathbf{U}^{\intercal}\mathbf{G}) \\
\approx\ &(r_{\mathrm{pub}}, \mathbf{A}, \mathbf{B}, \mathbf{K}, \quad \boldsymbol{\Delta}^{\intercal}_1 \quad , \quad \boldsymbol{\Delta}^{\intercal}_0) \\
\equiv\ &(r_{\mathrm{pub}}, \mathbf{A}, \mathbf{B}, \mathbf{K}, \quad \boldsymbol{\Delta}^{\intercal}_1 \quad , \quad \boldsymbol{\Delta}^{\intercal}_0 + \mathbf{U}^{\intercal}_{\$}\mathbf{G}) \\
\approx\ &(r_{\mathrm{pub}}, \mathbf{A}, \mathbf{B}, \mathbf{K}, \mathbf{D}^{\intercal}\mathbf{B} + \mathbf{E}^{\intercal}_{\mathrm{post},1}, \mathbf{D}^{\intercal}\mathbf{A} + \mathbf{E}^{\intercal}_{\mathrm{post},0} + (\mathbf{U} + \mathbf{U}_{\$})^{\intercal}\mathbf{G}) \\
\equiv\ &(r_{\mathrm{pub}}, \mathbf{A}, \mathbf{B}, \mathbf{K}, \mathbf{D}^{\intercal}\mathbf{B} + \mathbf{E}^{\intercal}_{\mathrm{post},1}, \mathbf{D}^{\intercal}\mathbf{A} + \mathbf{E}^{\intercal}_{\mathrm{post},0} + \mathbf{U}^{\intercal}_{\$}\mathbf{G}),
\end{aligned}$$

where $\mathbf{U}_{\$} = \mathbf{U}_{\$,1,\mathfrak{o}} \xleftarrow{\$} \mathbb{Z}_q^{Z \times I}$. Opening up $\mathsf{IPFEsec}^{\mathcal{S}}_{\mathrm{post}}$ for Construction 1, we see that

$$\underbrace{r_{\mathrm{pub}}}_{\mathcal{S}} = \underbrace{r_{\mathrm{pub}}}_{\mathcal{S}'}, \qquad \begin{aligned}
\mathsf{impk} &= (\overbrace{1^n, 1^m, q, 1^Z, \sigma_{-1}, \sigma_{\mathrm{post}}, 1^\kappa}^{\text{efficiently computable from } r_{\mathrm{pub}}}, \mathbf{A}, \mathbf{B}), \qquad \mathsf{isk}_1 = \mathbf{K}, \\
\mathsf{ict}^{*}_{1,\mathfrak{o}} &= (\mathbf{D}^{\intercal}\mathbf{B} + \mathbf{E}^{\intercal}_{\mathrm{post},1}, \mathbf{D}^{\intercal}\mathbf{A} + \mathbf{E}^{\intercal}_{\mathrm{post},0} + \mathbf{U}^{\intercal}\mathbf{G}), \\
\mathsf{ict}^{\$}_{1,\mathfrak{o}} &= (\mathbf{D}^{\intercal}\mathbf{B} + \mathbf{E}^{\intercal}_{\mathrm{post},1}, \mathbf{D}^{\intercal}\mathbf{A} + \mathbf{E}^{\intercal}_{\mathrm{post},0} + \mathbf{U}^{\intercal}_{\$}\mathbf{G}),
\end{aligned}$$

[2] For the first part, explicitly requiring lower bounds on $n, \sigma_{\mathrm{post}}$ is necessary, because $\mathsf{evIPFE}_{\mathrm{ipre}}$ could be unconditional when $J < Z$ yet $\mathsf{evIPFE}_{\mathrm{post}}$ is always computational. For the second part, $\mathsf{evIPFE}_{\mathrm{spre}}$ implies LWE, so the lower bounds on $n, \sigma_{\mathrm{post}}$ are implicit.

[3] Without loss of generality, we may assume that $\mathcal{S}_{\mathrm{V}}$ always outputs $\mathsf{ID} = \{1\}$ and $I_{1,\mathfrak{g}} = 0$. The other cases can be converted to $\mathsf{ID} = \{1\}$ and $I_{1,\mathfrak{o}} = J_1 = 0$, with $\mathbf{U}^{\intercal}_{1,\mathfrak{g}}$ discarded, or both. The quantities $n, m, \sigma_{-1}, \sigma_{\mathrm{post}}$ are determined by q, Z, so $\mathcal{S}'_{\mathrm{V}}$ does not rely on the randomness of Setup of Construction 1.

except for noise truncation only up to a negligible statistical error by Lemma 1. Therefore, $\mathsf{IPFEsec}^{\mathcal{S}}_{\mathrm{post}}$ follows from the previous indistinguishability. □

Full and Conservative Versions. The evasive IPFE assumption might appear custom-made for the scheme, rendering the security proof almost tautological, and it looks far-fetched from and bears little resemblance to the evasive LWE assumption. We justify it below.

Intuitively, the only meaningful way of using $\mathbf{B}^{-1}(\mathbf{A}\mathbf{G}^{-1}(\mathbf{V}))$ is to compute

$$-\underset{\sim}{\mathbf{D}^\intercal\mathbf{B}} \cdot \mathbf{B}^{-1}(\mathbf{A}\mathbf{G}^{-1}(\mathbf{V})) + (\underset{\sim}{\mathbf{D}^\intercal\mathbf{A}} + \mathbf{U}^\intercal\mathbf{G})\mathbf{G}^{-1}(\mathbf{V}) = \underset{\sim}{\mathbf{U}^\intercal\mathbf{V}},$$

where wavy underlines indicate noises. In this computation, it appears that *some* cancellation has happened so that $\mathbf{U}^\intercal\mathbf{V}$ is approximately recovered, which is different from evasive LWE, where *no* cancellation happens. But the precondition requires that the inner products with noise be pseudorandom, i.e., the "cancellation" only leads to pseudorandom results, morally equivalent to that there is no *actual* cancellation. It excludes the obvious zeroizing attacks.

The *conservative* evasive IPFE assumption is more along the lines of the evasive LWE assumption, for which the only addition is the ability to combine privately sampled value $\mathbf{U}$ with LWE sample $\mathbf{D}^\intercal\mathbf{A}$. It appears to be a smaller leap of faith than the full version, yet this is actually superfluous.

In fact, the full version does not demand more than the conservative version in suitable parameter regimes. We show that $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{ipre}}$ implies $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{spre}}$ under the LWE assumption with noise super-polynomially smaller than σ_{pre}:

Theorem 6 (¶). *If $\mathcal{S}$ (Assumption 5) is efficient and $\mathsf{LWE}_{n,m+Z\lceil\log_2 q\rceil,q,\sigma_{\mathrm{help}}}$ (Assumption 1) holds for some $\sigma_{\mathrm{help}} \le \frac{\sigma_{\mathrm{pre}}}{2^{\kappa+6}\sqrt{\kappa}\cdot Z\lceil\log_2 q\rceil}$, where $q, Z, \sigma_{\mathrm{pre}}$ are those picked by $\mathcal{S}$, then $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{ipre}}$ implies $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{spre}}$.*

We obtain a conservative version of Theorem 5 as a corollary.

Corollary 7 (¶). *Construction 1 is σ_{post}-secure (Definition 5) under* conservative *evasive IPFE (Assumption 5) and $\mathsf{LWE}_{n,m+Z\lceil\log_2 q\rceil,q,\sigma_{\mathrm{help}}}$ (Assumption 1) for some $\sigma_{\mathrm{help}} \le \frac{\sigma_{\mathrm{post}}}{(2^{\kappa+6}\sqrt{\kappa})^2 Z\lceil\log_2 q\rceil}$, where q, Z are those picked by $\mathcal{S}$ and $n, m, \sigma_{\mathrm{post}}, \kappa$ are those picked by* Setup.

The idea for the proof of Theorem 6 is

$$\begin{aligned}
& \left(\underset{\sim}{\mathbf{D}^\intercal\mathbf{A}\mathbf{G}^{-1}(\mathbf{V})}, \underset{\sim}{\mathbf{D}^\intercal\mathbf{B}}, \underset{\sim}{\mathbf{D}^\intercal\mathbf{A}} + \mathbf{U}^\intercal\mathbf{G}\right) \\
\text{(noise flooding)} \quad \approx_{\mathrm{s}} & \left((\underset{\sim}{\mathbf{D}^\intercal\mathbf{A}} + \mathbf{U}^\intercal\mathbf{G})\mathbf{G}^{-1}(\mathbf{V}) - \underset{\sim}{\mathbf{U}^\intercal\mathbf{V}}, \underset{\sim}{\mathbf{D}^\intercal\mathbf{B}}, \underset{\sim}{\mathbf{D}^\intercal\mathbf{A}} + \mathbf{U}^\intercal\mathbf{G}\right) \\
\text{(LWE)} \quad \approx & \begin{pmatrix} \mathbf{\Delta}_0^\intercal\mathbf{G}^{-1}(\mathbf{V}) - \underset{\sim}{\mathbf{U}^\intercal\mathbf{V}}, & \mathbf{\Delta}_1^\intercal, & \mathbf{\Delta}_0^\intercal \end{pmatrix} \\
\text{(precondition)} \quad \approx & \begin{pmatrix} \mathbf{\Delta}_2^\intercal, & \mathbf{\Delta}_1^\intercal, & \mathbf{\Delta}_0^\intercal \end{pmatrix},
\end{aligned}$$

where wavy underlines indicate noises. The rewriting in the first step is essentially IPFE simulation [Wee17,ACF+18,ALMT20,LL20b], i.e., $\underset{\sim}{\mathbf{D}^\intercal\mathbf{A}\mathbf{G}^{-1}(\mathbf{V})}$ contains exactly the information of $\underset{\sim}{\mathbf{U}^\intercal\mathbf{V}}$, when given $\underset{\sim}{\mathbf{D}^\intercal\mathbf{B}}$ and $(\underset{\sim}{\mathbf{D}^\intercal\mathbf{A}} + \mathbf{U}^\intercal\mathbf{G})$.

Proof (Theorem 6). We consider the following hybrids.

- H_0 is the first distribution in $\mathsf{evIPFE}^{\mathcal{S}}_{\mathsf{spre}}$, i.e.,

$$r_{\mathsf{pub}}, \mathbf{A}, \mathbf{B}, \quad \mathbf{D}^\intercal\mathbf{A}\mathbf{G}^{-1}(\mathbf{V}) + \mathbf{E}^\intercal_{\mathsf{pre},2},$$
$$\mathbf{D}^\intercal\mathbf{B} + \mathbf{E}^\intercal_{\mathsf{pre},1}, \quad \mathbf{D}^\intercal\mathbf{A} + \mathbf{E}^\intercal_{\mathsf{pre},0} + \mathbf{U}^\intercal\mathbf{G}.$$

- In H_1, the matrix $\mathbf{B}$ is sampled uniformly at random, without a known trapdoor. By Lemma 3, we have $\mathsf{H}_0 \approx_{\mathsf{s}} \mathsf{H}_1$.

- In H_2, helper noises are inserted and flooded by $\mathbf{E}_{\mathsf{pre}}$'s, i.e.,

$$r_{\mathsf{pub}}, \mathbf{A}, \mathbf{B}, \quad \mathbf{D}^\intercal\mathbf{A}\mathbf{G}^{-1}(\mathbf{V}) + (\mathbf{E}^\intercal_{\mathsf{pre},2} + \mathbf{E}^\intercal_{\mathsf{help},0}\mathbf{G}^{-1}(\mathbf{V})),$$
$$\mathbf{D}^\intercal\mathbf{B} + (\mathbf{E}_{\mathsf{pre},1} + \mathbf{E}_{\mathsf{help},1})^\intercal, \quad \mathbf{D}^\intercal\mathbf{A} + \mathbf{U}^\intercal\mathbf{G} + (\mathbf{E}_{\mathsf{pre},0} + \mathbf{E}_{\mathsf{help},0})^\intercal,$$

  where the entries of $\mathbf{E}_{\mathsf{help}}$'s are sampled independently from $\mathcal{D}_{\mathbb{Z},\sigma_{\mathsf{help}},\le\sigma_{\mathsf{help}}\sqrt{\kappa}}$. Since $\|(\mathbf{G}^{-1}(\mathbf{V}))^\intercal\| \le Z\lceil\log_2 q\rceil$, the entries of ($\mathbf{E}_{\mathsf{help}}$'s and) $\mathbf{E}^\intercal_{\mathsf{help},0}\mathbf{G}^{-1}(\mathbf{V})$ are bounded by $\sigma_{\mathsf{help}}\sqrt{\kappa}\cdot Z\lceil\log_2 q\rceil$. Together with $\sigma_{\mathsf{pre}} \ge 2^{\kappa+6}\sigma_{\mathsf{help}}\sqrt{\kappa}\cdot Z\lceil\log_2 q\rceil$, it follows from Lemma 2 that $\mathsf{H}_1 \approx_{\mathsf{s}} \mathsf{H}_2$.

- In H_3, we change $\mathbf{E}_{\mathsf{pre},2}$ to $-\mathbf{E}_{\mathsf{pre},2}$, rearrange some terms, and derive the noisy $\mathbf{D}^\intercal\mathbf{A}\mathbf{G}^{-1}(\mathbf{V})$ from the IPFE decryption relation, i.e.,

$$r_{\mathsf{pub}}, \mathbf{A}, \mathbf{B}, \quad (\mathbf{D}^\intercal\mathbf{A} + \mathbf{E}^\intercal_{\mathsf{help},0} + \mathbf{U}^\intercal\mathbf{G})\mathbf{G}^{-1}(\mathbf{V}) - (\mathbf{U}^\intercal\mathbf{V} + \mathbf{E}^\intercal_{\mathsf{pre},2}),$$
$$(\mathbf{D}^\intercal\mathbf{B} + \mathbf{E}^\intercal_{\mathsf{help},1}) + \mathbf{E}^\intercal_{\mathsf{pre},1}, \quad (\mathbf{D}^\intercal\mathbf{A} + \mathbf{E}^\intercal_{\mathsf{help},0} + \mathbf{U}^\intercal\mathbf{G}) + \mathbf{E}^\intercal_{\mathsf{pre},0}.$$

  Since $\mathbf{E}_{\mathsf{pre},2}$ follows a symmetric distribution, $\mathsf{H}_2 \equiv \mathsf{H}_3$.

- In H_4, the $\mathbf{E}_{\mathsf{help}}$'s are no longer truncated, i.e., their entries are sampled independently from $\mathcal{D}_{\mathbb{Z},\sigma_{\mathsf{help}}}$. By Lemma 1, it holds that $\mathsf{H}_3 \approx_{\mathsf{s}} \mathsf{H}_4$.

- In H_5, we invoke $\mathsf{LWE}_{n,m+Z\lceil\log_2 q\rceil,q,\sigma_{\mathsf{help}}}$ for secrets[4] $\mathbf{D}$ and public matrix $(\mathbf{B}, \mathbf{A})$, and the distribution becomes

$$r_{\mathsf{pub}}, \mathbf{A}, \mathbf{B}, \quad (\boldsymbol{\Delta}^\intercal_0 + \mathbf{U}^\intercal\mathbf{G})\mathbf{G}^{-1}(\mathbf{V}) - (\mathbf{U}^\intercal\mathbf{V} + \mathbf{E}^\intercal_{\mathsf{pre},2}),$$
$$\boldsymbol{\Delta}^\intercal_1 + \mathbf{E}^\intercal_{\mathsf{pre},1}, \quad (\boldsymbol{\Delta}^\intercal_0 + \mathbf{U}^\intercal\mathbf{G}) + \mathbf{E}^\intercal_{\mathsf{pre},0}.$$

  By the LWE assumption, $\mathsf{H}_4 \approx \mathsf{H}_5$.[5]

- In H_6, we make $\boldsymbol{\Delta}$'s absorb terms added to them, i.e.,

$$r_{\mathsf{pub}}, \mathbf{A}, \mathbf{B}, \quad (\boldsymbol{\Delta}^\intercal_0 - \mathbf{E}^\intercal_{\mathsf{pre},0})\mathbf{G}^{-1}(\mathbf{V}) - (\mathbf{U}^\intercal\mathbf{V} + \mathbf{E}^\intercal_{\mathsf{pre},2}), \quad \boldsymbol{\Delta}^\intercal_1, \quad \boldsymbol{\Delta}^\intercal_0.$$

  Clearly, $\mathsf{H}_5 \equiv \mathsf{H}_6$.

- In H_7, we invoke $\mathsf{evIPFE}^{\mathcal{S}}_{\mathsf{ipre}}$ so the distribution becomes

$$r_{\mathsf{pub}}, \mathbf{A}, \mathbf{B}, \quad (\boldsymbol{\Delta}^\intercal_0 - \mathbf{E}^\intercal_{\mathsf{pre},0})\mathbf{G}^{-1}(\mathbf{V}) - \boldsymbol{\Delta}^\intercal_2, \quad \boldsymbol{\Delta}^\intercal_1, \quad \boldsymbol{\Delta}^\intercal_0.$$

  By the assumption, $\mathsf{H}_6 \approx \mathsf{H}_7$.

[4] Technically, there are a series of hybrids over the columns of $\mathbf{D}$.
[5] This step requires that $\mathcal{S}$ (part of the reduction algorithm) be efficient.

- H_8 is the second distribution in $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{spre}}$, i.e.,

$$r_{\mathrm{pub}}, \mathbf{A}, \mathbf{B}, \qquad \boldsymbol{\Delta}_2^\intercal, \qquad\qquad\qquad\qquad \boldsymbol{\Delta}_1^\intercal, \qquad \boldsymbol{\Delta}_0^\intercal.$$

  Clearly, $\mathsf{H}_7 \equiv \mathsf{H}_8$.

By a hybrid argument, $\mathsf{H}_0 \approx \mathsf{H}_8$, i.e., $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{spre}}$ holds. □

Proof (Corollary 7). Given an efficient $\mathcal{S}$ (Definition 5) satisfying $\sigma_{\mathrm{pre}} \leq \sigma_{\mathrm{post}}$ and $\mathsf{IPFEsec}^{\mathcal{S}}_{\mathrm{pre}}$, we do the following:

1. construct $\mathcal{S}'$ (Assumption 5) *similarly* to the proof of Theorem 5 and show that $\mathsf{evIPFE}^{\mathcal{S}'}_{\mathrm{ipre}}$ holds,
2. apply Theorem 6 to $\mathcal{S}'$ for proving $\mathsf{evIPFE}^{\mathcal{S}'}_{\mathrm{spre}}$,
3. derive $\mathsf{evIPFE}^{\mathcal{S}'}_{\mathrm{post}}$ from the conservative evasive IPFE assumption, and
4. conclude $\mathsf{IPFEsec}^{\mathcal{S}}_{\mathrm{post}}$ in the same way as the final parts of the proof of Theorem 5.

Let $\sigma^{\mathcal{S}}_{\mathrm{pre}}, \sigma^{\mathcal{S}'}_{\mathrm{pre}}$ be the σ_{pre} of $\mathcal{S}, \mathcal{S}'$. We set

$$\sigma^{\mathcal{S}'}_{\mathrm{pre}} = \begin{cases} \sigma^{\mathcal{S}}_{\mathrm{pre}}, & \text{if } \sigma^{\mathcal{S}}_{\mathrm{pre}} \geq \frac{\sigma_{\mathrm{post}}}{2^{\kappa+6}\sqrt{\kappa}}; \\ \sigma_{\mathrm{post}}, & \text{otherwise.} \end{cases}$$

In the first case, $\mathsf{evIPFE}^{\mathcal{S}'}_{\mathrm{ipre}}$ is just $\mathsf{IPFEsec}^{\mathcal{S}}_{\mathrm{pre}}$ and Theorem 6 applies because

$$\sigma_{\mathrm{help}} \leq \frac{\sigma_{\mathrm{post}}}{(2^{\kappa+6}\sqrt{\kappa})^2 Z \lceil \log_2 q \rceil} \leq \frac{\sigma^{\mathcal{S}'}_{\mathrm{pre}}}{2^{\kappa+6}\sqrt{\kappa} \cdot Z \lceil \log_2 q \rceil}.$$

In the second case, we cannot set $\sigma^{\mathcal{S}'}_{\mathrm{pre}}$ to $\sigma^{\mathcal{S}}_{\mathrm{pre}}$, which might not be large enough compared to σ_{help} per the premise of Theorem 6. Instead, with our choice, $\mathsf{evIPFE}^{\mathcal{S}'}_{\mathrm{ipre}}$ follows from $\mathsf{IPFEsec}^{\mathcal{S}}_{\mathrm{pre}}$ by noise truncation and flooding (Lemmas 1 and 2), and Theorem 6 again applies. □

We remark that for security against $\mathcal{S}$ with particular σ_{pre}'s (e.g., $\sigma_{\mathrm{pre}} = \sigma_{\mathrm{post}}$), the bound on σ_{help} can be tuned accordingly (e.g., without the quadratic blow-up).

3.3 Security for Restricted Samplers from Evasive LWE

We prove that Construction 1 is $\mathcal{S}$-secure for a restricted class of samplers under the evasive LWE assumption. These samplers are efficient, satisfy $\mathsf{IPFEsec}^{\mathcal{S}}_{\mathrm{pre}}$, and their $\mathcal{S}_{\mathrm{U}}$'s output a uniformly random vector in a publicly known subspace. This will suffice for our ABE applications.

Definition 6 (restricted IB-evIPFE-w/sn security)**.** A sampler $\mathcal{S} = (\mathcal{S}_{\mathrm{V}}, \mathcal{S}_{\mathrm{U}})$ per Definition 5 is *restricted* if $\mathcal{S}_{\mathrm{U}}(1^\lambda, r_{\mathrm{pub}}; r_{\mathrm{priv}})$ works as follows.

1. It uses a fixed deterministic algorithm $\mathcal{S}_{\mathrm{A0}}$ to compute

$$(1^{n'}, \{\mathbf{A}_{\mathsf{id},\mathfrak{s},0,i}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{v},\mathfrak{g}\},i\in[I_{\mathsf{id},\mathfrak{s}}]}) \leftarrow \mathcal{S}_{\mathrm{A0}}(1^\lambda, r_{\mathrm{pub}}),$$

   where $\mathbf{A}_{\mathsf{id},\mathfrak{s},0,i} \in \mathbb{Z}_q^{n'\times Z}$ for all $\mathsf{id} \in \mathsf{ID}$, $\mathfrak{s} \in \{\mathfrak{v},\mathfrak{g}\}$, $i \in [I_{\mathsf{id},\mathfrak{s}}]$, and the values of q, Z, ID, $\{I_{\mathsf{id},\mathfrak{s}}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{v},\mathfrak{g}\}}$ agree with the output of $\mathcal{S}_{\mathrm{V}}(1^\lambda; r_{\mathrm{pub}})$.

2. It uses r_{priv} to sample $\mathbf{d}_0 \xleftarrow{\$} \mathbb{Z}_q^{n'}$. The algorithm outputs $\{\mathbf{U}_{\mathsf{id},\mathfrak{s}}^\intercal\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{v},\mathfrak{g}\}}$

$$\begin{aligned} &\text{for} && \mathbf{U}_{\mathsf{id},\mathfrak{s}} = (\mathbf{u}_{\mathsf{id},\mathfrak{s},1}, \ldots, \mathbf{u}_{\mathsf{id},\mathfrak{s},I_{\mathsf{id},\mathfrak{s}}}) \\ &\text{with} && \mathbf{u}_{\mathsf{id},\mathfrak{s},i}^\intercal = \mathbf{d}_0^\intercal \mathbf{A}_{\mathsf{id},\mathfrak{s},0,i} \quad \text{for all } \mathsf{id} \in \mathsf{ID},\ \mathfrak{s} \in \{\mathfrak{v},\mathfrak{g}\},\ i \in [I_{\mathsf{id},\mathfrak{s}}]. \end{aligned}$$

An IB-evIPFE-w/sn scheme (Definition 4) is *restricted-$\sigma_{\mathrm{pre},0}$-secure* if it is $\mathcal{S}$-secure (Definition 5) for all efficient restricted $\mathcal{S}$ such that $\sigma_{\mathrm{pre}} \le \sigma_{\mathrm{pre},0}$ and $\mathsf{IPFEsec}_{\mathrm{pre}}^{\mathcal{S}}$ holds.

Parallel to restricted security, we formulate the following:

Assumption 6 (restricted conservative evasive IPFE)**.** A sampler $\mathcal{S} = (\mathcal{S}_{\mathrm{V}}, \mathcal{S}_{\mathrm{U}})$ for Assumption 5 is *restricted* if $\mathcal{S}_{\mathrm{U}}(1^\lambda, r_{\mathrm{pub}}; r_{\mathrm{priv}})$ works as follows.

1. It uses a fixed deterministic algorithm $\mathcal{S}_{\mathrm{A0}}$ to compute

$$(1^{n'}, \{\mathbf{A}'_{0,i}\}_{i\in[I]}) \leftarrow \mathcal{S}_{\mathrm{A0}}(1^\lambda, r_{\mathrm{pub}}) \quad \text{with} \quad \mathbf{A}'_{0,i} \in \mathbb{Z}_q^{n'\times Z} \text{ for all } i \in [I],$$

   where q, Z, I agree with the output of $\mathcal{S}_{\mathrm{V}}(1^\lambda; r_{\mathrm{pub}})$.

2. It uses r_{priv} to sample $\mathbf{d}_0 \xleftarrow{\$} \mathbb{Z}_q^{n'}$. The algorithm outputs $\mathbf{U}^\intercal$ for

$$\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_I) \quad \text{and} \quad \mathbf{u}_i^\intercal = \mathbf{d}_0^\intercal \mathbf{A}'_{0,i} \text{ for all } i \in [I].$$

The *restricted conservative evasive IPFE assumption* states that $\mathsf{evIPFE}_{\mathrm{spre}}^{\mathcal{S}}$ implies $\mathsf{evIPFE}_{\mathrm{post}}^{\mathcal{S}}$ for all efficient restricted $\mathcal{S}$.

Lemma 8. *The sampler (Assumption 5) constructed in the proof of Corollary 7 is restricted (Assumption 6) if so is the original sampler (Definitions 5 and 6).*

Lemma 8 can be easily verified. The following implication holds:

Theorem 9 (¶)**.** *Evasive LWE (Assumption 4) implies restricted conservative evasive IPFE (Assumption 6).*

Combining Theorem 6, the proof of Corollary 7, Lemma 8, and Theorem 9:

Corollary 10. *Construction 1 is restricted-σ_{post}-secure (Definition 6) under evasive LWE (Assumption 4) and $\mathsf{LWE}_{n,m+Z\lceil\log_2 q\rceil,q,\sigma_{\mathrm{help}}}$ (Assumption 1) for some $\sigma_{\mathrm{help}} \le \frac{\sigma_{\mathrm{post}}}{(2^{\kappa+6}\sqrt{\kappa})^2 Z\lceil\log_2 q\rceil}$, where q, Z are those picked by $\mathcal{S}$ and $n, m, \sigma_{\mathrm{post}}, \kappa$ are those picked by* Setup.

Proof (Theorem 9). Let $\mathcal{S} = (\mathcal{S}_{\mathrm{V}}, \mathcal{S}_{\mathrm{U}})$ be an efficient restricted sampler with $\mathcal{S}_{\mathrm{A0}}$ (Assumptions 5 and 6) such that $\mathsf{evIPFE}_{\mathrm{spre}}^{\mathcal{S}}$ holds. Construct the following $\mathcal{S}'(r)$ for Assumption 4. It parses $r = (r_{\mathrm{pub}}, r'_{\mathrm{pub}})$ and runs

$$\begin{aligned} (1^n, 1^m, q, 1^Z, 1^J, 1^I, \sigma_{-1}, \sigma_{\mathrm{post}}, \sigma_{\mathrm{pre}}, \mathbf{V}) &\leftarrow \mathcal{S}_{\mathrm{V}}(r_{\mathrm{pub}}), \\ (1^{n'}, \{\mathbf{A}'_{0,i}\}_{i\in[I]}) &\leftarrow \mathcal{S}_{\mathrm{A0}}(r_{\mathrm{pub}}). \end{aligned}$$

The algorithm samples $\mathbf{A}$ uniformly random over $\mathbb{Z}_q^{n\times Z\lceil\log_2 q\rceil}$ using r'_{pub} in the straightforward manner,[6] and sets $K=0$ and $m'=Z\lceil\log_2 q\rceil$. It outputs

$$1^n, 1^m, 1^K, q, 1^{n'}, 1^J, 1^I, 1^{m'}, \qquad \sigma_{-1}, \quad \sigma_{\mathrm{post}}, \quad \sigma_{\mathrm{pre}},$$
$$\widetilde{\mathbf{P}} = \mathbf{A}\mathbf{G}^{-1}(\mathbf{V}), \quad \{\mathbf{A}_{1,i}, \mathbf{A}_{0,i}\}_{i\in[I]} \quad (\text{with } \mathbf{A}_{1,i}=\mathbf{A} \text{ and } \mathbf{A}_{0,i}=\mathbf{A}'_{0,i}\mathbf{G}),$$

By setting

$$\mathbf{D}=(\mathbf{d}_1,\dots,\mathbf{d}_I), \qquad \boldsymbol{\Delta}_\alpha=(\boldsymbol{\delta}_{1,\alpha},\dots,\boldsymbol{\delta}_{I,\alpha}) \text{ for all } \alpha\in\{0,1,2\},$$
$$\mathbf{E}_{p,\alpha}=(\mathbf{e}_{p,1,\alpha},\dots,\mathbf{e}_{p,I,\alpha}) \text{ for all } (p,\alpha)\in\left\{\begin{matrix}(\mathrm{pre},2),(\mathrm{pre},1),(\mathrm{pre},0),\\ (\mathrm{post},1),(\mathrm{post},0)\end{matrix}\right\},$$

the components in $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{spre/post}}, \mathsf{evLWE}^{\mathcal{S}'}_{\mathrm{pre/post}}$ are perfectly lined up, except that $\mathbf{A}$ corresponds to r'_{pub}. It is readily verified that $\mathsf{evLWE}^{\mathcal{S}'}_{\mathrm{pre}}$ follows from $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{spre}}$. By the evasive LWE assumption, $\mathsf{evLWE}^{\mathcal{S}'}_{\mathrm{post}}$ holds, implying $\mathsf{evIPFE}^{\mathcal{S}}_{\mathrm{post}}$.[7] □

3.4 Identity-Based Scheme

Construction 1 only supports a single identity. We employ pairing-based techniques to generically obtain an exponentially large identity space. The core idea is to generate identity-based keys and ciphertexts using a single-identity IPFE (denoted with primes), e.g., when not considering structured noises,

$$\left.\begin{matrix}\mathsf{isk}_{\mathsf{id}'}(\mathbf{v}):\ \mathsf{isk}'(\ \mathbf{v}, & \psi, \mathsf{id}'\cdot\psi\ )\\ \mathsf{ict}_{\mathsf{id}}(\mathbf{u}):\ \mathsf{ict}'(\ \mathbf{u}, \mathsf{id}\cdot\varphi, & -\varphi\ )\end{matrix}\right\} \xrightarrow{\mathsf{Dec}'} \mathbf{u}^\intercal\mathbf{v}+(\mathsf{id}-\mathsf{id}')\varphi\psi$$

so that decryption yields $\mathbf{u}^\intercal\mathbf{v}$ if $\mathsf{id}=\mathsf{id}'$ and the result is masked by a blinding factor $(\mathsf{id}-\mathsf{id}')\varphi\psi$ if $\mathsf{id}\neq\mathsf{id}'$. Intuitively, the masks are pseudorandom by the DDH assumption (this idea can be materialized in various ways). In our adaptation, we use the LWE assumption to create the pseudorandom blinding factors.

We note that the transformation of [GKW16] compressing exponentially many copies of a single-identity scheme into an identity-based scheme is not as desirable as the upcoming construction. Security of evIPFE(-w/sn) is knowledge-type. Since Definition 5 requires fully private-coin sampling of plaintext vectors (see Remark 3), it cannot be applied simultaneously for two copies unless the two copies only encrypt independent plaintext vectors — but in our applications, plaintext vectors under different identities are correlated. Moreover, even if we consider an alternative definition with arbitrary (efficiently computable) auxiliary information about the plaintext vectors, we would have to assume evasive LWE with very tight pre/postcondition distinguishing gap for it to be applied super-constantly many times (corresponding to the number of identities; used in CP-ABE for circuits).

[6] Precisely speaking, the distribution of r'_{pub} conditioned on $\mathbf{A}$ should be efficiently sampleable from $\mathbf{A}$ so that it does not contain a trapdoor of $\mathbf{A}$ or otherwise impedes security. If we insist that r'_{pub} be a binary string, we can make r'_{pub} at least $\max_{r_{\mathrm{pub}}}((\kappa+\lceil\log_2 q\rceil)\cdot nZ\lceil\log_2 q\rceil)$ bits long and set $\mathbf{A}[i,j]$ to be its $((i-1)+(j-1)n+1)^{\mathrm{st}}$ chunk of $(\kappa+\lceil\log_2 q\rceil)$ bits interpreted as an integer in binary then reduced modulo q, incurring a statistical error of $2^{-\kappa}nZ\lceil\log_2 q\rceil$.

[7] The reduction algorithms reshape components between vectors and matrices, and either reversely sample r'_{pub} from $\mathbf{A}$ or computes $\mathbf{A}$ from r'_{pub}.

Ingredients of Construction 2. Let $\mathsf{IPFE}' = (\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{Enc}', \mathsf{Dec}')$ be an underlying evIPFE(-w/sn) such that $1 \in \mathcal{I}'_{q,K,Z}$ (e.g., Constructions 1 and 3).

Construction 2 (IB-evIPFE(-w/sn))**.** Our scheme works as follows.

- $\mathsf{Setup}(q, 1^K, 1^Z)$ takes as input q, K, Z. If q is not a prime or $1 \notin \mathcal{I}'_{q,K,Z}$, the algorithm sets $\mathcal{I}_{q,K,Z} = \emptyset$ and terminates (aborting case). Otherwise, it sets $\mathcal{I}_{q,K,Z} = [q]$[8] and *deterministically* picks n.[9] The algorithm runs
$$(\mathsf{impk}', \mathsf{imsk}') \xleftarrow{\$} \mathsf{Setup}'(q, 1^K, 1^{Z+2n})$$
and outputs $\mathsf{impk} = (q, 1^K, 1^Z, 1^n, \mathsf{impk}')$ and $\mathsf{imsk} = (q, 1^K, 1^Z, 1^n, \mathsf{imsk}')$.
- $\mathsf{KeyGen}(\mathsf{imsk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g})$ takes as input imsk, $\mathsf{id} \in [q]$, $\mathbf{v}_\mathfrak{o} \in \mathbb{Z}_q^Z$, and $\mathbf{V}_\mathfrak{g} \in \mathbb{Z}_q^{Z \times K}$. It samples $\boldsymbol{\psi}_\mathfrak{o} \xleftarrow{\$} \mathbb{Z}_q^n$ and $\boldsymbol{\Psi}_\mathfrak{g} \xleftarrow{\$} \mathbb{Z}_q^{n \times K}$, runs
$$\mathsf{isk}' \xleftarrow{\$} \mathsf{KeyGen}'\left(\mathsf{imsk}', 1, \begin{pmatrix} \mathbf{v}_\mathfrak{o} \\ \boldsymbol{\psi}_\mathfrak{o} \\ \mathsf{id} \cdot \boldsymbol{\psi}_\mathfrak{o} \end{pmatrix}, \begin{pmatrix} \mathbf{V}_\mathfrak{g} \\ \boldsymbol{\Psi}_\mathfrak{g} \\ \mathsf{id} \cdot \boldsymbol{\Psi}_\mathfrak{g} \end{pmatrix}\right),$$
and outputs $\mathsf{isk} = (\boldsymbol{\psi}_\mathfrak{o}, \boldsymbol{\Psi}_\mathfrak{g}, \mathsf{isk}')$.
- $\mathsf{Enc}(\mathsf{impk}, \mathsf{id}, \mathfrak{s}, \mathbf{u}^\mathsf{T})$ takes as input imsk, $\mathsf{id} \in [q]$, $\mathfrak{s} \in \{\mathfrak{o}, \mathfrak{g}\}$, and $\mathbf{u} \in \mathbb{Z}_q^Z$. It samples $\boldsymbol{\varphi} \xleftarrow{\$} \mathbb{Z}_q^n$, and runs and outputs
$$\mathsf{ict} = \mathsf{ict}' \xleftarrow{\$} \mathsf{Enc}'(\mathsf{impk}', 1, \mathfrak{s}, (\mathbf{u}^\mathsf{T}, \mathsf{id} \cdot \boldsymbol{\varphi}^\mathsf{T}, -\boldsymbol{\varphi}^\mathsf{T})).$$
- $\mathsf{Dec}(\mathsf{impk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g}, \mathsf{isk}, \mathfrak{s}, \mathsf{ict})$ runs and outputs
$$\mathsf{Dec}'\left(\mathsf{impk}', 1, \begin{pmatrix} \mathbf{v}_\mathfrak{o} \\ \boldsymbol{\psi}_\mathfrak{o} \\ \mathsf{id} \cdot \boldsymbol{\psi}_\mathfrak{o} \end{pmatrix}, \begin{pmatrix} \mathbf{V}_\mathfrak{g} \\ \boldsymbol{\Psi}_\mathfrak{g} \\ \mathsf{id} \cdot \boldsymbol{\Psi}_\mathfrak{g} \end{pmatrix}, \mathsf{isk}', \mathfrak{s}, \mathsf{ict}'\right).$$

Correctness. Suppose IPFE' is $B'(q', K', Z')$-correct, then for noise structure $\mathfrak{o}$,
$$\begin{aligned} \mathsf{Dec}(\mathsf{impk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g}, \mathsf{isk}, \mathfrak{o}, \mathsf{ict}) &= \mathbf{u}^\mathsf{T} \cdot \mathbf{v}_\mathfrak{o} + (\mathsf{id} \cdot \boldsymbol{\varphi}^\mathsf{T}) \cdot \boldsymbol{\psi}_\mathfrak{o} + (-\boldsymbol{\varphi}^\mathsf{T}) \cdot (\mathsf{id} \cdot \boldsymbol{\psi}_\mathfrak{o}) + e_\mathfrak{o} \\ &= \mathbf{u}^\mathsf{T} \mathbf{v}_\mathfrak{o} + e_\mathfrak{o} \end{aligned}$$
for some $e_\mathfrak{o} \in [-B'(q, K, Z+2n), B'(q, K, Z+2n)]$, and similarly for the case of $\mathfrak{g}$. In summary, Construction 2 is B-correct for $B(q, K, Z) = B'(q, K, Z+2n)$.

Security. We state and prove the security of Construction 2.

Theorem 11 (¶). *Suppose* IPFE' *is* $\sigma_{\mathrm{pre},0}$*-secure (Definition 5) and* $\mathsf{LWE}_{n, J'_1, q, \sigma_{\mathrm{help}}}$ *(Assumption 1) holds for some* $\sigma_{\mathrm{help}} \le \frac{\sigma_{\mathrm{pre},0}}{(2^{\kappa+6}\sqrt{\kappa})^2}$,[10] *then Construction 2 is also* $\sigma_{\mathrm{pre},0}$*-secure, where* q *and* $J'_1 = \sum_{\mathsf{id} \in \mathsf{ID}} J_{\mathsf{id}}$ *are picked by* $\mathcal{S}$ *and* n *is picked by* Setup.

Suppose instead IPFE' *is restricted-*$\sigma_{\mathrm{pre},0}$*-secure (Definition 6) while keeping the other conditions intact, then Construction 2 is is also restricted-*$\sigma_{\mathrm{pre},0}$*-secure.*

[8] Due to homomorphic noise growth and noise flooding, q is super-polynomial in our work. When q is polynomially large, the identity space can be enlarged using full-rank difference hash [ABB10].

[9] This construction is generic. The quantity n here is unrelated to those in Construction 1 or 3, although they can be set to the same value.

[10] Here, κ is unrelated to that in Construction 1, although they can be set to the same value.

The proof works by arguing that when the noisy inner products for the matching identities are pseudorandom, so are the inner products concerned by IPFE′, e.g., for $\mathfrak{o}$,

$$\begin{cases} \mathsf{id} \neq \mathsf{id}' : \ \mathbf{U}_{\mathsf{id},\mathfrak{o}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id}',\mathfrak{o}} + (\mathsf{id}-\mathsf{id}')\mathbf{\Phi}_{\mathsf{id},\mathfrak{o}}^{\mathsf{T}}\mathbf{\Psi}_{\mathsf{id}',\mathfrak{o}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{o}})^{\mathsf{T}} \\ \mathsf{id} = \mathsf{id}' : \ \mathbf{U}_{\mathsf{id},\mathfrak{o}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id},\mathfrak{o}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{o}})^{\mathsf{T}} \end{cases} \begin{matrix} \approx \\ \uparrow \\ \text{LWE} \end{matrix} \mathbf{U}_{\mathsf{id},\mathfrak{o}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id}',\mathfrak{o}} + \$ \ \equiv \$, \begin{matrix} \mathsf{IPFEsec}_{\mathsf{pre}}^{\mathcal{S}} \\ \downarrow \\ \approx \end{matrix} \$.$$

When the noise widths of $\mathsf{IPFEsec}_{\mathsf{pre}}^{\mathcal{S}}$ and LWE do not match, we resolve the issue similarly to the proof of Corollary 7.

Proof (Theorem 11). Let $\mathcal{S} = (\mathcal{S}_{\mathrm{V}}, \mathcal{S}_{\mathrm{U}})$ be an efficient sampler (Definition 5) such that $\sigma_{\mathrm{pre}} \leq \sigma_{\mathrm{pre},0}$ and $\mathsf{IPFEsec}_{\mathsf{pre}}^{\mathcal{S}}$ holds against Construction 2. Consider the following efficient $\mathcal{S}' = (\mathcal{S}'_{\mathrm{V}}, \mathcal{S}'_{\mathrm{U}})$.

- $\mathcal{S}'_{\mathrm{V}}(r''_{\mathrm{pub}})$ parses $r''_{\mathrm{pub}} = (r_{\mathrm{pub}}, r'_{\mathrm{pub}})$ and runs

  $$(q, 1^K, 1^Z, \mathsf{ID}, \{1^{J_{\mathsf{id}}}, 1^{I_{\mathsf{id},\mathfrak{o}}}, 1^{I_{\mathsf{id},\mathfrak{g}}}\}_{\mathsf{id}\in\mathsf{ID}}, \sigma_{\mathrm{pre}}, \{\mathbf{V}_{\mathsf{id},\mathfrak{o}}, \mathbf{V}_{\mathsf{id},\mathfrak{g}}\}_{\mathsf{id}\in\mathsf{ID}}) \leftarrow \mathcal{S}_{\mathrm{V}}(r_{\mathrm{pub}}).$$

  The algorithm aborts if q is not a prime or $1 \notin \mathcal{I}'_{q,K,Z}$. Otherwise, it computes n used by Setup of Construction 2, and samples $\{\mathbf{\Psi}_{\mathsf{id},\mathfrak{s}}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}$, with $\mathbf{\Psi}_{\mathsf{id},\mathfrak{o}}$ (resp. $\mathbf{\Psi}_{\mathsf{id},\mathfrak{g}}$) uniformly random over $\mathbb{Z}_q^{n\times J_{\mathsf{id}}}$ (resp. $\mathbb{Z}_q^{n\times KJ_{\mathsf{id}}}$) for all $\mathsf{id}\in\mathsf{ID}$, using r''_{pub} in the straight-forward manner.[11] The algorithm sets

  $$Z' = Z + 2n, \qquad \mathsf{ID}' = \{1\}, \qquad J'_1 = \sum_{\mathsf{id}\in\mathsf{ID}} J_{\mathsf{id}}, \qquad I'_{1,\mathfrak{s}} = \sum_{\mathsf{id}\in\mathsf{ID}} I_{\mathsf{id},\mathfrak{s}},$$

  $$\sigma'_{\mathrm{pre}} = \begin{cases} \sigma_{\mathrm{pre}}, & \text{if } \sigma_{\mathrm{pre}} \geq \frac{\sigma_{\mathrm{pre},0}}{2^{\kappa+6}\sqrt{\kappa}}; \\ \sigma_{\mathrm{pre},0}, & \text{otherwise;} \end{cases} \qquad \mathbf{V}'_{1,\mathfrak{s}} = \begin{pmatrix} \cdots & \mathbf{V}_{\mathsf{id},\mathfrak{s}} & \cdots \\ \cdots & \mathbf{\Psi}_{\mathsf{id},\mathfrak{s}} & \cdots \\ \cdots & \mathsf{id}\cdot\mathbf{\Psi}_{\mathsf{id},\mathfrak{s}} & \cdots \end{pmatrix}_{\mathsf{id}\in\mathsf{ID}},$$

  and outputs $(q, 1^K, 1^{Z'}, \mathsf{ID}', 1^{J'_1}, 1^{I'_{1,\mathfrak{o}}}, 1^{I'_{1,\mathfrak{g}}}, \sigma'_{\mathrm{pre}}, \mathbf{V}'_{1,\mathfrak{o}}, \mathbf{V}'_{1,\mathfrak{g}})$.

- $\mathcal{S}'_{\mathrm{U}}(r''_{\mathrm{pub}}; r''_{\mathrm{priv}})$ parses $r''_{\mathrm{pub}} = (r_{\mathrm{pub}}, r'_{\mathrm{pub}})$ and $r''_{\mathrm{priv}} = (r_{\mathrm{priv}}, r'_{\mathrm{priv}})$. It recomputes q, ID, $\{I_{\mathsf{id},\mathfrak{s}}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}$ and n as in $\mathcal{S}'_{\mathrm{V}}$. The algorithm runs

  $$\{\mathbf{U}_{\mathsf{id},\mathfrak{s}}^{\mathsf{T}}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}} \leftarrow \mathcal{S}_{\mathrm{U}}(r_{\mathrm{pub}}; r_{\mathrm{priv}}),$$

  and samples $\mathbf{\Phi}_{\mathsf{id},\mathfrak{s}}$ uniformly random over $\mathbb{Z}_q^{n\times I_{\mathsf{id},\mathfrak{s}}}$ for all $\mathsf{id}\in\mathsf{ID}$ and $\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}$ using r'_{priv}. It sets

  $$(\mathbf{U}'_{1,\mathfrak{s}})^{\mathsf{T}} = \begin{pmatrix} \vdots & \vdots & \vdots \\ \mathbf{U}_{\mathsf{id},\mathfrak{s}}^{\mathsf{T}} & \mathsf{id}\cdot\mathbf{\Phi}_{\mathsf{id},\mathfrak{s}}^{\mathsf{T}} & -\mathbf{\Phi}_{\mathsf{id},\mathfrak{s}}^{\mathsf{T}} \\ \vdots & \vdots & \vdots \end{pmatrix}_{\mathsf{id}\in\mathsf{ID}}$$

  and outputs $\{(\mathbf{U}'_{1,\mathfrak{s}})^{\mathsf{T}}\}_{\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}$.

Claim 12 (¶). $\mathsf{IPFEsec}_{\mathsf{pre}}^{\mathcal{S}'}$ *(Definition 5) holds.*

[11] The distribution of r''_{pub} conditioned on $\{\mathbf{\Psi}_{\mathsf{id},\mathfrak{s}}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}$ must be efficiently sampleable from $\{\mathbf{\Psi}_{\mathsf{id},\mathfrak{s}}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}$. See Footnote 6.

From the $\sigma_{\mathrm{pre},0}$-security of IPFE', it follows that

$$\begin{aligned}&\left(r''_{\mathrm{pub}}, \mathsf{impk}', \mathsf{isk}'_1, \{\mathsf{Enc}'(\mathsf{impk}', 1, \mathfrak{s}, (\mathbf{U}'_{1,\mathfrak{s}})^\top)\}_{\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}\right)\\ \approx\ &\left(r''_{\mathrm{pub}}, \mathsf{impk}', \mathsf{isk}'_1, \{\mathsf{Enc}'(\mathsf{impk}', 1, \mathfrak{s}, \quad \mathbf{\Delta}_{\mathfrak{s}}^\top \quad)\}_{\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}\right),\end{aligned}$$

where $\mathbf{\Delta}_{\mathfrak{s}} \xleftarrow{\$} \mathbb{Z}_q^{Z'\times I'_{1,\mathfrak{s}}}$. (This is the "initial part" of this proof.)

Now consider this efficient $\mathcal{S}_{\$} = (\mathcal{S}_{\$,\mathrm{V}}, \mathcal{S}_{\$,\mathrm{U}})$:

- $\mathcal{S}_{\$,\mathrm{V}}$ is just $\mathcal{S}_{\mathrm{V}}$.
- $\mathcal{S}_{\$,\mathrm{U}}(r_{\mathrm{pub}}; r_{\$,\mathrm{priv}})$ recomputes $q, Z, \mathsf{ID}, \{I_{\mathsf{id},\mathfrak{s}}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}$ as in $\mathcal{S}_{\$,\mathrm{V}}$, then uses $r_{\$,\mathrm{priv}}$ to sample $\mathbf{U}_{\$,\mathsf{id},\mathfrak{s}} \xleftarrow{\$} \mathbb{Z}_q^{Z\times I_{\mathsf{id},\mathfrak{s}}}$ for all $\mathsf{id}\in\mathsf{ID}$ and $\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}$, and lastly outputs $\{\mathbf{U}_{\$,\mathsf{id},\mathfrak{s}}^\top\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}$.

It can be readily verified that $\mathsf{IPFEsec}_{\mathrm{pre}}^{\mathcal{S}_{\$}}$ follows from $\mathsf{IPFEsec}_{\mathrm{pre}}^{\mathcal{S}}$:

$$\begin{aligned}&\left(r_{\mathrm{pub}}, \left\{\begin{array}{r}\mathbf{U}_{\$,\mathsf{id},\mathfrak{o}}^\top\mathbf{V}_{\mathsf{id},\mathfrak{o}} + \mathbf{E}_{\mathsf{id},\mathfrak{o}}^\top,\\ \mathbf{U}_{\$,\mathsf{id},\mathfrak{g}}^\top\mathbf{V}_{\mathsf{id},\mathfrak{g}} + \widetilde{\mathbf{g}}^\top\otimes\mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{g}}^\top + \mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{o}}^\top\end{array}\right\}_{\mathsf{id}\in\mathsf{ID}}\right)\\ \equiv\ &\left(r_{\mathrm{pub}}, \left\{\begin{array}{r}(\mathbf{U}_{\$,\mathsf{id},\mathfrak{o}} + \mathbf{U}_{\mathsf{id},\mathfrak{o}})^\top\mathbf{V}_{\mathsf{id},\mathfrak{o}} + \mathbf{E}_{\mathsf{id},\mathfrak{o}}^\top,\\ (\mathbf{U}_{\$,\mathsf{id},\mathfrak{g}} + \mathbf{U}_{\mathsf{id},\mathfrak{g}})^\top\mathbf{V}_{\mathsf{id},\mathfrak{g}} + \widetilde{\mathbf{g}}^\top\otimes\mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{g}}^\top + \mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{o}}^\top\end{array}\right\}_{\mathsf{id}\in\mathsf{ID}}\right)\\ \approx\ &\left(r_{\mathrm{pub}}, \left\{\begin{array}{l}\mathbf{U}_{\$,\mathsf{id},\mathfrak{o}}^\top\mathbf{V}_{\mathsf{id},\mathfrak{o}} + \mathbf{\Delta}_{\mathrm{pre},\mathsf{id},\mathfrak{o}}^\top,\\ \mathbf{U}_{\$,\mathsf{id},\mathfrak{g}}^\top\mathbf{V}_{\mathsf{id},\mathfrak{g}} + \mathbf{\Delta}_{\mathrm{pre},\mathsf{id},\mathfrak{g}}^\top\end{array}\right\}_{\mathsf{id}\in\mathsf{ID}}\right) \equiv \left(r_{\mathrm{pub}}, \left\{\begin{array}{l}\mathbf{\Delta}_{\mathrm{pre},\mathsf{id},\mathfrak{o}}^\top,\\ \mathbf{\Delta}_{\mathrm{pre},\mathsf{id},\mathfrak{g}}^\top\end{array}\right\}_{\mathsf{id}\in\mathsf{ID}}\right).\end{aligned}$$

Applying the initial part of the proof to $\mathcal{S}_{\$}$, we obtain

$$\begin{aligned}&\left(r''_{\mathrm{pub}}, \mathsf{impk}', \mathsf{isk}'_1, \{\mathsf{Enc}'(\mathsf{impk}', 1, \mathfrak{s}, (\mathbf{U}'_{\$,1,\mathfrak{s}})^\top)\}_{\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}\right)\\ \approx\ &\left(r''_{\mathrm{pub}}, \mathsf{impk}', \mathsf{isk}'_1, \{\mathsf{Enc}'(\mathsf{impk}', 1, \mathfrak{s}, \quad \mathbf{\Delta}_{\mathfrak{s}}^\top \quad)\}_{\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}\right),\end{aligned}$$

where $\mathbf{U}'_{\$,1,\mathfrak{s}}$ is to $\mathbf{U}_{\$,\mathsf{id},\mathfrak{s}}$'s as $\mathbf{U}'_{1,\mathfrak{s}}$ is to $\mathbf{U}_{\mathsf{id},\mathfrak{s}}$'s for each $\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}$. Therefore,

$$\begin{aligned}&\left(r''_{\mathrm{pub}}, \mathsf{impk}', \mathsf{isk}'_1, \{\mathsf{Enc}'(\mathsf{impk}', 1, \mathfrak{s}, (\mathbf{U}'_{\$,1,\mathfrak{s}})^\top)\}_{\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}\right)\\ \approx\ &\left(r''_{\mathrm{pub}}, \mathsf{impk}', \mathsf{isk}'_1, \{\mathsf{Enc}'(\mathsf{impk}', 1, \mathfrak{s}, \ (\mathbf{U}'_{1,\mathfrak{s}})^\top \ )\}_{\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}\right).\end{aligned}$$

Opening up $\mathsf{IPFEsec}_{\mathrm{post}}^{\mathcal{S}}$, we see that

$$\begin{aligned}\overbrace{r_{\mathrm{pub}}}^{\text{part of } r''_{\mathrm{pub}}},\qquad \mathsf{impk} &= (\overbrace{q, 1^K, 1^Z, 1^n}^{\text{from } r_{\mathrm{pub}}}, \mathsf{impk}'),\\ \{\mathsf{ict}^*_{\mathsf{id},\mathfrak{s}}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}} &= \{\mathsf{Enc}'(\mathsf{impk}', 1, \mathfrak{s}, \ (\mathbf{U}'_{1,\mathfrak{s}})^\top \ )\}_{\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}},\\ \{\mathsf{ict}^{\$}_{\mathsf{id},\mathfrak{s}}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}} &= \{\mathsf{Enc}'(\mathsf{impk}', 1, \mathfrak{s}, (\mathbf{U}'_{\$,1,\mathfrak{s}})^\top)\}_{\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}},\\ \{\mathsf{isk}_{\mathsf{id}}\}_{\mathsf{id}\in\mathsf{ID}} &= (\{\underbrace{\mathbf{\Psi}_{\mathsf{id},\mathfrak{o}}, \mathbf{\Psi}_{\mathsf{id},\mathfrak{g}}}_{\text{from } r'_{\mathrm{pub}} \text{ in } r''_{\mathrm{pub}}}\}_{\mathsf{id}\in\mathsf{ID}}, \mathsf{isk}'_1),\end{aligned}$$

so it follows from the previous indistinguishability.

For the part regarding restricted security, it remains to verify that $\mathcal{S}'$ is restricted whenever $\mathcal{S}$ is restricted (below), and that $\mathcal{S}_{\$}$ is always restricted (clear by inspection). Let $\mathcal{S}_{\mathrm{A}0}$ be the algorithm in Definition 6 for $\mathcal{S}$. The $\mathcal{S}'_{\mathrm{A}0}(r''_{\mathrm{pub}})$ for $\mathcal{S}'$ parses r''_{pub} as $(r_{\mathrm{pub}}, r'_{\mathrm{pub}})$, recomputes $\mathsf{ID}, \{I_{\mathsf{id},\mathfrak{s}}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\}}, I'_{1,\mathfrak{o}}, I'_{1,\mathfrak{g}}, n$ as in $\mathcal{S}_{\mathrm{V}}$, runs and sets

$$(1^{n'}, \{\mathbf{A}_{\mathsf{id},\mathfrak{s},0,i}\}_{\mathsf{id}\in\mathsf{ID},\mathfrak{s}\in\{\mathfrak{o},\mathfrak{g}\},i\in[I_{\mathsf{id},\mathfrak{s}}]}) \leftarrow \mathcal{S}_{\mathrm{A}0}(r_{\mathrm{pub}}), \qquad n'' = n' + n(I'_{1,\mathfrak{o}} + I'_{1,\mathfrak{g}}),$$

$$\mathbf{A}'_{1,\mathfrak{v},0,i'_{\mathfrak{v}}} = \begin{pmatrix} \mathbf{A}_{\mathsf{id},\mathfrak{v},0,i} & & \\ & \mathbf{0}_{n(i'_{\mathfrak{v}}-1)\times n} & \\ & \mathsf{id}\cdot\mathbf{I}_n & -\mathbf{I}_n \\ & & \mathbf{0}_{n(I'_{1,\mathfrak{v}}-i'_{\mathfrak{v}}+I'_{1,\mathfrak{g}})\times n} \end{pmatrix}, \quad \mathbf{A}'_{1,\mathfrak{g},0,i'_{\mathfrak{g}}} = \begin{pmatrix} \mathbf{A}_{\mathsf{id},\mathfrak{g},0,i} & & \\ & \mathbf{0}_{n(I'_{1,\mathfrak{v}}+i'_{\mathfrak{g}}-1)\times n} & \\ & \mathsf{id}\cdot\mathbf{I}_n & -\mathbf{I}_n \\ & & \mathbf{0}_{n(I'_{1,\mathfrak{g}}-i'_{\mathfrak{g}})\times n} \end{pmatrix},$$

with (id, i) being the $(i'_{\mathfrak{s}})^{\text{th}}$ item among all such that $\mathsf{id} \in \mathsf{ID}$ and $i \in [I_{\mathsf{id},\mathfrak{s}}]$. It outputs $(1^{n''}, \{\mathbf{A}'_{1,\mathfrak{s},0,i'_{\mathfrak{s}}}\}_{\mathfrak{s}\in\{\mathfrak{v},\mathfrak{g}\}, i'_{\mathfrak{s}}\in[I'_{1,\mathfrak{s}}]})$. Conceptually, let

$$(\mathbf{d}'_0)^\top = (\mathbf{d}_0^\top, \underbrace{\ldots, \boldsymbol{\varphi}_{\mathsf{id},\mathfrak{v},i}^\top, \ldots}_{\mathsf{id}\in\mathsf{ID},\, i\in[I_{\mathsf{id},\mathfrak{v}}]}, \underbrace{\ldots, \boldsymbol{\varphi}_{\mathsf{id},\mathfrak{g},i}^\top, \ldots}_{\mathsf{id}\in\mathsf{ID},\, i\in[I_{\mathsf{id},\mathfrak{g}}]}),$$

then

$$\begin{aligned}(\mathbf{u}'_{1,\mathfrak{s},i'_{\mathfrak{s}}})^\top &= (\mathbf{u}_{\mathsf{id},\mathfrak{s},i}^\top, \mathsf{id}\cdot\boldsymbol{\varphi}_{\mathsf{id},\mathfrak{s},i}^\top, -\boldsymbol{\varphi}_{\mathsf{id},\mathfrak{s},i}^\top) \\ &= (\mathbf{d}_0^\top\mathbf{A}_{\mathsf{id},\mathfrak{s},0,i}, \mathsf{id}\cdot\boldsymbol{\varphi}_{\mathsf{id},\mathfrak{s},i}^\top, -\boldsymbol{\varphi}_{\mathsf{id},\mathfrak{s},i}^\top) = (\mathbf{d}'_0)^\top\mathbf{A}'_{1,\mathfrak{s},0,i'_{\mathfrak{s}}}.\end{aligned}$$

□

Proof (Claim 12). Consider the following hybrids.

- H_0 is the first distribution in $\mathsf{IPFEsec}^{\mathcal{S}'}_{\mathrm{pre}}$, i.e.,

  $$r''_{\mathrm{pub}}, \qquad (\mathbf{U}'_{1,\mathfrak{v}})^\top\mathbf{V}'_{1,\mathfrak{v}} + (\mathbf{E}'_{1,\mathfrak{v}})^\top, \qquad (\mathbf{U}'_{1,\mathfrak{g}})^\top\mathbf{V}'_{1,\mathfrak{g}} + \widetilde{\mathbf{g}}^\top \otimes (\mathbf{E}'_{1,\mathfrak{g}\mathfrak{g}})^\top + (\mathbf{E}'_{1,\mathfrak{g}\mathfrak{v}})^\top.$$

  The components can be rearranged into

  $$\begin{aligned}&r_{\mathrm{pub}}, \qquad r'_{\mathrm{pub}}, \\ &\left\{\mathbf{U}_{\mathsf{id},\mathfrak{v}}^\top\mathbf{V}_{\mathsf{id}',\mathfrak{v}} + (\mathsf{id}-\mathsf{id}')\boldsymbol{\Phi}_{\mathsf{id},\mathfrak{v}}^\top\boldsymbol{\Psi}_{\mathsf{id}',\mathfrak{v}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{v}})^\top\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}'}, \\ &\left\{\mathbf{U}_{\mathsf{id},\mathfrak{g}}^\top\mathbf{V}_{\mathsf{id}',\mathfrak{g}} + (\mathsf{id}-\mathsf{id}')\boldsymbol{\Phi}_{\mathsf{id},\mathfrak{g}}^\top\boldsymbol{\Psi}_{\mathsf{id}',\mathfrak{g}} + \widetilde{\mathbf{g}}^\top \otimes (\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}\mathfrak{g}})^\top + (\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}\mathfrak{v}})^\top\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}'}, \\ &\left\{\mathbf{U}_{\mathsf{id},\mathfrak{v}}^\top\mathbf{V}_{\mathsf{id},\mathfrak{v}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{v}})^\top\right\}_{\mathsf{id}\in\mathsf{ID}}, \\ &\left\{\mathbf{U}_{\mathsf{id},\mathfrak{g}}^\top\mathbf{V}_{\mathsf{id},\mathfrak{g}} + \widetilde{\mathbf{g}}^\top \otimes (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{g}})^\top + (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{v}})^\top\right\}_{\mathsf{id}\in\mathsf{ID}},\end{aligned}$$

  where

  $$\mathbf{E}'_{1,\mathfrak{e}} = \begin{pmatrix} \mathbf{E}'_{*,*,\mathfrak{e}} & \mathbf{E}'_{\mathsf{id},*,\mathfrak{e}} & \cdots \\ \mathbf{E}'_{*,\mathsf{id}',\mathfrak{e}} & \mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{e}} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}_{\mathsf{id}',\mathsf{id}\in\mathsf{ID}} \qquad \text{for all } \mathfrak{e} \in \{\mathfrak{v}, \mathfrak{g}\mathfrak{g}, \mathfrak{g}\mathfrak{v}\}$$

  with $\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{v}} \in \mathbb{Z}^{J_{\mathsf{id}'}\times I_{\mathsf{id},\mathfrak{v}}}$, $\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}\mathfrak{g}} \in \mathbb{Z}^{J_{\mathsf{id}'}\times I_{\mathsf{id},\mathfrak{g}}}$, $\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}\mathfrak{v}} \in \mathbb{Z}^{KJ_{\mathsf{id}'}\times I_{\mathsf{id},\mathfrak{g}}}$ for all $\mathsf{id}, \mathsf{id}' \in \mathsf{ID}$.

- In H_1, we insert noises $\mathbf{E}^{\mathrm{help}}$ into the inner products for $\mathfrak{v}$ between non-matching identities, i.e.,

  $$\begin{aligned}&r_{\mathrm{pub}}, \qquad r'_{\mathrm{pub}}, \\ &\left\{\mathbf{U}_{\mathsf{id},\mathfrak{v}}^\top\mathbf{V}_{\mathsf{id}',\mathfrak{v}} + (\mathsf{id}-\mathsf{id}')\boldsymbol{\Phi}_{\mathsf{id},\mathfrak{v}}^\top\boldsymbol{\Psi}_{\mathsf{id}',\mathfrak{v}} + (\mathbf{E}^{\mathrm{help}}_{\mathsf{id},\mathsf{id}'})^\top + (\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{v}})^\top\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}'}, \\ &\left\{\mathbf{U}_{\mathsf{id},\mathfrak{g}}^\top\mathbf{V}_{\mathsf{id}',\mathfrak{g}} + (\mathsf{id}-\mathsf{id}')\boldsymbol{\Phi}_{\mathsf{id},\mathfrak{g}}^\top\boldsymbol{\Psi}_{\mathsf{id}',\mathfrak{g}} + \widetilde{\mathbf{g}}^\top \otimes (\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}\mathfrak{g}})^\top + (\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}\mathfrak{v}})^\top\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}'}, \\ &\left\{\mathbf{U}_{\mathsf{id},\mathfrak{v}}^\top\mathbf{V}_{\mathsf{id},\mathfrak{v}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{v}})^\top\right\}_{\mathsf{id}\in\mathsf{ID}}, \\ &\left\{\mathbf{U}_{\mathsf{id},\mathfrak{g}}^\top\mathbf{V}_{\mathsf{id},\mathfrak{g}} + \widetilde{\mathbf{g}}^\top \otimes (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{g}})^\top + (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{v}})^\top\right\}_{\mathsf{id}\in\mathsf{ID}},\end{aligned}$$

  where $\mathbf{E}^{\mathrm{help}}_{\mathsf{id},\mathsf{id}'} \xleftarrow{\$} \mathcal{D}^{J_{\mathsf{id}'}\times I_{\mathsf{id},\mathfrak{v}}}_{\mathbb{Z},\sigma_{\mathrm{help}},\le\sigma_{\mathrm{help}}\sqrt{\kappa}}$, each of which is flooded by $\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{v}}$.

  By Lemma 2, we have $\mathsf{H}_0 \approx_{\mathrm{s}} \mathsf{H}_1$.

- In $\mathsf{H}_{2,\alpha}$ (with $0 \le \alpha \le |\mathsf{ID}|$), we no longer truncate $\mathbf{E}^{\mathrm{help}}$, and change the non-matching inner products with the ciphertexts under the first α identities for $\mathfrak{o}$ to random, i.e.,

$$
\begin{aligned}
&r_{\mathrm{pub}}, \qquad r'_{\mathrm{pub}},\\
&\left\{\mathbf{U}_{\mathsf{id},\mathfrak{o}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id}',\mathfrak{o}} + (\boldsymbol{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{o}})^{\mathsf{T}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}',\mathsf{id}\le\mathsf{id}_\alpha},\\
&\left\{\mathbf{U}_{\mathsf{id},\mathfrak{o}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id}',\mathfrak{o}} + (\mathsf{id}-\mathsf{id}')\boldsymbol{\Phi}_{\mathsf{id},\mathfrak{o}}^{\mathsf{T}}\boldsymbol{\Psi}_{\mathsf{id}',\mathfrak{o}} + (\mathbf{E}^{\mathrm{help}}_{\mathsf{id},\mathsf{id}'})^{\mathsf{T}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}',\mathsf{id}>\mathsf{id}_\alpha},\\
&\left\{\mathbf{U}_{\mathsf{id},\mathfrak{g}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id}',\mathfrak{g}} + (\mathsf{id}-\mathsf{id}')\boldsymbol{\Phi}_{\mathsf{id},\mathfrak{g}}^{\mathsf{T}}\boldsymbol{\Psi}_{\mathsf{id}',\mathfrak{g}} + \widetilde{\mathbf{g}}^{\mathsf{T}}\otimes(\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}\mathfrak{g}})^{\mathsf{T}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}'},\\
&\left\{\mathbf{U}_{\mathsf{id},\mathfrak{o}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id},\mathfrak{o}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id}\in\mathsf{ID}},\\
&\left\{\mathbf{U}_{\mathsf{id},\mathfrak{g}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id},\mathfrak{g}} + \widetilde{\mathbf{g}}^{\mathsf{T}}\otimes(\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{g}})^{\mathsf{T}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id}\in\mathsf{ID}},
\end{aligned}
$$

  where $\mathbf{E}^{\mathrm{help}}_{\mathsf{id},\mathsf{id}'} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{help}}}^{J_{\mathsf{id}'}\times I_{\mathsf{id},\mathfrak{o}}}$ and $\boldsymbol{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{o}} \xleftarrow{\$} \mathbb{Z}_q^{J_{\mathsf{id}'}\times I_{\mathsf{id},\mathfrak{o}}}$ for all $\mathsf{id},\mathsf{id}'\in\mathsf{ID}$. Here, $\mathsf{id}_\alpha\in\mathsf{ID}$ is the α^{th}-smallest among all $\mathsf{id}\in\mathsf{ID}$ in some fixed efficiently comparable total order for all $\alpha\in[|\mathsf{ID}|]$, and id_0 is a sentinel value that compares less than all $\mathsf{id}\in\mathsf{ID}$.

  By Lemma 1, we have $\mathsf{H}_1 \approx_{\mathrm{s}} \mathsf{H}_{2,0}$. By a hybrid argument, from $\mathsf{LWE}_{n,J'_1,q,\sigma_{\mathrm{help}}}$ it follows that $\mathsf{H}_{2,\alpha}\approx\mathsf{H}_{2,\alpha+1}$ for all $0\le\alpha<|\mathsf{ID}|$, with the LWE public matrix being

$$\left(\ldots,\ (\mathsf{id}_{\alpha+1}-\mathsf{id}')\boldsymbol{\Psi}_{\mathsf{id}',\mathfrak{o}},\ \ldots\right)_{\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}'\neq\mathsf{id}_{\alpha+1}}$$

  and the LWE secret being each column of $\boldsymbol{\Phi}_{\mathsf{id},\mathfrak{o}}$. Roughly speaking, the reduction algorithm computes[12] $\boldsymbol{\Psi}_{\mathsf{id}',\mathfrak{o}}$ for all $\mathsf{id}'\in\mathsf{ID}$ from the public matrices, and samples the other components by itself.

- In H_3, we change all non-matching inner products for $\mathfrak{o}$ to random, i.e.,

$$
\begin{aligned}
&r_{\mathrm{pub}}, \qquad r'_{\mathrm{pub}}, \qquad \left\{(\boldsymbol{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}'},\\
&\left\{\mathbf{U}_{\mathsf{id},\mathfrak{g}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id}',\mathfrak{g}} + (\mathsf{id}-\mathsf{id}')\boldsymbol{\Phi}_{\mathsf{id},\mathfrak{g}}^{\mathsf{T}}\boldsymbol{\Psi}_{\mathsf{id}',\mathfrak{g}} + \widetilde{\mathbf{g}}^{\mathsf{T}}\otimes(\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}\mathfrak{g}})^{\mathsf{T}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}'},\\
&\left\{\mathbf{U}_{\mathsf{id},\mathfrak{o}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id},\mathfrak{o}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id}\in\mathsf{ID}},\\
&\left\{\mathbf{U}_{\mathsf{id},\mathfrak{g}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id},\mathfrak{g}} + \widetilde{\mathbf{g}}^{\mathsf{T}}\otimes(\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{g}})^{\mathsf{T}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id}\in\mathsf{ID}}.
\end{aligned}
$$

  Comparing $\mathsf{H}_{2,|\mathsf{ID}|}$ and H_3, we see that $(\boldsymbol{\Delta}')$'s absorb the terms added to them without changing the distribution, so $\mathsf{H}_{2,|\mathsf{ID}|}\equiv\mathsf{H}_3$.

- In H_4, we change all non-matching inner products for $\mathfrak{g}$ to random, i.e.,

$$
\begin{aligned}
&r_{\mathrm{pub}}, \qquad r'_{\mathrm{pub}}, \qquad \left\{(\boldsymbol{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}'}, \quad \left\{(\boldsymbol{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}})^{\mathsf{T}}\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}'},\\
&\left\{\mathbf{U}_{\mathsf{id},\mathfrak{o}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id},\mathfrak{o}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id}\in\mathsf{ID}},\\
&\left\{\mathbf{U}_{\mathsf{id},\mathfrak{g}}^{\mathsf{T}}\mathbf{V}_{\mathsf{id},\mathfrak{g}} + \widetilde{\mathbf{g}}^{\mathsf{T}}\otimes(\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{g}})^{\mathsf{T}} + (\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id}\in\mathsf{ID}}.
\end{aligned}
$$

  The transition from H_3 to H_4 is analogous to that from H_0 to H_3, and $\mathsf{H}_3\approx\mathsf{H}_4$.

- In H_5, we alter the noises attached to the matching inner products into

$$
\begin{aligned}
&r_{\mathrm{pub}}, \qquad r'_{\mathrm{pub}}, \qquad \left\{(\boldsymbol{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{o}})^{\mathsf{T}}\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}'}, \quad \left\{(\boldsymbol{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}})^{\mathsf{T}}\right\}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}^{\mathsf{id}\neq\mathsf{id}'},
\end{aligned}
$$

[12] This step relies on q being a prime.

$\{\mathbf{U}_{\mathsf{id},\mathfrak{o}}^\intercal \mathbf{V}_{\mathsf{id},\mathfrak{o}} + \widetilde{\mathbf{E}}_{\mathsf{id},\mathfrak{o}}^\intercal\}_{\mathsf{id}\in\mathsf{ID}}$,
$\{\mathbf{U}_{\mathsf{id},\mathfrak{g}}^\intercal \mathbf{V}_{\mathsf{id},\mathfrak{g}} + \widetilde{\mathbf{g}}^\intercal \otimes \widetilde{\mathbf{E}}_{\mathsf{id},\mathfrak{g}\mathfrak{g}}^\intercal + \widetilde{\mathbf{E}}_{\mathsf{id},\mathfrak{g}\mathfrak{o}}^\intercal\}_{\mathsf{id}\in\mathsf{ID}}$,

where for all $\mathsf{id} \in \mathsf{ID}$ and $\mathfrak{e} \in \{\mathfrak{o}, \mathfrak{g}\mathfrak{g}, \mathfrak{g}\mathfrak{o}\}$,

$$\widetilde{\mathbf{E}}_{\mathsf{id},\mathfrak{e}} = \begin{cases} \mathbf{E}_{\mathsf{id},\mathfrak{e}}, & \text{if } \sigma_{\mathsf{pre}} \geq \frac{\sigma_{\mathsf{pre},0}}{2^{\kappa+6}\sqrt{\kappa}} \text{ so } \sigma'_{\mathsf{pre}} = \sigma_{\mathsf{pre}}; \\ \mathbf{E}_{\mathsf{id},\mathfrak{e}} + \mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{e}}, & \text{otherwise;} \end{cases}$$

with $\mathbf{E}_{\mathsf{id},\mathfrak{o}} \xleftarrow{\$} \mathcal{D}^{J_{\mathsf{id}}\times I_{\mathsf{id},\mathfrak{o}}}_{\mathbb{Z},\sigma_{\mathsf{pre}},\leq\sigma_{\mathsf{pre}}\sqrt{\kappa}}$, $\mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{g}} \xleftarrow{\$} \mathcal{D}^{J_{\mathsf{id}}\times I_{\mathsf{id},\mathfrak{g}}}_{\mathbb{Z},\sigma_{\mathsf{pre}},\leq\sigma_{\mathsf{pre}}\sqrt{\kappa}}$, $\mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{o}} \xleftarrow{\$} \mathcal{D}^{KJ_{\mathsf{id}}\times I_{\mathsf{id},\mathfrak{g}}}_{\mathbb{Z},\sigma_{\mathsf{pre}},\leq\sigma_{\mathsf{pre}}\sqrt{\kappa}}$.

In the first case, the two hybrids are statistically close by Lemma 1, and in the second case, by Lemma 2, as $\sigma'_{\mathsf{pre}} = \sigma_{\mathsf{pre},0} \geq 2^{\kappa+6}\sigma_{\mathsf{pre}}\sqrt{\kappa}$. Therefore, $\mathsf{H}_4 \approx_{\mathsf{s}} \mathsf{H}_5$.

- In H_6, we no longer truncate $\mathbf{E}_{\mathsf{id},\mathfrak{e}}$'s, i.e.,

  r_{pub}, r'_{pub}, $\{(\mathbf{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{o}})^\intercal\}^{\mathsf{id}\neq\mathsf{id}'}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}$, $\{(\mathbf{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}})^\intercal\}^{\mathsf{id}\neq\mathsf{id}'}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}$,
  $\{\mathbf{U}_{\mathsf{id},\mathfrak{o}}^\intercal \mathbf{V}_{\mathsf{id},\mathfrak{o}} + \mathbf{E}_{\mathsf{id},\mathfrak{o}}^\intercal + (\mathbf{E}''_{\mathsf{id},\mathfrak{o}})^\intercal\}_{\mathsf{id}\in\mathsf{ID}}$,
  $\{\mathbf{U}_{\mathsf{id},\mathfrak{g}}^\intercal \mathbf{V}_{\mathsf{id},\mathfrak{g}} + \widetilde{\mathbf{g}}^\intercal \otimes \mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{g}}^\intercal + \mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{o}}^\intercal + (\mathbf{E}''_{\mathsf{id},\mathfrak{g}})^\intercal\}_{\mathsf{id}\in\mathsf{ID}}$,

  where $\mathbf{E}_{\mathsf{id},\mathfrak{o}} \xleftarrow{\$} \mathcal{D}^{J_{\mathsf{id}}\times I_{\mathsf{id},\mathfrak{o}}}_{\mathbb{Z},\sigma_{\mathsf{pre}}}$, $\mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{g}} \xleftarrow{\$} \mathcal{D}^{J_{\mathsf{id}}\times I_{\mathsf{id},\mathfrak{g}}}_{\mathbb{Z},\sigma_{\mathsf{pre}}}$, $\mathbf{E}_{\mathsf{id},\mathfrak{g}\mathfrak{o}} \xleftarrow{\$} \mathcal{D}^{KJ_{\mathsf{id}}\times I_{\mathsf{id},\mathfrak{g}}}_{\mathbb{Z},\sigma_{\mathsf{pre}}}$, and $\mathbf{E}''_{\mathsf{id},\mathfrak{o}}$ (resp. $\mathbf{E}''_{\mathsf{id},\mathfrak{g}}$) is either $\mathbf{0}$ or $\mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{o}}$ (resp. $\widetilde{\mathbf{g}} \otimes \mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{g}} + \mathbf{E}'_{\mathsf{id},\mathsf{id},\mathfrak{g}\mathfrak{o}}$) for all $\mathsf{id} \in \mathsf{ID}$.

  By Lemma 1, we have $\mathsf{H}_5 \approx_{\mathsf{s}} \mathsf{H}_6$.

- H_7 is the second distribution in $\mathsf{IPFEsec}^{\mathcal{S}'}_{\mathsf{pre}}$, i.e.,

  r_{pub}, r'_{pub}, $\{(\mathbf{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{o}})^\intercal\}^{\mathsf{id}\neq\mathsf{id}'}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}$, $\{(\mathbf{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}})^\intercal\}^{\mathsf{id}\neq\mathsf{id}'}_{\mathsf{id},\mathsf{id}'\in\mathsf{ID}}$,
  $\{(\mathbf{\Delta}'_{\mathsf{id},\mathsf{id},\mathfrak{o}})^\intercal\}_{\mathsf{id}\in\mathsf{ID}}$, $\{(\mathbf{\Delta}'_{\mathsf{id},\mathsf{id},\mathfrak{g}})^\intercal\}_{\mathsf{id}\in\mathsf{ID}}$,

  where

  $$\mathbf{\Delta}'_{1,\mathfrak{s}} = \begin{pmatrix} \mathbf{\Delta}'_{*,*,\mathfrak{s}} & \mathbf{\Delta}'_{\mathsf{id},*,\mathfrak{s}} & \cdots \\ \mathbf{\Delta}'_{*,\mathsf{id}',\mathfrak{s}} & \mathbf{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{s}} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}_{\mathsf{id}',\mathsf{id}\in\mathsf{ID}} \qquad \text{for all } \mathfrak{s} \in \{\mathfrak{o}, \mathfrak{g}\}$$

  with $\mathbf{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{o}} \xleftarrow{\$} \mathbb{Z}_q^{J_{\mathsf{id}'}\times I_{\mathsf{id},\mathfrak{o}}}$, $\mathbf{\Delta}'_{\mathsf{id},\mathsf{id}',\mathfrak{g}} \xleftarrow{\$} \mathbb{Z}_q^{KJ_{\mathsf{id}'}\times I_{\mathsf{id},\mathfrak{g}}}$ for all $\mathsf{id}, \mathsf{id}' \in \mathsf{ID}$.

  $\mathsf{H}_6 \approx \mathsf{H}_7$ follows from $\mathsf{IPFEsec}^{\mathcal{S}}_{\mathsf{pre}}$, with $\mathbf{\Delta}'_{\mathsf{id},\mathsf{id},\mathfrak{s}}$ being either $\mathbf{\Delta}_{\mathsf{id},\mathfrak{s}}$ or $(\mathbf{\Delta}_{\mathsf{id},\mathfrak{s}} - \mathbf{E}''_{\mathsf{id},\mathsf{id},\mathfrak{s}})$.

By hybrid argument, $\mathsf{H}_0 \approx \mathsf{H}_7$, which is exactly $\mathsf{IPFEsec}^{\mathcal{S}'}_{\mathsf{pre}}$. □

3.5 Scheme with Structured Noises

To support structured noises, we modify Construction 1, fitting it into the shape of evasive learning with structured errors. We explain a subtlety in the adaptation. In the basic scheme, a key $\mathbf{k}$ for $\mathbf{0}$ makes $\mathbf{d}^\intercal\mathbf{B}\mathbf{k}$ small (not pseudorandom). Therefore, in the presence of such a key, no ciphertext can be provably secure from evasive LWE. This is not a problem for Construction 1, because every (plaintext) vector has non-pseudorandom noisy inner product with $\mathbf{0}$. Similar properties of keys are not allowed in a scheme with structured noises, as security should hold for a ciphertext of random vector for $\mathfrak{g}$, given a key for $\mathbf{v}_{\mathfrak{o}} = \mathbf{0}$ and random $\mathbf{V}_{\mathfrak{g}}$. Preventing noise structure mix-and-match is akin to having two identities, for which we employ a mechanism similar to yet simpler than that of Construction 2.

Ingredients of Construction 3. We rely on Lemma 3 for correctness. For security, we need LWE (Assumption 1) and evasive learning with structured errors (Assumption 3).

Construction 3 (evIPFE-w/sn)**.** Our scheme works as follows.

- $\mathsf{Setup}(q, 1^K, 1^Z)$ takes as input q, K, Z. It sets $\mathcal{I}_{q,K,Z} = \{1\}$, and *deterministically* picks n and $m \geq m_0(n(K+1), q)$ and $\sigma_{-1} \geq \sigma_0(n, m)$ appropriate for Lemma 3, as well as $\sigma_{\mathrm{post}}, \kappa$. The algorithm samples and sets

$$(\widetilde{\mathbf{B}}, \tau) \xleftarrow{\$} \mathsf{TrapGen}(1^{n(K+1)}, 1^m, q), \qquad \mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times 2(Z+n)\lceil \log_2 q \rceil},$$
$$\mathsf{impk} = (1^n, 1^m, q, 1^K, 1^Z, \sigma_{-1}, \sigma_{\mathrm{post}}, 1^\kappa, \mathbf{A}, \mathbf{B}), \qquad \mathsf{imsk} = (\mathsf{impk}, \tau).$$

  where $\mathbf{B} = (\mathbf{B}_{-1}, \ldots, \mathbf{B}_{K-1})$ for $\widetilde{\mathbf{B}} = (\mathbf{B}_{-1}^\intercal, \ldots, \mathbf{B}_{K-1}^\intercal)^\intercal$. It outputs $(\mathsf{impk}, \mathsf{imsk})$.

- $\mathsf{KeyGen}(\mathsf{imsk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g})$ imsk, $\mathsf{id} = 1$, $\mathbf{v}_\mathfrak{o} \in \mathbb{Z}_q^Z$, and $\mathbf{V}_\mathfrak{g} \in \mathbb{Z}_q^{Z \times K}$. It samples and runs

$$\boldsymbol{\psi}_\mathfrak{o} \xleftarrow{\$} \mathbb{Z}_q^n, \qquad \boldsymbol{\Psi}_\mathfrak{g} \xleftarrow{\$} \mathbb{Z}_q^{n \times K},$$
$$\mathbf{K} \xleftarrow{\$} \mathsf{SampleD}\left(\widetilde{\mathbf{B}}, \tau, \widetilde{\mathbf{P}}, \sigma_{-1}\right),$$

  where $\widetilde{\mathbf{P}} = (\mathbf{P}_{-1}^\intercal, \ldots, \mathbf{P}_{K-1}^\intercal)^\intercal$ for

$$\mathbf{P} = (\mathbf{P}_{-1}, \ldots, \mathbf{P}_{K-1}) = \mathbf{A}\mathbf{G}^{-1}\begin{pmatrix} \mathbf{v}_\mathfrak{o} & \\ \boldsymbol{\psi}_\mathfrak{o} & \\ & \mathbf{V}_\mathfrak{g} \\ & \boldsymbol{\Psi}_\mathfrak{g} \end{pmatrix}.$$

  The algorithm outputs $\mathsf{isk} = (\boldsymbol{\psi}_\mathfrak{o}, \boldsymbol{\Psi}_\mathfrak{g}, \mathbf{K})$.

- $\mathsf{Enc}(\mathsf{impk}, \mathsf{id}, \mathfrak{s}, \mathbf{u}^\intercal)$ takes as input impk, $\mathsf{id} = 1$, $\mathfrak{s} \in \{\mathfrak{o}, \mathfrak{g}\}$, and $\mathbf{u} \in \mathbb{Z}_q^Z$. It samples and sets

$$\boldsymbol{\varphi} \xleftarrow{\$} \mathbb{Z}_q^n, \quad \mathbf{d} \xleftarrow{\$} \mathbb{Z}_q^n, \quad \mathbf{e}_0 \xleftarrow{\$} \mathcal{D}^{2(Z+n)\lceil \log_2 q \rceil}_{\mathbb{Z}, \sigma_{\mathrm{post}}, \leq \sigma_{\mathrm{post}}\sqrt{\kappa}},$$
$$\mathbf{e}_1 \xleftarrow{\$} \mathcal{D}^{m(K+1)}_{\mathbb{Z}, \sigma_{\mathrm{post}}, \leq \sigma_{\mathrm{post}}\sqrt{\kappa}}, \quad \mathbf{e}_3 \xleftarrow{\$} \mathcal{D}^{m}_{\mathbb{Z}, \sigma_{\mathrm{post}}, \leq \sigma_{\mathrm{post}}\sqrt{\kappa}},$$
$$(\mathbf{u}')^\intercal \leftarrow \begin{cases} (\mathbf{u}^\intercal, \mathbf{0}_{1\times n}, \mathbf{0}_{1\times Z}, \boldsymbol{\varphi}^\intercal), & \text{if } \mathfrak{s} = \mathfrak{o}; \\ (\mathbf{0}_{1\times Z}, \boldsymbol{\varphi}^\intercal\mathbf{G}, \mathbf{u}^\intercal\mathbf{G}, \mathbf{0}_{1\times n}), & \text{if } \mathfrak{s} = \mathfrak{g}. \end{cases}$$

  The algorithm outputs $\mathsf{ict} = (\mathbf{d}^\intercal\mathbf{B} + (\mathbf{0}_{1\times m}, \widetilde{\mathbf{g}}^\intercal \otimes \mathbf{e}_3^\intercal) + \mathbf{e}_1^\intercal,\ \mathbf{d}^\intercal\mathbf{A} + (\mathbf{u}')^\intercal\mathbf{G} + \mathbf{e}_0^\intercal)$.

- $\mathsf{Dec}(\mathsf{impk}, \mathsf{id}, \mathbf{v}_\mathfrak{o}, \mathbf{V}_\mathfrak{g}, \mathsf{isk}, \mathfrak{s}, \mathsf{ict})$ computes and outputs

$$\mathsf{ict} \cdot \begin{pmatrix} -\mathbf{I}_{K+1} \otimes \mathbf{K} \\ \mathbf{G}^{-1}\begin{pmatrix} \mathbf{v}_\mathfrak{o} & \\ \boldsymbol{\psi}_\mathfrak{o} & \\ & \mathbf{V}_\mathfrak{g} \\ & \boldsymbol{\Psi}_\mathfrak{g} \end{pmatrix} \end{pmatrix} \cdot \begin{cases} (1, \mathbf{0}_{1\times K})^\intercal, & \text{if } \mathfrak{s} = \mathfrak{o}; \\ (\mathbf{0}_{K\times 1}, \mathbf{I}_K)^\intercal, & \text{if } \mathfrak{s} = \mathfrak{g}. \end{cases}$$

Correctness. It is readily verified, similarly to Construction 1, that the scheme is B-correct for

$$B = \kappa^{1/2}\sigma_{\mathrm{post}}(m^{3/2}\sigma_{-1} + 2(Z+n)\lceil \log_2 q \rceil).$$

Theorem 13. *Construction 3 is restricted-σ_{post}-secure (Definition 6) under evasive learning with structured errors (Assumption 3) and $\mathsf{LWE}_{n,m(K+1)+2(Z+n)\lceil \log_2 q\rceil,q,\sigma_{\mathrm{help}}}$ (Assumption 1) for some $\sigma_{\mathrm{help}} \leq \frac{\sigma_{\mathrm{post}}}{(2^{\kappa+6}\sqrt{\kappa})^2 \cdot 2(Z+n)\lceil \log_2 q\rceil}$, where q, K, Z are those picked by $\mathcal{S}$ and $n, m, \sigma_{\mathrm{post}}, \kappa$ are those picked by* Setup.

Proof Sketch. The proof done by combining the techniques demonstrated earlier in this section. Roughly speaking, the desired can be cast as a postcondition of evLWsE. In the precondition of evLWsE, first use IPFE simulation as in the proof of Theorem 6 so that we only consider the inner products, both for matching and non-matching noise structures. Then, use the proof of Claim 12 to argue that non-matching inner products, due to the presence of $\boldsymbol{\psi}^\intercal \boldsymbol{\varphi}_{\mathfrak{o}}$ or $\boldsymbol{\psi}^\intercal \boldsymbol{\Phi}_{\mathfrak{g}}$, are pseudorandom. Lastly, use the premise that the matching inner products are pseudorandom to conclude the proof. □

Applying Construction 2 to Construction 3, we obtain the following:

Corollary 14. *Under $\mathsf{LWE}_{n,\mathrm{poly}(\lambda),q,\sigma_{\mathrm{help}}}$ (Assumption 1) and evasive learning with structure errors (Assumption 3), there exists an IB-evIPFE-w/sn (Definition 4) with identity space $[q]$ and correctness bound*

$$B = \kappa^{1/2}\sigma_{\mathrm{post}}(m^{3/2}\sigma_{-1} + 2(Z+3n)\lceil \log_2 q \rceil)$$

that is restricted-σ_{post}-secure (Definition 6), where $\sigma_{\mathrm{help}} \leq \frac{\sigma_{\mathrm{post}}}{(2^{\kappa+6}\sqrt{\kappa})^2 \cdot 2(Z+3n)\lceil \log_2 q\rceil}$.

4 Noisy Linear Garbling

We consider a notion of noisy linear garbling with authenticity (i.e., conditional disclosure of secrets)[13] reusable for multiple inputs. Our formulation characterizes a *promise* primitive. Given a function $f : \mathbb{Z}^L \to \{0, 1, \bot\}$, the garbling consists of labels decomposable and affine in the input $\mathbf{x}$, a garbled table, a secret, and some auxiliary information. For correctness, if $f(\mathbf{x}) = 1$, the secret can be approximately recovered from the other information; for security, if $f(\mathbf{x}) = 0$, the secret remains hidden; when $f(\mathbf{x}) = \bot$, we require neither correctness nor security. The *promise* feature of the definition is used to exclude $\mathbf{x}$ leading to out-of-bound wire values in arithmetic computations, and $\mathbf{x} \notin \{0, 1\}^\star$ in Boolean computations.

Definition 7 (noisy linear garbling)**.** Let $F = \{F_{\lambda,\mathsf{param}}\}_{\lambda\in\mathbb{N},\mathsf{param}\in\mathsf{Params}_\lambda}$ be a sequence of function families, where $\mathsf{Params} = \{\mathsf{Params}_\lambda\}_{\lambda\in\mathbb{N}}$ is a sequence of function family description sets and each $f \in F_{\lambda,\mathsf{param}}$ is a function $\mathbb{Z}^L \to \{0, 1, \bot\}$ (with potentially different L for each f). A *noisy linear garbling scheme* for F consists of three efficient algorithms.

[13] The terminology is changed from "secret sharing" in introduction and technical overview, because the policy function might not be monotone.

- $\mathsf{Setup}(1^\lambda, \mathsf{param})$ takes the function family description $\mathsf{param} \in \mathsf{Params}_\lambda$ as input. It outputs some public parameter $\mathsf{pp} = (q, \dots)$, where $q \in \mathbb{N}_{\geq 2}$ is the modulus.[14]
- $\mathsf{GenF}(1^\lambda, \mathsf{pp}, f)$ takes as input pp and $f \in F_{\lambda,\mathsf{param}} : \mathbb{Z}^L \to \{0, 1, \bot\}$. It outputs a deterministic dimension 1^{n_f} of garbling randomness, a secret vector $\mathbf{w}_{\mathrm{out}}$, label functions $\{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}$, a garbled table $\mathbf{T}$, and some reusable information $\mathbf{R}$. Here, $\mathbf{w}_{\mathrm{out}}$, $\mathbf{W}$'s, and $\mathbf{T}$ are n_f-row matrices over $\mathbb{Z}$, and their shapes, including n_f, are fully determined by pp, f.
- $\mathsf{Eval}(1^\lambda, \mathsf{pp}, f, \mathbf{R}, \mathbf{x}, \{\mathbf{w}_\ell^\intercal\}_{\ell\in[L]}, \mathbf{t}^\intercal)$ takes as input pp, f, $\mathbf{R}$, input $\mathbf{x}$ (to f), one set of randomized labels, and the randomized garbled table $\mathbf{t}$. If $f(\mathbf{x}) = 1$, it is supposed to output an approximation of the randomized secret w_{out}. Here, w_{out}, $\{\mathbf{w}_\ell\}_{\ell\in[L]}, \mathbf{t}$ are over $\mathbb{Z}_q$.

Let B_{in} and B_{out} be functions mapping $(\lambda, \mathsf{param})$ to natural numbers. The scheme is $(B_{\mathrm{in}}, B_{\mathrm{out}})$-*correct* if for all $\lambda \in \mathbb{N}$, $\mathsf{param} \in \mathsf{Params}_\lambda$, $f \in F_{\lambda,\mathsf{param}} : \mathbb{Z}^L \to \{0, 1, \bot\}$, $\mathbf{x} \in \mathbb{Z}^L$, $e_{\mathrm{out}}, \mathbf{e}_1, \dots, \mathbf{e}_L, \mathbf{e}_{\mathrm{t}} \in [-B_{\mathrm{in}}(\lambda, \mathsf{param}), B_{\mathrm{in}}(\lambda, \mathsf{param})]^\star$ (of suitable dimensions)[15] such that $f(\mathbf{x}) = 1$, it holds that

$$\Pr\left[\begin{array}{r} \mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda, \mathsf{param}) \\ \left(\begin{smallmatrix} 1^{n_f}, \mathbf{w}_{\mathrm{out}}, \\ \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}, \\ \mathbf{T}, \mathbf{R} \end{smallmatrix}\right) \xleftarrow{\$} \mathsf{GenF}(1^\lambda, \mathsf{pp}, f) \\ \mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^{n_f} \end{array} : \begin{array}{l} \mathsf{Eval}\left(\begin{smallmatrix} 1^\lambda, \mathsf{pp}, f, \mathbf{R}, \mathbf{x}, \\ \{\mathbf{s}^\intercal(\mathbf{W}_{\ell,0} + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_\ell^\intercal\}_{\ell\in[L]}, \\ \mathbf{s}^\intercal\mathbf{T} + \mathbf{e}_{\mathrm{t}}^\intercal \end{smallmatrix}\right) \\ \qquad - (\mathbf{s}^\intercal\mathbf{w}_{\mathrm{out}} + e_{\mathrm{out}}) \\ \in [-B_{\mathrm{out}}(\lambda, \mathsf{param}), B_{\mathrm{out}}(\lambda, \mathsf{param})] \end{array}\right] = 1.$$

Definition 8 (shortness)**.** A noisy linear garbling scheme (Definition 7) is B_{short}-*short* (for some function B_{short} mapping $(\lambda, \mathsf{param})$ to natural numbers) if for all $\lambda \in \mathbb{N}$, $\mathsf{param} \in \mathsf{Params}_\lambda$, $f \in F_{\lambda,\mathsf{param}} : \mathbb{Z}^L \to \{0, 1, \bot\}$, $\mathbf{x} \in \mathbb{Z}^L$ such that $f(\mathbf{x}) = 0$,

$$\Pr\left[\begin{array}{r} \mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda, \mathsf{param}) \\ \left(\begin{smallmatrix} 1^{n_f}, \mathbf{w}_{\mathrm{out}}, \\ \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}, \\ \mathbf{T}, \mathbf{R} \end{smallmatrix}\right) \xleftarrow{\$} \mathsf{GenF}(1^\lambda, \mathsf{pp}, f) \end{array} : \begin{array}{r} \|\mathbf{w}_{\mathrm{out}}^\intercal\|, \|\mathbf{T}^\intercal\| \leq B_{\mathrm{short}}(\lambda, \mathsf{param}), \\ \|\mathbf{W}_{\ell,0}^\intercal + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times}^\intercal\| \leq B_{\mathrm{short}}(\lambda, \mathsf{param}) \\ \text{for all } \ell \in [L] \end{array}\right] = 1.$$

Definition 9 (fixed randomness dimension)**.** A noisy linear garbling scheme (Definition 7) has *fixed randomness dimension* if param is of the form $(q, 1^n, \dots)$ and $n_f = n$ always holds.

Security. We require that the randomized (by noisy linear combination) secret, labels, and garbled table be pseudorandom if $f(\mathbf{x}) = 0$.

Definition 10 (noisy linear garbling security)**.** Let $(\mathsf{Setup}, \mathsf{GenF}, \mathsf{Eval})$ be a noisy linear garbling scheme for F (Definition 7) and GenNoise an efficient algorithm with suitable input/output formats. The scheme is GenNoise-*secure* if $\mathsf{Exp}^{\mathrm{real}}_{\mathrm{garble}} \approx \mathsf{Exp}^{\mathrm{random}}_{\mathrm{garble}}$, where $\mathsf{Exp}^{\mathrm{real}}_{\mathrm{garble}}(1^\lambda, \mathcal{A})$ and $\mathsf{Exp}^{\mathrm{random}}_{\mathrm{garble}}(1^\lambda, \mathcal{A})$ proceed as follows.

[14] In the conference version [HLL24], the definition requires q be deterministic. While that version still works (by smuggling q into param and checking its suitability in Setup), it was an unintended error — it is easier and more consistent (with the other definitions in this paper) to let Setup sample q instead. This is currently necessary since we need prime q and it is unknown [TCH12] how to generate large primes in deterministic polynomial time.

[15] The dimensions could depend on the randomness of Setup. Formally, they can be quantified over sufficiently large dimensions then truncated appropriately.

- **Setup.** Launch $\mathcal{A}(1^\lambda)$ and receive $\mathsf{param} \in \mathsf{Params}_\lambda$ from it. Sample r_{Setup}, run

$$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{param}; r_{\mathsf{Setup}}),$$

  and send r_{Setup} to $\mathcal{A}$.

- **Challenge.** $\mathcal{A}$ chooses $f \in F_{\lambda,\mathsf{param}} : \mathbb{Z}^L \to \{0, 1, \perp\}$ and $\mathbf{x} \in \mathbb{Z}^L$. Sample r_{GenF} and run

$$(1^{n_f}, \mathbf{w}_{\mathrm{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}, \mathbf{T}, \mathbf{R}) \leftarrow \mathsf{GenF}(1^\lambda, \mathsf{pp}, f; r_{\mathsf{GenF}}).$$

  Sample $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^{n_f}$, run

$$(e_{\mathrm{out}}, \mathbf{e}_1, \ldots, \mathbf{e}_L, \mathbf{e}_{\mathrm{t}}) \xleftarrow{\$} \mathsf{GenNoise}(1^\lambda, \mathsf{param}, \mathsf{pp}, f, \mathbf{w}_{\mathrm{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}, \mathbf{T}, \mathbf{R}, \mathbf{x}),$$

  and compute

$$\begin{cases} w_{\mathrm{out}} \leftarrow \mathbf{s}^\intercal \mathbf{w}_{\mathrm{out}} + e_{\mathrm{out}}, & \mathbf{w}_\ell^\intercal \leftarrow \mathbf{s}^\intercal(\mathbf{W}_{\ell,0} + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_\ell^\intercal, & \mathbf{t}^\intercal \leftarrow \mathbf{s}^\intercal \mathbf{T} + \mathbf{e}_{\mathrm{t}}^\intercal, & \text{in } \mathsf{Exp}^{\mathrm{real}}_{\mathrm{garble}}; \\ w_{\mathrm{out}} \xleftarrow{\$} \mathbb{Z}_q, & \mathbf{w}_\ell \xleftarrow{\$} \mathbb{Z}_q^\star \quad (\text{for all } \ell \in [L]), & \mathbf{t} \xleftarrow{\$} \mathbb{Z}_q^\star, & \text{in } \mathsf{Exp}^{\mathrm{random}}_{\mathrm{garble}}; \end{cases}$$

  where $\{\mathbf{w}_\ell\}_{\ell\in[L]}$ and $\mathbf{t}$ in $\mathsf{Exp}^{\mathrm{random}}_{\mathrm{garble}}$ are of the same dimensions as their counterparts in $\mathsf{Exp}^{\mathrm{real}}_{\mathrm{garble}}$. Send $(r_{\mathsf{GenF}}, w_{\mathrm{out}}, \{\mathbf{w}_\ell\}_{\ell\in[L]}, \mathbf{t})$ to $\mathcal{A}$.

- **Guess.** $\mathcal{A}$ outputs $\beta' \in \{0, 1\}$. The output of the experiment is β' if $f(\mathbf{x}) = 0$. Otherwise, the output is set to 0.

5 Noisy Linear Garbling for Circuits

In this section, we construct a noisy linear garbling scheme for arithmetic circuits of bounded wire values. It is a variant of [AIK11,IW14,LL20a]. The garbling scheme serves as the central component in our CP-ABE for bounded-arithmetic circuits. (The notations have undergone some major changes since the conference version [HLL24] — see Remark 4.)

Construction 4 (noisy linear garbling for bounded-arithmetic circuits)**.** The function family of bounded-arithmetic circuits is

$$\begin{aligned} \mathsf{Params} &= \left\{ (M, 1^d) \,\middle|\, M \in \mathbb{N}_+ \text{ and } d \in \mathbb{N} \right\}, \\ F_{M,1^d} &= \left\{ f_{M,C} \,\middle|\, C \text{ is an arithmetic circuit of depth no more than } d \right\}, \\ f_{M,C}(\mathbf{x}) &= \begin{cases} \perp, & \text{if some wire value} \notin [-M, M] \text{ when evaluating } C(\mathbf{x}); \\ 1, & \text{if } C(\mathbf{x}) = 0 \text{ and all wire values} \in [-M, M]; \\ 0, & \text{if } C(\mathbf{x}) \neq 0 \text{ and all wire values} \in [-M, M]; \end{cases} \\ & \qquad \text{for } C : \mathbb{Z}^L \to \mathbb{Z} \text{ and } \mathbf{x} \in \mathbb{Z}^L. \end{aligned}$$

Since M is known from param, the function $f_{M,C}$ is simply represented by C. The scheme works as follows.

- $\mathsf{Setup}(M, 1^d)$ picks suitable q, N and outputs $\mathsf{pp} = (q, M, 1^d, 1^N)$. The constraints of q, N are specified in Theorems 15, 16, and 17.

- $\mathsf{GenF}(\mathsf{pp}, C)$ generates the garbling for $C : \mathbb{Z}^L \to \mathbb{Z}$ in the following steps.
  - It parses C into L input gates $1, \ldots, L$ and $(|C| - L)$ arithmetic gates $L+1, \ldots, |C|$. They are sorted in topological order so that gate $|C|$ is the output gate of C and the inputs to every non-input gate i are connected to gates $\mathsf{gin}[i,1], \mathsf{gin}[i,2] < i$.
  - For each gate i, the algorithm samples its shrunken label function

    $$\mathbf{W}_{i,0} \xleftarrow{\$} \{0,1\}^{N \times N}, \qquad \mathbf{W}_{i,\times} \xleftarrow{\$} \{0,1\}^{N \times N}.$$

    We write $\mathbf{W}_{i,x} = \mathbf{W}_{i,0} + x\mathbf{W}_{i,\times}$, where $x \in [-M, M]$ is a potential output value of gate i, for the corresponding shrunken label (before randomization).
  - For each non-input gate i, the algorithm samples $\widetilde{\mathbf{W}}^{(i,1)}_{\mathsf{gin}[i,1],0}, \widetilde{\mathbf{W}}^{(i,1)}_{\mathsf{gin}[i,1],\times}$, $\widetilde{\mathbf{W}}^{(i,2)}_{\mathsf{gin}[i,2],0}, \widetilde{\mathbf{W}}^{(i,2)}_{\mathsf{gin}[i,2],\times}$, which contribute to the expanded label function of each input to gate i. Here, the superscript indicates the purpose of the part (due to being an input to gate i), and the subscript indicates the matrix containing that part, so

    $$\widetilde{\mathbf{W}}_{i',0} = \left(\ldots, \widetilde{\mathbf{W}}^{(i,\gamma)}_{i',0}, \ldots\right)_{\mathsf{gin}[i,\gamma]=i'}, \qquad \widetilde{\mathbf{W}}_{i',\times} = \left(\ldots, \widetilde{\mathbf{W}}^{(i,\gamma)}_{i',\times}, \ldots\right)_{\mathsf{gin}[i,\gamma]=i'}.$$

    Similarly, we write $\widetilde{\mathbf{W}}_{i',x} = \widetilde{\mathbf{W}}_{i',0} + x\widetilde{\mathbf{W}}_{i',\times}$ for the expanded label (before randomization) corresponding to a potential output $x \in [-M, M]$ of gate i'. Suppose the input values to gate i are x_1, x_2, the goal is to compute the shrunken label $\mathbf{W}_{i,x}$ from the expanded labels $\widetilde{\mathbf{W}}_{\mathsf{gin}[i,1],x_1}, \widetilde{\mathbf{W}}_{\mathsf{gin}[i,2],x_2}$. Following [AIK11] (adapted for authenticity only), first sample $\mathbf{U}_i \xleftarrow{\$} \{0,1\}^{N \times N}$.
    - If gate i is addition, set

      $$\begin{aligned} \widetilde{\mathbf{W}}^{(i,1)}_{\mathsf{gin}[i,1],0} &\leftarrow \mathbf{U}_i, & \widetilde{\mathbf{W}}^{(i,1)}_{\mathsf{gin}[i,1],\times} &\leftarrow \mathbf{W}_{i,\times}, \\ \widetilde{\mathbf{W}}^{(i,2)}_{\mathsf{gin}[i,2],0} &\leftarrow \mathbf{W}_{i,0} - \mathbf{U}_i, & \widetilde{\mathbf{W}}^{(i,2)}_{\mathsf{gin}[i,2],\times} &\leftarrow \mathbf{W}_{i,\times}, \end{aligned}$$

      which are subject to $\widetilde{\mathbf{W}}^{(i,1)}_{\mathsf{gin}[i,1],x_1} + \widetilde{\mathbf{W}}^{(i,2)}_{\mathsf{gin}[i,2],x_2} = \mathbf{W}_{i,x_1+x_2}$.
    - If gate i is multiplication, set

      $$\begin{aligned} \widetilde{\mathbf{W}}^{(i,1)}_{\mathsf{gin}[i,1],0} &\leftarrow \mathbf{U}_i, & \widetilde{\mathbf{W}}^{(i,1)}_{\mathsf{gin}[i,1],\times} &\leftarrow \mathbf{W}_{i,\times}, \\ \widetilde{\mathbf{W}}^{(i,2)}_{\mathsf{gin}[i,2],0} &\leftarrow \mathbf{W}_{i,0}, & \widetilde{\mathbf{W}}^{(i,2)}_{\mathsf{gin}[i,2],\times} &\leftarrow -\mathbf{U}_i, \end{aligned}$$

      which are subject to $x_2\widetilde{\mathbf{W}}^{(i,1)}_{\mathsf{gin}[i,1],x_1} + \widetilde{\mathbf{W}}^{(i,2)}_{\mathsf{gin}[i,2],x_2} = \mathbf{W}_{i,x_1x_2}$.
  - For the output gate $|C|$, sample the expanded label function

    $$\widetilde{\mathbf{W}}_{|C|,0} \xleftarrow{\$} \{0,1\}^{N \times 1}, \qquad \widetilde{\mathbf{W}}_{|C|,\times} \xleftarrow{\$} \{0,1\}^{N \times 1}.$$

    Here, $\widetilde{\mathbf{W}}_{|C|,0}, \widetilde{\mathbf{W}}_{|C|,\times}$ are vectors (single-column matrices; still in upper case for consistency with the other components).
  - The algorithm computes the garbled table, which enables conversion to the expanded label $\widetilde{\mathbf{W}}_{i',x}$ from the shrunken label $\mathbf{W}_{i',x}$ for each gate i' with output value x, akin to the key-shrinking gadget in [AIK11]. For each tuple

(i', i, γ) with $\mathsf{gin}[i, \gamma] = i'$ (i.e., gate i' is the γ^{th} input to gate i), sample and set

$$\mathbf{R}_{i'}^{(i,\gamma)} \xleftarrow{\$} \{0,1\}^{N\times N},$$
$$\mathbf{T}_{i',0}^{(i,\gamma)} \leftarrow \mathbf{W}_{i',0}\mathbf{R}_{i'}^{(i,\gamma)} + \widetilde{\mathbf{W}}_{i',0}^{(i,\gamma)}, \qquad \mathbf{T}_{i',\times}^{(i,\gamma)} \leftarrow \mathbf{W}_{i',\times}\mathbf{R}_{i'}^{(i,\gamma)} + \widetilde{\mathbf{W}}_{i',\times}^{(i,\gamma)}.$$

Similarly define $\mathbf{T}_{i',x}^{(i,\gamma)}$, then $\mathbf{T}_{i',x}^{(i,\gamma)} = \mathbf{W}_{i',x}\mathbf{R}_{i'}^{(i,\gamma)} + \widetilde{\mathbf{W}}_{i',x}^{(i,\gamma)}$. For the output gate, sample $\mathbf{R}_{|C|} \xleftarrow{\$} \{0,1\}^{N\times 1}$ and set

$$\mathbf{T}_{|C|,0} \leftarrow \mathbf{W}_{|C|,0}\mathbf{R}_{|C|} + \widetilde{\mathbf{W}}_{|C|,0}, \qquad \mathbf{T}_{|C|,\times} \leftarrow \mathbf{W}_{|C|,\times}\mathbf{R}_{|C|} + \widetilde{\mathbf{W}}_{|C|,\times}.$$

Here, $\mathbf{R}_{|C|}, \mathbf{T}_{|C|,0}, \mathbf{T}_{|C|,\times}$ are vectors (single-column matrices; still in upper case for consistency with the other components).

All the samplings are done in a straight-forward way.[16] The algorithm outputs

$$1^{n_C} = 1^N, \qquad \mathbf{w}_{\text{out}} = \widetilde{\mathbf{W}}_{|C|,0}, \qquad \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]},$$
$$\mathbf{T} = \left(\ldots, \mathbf{T}_{i',0}^{(i,\gamma)}, \mathbf{T}_{i',\times}^{(i,\gamma)}, \ldots\right)_{\mathsf{gin}[i,\gamma]=i'}, \qquad \mathbf{R} = \left(\ldots, \mathbf{R}_{i'}^{(i,\gamma)}, \ldots\right)_{\mathsf{gin}[i,\gamma]=i'}.$$

- $\mathsf{Eval}(\mathsf{pp}, C, \mathbf{R}, \mathbf{x}, \{\mathbf{w}_\ell^\intercal\}_{\ell\in[L]}, \mathbf{t}^\intercal)$ parses $C, \mathbf{R}$ as in GenF and

$$\mathbf{t}^\intercal = \left(\ldots, (\mathbf{t}_{i',0}^{(i,\gamma)})^\intercal, (\mathbf{t}_{i',\times}^{(i,\gamma)})^\intercal, \ldots\right)_{\mathsf{gin}[i,\gamma]=i'}.$$

It evaluates $C(\mathbf{x})$ for the wire values $\{x_i\}_{i\in[|C|]}$ of each gate i, namely,

$$x_i \leftarrow \begin{cases} \mathbf{x}[\ell], & \text{if gate } i \text{ is an input gate } (i = \ell \in [L]); \\ x_{\mathsf{gin}[i,1]} + x_{\mathsf{gin}[i,2]}, & \text{if gate } i \text{ is addition;} \\ x_{\mathsf{gin}[i,1]} x_{\mathsf{gin}[i,2]}, & \text{if gate } i \text{ is multiplication.} \end{cases}$$

The algorithm has the (shrunken) input labels $\{\mathbf{w}_\ell^\intercal\}_{\ell\in[L]}$ as input. It computes the label of each non-input gate in increasing order by its index i.

  - The algorithm decrypts for the expanded label using

$$(\widetilde{\mathbf{w}}_{\mathsf{gin}[i,1]}^{(i,1)})^\intercal \leftarrow (\mathbf{t}_{\mathsf{gin}[i,1],x_{\mathsf{gin}[i,1]}}^{(i,1)})^\intercal - \mathbf{w}_{\mathsf{gin}[i,1]}^\intercal \mathbf{R}_{\mathsf{gin}[i,1]}^{(i,1)},$$
$$(\widetilde{\mathbf{w}}_{\mathsf{gin}[i,2]}^{(i,2)})^\intercal \leftarrow (\mathbf{t}_{\mathsf{gin}[i,2],x_{\mathsf{gin}[i,2]}}^{(i,2)})^\intercal - \mathbf{w}_{\mathsf{gin}[i,2]}^\intercal \mathbf{R}_{\mathsf{gin}[i,2]}^{(i,2)},$$

    where $\mathbf{t}_{\mathsf{gin}[i,1],x_{\mathsf{gin}[i,1]}}^{(i,1)} = \mathbf{t}_{\mathsf{gin}[i,1],0}^{(i,1)} + x_{\mathsf{gin}[i,1]}\mathbf{t}_{\mathsf{gin}[i,1],\times}^{(i,1)}$ (similarly for $\mathbf{t}_{\mathsf{gin}[i,2],x_{\mathsf{gin}[i,2]}}^{(i,2)}$) can be computed from $\mathbf{t}$ (input to Eval) and x's.
  - The algorithm recovers the shrunken label as

$$\mathbf{w}_i \leftarrow \begin{cases} \widetilde{\mathbf{w}}_{\mathsf{gin}[i,1]}^{(i,1)} + \widetilde{\mathbf{w}}_{\mathsf{gin}[i,2]}^{(i,2)}, & \text{if gate } i \text{ is addition;} \\ x_{\mathsf{gin}[i,2]}\widetilde{\mathbf{w}}_{\mathsf{gin}[i,1]}^{(i,1)} + \widetilde{\mathbf{w}}_{\mathsf{gin}[i,2]}^{(i,2)}, & \text{if gate } i \text{ is multiplication.} \end{cases}$$

[16] Precisely speaking, r_{GenF} conditioned on the sampled matrices must be efficiently sampleable given those matrices (with negligible statistical error), e.g., when the matrices are just the bits read sequentially from r_{GenF}. This ensures that no sampled matrix has a known trapdoor, and is important because r_{GenF} is incorporated into the sampler's randomness in a security proof.

Lastly, the algorithm computes and outputs the secret

$$w_{\mathrm{out}} \leftarrow \mathbf{t}_{|C|,x_{|C|}}^{\intercal} - \mathbf{w}_{|C|}^{\intercal}\mathbf{R}_{|C|}.$$

Here, $\mathbf{t}_{|C|,x_{|C|}}$ is a scalar (one-dimensional vector; still boldfaced for consistency with the other components).

Remark 4 (changes since the conference version [HLL24]). We list major changes for those who have read the conference version of Construction 4.

- In Params, the wire value bound M is now encoded in binary (instead of unary) for stronger functionality since it need not be $\mathrm{poly}(\lambda)$-bounded.
- The dimension of secret is N (instead of n, m) since it is for flipped LWE secret. It is now encoded in unary in pp to emphasize $\mathrm{poly}(\lambda)$-boundedness.
- Gates are named simply by their indices (instead of g subscript index).
- The notations i, i' are more consistent. When both appear in a context, gate i' is always an input to gate i, and i' never appears as component superscript. The notations in the proof are also improved for consistency.
- The expanded label function of the output gate is now $\widetilde{\mathbf{W}}$ (instead of $\widetilde{\mathbf{w}}$) to reduce ambiguity.

5.1 Correctness and Shortness

Theorem 15 (¶). *Construction 4 is $(B_{\mathrm{in}}, B_{\mathrm{out}})$-correct (Definition 7) if*

$$(6NM^2)^{d+1}B_{\mathrm{in}}(M,d) \le B_{\mathrm{out}}(M,d).$$

Proof (Theorem 15). Let $C : \mathbb{Z}^L \to \mathbb{Z}$ be an arithmetic circuit of depth at most d and $\mathbf{x} \in \mathbb{Z}^L$ an input with $C(\mathbf{x}) = 0$ such that all wire values in $C(\mathbf{x})$ are bounded by M. Let $\mathbf{s} \in \mathbb{Z}_q^N$ be a secret and $e_{\mathrm{out}}, \mathbf{e}_1, \ldots, \mathbf{e}_L, \mathbf{e}_{\mathrm{t}}$ any B_{in}-bounded errors. We extend the notation of $\mathbf{e}$'s by $\mathbf{e}_i = \mathbf{w}_i - (\mathbf{s}^{\intercal}\mathbf{W}_{i,x_i})^{\intercal}$ for all $L < i \le |C|$, where x_i's are the wire values. The extension symbolically agrees with $\mathbf{e}_1, \ldots, \mathbf{e}_L$.

We show by induction that during evaluation, $\|\mathbf{e}_i\| \le (6NM^2)^{d_i}B_{\mathrm{in}}$ for every gate i, where d_i is the depth of gate i (the input gates are of depth 0). The base case (for $\mathbf{e}_1, \ldots, \mathbf{e}_L$) is by assumption. For the inductive case, let $L < i \le |C|$. Assuming the induction hypothesis for all $i' < i$, then it applies to $i' = \mathsf{gin}[i,\gamma] < i$ for both $\gamma \in \{1,2\}$, and together with $d_{i'} \le d_i - 1$, we have

$$\|\mathbf{e}_{i'}\| \le (6NM^2)^{d_{i'}}B_{\mathrm{in}} \le (6NM^2)^{d_i-1}B_{\mathrm{in}}.$$

By the definition of $\mathbf{t}$ and how Eval proceeds,

$$\begin{aligned}\left(\widetilde{\mathbf{w}}_{i'}^{(i,\gamma)}\right)^{\intercal} &= \left(\mathbf{s}^{\intercal}\mathbf{T}_{i',0}^{(i,\gamma)} + (\mathbf{e}_{\mathrm{t},i',0}^{(i,\gamma)})^{\intercal}\right) + x_{i'}\left(\mathbf{s}^{\intercal}\mathbf{T}_{i',\times}^{(i,\gamma)} + (\mathbf{e}_{\mathrm{t},i',\times}^{(i,\gamma)})^{\intercal}\right) - (\mathbf{s}^{\intercal}\mathbf{W}_{i',x_{i'}} + \mathbf{e}_{i'}^{\intercal})\mathbf{R}_{i'}^{(i,\gamma)} \\ &= \mathbf{s}^{\intercal}\widetilde{\mathbf{W}}_{i',x_{i'}}^{(i,\gamma)} + \left(\mathbf{e}_{\mathrm{t},i',0}^{(i,\gamma)} + x_{i'}\mathbf{e}_{\mathrm{t},i',\times}^{(i,\gamma)}\right)^{\intercal} - \mathbf{e}_{i'}^{\intercal}\mathbf{R}_{i'}^{(i,\gamma)} = \mathbf{s}^{\intercal}\widetilde{\mathbf{W}}_{i',x_{i'}}^{(i,\gamma)} + \left(\widetilde{\mathbf{e}}_{i'}^{(i,\gamma)}\right)^{\intercal},\end{aligned}$$

where superscripts/subscripts follow the usual meaning of taking the relevant parts. It follows that the errors on the expanded labels are

$$\begin{aligned}\left\|\widetilde{\mathbf{e}}_{i'}^{(i,\gamma)}\right\| &\le \left\|\mathbf{e}_{\mathrm{t},i',0}^{(i,\gamma)}\right\| + |x_{i'}| \cdot \left\|\mathbf{e}_{\mathrm{t},i',\times}^{(i,\gamma)}\right\| + \left\|(\mathbf{R}_{i'}^{(i,\gamma)})^{\mathsf{T}}\right\| \cdot \|\mathbf{e}_{i'}\| \\ &\le B_{\mathrm{in}} + M \cdot B_{\mathrm{in}} + N \cdot (6NM^2)^{d_i-1} B_{\mathrm{in}} \\ &\le (N+M+1)(6NM^2)^{d_i-1} B_{\mathrm{in}}.\end{aligned}$$

We proceed to a case analysis.

- If gate i is addition,

$$\begin{aligned}\mathbf{e}_i^{\mathsf{T}} &= \overbrace{\left(\mathbf{s}^{\mathsf{T}}\widetilde{\mathbf{W}}^{(i,1)}_{\mathsf{gin}[i,1],x_{\mathsf{gin}[i,1]}} + (\widetilde{\mathbf{e}}^{(i,1)}_{\mathsf{gin}[i,1]})^{\mathsf{T}}\right) + \left(\mathbf{s}^{\mathsf{T}}\widetilde{\mathbf{W}}^{(i,2)}_{\mathsf{gin}[i,2],x_{\mathsf{gin}[i,2]}} + (\widetilde{\mathbf{e}}^{(i,2)}_{\mathsf{gin}[i,2]})^{\mathsf{T}}\right)}^{\mathbf{w}_i = \widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1]} + \widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2]}} - \mathbf{s}^{\mathsf{T}}\mathbf{W}_{i,x_i} \\ &= \left(\mathbf{e}^{(i,1)}_{\mathsf{gin}[i,1]} + \mathbf{e}^{(i,2)}_{\mathsf{gin}[i,2]}\right)^{\mathsf{T}},\end{aligned}$$

  so $\|\mathbf{e}_i\| \le 2(N+M+1)(6NM^2)^{d_i-1} B_{\mathrm{in}} \le (6NM^2)^{d_i} B_{\mathrm{in}}$.

- If gate i is multiplication,

$$\begin{aligned}\mathbf{e}_i^{\mathsf{T}} &= \overbrace{x_{\mathsf{gin}[i,2]}\left(\mathbf{s}^{\mathsf{T}}\widetilde{\mathbf{W}}^{(i,1)}_{\mathsf{gin}[i,1],x_{\mathsf{gin}[i,1]}} + (\widetilde{\mathbf{e}}^{(i,1)}_{\mathsf{gin}[i,1]})^{\mathsf{T}}\right) + \left(\mathbf{s}^{\mathsf{T}}\widetilde{\mathbf{W}}^{(i,2)}_{\mathsf{gin}[i,2],x_{\mathsf{gin}[i,2]}} + (\widetilde{\mathbf{e}}^{(i,2)}_{\mathsf{gin}[i,2]})^{\mathsf{T}}\right)}^{\mathbf{w}_i = x_{\mathsf{gin}[i,2]}\widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1]} + \widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2]}} - \mathbf{s}^{\mathsf{T}}\mathbf{W}_{i,x_i} \\ &= \left(x_{\mathsf{gin}[i,2]}\mathbf{e}^{(i,1)}_{\mathsf{gin}[i,1]} + \mathbf{e}^{(i,2)}_{\mathsf{gin}[i,2]}\right)^{\mathsf{T}},\end{aligned}$$

  so $\|\mathbf{e}_i\| \le (M+1)(N+M+1)(6NM^2)^{d_i-1} B_{\mathrm{in}} \le (6NM^2)^{d_i} B_{\mathrm{in}}$.

This completes the induction.

Applying the error bound to the output gate, we have

$$\|\mathbf{e}_{|C|}\| \le (6NM^2)^{d_{|C|}} B_{\mathrm{in}} \le (6NM^2)^{d} B_{\mathrm{in}}.$$

Note that $x_{|C|} = C(\mathbf{x}) = 0$ by assumption, and we have

$$\begin{aligned}&w_{\mathrm{out}} - (\mathbf{s}^{\mathsf{T}}\mathbf{w}_{\mathrm{out}} + e_{\mathrm{out}}) \\ &= \left(\mathbf{t}_{|C|,0}^{\mathsf{T}} - \mathbf{w}_{|C|}^{\mathsf{T}}\mathbf{R}_{|C|}\right) - (\mathbf{s}^{\mathsf{T}}\mathbf{w}_{\mathrm{out}} + e_{\mathrm{out}}) \\ &= \left(\mathbf{s}^{\mathsf{T}}\mathbf{T}_{|C|,0} + \mathbf{e}_{\mathrm{t},|C|,0}^{\mathsf{T}}\right) - \left(\mathbf{s}^{\mathsf{T}}\mathbf{W}_{|C|,0} + \mathbf{e}_{|C|}^{\mathsf{T}}\right)\mathbf{R}_{|C|} - \mathbf{s}^{\mathsf{T}}\mathbf{w}_{\mathrm{out}} - e_{\mathrm{out}} \\ &= \mathbf{e}_{\mathrm{t},|C|,0}^{\mathsf{T}} - \mathbf{e}_{|C|}^{\mathsf{T}}\mathbf{R}_{|C|} - e_{\mathrm{out}} + \mathbf{s}^{\mathsf{T}}\underbrace{\left(\mathbf{W}_{|C|,0}\mathbf{R}_{|C|} + \widetilde{\mathbf{W}}_{|C|,0} - \mathbf{W}_{|C|,0}\mathbf{R}_{|C|} - \widetilde{\mathbf{W}}_{|C|,0}\right)}_{\mathbf{0}}.\end{aligned}$$

Therefore, the output error is bounded by

$$\begin{aligned}\|\mathbf{e}_{\mathrm{t},|C|,0}\| + \left\|\mathbf{R}_{|C|}^{\mathsf{T}}\right\| \cdot \|\mathbf{e}_{|C|}\| + |e_{\mathrm{out}}| &\le B_{\mathrm{in}} + N \cdot (6NM^2)^{d} B_{\mathrm{in}} + B_{\mathrm{in}} \\ &\le (6NM^2)^{d+1} B_{\mathrm{in}} \le B_{\mathrm{out}}.\end{aligned}$$

□

Theorem 16 (¶). *Construction 4 is B_{short}-short (Definition 8) if*

$$B_{\mathrm{short}}(M,d) \ge (N+M+2)N.$$

Proof (Theorem 16). By definition, $\mathbf{w}_{\mathrm{out}}^{\intercal} \in \{0,1\}^{1\times N}$, so

$$\|\mathbf{w}_{\mathrm{out}}^{\intercal}\| \le N \le (N+M+2)N \le B_{\mathrm{short}}.$$

For all $\ell \in [L]$, when $|\mathbf{x}[\ell]| \le M$,

$$\|\mathbf{W}_{\ell,\mathbf{x}[\ell]}^{\intercal}\| \le \|\mathbf{W}_{\ell,0}^{\intercal}\| + M\|\mathbf{W}_{\ell,\times}^{\intercal}\| \le (M+1)N \le (N+M+2)N \le B_{\mathrm{short}},$$

because $\mathbf{W}$'s are in $\{0,1\}^{N\times N}$. For $\mathbf{T}$, observe that each $\widetilde{\mathbf{W}}$ is a signed sum of at most two matrices in $\{0,1\}^{N\times N}$, so each $\widetilde{\mathbf{W}}^{\intercal}$ has norm bound $2N$. Therefore,

$$\|\mathbf{T}^{\intercal}\| \le \overbrace{M}^{\mathbf{R}^{\intercal}} \cdot \overbrace{N}^{\mathbf{W}^{\intercal}} + \overbrace{2N}^{\widetilde{\mathbf{W}}^{\intercal}} = (M+2)N \le (N+M+2)N \le B_{\mathrm{short}}. \qquad \square$$

5.2 Security with Gaussian Noise

Theorem 17 (¶). *Suppose*

$$\mathsf{GenGaussian}(M, 1^d, (q, M, 1^d, 1^N), C, \mathbf{w}_{\mathrm{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}, \mathbf{T}, \mathbf{R}, \mathbf{x})$$

samples (truncated) Gaussian noises with appropriate width σ of suitable shape

$$(e_{\mathrm{out}}, \mathbf{e}_1, \ldots, \mathbf{e}_L, \mathbf{e}_{\mathrm{t}}) \xleftarrow{\$} \mathcal{D}^{\star}_{\mathbb{Z},\sigma,\le\sigma\sqrt{\kappa}},$$

then Construction 4 is GenGaussian*-secure (Definition 10) if q is always a prime and* $\mathsf{FlipLWE}_{N,\mathrm{poly}(\lambda),q,\sigma'}$ *(Assumption 2) holds for some $\sigma' \le \frac{\sigma}{2^{\kappa+6}(2N+M+2)\sqrt{\kappa}}$.*[17]

It is known [AIK11,IW14,LL20a] that the information-theoretic version of arithmetic garbling satisfies the following simple simulation property — the labels are uniformly random conditioned on evaluation correctness. Construction 4 is similar in this regard. The garbling is a noisy linear randomization of the public matrices. Once LWE is applied:

- the expanded labels of fan-ins are uniformly random conditioned on the correct evaluation into the shrunken labels (of fan-outs),
- the garbled table is uniformly random conditioned on the correct recovery of the expanded labels from the shrunken labels (of the same gate), and
- the expanded labels of the output gate are uniformly random conditioned on the approximate recovery of $\mathbf{s}^{\intercal}(\widetilde{\mathbf{W}}_{|C|,0} + C(\mathbf{x}) \cdot \widetilde{\mathbf{W}}_{|C|,\times})$.

Combining these points, the input labels and the garbled table are jointly random, conditioned on the approximate recovery of $\mathbf{s}^{\intercal}(\widetilde{\mathbf{W}}_{|C|,0} + C(\mathbf{x}) \cdot \widetilde{\mathbf{W}}_{|C|,\times})$. When this latter value itself is random, the overall distribution of input labels and garbled table is simply random. Recall that security is considered only when $C(\mathbf{x}) \neq 0$. Since q is a prime, the evaluation result will be independent of the secret approximating $\mathbf{s}^{\intercal}\widetilde{\mathbf{W}}_{|C|,0}$ once LWE is applied. Therefore, the revealed components (secret, input labels, garbled table) are jointly pseudorandom. The proof below formalizes this idea.

[17] The constant has been changed from the conference version [HLL24]. Both versions are correct. The new version fits the proof better.

Proof (Theorem 17). We show $\mathsf{Exp}^{\mathrm{real}}_{\mathrm{garble}} \approx \mathsf{Exp}^{\mathrm{random}}_{\mathrm{garble}}$ with the following hybrids.

- H_0 is $\mathsf{Exp}^{\mathrm{real}}_{\mathrm{garble}}$. The adversary sees $r_{\mathsf{Setup}}, r_{\mathsf{GenF}}$ and

$$w_{\mathrm{out}} = \mathbf{s}^\intercal \widetilde{\mathbf{W}}_{|C|,0} + e_{\mathrm{out}}, \qquad \mathbf{w}_\ell^\intercal = \mathbf{s}^\intercal(\mathbf{W}_{\ell,0} + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_\ell^\intercal,$$
$$(\mathbf{t}_{i',0}^{(i,\gamma)})^\intercal = \mathbf{s}^\intercal \mathbf{T}_{i',0}^{(i,\gamma)} + (\mathbf{e}_{i',0}^{(i,\gamma)})^\intercal, \qquad (\mathbf{t}_{i',\times}^{(i,\gamma)})^\intercal = \mathbf{s}^\intercal \mathbf{T}_{i',\times}^{(i,\gamma)} + (\mathbf{e}_{i',\times}^{(i,\gamma)})^\intercal,$$

  where the indices for $\mathbf{t}$ are constrained by $\mathsf{gin}[i,\gamma] = i'$ (henceforth same and omitted).

- In H_1, we rewrite the garbling using distributivity and associativity. Recall that each block in $\mathbf{T}$ is a signed sum of $\mathbf{W}$'s and $\mathbf{U}$'s. Instead of computing $\mathbf{T}$, multiplying $\mathbf{s}$, then adding the noise, we compute $\mathbf{s}$ multiplied by $\mathbf{W}$'s and $\mathbf{U}$'s, sum them with appropriate signs, then add the noise. Namely, for all $i \in [|C|]$, compute

$$\mathbf{w}_{i,0}^\intercal = \mathbf{s}^\intercal \mathbf{W}_{i,0}, \qquad \mathbf{w}_{i,\times}^\intercal = \mathbf{s}^\intercal \mathbf{W}_{i,\times}, \qquad (\text{if } i > L) \quad \mathbf{u}_i^\intercal = \mathbf{s}^\intercal \mathbf{U}_i,$$

  and $\widetilde{w}_{\mathrm{out},0} = \mathbf{s}^\intercal \widetilde{\mathbf{W}}_{|C|,0}$ and $\widetilde{w}_{\mathrm{out},\times} = \mathbf{s}^\intercal \widetilde{\mathbf{W}}_{|C|,\times}$. Those components are called "initial $\mathbf{w}$'s and $\mathbf{u}$'s", which are not given directly to the adversary. Next, compute the randomized expanded label functions as follows (compare with GenF of Construction 4).

  - If gate i is addition, set

$$\widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1],0} = \mathbf{u}_i, \qquad \widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1],\times} = \mathbf{w}_{i,\times},$$
$$\widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2],0} = \mathbf{w}_{i,0} - \mathbf{u}_i, \qquad \widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2],\times} = \mathbf{w}_{i,\times}.$$

  - If gate i is multiplication, set

$$\widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1],0} = \mathbf{u}_i, \qquad \widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1],\times} = \mathbf{w}_{i,\times},$$
$$\widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2],0} = \mathbf{w}_{i,0}, \qquad \widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2],\times} = -\mathbf{u}_i.$$

  Then, compute the $\mathbf{s}$-linear parts (denoted by $\widetilde{\mathbf{t}}$) of $\mathbf{t}$ as

$$(\widetilde{\mathbf{t}}_{i',0}^{(i,\gamma)})^\intercal = \mathbf{w}_{i',0}^\intercal \mathbf{R}_{i'}^{(i,\gamma)} + (\widetilde{\mathbf{w}}_{i',0}^{(i,\gamma)})^\intercal, \qquad (\widetilde{\mathbf{t}}_{i',\times}^{(i,\gamma)})^\intercal = \mathbf{w}_{i',\times}^\intercal \mathbf{R}_{i'}^{(i,\gamma)} + (\widetilde{\mathbf{w}}_{i',\times}^{(i,\gamma)})^\intercal.$$

  The randomized secret, input labels, and garble table are

$$w_{\mathrm{out}} = \widetilde{w}_{\mathrm{out},0} + e_{\mathrm{out}}, \qquad \mathbf{w}_\ell = \mathbf{w}_{\ell,0} + \mathbf{x}[\ell]\mathbf{w}_{\ell,\times} + \mathbf{e}_\ell,$$
$$\mathbf{t}_{i',0}^{(i,\gamma)} = \widetilde{\mathbf{t}}_{i',0}^{(i,\gamma)} + \mathbf{e}_{\mathrm{t},i',0}^{(i,\gamma)}, \qquad \mathbf{t}_{i',\times}^{(i,\gamma)} = \widetilde{\mathbf{t}}_{i',\times}^{(i,\gamma)} + \mathbf{e}_{\mathrm{t},i',\times}^{(i,\gamma)}.$$

  Clearly, $\mathsf{H}_0 \equiv \mathsf{H}_1$.

- In H_2, additional small noises are attached to the initial $\mathbf{w}$'s and $\mathbf{u}$'s as follows. Sample $\overline{\mathbf{e}}_{i,0}, \overline{\mathbf{e}}_{i,\times}, \overline{\mathbf{e}}_{\mathrm{U},i}$'s of appropriate dimensions and $\overline{e}_{\mathrm{out},0}, \overline{e}_{\mathrm{out},\times}$, each entry from $\mathcal{D}_{\mathbb{Z},\sigma',\le\sigma'\sqrt{\kappa}}$, and set

$$\mathbf{w}_{i,0}^\intercal = \mathbf{s}^\intercal \mathbf{W}_{i,0} + \overline{\mathbf{e}}_{i,0}^\intercal, \qquad \mathbf{w}_{i,\times}^\intercal = \mathbf{s}^\intercal \mathbf{W}_{i,\times} + \overline{\mathbf{e}}_{i,\times}^\intercal, \qquad (i > L) \quad \mathbf{u}_i^\intercal = \mathbf{s}^\intercal \mathbf{U}_i + \overline{\mathbf{e}}_{\mathrm{U},i}^\intercal,$$
$$\widetilde{w}_{\mathrm{out},0} = \mathbf{s}^\intercal \widetilde{\mathbf{W}}_{|C|,0} + \overline{e}_{\mathrm{out},0}, \quad \widetilde{w}_{\mathrm{out},\times} = \mathbf{s}^\intercal \widetilde{\mathbf{W}}_{|C|,\times} + \overline{e}_{\mathrm{out},\times}.$$

Recall that the initial $\mathbf{w}$'s and $\mathbf{u}$'s are not directly given to the adversary. In the revealed components (secret, input labels, garbled table), the magnitude of errors contributed by the additional small noises is bounded by

$$\sigma'\sqrt{\kappa} \cdot \max\{M+1, N+2\} \leq (N+M+2)\sigma'\sqrt{\kappa} \leq 2^{-\kappa-6}\sigma.$$

Therefore, they are flooded (Lemma 2) by the randomizing noises ($\mathbf{e}$'s and e's, without bars), i.e., $\mathsf{H}_1 \approx_s \mathsf{H}_2$.

- In H_3, the small noises $\overline{\mathbf{e}}_{i,0}, \overline{\mathbf{e}}_{i,\times}, \overline{\mathbf{e}}_{\mathrm{U},i}$'s and $\overline{e}_{\mathrm{out},0}, \overline{e}_{\mathrm{out},\times}$ are no longer truncated, i.e., the entries are from $\mathcal{D}_{\mathbb{Z},\sigma'}$. We have $\mathsf{H}_2 \approx_s \mathsf{H}_3$ by Lemma 1.
- In H_4, the initial $\mathbf{w}$'s and $\mathbf{u}$'s are replaced by random values, i.e.,

$$\mathbf{w}_{i,0} \xleftarrow{\$} \mathbb{Z}_q^N, \quad \mathbf{w}_{i,\times} \xleftarrow{\$} \mathbb{Z}_q^N, \quad (i > L)\ \mathbf{u}_i \xleftarrow{\$} \mathbb{Z}_q^N, \quad \widetilde{w}_{\mathrm{out},0} \xleftarrow{\$} \mathbb{Z}_q, \quad \widetilde{w}_{\mathrm{out},\times} \xleftarrow{\$} \mathbb{Z}_q.$$

  $\mathsf{H}_3 \approx \mathsf{H}_4$ follows from $\mathsf{FlipLWE}_{N,N(L+|C|)+2,q,\sigma'}$.

  For this step, we rely on the "straight-forwardness", i.e., r_{GenF}, which must be produced by the reduction algorithm for the underlying adversary, is efficiently sampleable conditioned on and given $\mathbf{W}_{i,0}, \mathbf{W}_{i,\times}, \mathbf{U}_i$'s and $\widetilde{\mathbf{W}}_{|C|,0}, \widetilde{\mathbf{W}}_{|C|,\times}$, which are LWE public matrices received by the reduction algorithm.
- In H_5, instead of sampling $\mathbf{w}_{i,0}$'s and $\mathbf{u}_i$'s, we sample and set

$$\begin{aligned} & \mathbf{w}_{i,x_i} \xleftarrow{\$} \mathbb{Z}_q^N, && \mathbf{w}_{i,0} = \mathbf{w}_{i,x_i} - x_i\mathbf{w}_{i,\times}, \\ (i > L) \qquad & \overline{\mathbf{u}}_i \xleftarrow{\$} \mathbb{Z}_q^N, && \mathbf{u}_i = \overline{\mathbf{u}}_i - x_{\mathsf{gin}[i,1]}\mathbf{w}_{i,\times}, \end{aligned}$$

  where x_i's are the wire values in the evaluation of $C(\mathbf{x})$. This is simply a change of variable, so $\mathsf{H}_4 \equiv \mathsf{H}_5$.

  Approximations of $\mathbf{w}_{i,x_i}$'s would be computed during an honest evaluation — we call $\mathbf{w}_{i,x_i}$'s the "active" shrunken labels. Note that the input labels given to the adversary are just the active ones, i.e., $\mathbf{w}_\ell = \mathbf{w}_{i,0} + \mathbf{x}[\ell]\mathbf{w}_{i,\times} + \mathbf{e}_\ell = \mathbf{w}_{i,x_i} + \mathbf{e}_\ell$. For the "active" expanded labels, we use a case analysis. If gate i is addition,

$$\begin{aligned} \widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1],x_{\mathsf{gin}[i,1]}} &= \widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1],0} + x_{\mathsf{gin}[i,1]}\widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1],\times} = \mathbf{u}_i + x_{\mathsf{gin}[i,1]}\mathbf{w}_{i,\times} = \overline{\mathbf{u}}_i, \\ \widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2],x_{\mathsf{gin}[i,2]}} &= \widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2],0} + x_{\mathsf{gin}[i,2]}\widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2],\times} \\ &= (\mathbf{w}_{i,0} - \mathbf{u}_i) + x_{\mathsf{gin}[i,2]}\mathbf{w}_{i,\times} \\ &= \mathbf{w}_{i,0} + (x_{\mathsf{gin}[i,1]} + x_{\mathsf{gin}[i,2]})\mathbf{w}_{i,\times} - (\mathbf{u}_i + x_{\mathsf{gin}[i,1]}\mathbf{w}_{i,\times}) \\ (x_i = x_{\mathsf{gin}[i,1]} + x_{\mathsf{gin}[i,2]}) \quad &= \mathbf{w}_{i,x_i} - \overline{\mathbf{u}}_i. \end{aligned}$$

  If gate i is multiplication,

$$\begin{aligned} \widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1],x_{\mathsf{gin}[i,1]}} &= \widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1],0} + x_{\mathsf{gin}[i,1]}\widetilde{\mathbf{w}}^{(i,1)}_{\mathsf{gin}[i,1],\times} = \mathbf{u}_i + x_{\mathsf{gin}[i,1]}\mathbf{w}_{i,\times} = \overline{\mathbf{u}}_i, \\ \widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2],x_{\mathsf{gin}[i,2]}} &= \widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2],0} + x_{\mathsf{gin}[i,2]}\widetilde{\mathbf{w}}^{(i,2)}_{\mathsf{gin}[i,2],\times} \\ &= \mathbf{w}_{i,0} - x_{\mathsf{gin}[i,2]}\mathbf{u}_i \\ &= \mathbf{w}_{i,0} + x_{\mathsf{gin}[i,1]}x_{\mathsf{gin}[i,2]}\mathbf{w}_{i,\times} - x_{\mathsf{gin}[i,2]}(\mathbf{u}_i + x_{\mathsf{gin}[i,1]}\mathbf{w}_{i,\times}) \\ (x_i = x_{\mathsf{gin}[i,1]}x_{\mathsf{gin}[i,2]}) \quad &= \mathbf{w}_{i,x_i} - x_{\mathsf{gin}[i,2]}\overline{\mathbf{u}}_i. \end{aligned}$$

The point is to see that the active expanded labels are random subject to correct evaluation into active shrunken labels, and that they are independent of values with subscript "×". Moreover, $\widetilde{\mathbf{t}}$ components with subscript "0" now become

$$\begin{aligned}(\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',0})^\intercal &= \mathbf{w}^\intercal_{i',0}\mathbf{R}^{(i,\gamma)}_{i'} + (\widetilde{\mathbf{w}}^{(i,\gamma)}_{i',0})^\intercal \\ &= (\mathbf{w}^\intercal_{i',x_{i'}} - x_{i'}\mathbf{w}^\intercal_{i',\times})\mathbf{R}^{(i,\gamma)}_{i'} + \big((\widetilde{\mathbf{w}}^{(i,\gamma)}_{i',x_{i'}})^\intercal - x_{i'}(\widetilde{\mathbf{w}}^{(i,\gamma)}_{i',\times})^\intercal\big) \\ &= \mathbf{w}^\intercal_{i',x_{i'}}\mathbf{R}^{(i,\gamma)}_{i'} + (\widetilde{\mathbf{w}}^{(i,\gamma)}_{i',x_{i'}})^\intercal - x_{i'}(\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',\times})^\intercal,\end{aligned}$$

i.e., the active expanded label $(\widetilde{\mathbf{w}}^{(i,\gamma)}_{i',x_{i'}})$ padded by the active shrunken label ($\mathbf{w}_{i',x_{i'}}$ stretched by $\mathbf{R}$), shifted by some known amount $(x_{i'}\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',\times})$.

- In H_6, additional small noises are attached to $\widetilde{\mathbf{t}}$ components with subscript "×" as follows. Sample $\mathbf{w}_{i,x_i}, \mathbf{w}_{i,\times}, \overline{\mathbf{u}}_i$'s at random, compute $\widetilde{\mathbf{w}}^{(i,\gamma)}_{x',x_{i'}}$'s as explained in H_5, and compute $\widetilde{\mathbf{w}}^{(i,\gamma)}_{x',\times}$ as usual. Sample $\overline{\mathbf{e}}^{(i,\gamma)}_{\mathrm{t},i',\times}$'s with entries independently from $\mathcal{D}_{\mathbb{Z},\sigma',\le\sigma'\sqrt{\kappa}}$ and set

$$(\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',\times})^\intercal = \mathbf{w}^\intercal_{i',\times}\mathbf{R}^{(i,\gamma)}_{i'} + (\widetilde{\mathbf{w}}^{(i,\gamma)}_{i',\times})^\intercal + (\overline{\mathbf{e}}^{(i,\gamma)}_{\mathrm{t},i',\times})^\intercal.$$

  Compute $\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',0}$'s as explained in H_5, and lastly the revealed components (secret, input labels, garbled table). The magnitude of inserted noises in the revealed components (in fact, just the randomized garbled table $\mathbf{t}$) is bounded by

$$\sigma'\sqrt{\kappa}\cdot\max\{|x_i|\} \le (N+M+2)\sigma'\sqrt{\kappa} \le 2^{-\kappa-6}\sigma,$$

  so $\mathsf{H}_5 \approx_\mathrm{s} \mathsf{H}_6$ by Lemma 2.

- In H^0_7, the small noises in $\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',\times}$'s are no longer truncated, i.e., entries of $\overline{\mathbf{e}}^{(i,\gamma)}_{\mathrm{t},i',\times}$ follow $\mathcal{D}_{\mathbb{Z},\sigma'}$. We have $\mathsf{H}_6 \approx_\mathrm{s} \mathsf{H}^0_7$ by Lemma 1.

- In $\mathsf{H}^{\mathfrak{i}'}_7$ (for $\mathfrak{i}'$ from 1 up to $|C|$), the components $\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',\times}$ with $i' \le \mathfrak{i}'$ are replaced by random. Comparing $\mathsf{H}^{\mathfrak{i}'-1}_7$ and $\mathsf{H}^{\mathfrak{i}'}_7$, the only change is

$$\begin{array}{lll}(\widetilde{\mathbf{t}}^{(i,\gamma)}_{\mathfrak{i}',\times})^\intercal & \text{from} & \mathbf{w}^\intercal_{\mathfrak{i}',\times}\mathbf{R}^{(i,\gamma)}_{\mathfrak{i}'} + (\widetilde{\mathbf{w}}^{(i,\gamma)}_{\mathfrak{i}',\times})^\intercal + (\overline{\mathbf{e}}^{(i,\gamma)}_{\mathrm{t},\mathfrak{i}',\times})^\intercal \quad \text{in } \mathsf{H}^{\mathfrak{i}'-1}_7 \\ & \text{to} & \text{random} \qquad\qquad\qquad\qquad\qquad\qquad\quad \text{in } \mathsf{H}^{\mathfrak{i}'}_7, \text{ where } \mathsf{gin}[i,\gamma] = \mathfrak{i}'.\end{array}$$

  The indistinguishability follows from flipped LWE with secret $\mathbf{w}_{\mathfrak{i}',\times}$ and public matrix $(\ldots, \mathbf{R}^{(i,\gamma)}_{\mathfrak{i}'}, \ldots)_{\mathsf{gin}[i,\gamma]=\mathfrak{i}'}$. To verify that the reduction works, we inspect where $\mathbf{w}_{\mathfrak{i}',\times}$ might appear in the revealed components.

  - The secret is $w_\mathrm{out} = \widetilde{w}_{\mathrm{out},0} + e_\mathrm{out}$, independent of $\mathbf{w}_{\mathfrak{i}',\times}$.
  - The input labels are $\mathbf{w}_\ell = \mathbf{w}_{\ell,x_\ell} + \mathbf{e}_\ell$, independent of $\mathbf{w}_{\mathfrak{i}',\times}$.
  - The garbled table is $\widetilde{\mathbf{t}}$ plus randomizing noises (without bars, independent of $\mathbf{w}_{\mathfrak{i}',\times}$). The $\widetilde{\mathbf{t}}$ components with subscript "×" require a case analysis.
    * For $i' < \mathfrak{i}'$, the $\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',\times}$'s are already replaced by random, hence independent of $\mathbf{w}_{\mathfrak{i}',\times}$.

* For $i' \geq \mathfrak{i}'$, we have

$$(\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',\times})^\intercal = \mathbf{w}^\intercal_{i',\times}\mathbf{R}^{(i,\gamma)}_{i'} + (\widetilde{\mathbf{w}}^{(i,\gamma)}_{i',\times})^\intercal + (\overline{\mathbf{e}}^{(i,\gamma)}_{\mathrm{t},i',\times})^\intercal.$$

The first and third terms are either received together as input to the reduction algorithm (for $i' = \mathfrak{i}'$) or independent of $\mathbf{w}_{\mathfrak{i}',\times}$ (for $i' > \mathfrak{i}'$). The second term cannot contain $\mathbf{w}_{\mathfrak{i}',\times}$ since by construction (see H_1 and how $\overline{\mathbf{u}}$'s are set in H_5), it is computed from $\mathbf{w}_{i,\times}, \overline{\mathbf{u}}_i$ with subscript $i > i' \geq \mathfrak{i}'$ (topological sorting implies $i > i'$) as well as the wiring of C and the wire values of $C(\mathbf{x})$.

The $\widetilde{\mathbf{t}}$ components with subscript "0", as explained in H_5, are computed from $\mathbf{w}_{i,x_i}$'s and $\widetilde{\mathbf{w}}^{(i,\gamma)}_{i',x_{i'}}$'s (independent of $\mathbf{w}_{\mathfrak{i}',\times}$) and $\widetilde{\mathbf{t}}$ components with subscript "$\times$" (analyzed above, either received or sampled by reduction).

Therefore, $\mathsf{H}^{\mathfrak{i}'-1}_7 \approx \mathsf{H}^{\mathfrak{i}'}_7$ follows from $\mathsf{FlipLWE}_{N,m',q,\sigma'}$ (the minimum required m' varies by how gate $\mathfrak{i}'$ is connected in C; bounded by some fixed $\mathrm{poly}(N, |C|)$). This step also relies on the "straight-forwardness".

- In $\mathsf{H}^{|C|+1}_8$, we replace $\widetilde{\mathbf{t}}_{|C|,0}$ (actually a scalar) by random. Comparing $\mathsf{H}^{|C|}_7$ and $\mathsf{H}^{|C|+1}_8$, the only change is

$$\begin{array}{rcll} \widetilde{\mathbf{t}}^\intercal_{|C|,0} & \text{from} & \mathbf{w}^\intercal_{|C|,x_{|C|}}\mathbf{R}_{|C|} + \widetilde{\mathbf{w}}^\intercal_{|C|,x_{|C|}} - x_{|C|}\widetilde{\mathbf{t}}^\intercal_{|C|,\times} & \text{in } \mathsf{H}^{|C|}_7 \\ & \text{to} & \text{random} & \text{in } \mathsf{H}^{|C|+1}_8. \end{array}$$

Note that the active expanded label is $\widetilde{\mathbf{w}}_{|C|,x_{|C|}} = \widetilde{w}_{|C|,0} + x_{|C|}\widetilde{w}_{|C|,\times}$. Since q is a prime, $x_{|C|} = C(\mathbf{x}) \neq 0$ (constraint for security), and $\widetilde{w}_{|C|,\times}$ is not used elsewhere in the revealed components (secret, input labels, garbled table except $\mathbf{t}_{|C|,0}$), it follows that $x_{|C|}\widetilde{w}_{|C|,\times}$ is a one-time pad for $\widetilde{\mathbf{t}}_{|C|,0}$, so $\mathsf{H}^{|C|}_7 \equiv \mathsf{H}^{|C|+1}_8$.

- In $\mathsf{H}^{\mathfrak{i}}_8$ (for $\mathfrak{i}$ from $|C|$ down to $(L+1)$), the components $\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',0}$ with $i \geq \mathfrak{i}$ (note that the indices being compared are $i, \mathfrak{i}$, not $i', \mathfrak{i}'$ as in $\mathsf{H}^{\mathfrak{i}'}_7$'s) are replaced by random. Comparing $\mathsf{H}^{\mathfrak{i}+1}_8$ and $\mathsf{H}^{\mathfrak{i}}_8$, the only change is

$$\begin{array}{rcll} (\widetilde{\mathbf{t}}^{(\mathfrak{i},\gamma)}_{\mathsf{gin}[\mathfrak{i},\gamma],0})^\intercal & \text{from} & \mathbf{w}^\intercal_{\mathsf{gin}[\mathfrak{i},\gamma],x_{\mathsf{gin}[\mathfrak{i},\gamma]}}\mathbf{R}^{(\mathfrak{i},\gamma)}_{\mathsf{gin}[\mathfrak{i},\gamma]} + (\widetilde{\mathbf{w}}^{(\mathfrak{i},\gamma)}_{\mathsf{gin}[\mathfrak{i},\gamma],x_{\mathsf{gin}[\mathfrak{i},\gamma]}})^\intercal - x_{\mathsf{gin}[\mathfrak{i},\gamma]}(\widetilde{\mathbf{t}}^{(\mathfrak{i},\gamma)}_{\mathsf{gin}[\mathfrak{i},\gamma],\times})^\intercal & \\ & & & \text{in } \mathsf{H}^{\mathfrak{i}+1}_8 \\ & \text{to} & \text{random} & \text{in } \mathsf{H}^{\mathfrak{i}}_8, \text{ where } \gamma \in \{1,2\}. \end{array}$$

Observe that

$$(\widetilde{\mathbf{w}}^{(\mathfrak{i},1)}_{\mathsf{gin}[\mathfrak{i},1],x_{\mathsf{gin}[\mathfrak{i},1]}}, \widetilde{\mathbf{w}}^{(\mathfrak{i},2)}_{\mathsf{gin}[\mathfrak{i},2],x_{\mathsf{gin}[\mathfrak{i},2]}}) = \begin{cases} (\overline{\mathbf{u}}_\mathfrak{i}, \mathbf{w}_{\mathfrak{i},x_\mathfrak{i}} - \overline{\mathbf{u}}_\mathfrak{i}), & \text{if gate } i \text{ is addition;} \\ (\overline{\mathbf{u}}_\mathfrak{i}, \mathbf{w}_{\mathfrak{i},x_\mathfrak{i}} - x_{\mathsf{gin}[\mathfrak{i},2]}\overline{\mathbf{u}}_\mathfrak{i}), & \text{if gate } i \text{ is multiplication;} \end{cases}$$

so $\overline{\mathbf{u}}_\mathfrak{i}, \mathbf{w}_{\mathfrak{i},x_\mathfrak{i}}$ can serve as one-time pads if they do not appear elsewhere in the revealed components. We inspect their appearances.

  - The secret is $w_{\mathrm{out}} = \widetilde{w}_{\mathrm{out},0} + e_{\mathrm{out}}$, no appearance.
  - The input labels are $\mathbf{w}_\ell = \mathbf{w}_{\ell,x_\ell} + \mathbf{e}_\ell$, no appearance (note $\mathfrak{i} > L \geq \ell$).

– The garbled table is $\widetilde{\mathbf{t}}$ plus randomizing noises (without bars, no $\overline{\mathbf{u}}_{\mathfrak{i}}, \mathbf{w}_{\mathfrak{i},x_{\mathfrak{i}}}$). The $\widetilde{\mathbf{t}}$ components with subscript "×" are already replaced by random, so no appearance. The $\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',0}$ components require a case analysis.
  * For $i > \mathfrak{i}$, they are already replaced by random, so no appearance.
  * For $i \le \mathfrak{i}$, we have

$$(\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',0})^\intercal = \mathbf{w}^\intercal_{i',x_{i'}} \mathbf{R}^{(i,\gamma)}_{i'} + (\widetilde{\mathbf{w}}^{(i,\gamma)}_{i',x_{i'}})^\intercal - x_{i'} (\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',\times})^\intercal.$$

    The first term has no appearance of $\overline{\mathbf{u}}_{\mathfrak{i}}, \mathbf{w}_{\mathfrak{i},x_{\mathfrak{i}}}$, since $i' = \mathsf{gin}[i,\gamma] < i \le \mathfrak{i}$ by topological sorting. Neither does the third term. The second term either (if $i = \mathfrak{i}$) is among the ones being one-time-padded, or (if $i < \mathfrak{i}$) cannot contain $\overline{\mathbf{u}}_{\mathfrak{i}}, \mathbf{w}_{\mathfrak{i},x_{\mathfrak{i}}}$, because by construction (see H_5), they are computed from only $\overline{\mathbf{u}}_i, \mathbf{w}_{i,x_i}$ (note $i < \mathfrak{i}$), the wiring of C, and the wire values of $C(\mathbf{x})$.

Therefore, $\mathsf{H}^{\mathfrak{i}+1}_8 \equiv \mathsf{H}^{\mathfrak{i}}_8$.

- In H_9, all the revealed components (secret, input labels, garbled table) are replaced by random. We inspect them in H^L_8.
  – The secret is $w_{\mathrm{out}} = \widetilde{w}_{\mathrm{out},0} + e_{\mathrm{out}}$ and $\widetilde{w}_{\mathrm{out},0}$ serves as a one-time pad.
  – The input labels are $\mathbf{w}_\ell = \mathbf{w}_{\ell,x_\ell} + \mathbf{e}_\ell$ and $\mathbf{w}_{\ell,x_\ell}$'s serve as one-time pads.
  – The garbled table is $\widetilde{\mathbf{t}}$ plus randomizing noises. The $\widetilde{\mathbf{t}}$ components with subscript "×" are independent and random thanks to $\mathsf{H}^{i'}_7$'s, and those with subscript "0", thanks to $\mathsf{H}^{\mathfrak{i}}_8$'s — the output component is handled in $\mathsf{H}^{|C|+1}_8$ and the others must belong to some $\widetilde{\mathbf{t}}^{(i,\gamma)}_{i',0}$ (for some $L < i \le |C|$) hence indeed handled in H^i_8.

  Therefore, $\mathsf{H}^L_8 \equiv \mathsf{H}_9$.

Clearly, H_9 is just $\mathsf{Exp}^{\mathrm{random}}_{\mathrm{garble}}$. By hybrid argument, $\mathsf{Exp}^{\mathrm{real}}_{\mathrm{garble}} \equiv \mathsf{H}_0 \approx \mathsf{H}_9 \equiv \mathsf{Exp}^{\mathrm{random}}_{\mathrm{garble}}$. □

6 Ciphertext-Policy ABE from Short Noisy Linear Garbling

In this section, we show how to generically construct a CP-ABE scheme from a noisy linear garbling scheme and an identity-based evasive IPFE scheme.

Ingredients of Construction 5. We rely on

- a noisy linear garbling scheme NLG supporting $F = \{F_{\mathsf{param}'}\}_{\mathsf{param}' \in \mathsf{Params}'}$ that is $(B_{\mathrm{in}}, B_{\mathrm{out}})$-correct, B_{short}-short, and GenNoise-secure such that the output of GenNoise is in $[-B_{\mathsf{NLG}}, B_{\mathsf{NLG}}]^\star$ (Definitions 7, 8, and 10), and
- an identity-based evasive IPFE scheme IPFE for $\mathcal{I} \supseteq \{0, 1, \ldots, L\}$ that is B_{IPFE}-correct and restricted-σ_{IPFE}-secure (Definitions 4 and 6).

Construction 5 (CP-ABE)**.** Define

$$\mathsf{Params} = \left\{ \mathsf{param} = (\mathsf{param}', 1^L) \,\middle|\, \mathsf{param}' \in \mathsf{Params}', L \in \mathbb{N} \right\}, \quad X_{\mathsf{param}',1^L} = \mathbb{Z}^L,$$
$$Y_{\mathsf{param}',1^L} = \left\{ f : \mathbb{Z}^L \to \{0,1,\bot\} \,\middle|\, f \in F_{\mathsf{param}'} \right\}, \quad P_{\mathsf{param}',1^L}(\mathbf{x}, f) = f(\mathbf{x}).$$

Our CP-ABE for P works as follows.

- $\mathsf{Setup}(\mathsf{param}', 1^L)$ runs

$$\mathsf{pp} = (q, \dots) \xleftarrow{\$} \mathsf{NLG.Setup}(\mathsf{param}'),$$

deterministically picks a suitable n_{r} based on q, and sets $K = 0$ and $Z = 2n_{\mathrm{r}}$. It then sets up the IPFE scheme by

$$(\mathsf{impk}, \mathsf{imsk}) \xleftarrow{\$} \mathsf{IPFE.Setup}(q, 1^K, 1^Z).$$

The algorithm outputs $\mathsf{mpk} = (\mathsf{pp}, 1^{n_{\mathrm{r}}}, 1^K, 1^Z, \mathsf{impk})$ and $\mathsf{msk} = (\mathsf{mpk}, \mathsf{imsk})$.

- $\mathsf{KeyGen}(\mathsf{msk}, \mathbf{x})$ samples $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_q^{n_{\mathrm{r}}}$ and sets

$$\mathbf{v}_0 \leftarrow \begin{pmatrix} \mathbf{r} \\ 1 \\ \mathbf{0}_{n_{\mathrm{r}}-1} \end{pmatrix}, \qquad \mathbf{v}_\ell \leftarrow \begin{pmatrix} \mathbf{r} \\ \mathbf{x}[\ell]\mathbf{r} \end{pmatrix} \quad \text{for all } \ell \in [L].$$

It generates IPFE secret keys

$$\begin{aligned} \mathsf{isk}_0 &\xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{imsk}, \overbrace{0}^{\text{id}}, \mathbf{v}_0, \overbrace{\perp}^{\text{structured noises, unused}}), \\ \mathsf{isk}_\ell &\xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{imsk}, \ell, \mathbf{v}_\ell, \perp) \qquad \text{for all } \ell \in [L], \end{aligned}$$

and outputs $\mathsf{sk} = (\mathbf{r}, \mathsf{isk}_0, \mathsf{isk}_1, \dots, \mathsf{isk}_L)$.

- $\mathsf{Enc}(\mathsf{mpk}, f, \mu)$ generates the noisy linear garbling of f by

$$(1^{n_f}, \mathbf{w}_{\mathrm{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell \in [L]}, \mathbf{T}, \mathbf{R}) \xleftarrow{\$} \mathsf{GenF}(\mathsf{pp}, f).$$

It samples a random secret matrix $\mathbf{S} \xleftarrow{\$} \mathbb{Z}_q^{n_f \times n_{\mathrm{r}}}$ and a sufficiently large message encoding scalar $s_{\mathrm{msg}} \xleftarrow{\$} \mathbb{Z}_q \setminus [-2 \cdot 2^{-\kappa} q, 2 \cdot 2^{-\kappa} q]$. The algorithm computes

$$\begin{aligned} \mathbf{u}_{\mathrm{msg}}^\intercal &\leftarrow (\mathbf{w}_{\mathrm{out}}^\intercal \mathbf{S}, \mu s_{\mathrm{msg}}, \mathbf{0}_{1 \times (n_{\mathrm{r}}-1)}), & \mathbf{U}_{\mathrm{t}}^\intercal &\leftarrow (\mathbf{T}^\intercal \mathbf{S}, \mathbf{0}_{\star \times n_{\mathrm{r}}}), \\ \mathbf{U}_\ell^\intercal &\leftarrow (\mathbf{W}_{\ell,0}^\intercal \mathbf{S}, \mathbf{W}_{\ell,\times}^\intercal \mathbf{S}) \qquad \text{for all } \ell \in [L]. \end{aligned}$$

Lastly, it generates IPFE ciphertexts

$$\begin{aligned} \mathsf{ict}_{\mathrm{msg}} &\xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, 0, \mathfrak{o}, \mathbf{u}_{\mathrm{msg}}^\intercal), \qquad \mathsf{ict}_{\mathrm{t}} \xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, 0, \mathfrak{o}, \mathbf{U}_{\mathrm{t}}^\intercal), \\ \mathsf{ict}_\ell &\xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, \ell, \mathfrak{o}, \mathbf{U}_\ell^\intercal) \qquad \text{for all } \ell \in [L], \end{aligned}$$

and outputs $\mathsf{ct} = (\mathbf{R}, \mathsf{ict}_{\mathrm{msg}}, \mathsf{ict}_{\mathrm{t}}, \mathsf{ict}_1, \dots, \mathsf{ict}_L)$.

- $\mathsf{Dec}(\mathsf{mpk}, \mathbf{x}, \mathsf{sk}, f, \mathsf{ct})$ outputs $\perp$ if $f(\mathbf{x}) \neq 1$. Otherwise, it parses sk, ct as in $\mathsf{KeyGen}, \mathsf{Enc}$, and recomputes $\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_L$ as in KeyGen. The algorithm decrypts all the IPFE ciphertexts for the rerandomized garbling instance, i.e.,

$$\begin{aligned} w_{\mathrm{msg}} &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, 0, \mathbf{v}_0, \perp, \mathsf{isk}_0, \mathfrak{o}, \mathsf{ict}_{\mathrm{msg}}), \\ \mathbf{t} &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, 0, \mathbf{v}_0, \perp, \mathsf{isk}_0, \mathfrak{o}, \mathsf{ict}_{\mathrm{t}}), \\ \mathbf{w}_\ell &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, \ell, \mathbf{v}_\ell, \perp, \mathsf{isk}_\ell, \mathfrak{o}, \mathsf{ict}_\ell) \qquad \text{for all } \ell \in [L]. \end{aligned}$$

It evaluates the garbling by

$$w_{\mathrm{out}} \leftarrow \mathsf{NLG.Eval}(\mathsf{pp}, f, \mathbf{R}, \mathbf{x}, \{\mathbf{w}_\ell^\intercal\}_{\ell \in [L]}, \mathbf{t}^\intercal)$$

and outputs the decryption result

$$\mu' \leftarrow \begin{cases} 0, & \text{if } w_{\mathrm{msg}} - w_{\mathrm{out}} \in [-2^{-\kappa} q, 2^{-\kappa} q], \\ 1, & \text{otherwise.} \end{cases}$$

6.1 Correctness

Theorem 18 (¶). *Construction 5 is correct (Definition 1) if*

- *the modulus q output by* $\mathsf{NLG.Setup}(\mathsf{param}')$ *is always a prime and always satisfies* $B_{\mathrm{out}}(\mathsf{param}') \le 2^{-\kappa} q$*, and*
- *it holds that* $B_{\mathsf{IPFE}}(q, 0, 2n_{\mathrm{r}}) \le B_{\mathrm{in}}(\mathsf{param}')$.

Proof (Theorem 18). The correctness of Construction 5 follows directly from the correctness of the underlying IPFE and garbling schemes. Since q is a prime, by the B_{IPFE}-correctness of the evasive IPFE scheme,

$$\begin{aligned} w_{\mathrm{msg}} &= \mathbf{u}_{\mathrm{msg}}^{\intercal}\mathbf{v}_0 + e_{\mathrm{out}} = \mathbf{w}_{\mathrm{out}}^{\intercal}\mathbf{Sr} + \mu s_{\mathrm{msg}} + e_{\mathrm{out}} = (\mathbf{Sr})^{\intercal}\mathbf{w}_{\mathrm{out}} + e_{\mathrm{out}} + \mu s_{\mathrm{msg}}, \\ \mathbf{t}^{\intercal} &= \mathbf{v}_0^{\intercal}\mathbf{U}_{\mathrm{t}} + \mathbf{e}_{\mathrm{t}}^{\intercal} = (\mathbf{Sr})^{\intercal}\mathbf{T} + \mathbf{e}_{\mathrm{t}}^{\intercal}, \\ \mathbf{w}_{\ell}^{\intercal} &= \mathbf{v}_{\ell}^{\intercal}\mathbf{U}_{\ell} + \mathbf{e}_{\ell}^{\intercal} = (\mathbf{Sr})^{\intercal}(\mathbf{W}_{\ell,0} + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_{\ell}^{\intercal}, \end{aligned}$$

where $\mathbf{Sr} \in \mathbb{Z}_q^{n_f}$ randomizes the garbling and the errors (from IPFE decryption) are bounded by $B_{\mathsf{IPFE}}(q, 0, 2n_{\mathrm{r}}) \le B_{\mathrm{in}}(\mathsf{param}')$. By the $(B_{\mathrm{in}}, B_{\mathrm{out}})$-correctness of garbling,

$$w_{\mathrm{out}} - ((\mathbf{Sr})^{\intercal}\mathbf{w}_{\mathrm{out}} + e_{\mathrm{out}}) \in [-B_{\mathrm{out}}(\mathsf{param}'), B_{\mathrm{out}}(\mathsf{param}')] \subseteq [-2^{-\kappa}q, 2^{-\kappa}q].$$

If $\mu = 0$, it always holds that $w_{\mathrm{msg}} - w_{\mathrm{out}} \in [-2^{-\kappa}q, 2^{-\kappa}q]$, which never holds when $\mu = 1$ by our choice of s_{msg}. We conclude that Construction 5 is correct. □

6.2 Security

Theorem 19 (¶). *Construction 5 is very selectively secure (Definition 2) if*[18]

- $\mathsf{LWE}_{n_{\mathrm{r}},\mathrm{poly}(\lambda),q,\sigma_{\mathrm{r}}}$ *(Assumption 1) holds for some* $\sigma_{\mathrm{r}} \le \frac{2^{-\kappa-6}\sigma_{\mathsf{IPFE}}}{\sqrt{\kappa}\cdot B_{\mathrm{short}}(\mathsf{param}')}$*, and*
- *it holds that* $2^{\kappa+6} B_{\mathsf{NLG}}(\mathsf{param}') \le \sigma_{\mathsf{IPFE}}$.

Proof (Theorem 19). Let $\mathcal{A}$ be an efficient adversary. We assume that it always chooses challenges such that $f(\mathbf{x}_j) = 0$ for all $j \in [J]$ — otherwise, we alter $\mathcal{A}$ so that when the condition fails, it resets J to zero and aborts when it receives back the ABE components (incomplete per its expectation), as its output is irrelevant to $\mathsf{Exp}^{\beta}_{\mathrm{ABE}}$ in that case. Consider the following evasive IPFE sampler $\mathcal{S}^{\beta} = (\mathcal{S}_{\mathrm{V}}, \mathcal{S}^{\beta}_{\mathrm{U}})$ per Definition 5.

- $\mathcal{S}_{\mathrm{V}}(r_{\mathrm{pub}})$ parses $r_{\mathrm{pub}} = (r_{\mathcal{A}}, r_{\mathsf{NLG.Setup}}, r_{\mathrm{r}}, r_{\mathsf{NLG.GenF}})$. It runs $\mathcal{A}(r_{\mathcal{A}})$ to obtain

  $$\mathsf{param} = (\mathsf{param}', 1^L), \qquad \{\mathbf{x}_j\}_{j\in[J]} \quad (\mathbf{x}_j \in \mathbb{Z}^L), \qquad f : \mathbb{Z}^L \to \{0, 1, \perp\}.$$

  The algorithm next sets up the noisy linear garbling by

  $$\mathsf{pp} = (q, \dots) \leftarrow \mathsf{NLG.Setup}(\mathsf{param}'; r_{\mathsf{NLG.Setup}}),$$

[18] The constants have been changed from the conference version [HLL24]. Both versions are correct. The new version fits the proof better.

and deterministically picks n_{r}, K, Z as in Setup of Construction 5. It then uses r_{r} to sample $\mathbf{r}_1, \ldots, \mathbf{r}_J \in \mathbb{Z}_q^{n_{\mathrm{r}}}$ uniformly at random in a straight-forward way.[19] The algorithm sets for all $j \in [J]$,

$$\mathbf{v}_{j,0} = \begin{pmatrix} \mathbf{r}_j \\ 1 \\ \mathbf{0}_{n_{\mathrm{r}}-1} \end{pmatrix}, \qquad \mathbf{v}_{j,\ell} = \begin{pmatrix} \mathbf{r}_j \\ \mathbf{x}_j[\ell]\mathbf{r}_j \end{pmatrix} \qquad (\text{for all } \ell \in [L]).$$

It runs

$$(1^{n_f}, \mathbf{w}_{\mathrm{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}, \mathbf{T}, \mathbf{R}) \leftarrow \mathsf{NLG.GenF}(\mathsf{pp}, f; r_{\mathsf{NLG.GenF}}).$$

Suppose $\mathbf{W}_{\ell,0} \in \mathbb{Z}_q^{n_f \times m_\ell}$ and $\mathbf{T} \in \mathbb{Z}_q^{n_f \times m_{\mathrm{t}}}$, the algorithm outputs

$$q, 1^K, 1^Z, \quad \mathsf{ID} = \{0, 1, \ldots, L\}, \quad 1^{J_{\mathsf{id}}} = 1^J, \quad 1^{I_{0,\mathfrak{o}}} = 1^{m_{\mathrm{t}}+1}, \quad 1^{I_{\ell,\mathfrak{o}}} = 1^{m_\ell},$$
$$1^{I_{\mathsf{id},\mathfrak{g}}} = 1^0, \quad \sigma_{\mathrm{pre}} = \sigma_{\mathsf{IPFE}}, \quad \mathbf{V}_{\mathsf{id},\mathfrak{o}} = (\mathbf{v}_{1,\mathsf{id}}, \ldots, \mathbf{v}_{J,\mathsf{id}}), \quad \mathbf{V}_{\mathsf{id},\mathfrak{g}} = \bot,$$

where the indices are $\mathsf{id} \in \mathsf{ID}$ and $\ell \in [L]$.

- $\mathcal{S}_{\mathrm{U}}^{\beta}(r_{\mathrm{pub}}; r_{\mathrm{priv}})$ first runs the deterministic subprocedure $\mathcal{S}_{\mathrm{A0}}^{\beta}(r_{\mathrm{pub}})$, which does the following. It first reruns $\mathcal{S}_{\mathrm{v}}(r_{\mathrm{pub}})$ to obtain $\mathbf{w}_{\mathrm{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}, \mathbf{T}$. The algorithm then rearranges them into matrices $\mathbf{A}_{\mathrm{msg}}, \{\mathbf{A}_{\mathrm{t},i}\}_{i\in[m_{\mathrm{t}}]}, \{\mathbf{A}_{\ell,i}\}_{\ell\in[L],i\in[m_\ell]}$ so that for all $\widetilde{\mathbf{s}}_1, \ldots, \widetilde{\mathbf{s}}_{n_{\mathrm{r}}} \in \mathbb{Z}_q^{n_f}$ and $s_{\mathrm{msg}} \in \mathbb{Z}_q$,

$$\mathbf{d}_0^{\mathsf{T}}\mathbf{A}_{\mathrm{msg}} = \mathbf{u}_{\mathrm{msg}}^{\mathsf{T}} = (\mathbf{w}_{\mathrm{out}}^{\mathsf{T}}\mathbf{S}, \beta s_{\mathrm{msg}}, \mathbf{0}_{1\times(n_{\mathrm{r}}-1)}),$$
$$(\mathbf{A}_{\mathrm{t},1}^{\mathsf{T}}\mathbf{d}_0, \ldots, \mathbf{A}_{\mathrm{t},m_{\mathrm{t}}}^{\mathsf{T}}\mathbf{d}_0)^{\mathsf{T}} = \mathbf{U}_{\mathrm{t}}^{\mathsf{T}} = (\mathbf{T}^{\mathsf{T}}\mathbf{S}, \mathbf{0}_{m_{\mathrm{t}}\times n_{\mathrm{r}}}),$$
$$(\mathbf{A}_{\ell,1}^{\mathsf{T}}\mathbf{d}_0, \ldots, \mathbf{A}_{\ell,m_\ell}^{\mathsf{T}}\mathbf{d}_0)^{\mathsf{T}} = \mathbf{U}_{\ell}^{\mathsf{T}} = (\mathbf{W}_{\ell,0}^{\mathsf{T}}\mathbf{S}, \mathbf{W}_{\ell,\times}^{\mathsf{T}}\mathbf{S}) \qquad (\text{for all } \ell \in [L]),$$

where $\mathbf{d}_0^{\mathsf{T}} = (s_{\mathrm{msg}}, \widetilde{\mathbf{s}}_1^{\mathsf{T}}, \ldots, \widetilde{\mathbf{s}}_{n_{\mathrm{r}}}^{\mathsf{T}})$ and $\mathbf{S} = (\widetilde{\mathbf{s}}_1, \ldots, \widetilde{\mathbf{s}}_{n_{\mathrm{r}}})$. This is possible since every entry on the right-hand side is linear in $\mathbf{d}_0$. The algorithm $\mathcal{S}_{\mathrm{A0}}$ outputs

$$1^{n'} = 1^{n_f n_{\mathrm{r}}+1}, \qquad \mathbf{A}_{0,\mathfrak{o},0,i} = \mathbf{A}_{\mathrm{t},i} \qquad (\text{for all } i \in [m_{\mathrm{t}}]),$$
$$\mathbf{A}_{0,\mathfrak{o},0,m_{\mathrm{t}}+1} = \mathbf{A}_{\mathrm{msg}}, \qquad \mathbf{A}_{\ell,\mathfrak{o},0,i} = \mathbf{A}_{\ell,i} \qquad (\text{for all } \ell \in [L],\ i \in [m_\ell]).$$

Completing $\mathcal{S}_{\mathrm{A0}}$, the algorithm $\mathcal{S}_{\mathrm{U}}$ samples $\mathbf{d}_0 \xleftarrow{\$} \mathbb{Z}_q^{n'}$ using r_{priv} and outputs

$$\mathbf{U}_{\mathsf{id},\mathfrak{o}}^{\mathsf{T}} = \begin{pmatrix} \mathbf{d}_0^{\mathsf{T}}\mathbf{A}_{\mathsf{id},\mathfrak{o},0,1} \\ \vdots \\ \mathbf{d}_0^{\mathsf{T}}\mathbf{A}_{\mathsf{id},\mathfrak{o},0,I_{\mathsf{id},\mathfrak{o}}} \end{pmatrix} \qquad \text{for all } \mathsf{id} \in \mathsf{ID}.$$

By definition, $\mathcal{S}^{\beta}$ is a restricted sampler (Definition 6). We have the following:

Claim 20 (¶). *$\mathcal{S}^{\beta}$ has pseudorandom noisy inner products, i.e., $\mathsf{IPFEsec}_{\mathrm{pre}}^{\mathcal{S}^{\beta}}$ (Definition 5) holds for both $\beta \in \{0, 1\}$.*

The very selective security of Construction 5 follows from Claim 20. Consider the following hybrids.

[19] Precisely speaking, r_{r} must be efficiently sampleable conditioned on and given $\mathbf{r}_1, \ldots, \mathbf{r}_J$.

- H_0^β is $\mathsf{Exp}_{\mathrm{ABE}}^\beta$ for Construction 5. Note that s_{msg} in the challenge ciphertext is uniformly random over $\mathbb{Z}_q \setminus [-2 \cdot 2^{-\kappa} q, 2 \cdot 2^{-\kappa} q]$.
- In H_1^β, the scalar s_{msg} is changed to be uniformly random over $\mathbb{Z}_q$. Clearly, $\mathsf{H}_0^\beta \approx_{\mathrm{s}} \mathsf{H}_1^\beta$. Moreover, H_1^β corresponds to the left distribution of $\mathsf{IPFEsec}_{\mathrm{post}}^{\mathcal{S}^\beta}$.
- In H_2^β, the vectors inside IPFE ciphertexts (components of the ABE challenge ciphertext) are replaced by random, which corresponds to the right distribution of $\mathsf{IPFEsec}_{\mathrm{post}}^{\mathcal{S}^\beta}$. By Claim 20 and the restricted-σ_{IPFE}-security of IPFE, we have $\mathsf{H}_1^\beta \approx \mathsf{H}_2^\beta$.

Lastly, $\mathsf{H}_2^0 \equiv \mathsf{H}_2^1$. By hybrid argument, we conclude $\mathsf{Exp}_{\mathrm{ABE}}^0 \approx \mathsf{Exp}_{\mathrm{ABE}}^1$, completing the proof. □

Proof (Claim 20). Fix $\beta \in \{0,1\}$ and consider the following hybrids.

- H_0 is the left distribution of $\mathsf{IPFEsec}_{\mathrm{pre}}^{\mathcal{S}^\beta}$. Recall that the public randomness is $r_{\mathrm{pub}} = (r_{\mathcal{A}}, r_{\mathsf{NLG.Setup}}, r_{\mathrm{r}}, r_{\mathsf{NLG.GenF}})$. Adopting the notations of the ABE decryption algorithm, the noisy inner products are

$$\begin{aligned} w_{\mathrm{msg},j} &= \mathbf{u}_{\mathrm{msg}}^\intercal \mathbf{v}_{j,0} + e_{\mathrm{out},j} = (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{w}_{\mathrm{out}} + \beta s_{\mathrm{msg}} + e_{\mathrm{out},j}, \\ \mathbf{t}_j^\intercal &= \mathbf{v}_{j,0}^\intercal \mathbf{U}_{\mathrm{t}} + \mathbf{e}_{\mathrm{t},j}^\intercal = (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{T} + \mathbf{e}_{\mathrm{t},j}^\intercal, \\ \mathbf{w}_{j,\ell}^\intercal &= \mathbf{v}_{j,\ell}^\intercal \mathbf{U}_\ell + \mathbf{e}_{j,\ell}^\intercal = (\mathbf{S}\mathbf{r}_j)^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_{j,\ell}^\intercal \qquad (\text{for all } \ell \in [L]), \end{aligned}$$

  where $j \in [J]$ and the entries of $\mathbf{e}$'s are independent $\mathcal{D}_{\mathbb{Z},\sigma_{\mathsf{IPFE}}}$.
- In H_1, additional small noises are attached to $\{\mathbf{S}\mathbf{r}_j\}_{j\in[J]}$. For all $j \in [J]$, sample $\mathbf{e}_{\mathrm{r},j} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{r}},\le\sigma_{\mathrm{r}}\sqrt{\kappa}}^{n_f}$ and set the inner products to

$$\begin{aligned} w_{\mathrm{msg},j} &= (\mathbf{S}\mathbf{r}_j + \mathbf{e}_{\mathrm{r},j})^\intercal \mathbf{w}_{\mathrm{out}} + \beta s_{\mathrm{msg}} + e_{\mathrm{out},j}, & \mathbf{t}_j^\intercal &= (\mathbf{S}\mathbf{r}_j + \mathbf{e}_{\mathrm{r},j})^\intercal \mathbf{T} + \mathbf{e}_{\mathrm{t},j}^\intercal, \\ \mathbf{w}_{j,\ell}^\intercal &= (\mathbf{S}\mathbf{r}_j + \mathbf{e}_{\mathrm{r},j})^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_{j,\ell}^\intercal & &(\text{for all } \ell \in [L]). \end{aligned}$$

  The errors introduced into the inner products are bounded by

$$B_{\mathrm{short}}(\mathsf{param}') \cdot \sigma_{\mathrm{r}}\sqrt{\kappa} \le 2^{-\kappa-6}\sigma_{\mathsf{IPFE}},$$

  so they are flooded (Lemma 2) by $\{e_{\mathrm{out},j}, \mathbf{e}_{\mathrm{t},j}, \{\mathbf{e}_{j,\ell}\}_{\ell\in[L]}\}_{j\in[J]}$, and $\mathsf{H}_0 \approx_{\mathrm{s}} \mathsf{H}_1$.
- In H_2, the $\mathbf{e}_{\mathrm{r},j}$'s are no longer truncated, i.e., $\mathbf{e}_{\mathrm{r},j} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{r}}}^{n_f}$. We have $\mathsf{H}_1 \approx_{\mathrm{s}} \mathsf{H}_2$ by Lemma 1.
- In H_3, the garblings use uniform randomness. For all $j \in [J]$, sample $\mathbf{s}_j \xleftarrow{\$} \mathbb{Z}_q^{n_f}$ and set the inner products to

$$\begin{aligned} w_{\mathrm{msg},j} &= \mathbf{s}_j^\intercal \mathbf{w}_{\mathrm{out}} + \beta s_{\mathrm{msg}} + e_{\mathrm{out},j}, & \mathbf{t}_j^\intercal &= \mathbf{s}_j^\intercal \mathbf{T} + \mathbf{e}_{\mathrm{t},j}^\intercal, \\ \mathbf{w}_{j,\ell}^\intercal &= \mathbf{s}_j^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_{j,\ell}^\intercal & &(\text{for all } \ell \in [L]). \end{aligned}$$

  By n_f hybrids of applying $\mathsf{LWE}_{n_{\mathrm{r}},J,q,\sigma_{\mathrm{r}}}$ over the rows of $\mathbf{S}$ with public matrix $(\mathbf{r}_1, \ldots, \mathbf{r}_J)$, we have

$$\mathbf{S}(\mathbf{r}_1, \ldots, \mathbf{r}_J) + (\mathbf{e}_{\mathrm{r},1}, \ldots, \mathbf{e}_{\mathrm{r},J}) \approx (\mathbf{s}_1, \ldots, \mathbf{s}_J),$$

  which implies $\mathsf{H}_2 \approx \mathsf{H}_3$. In this step, we rely on the "straight-forwardness" of sampling $\mathbf{r}_j$'s from r_{r} in $\mathcal{S}^\beta$.

- In H_4, additional noises generated by GenNoise are attached to the garblings. For all $j \in [J]$, run

$$(\overline{e}_{\mathrm{out},j}, \{\overline{\mathbf{e}}_{j,\ell}\}_{\ell\in[L]}, \overline{\mathbf{e}}_{\mathrm{t},j}) \xleftarrow{\$} \mathsf{GenNoise}(\mathsf{param}', \mathsf{pp}, f, \mathbf{w}_{\mathrm{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}\mathbf{T}, \mathbf{R}, \mathbf{x}_j),$$

  and set the inner products to

$$\begin{aligned} w_{\mathrm{msg},j} &= \mathbf{s}_j^\intercal \mathbf{w}_{\mathrm{out}} + \overline{e}_{\mathrm{out},j} + \beta s_{\mathrm{msg}} + e_{\mathrm{out},j}, & \mathbf{t}_j^\intercal &= \mathbf{s}_j^\intercal \mathbf{T} + \overline{\mathbf{e}}_{\mathrm{t},j}^\intercal + \mathbf{e}_{\mathrm{t},j}^\intercal, \\ \mathbf{w}_{j,\ell}^\intercal &= \mathbf{s}_j^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \overline{\mathbf{e}}_{j,\ell}^\intercal + \mathbf{e}_{j,\ell}^\intercal & &\text{(for all } \ell \in [L]). \end{aligned}$$

  The $\overline{e}, \overline{\mathbf{e}}$'s introduced into the inner products are bounded by $B_{\mathsf{NLG}} \le 2^{-\kappa-6}\sigma_{\mathsf{IPFE}}$, so they are flooded (Lemma 2) by $e, \mathbf{e}$'s. We have $\mathsf{H}_3 \approx_s \mathsf{H}_4$.

- In H_5, all the inner products are replaced by random, i.e., the right distribution of $\mathsf{IPFEsec}_{\mathrm{pre}}^{\mathcal{S}^\beta}$. Note that the inner products in H_4 are well-randomized garblings (by $\mathbf{s}, \overline{e}, \overline{\mathbf{e}}$'s) plus independent noises ($e, \mathbf{e}$'s). Therefore, by J hybrids of applying GenNoise-security of NLG, we conclude $\mathsf{H}_4 \approx \mathsf{H}_5$.

By hybrid argument, $\mathsf{H}_0 \approx \mathsf{H}_5$, i.e., $\mathsf{IPFEsec}_{\mathrm{pre}}^{\mathcal{S}^\beta}$ holds. □

6.3 Unbounded Attribute and Summary

Unbounded Attribute. Construction 5 can be modified for attributes of unbounded length. Before presenting the scheme, we shall clarify the predicate family of such a scheme, in particular, what $P(\mathbf{x}, f)$ is when $\mathbf{x}$ is shorter than an input to f. There are multiple approaches in the literature. The following is the discussion in [HLL23b; Section 5.4]. For monotone functions, the missing attribute bits can be assumed to be zero. For general functions: the specification is silent in many works, which should be interpreted as $P(\mathbf{x}, f) = \perp$ (neither correct nor secure); it might require exact length matching, or only reject $\mathbf{x}$ shorter than input to f (prefix-matching), or attach index sets to $\mathbf{x}, f$ and require equal-match or subset-match.

The "silent" specification is straight-forward to achieve. We explain how to achieve prefix-matching. For simplicity, we require that $f(\mathbf{x}) \neq \perp$ for all $\mathbf{x} \in \{0,1\}^L$. The predicate family is

$$\begin{aligned} \mathsf{Params} &= \{\, \mathsf{param} = \mathsf{param}' \mid \mathsf{param}' \in \mathsf{Params}', \,\}, \\ X_{\mathsf{param}'} &= \mathbb{Z}^{<2^\lambda}, \qquad Y_{\mathsf{param}'} = \{\, f : \mathbb{Z}^L \to \{0,1,\perp\} \mid f \in F_{\mathsf{param}'} \,\}, \\ P_{\mathsf{param}'}(\mathbf{x}, f) &= \begin{cases} f(\mathbf{x}'), & \text{if the input length of } f \text{ is } L \text{ and } \mathbf{x}' \text{ is the } L\text{-prefix of } \mathbf{x}; \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

Construction 5 with L (textually) removed is already quite close to an unbounded ABE — it becomes one with the following modifications similar to [HLL23b].

- The identity space satisfies $\mathcal{I} \supseteq \{0, 1, \ldots, \lambda, \lambda+1, \ldots, \lambda+2^\lambda-1\}$.
- When generating a key for $\mathbf{x}$, instead generate a key for $(L_{\mathbf{x}}, \mathbf{x})$, where $L_{\mathbf{x}}$ is the λ-bit encoding of $\mathbf{x}$.

- When generating a ciphertext for f, instead generate a ciphertext for

$$f'(L_{\mathrm{x}}, \mathbf{x}') = \begin{cases} 0, & \text{if } L_{\mathrm{x}} \text{ is less than the input length of } f; \\ f(\mathbf{x}'), & \text{otherwise.} \end{cases}$$

- The security reduces to that of Construction 5 by appropriately appending zeros to every key attribute.

Summary. By instantiating our CP-ABE with the garbling scheme of Construction 4 and the IPFE scheme of Construction 2, we obtain CP-ABE for bounded-arithmetic circuits.

Corollary 21. *Under LWE (Assumption 1) and evasive LWE (Assumption 4), there exist CP-ABE schemes (bounded/unbounded in attribute length) for bounded-arithmetic circuits with* $|\mathsf{mpk}| = \mathrm{poly}(\lambda, \log M, d)$ *and*

$$|\mathsf{sk}_{\mathbf{x}}| = (|\mathbf{x}| + 1)\,\mathrm{poly}(\lambda, \log M, d), \qquad |\mathsf{ct}_C| = (|C| + 1)\,\mathrm{poly}(\lambda, \log M, d),$$

where M is the maximum wire value and d is the maximum depth.

7 ABE for DFA

We define DFA in a minimal format convenient for garbling.

Definition 11 (DFA)**.** A *deterministic finite automaton* Γ over the binary alphabet is a tuple $(1^{\mathfrak{Q}}, \mathbf{\Gamma}_0, \mathbf{\Gamma}_1, \boldsymbol{\xi})$, where $\mathfrak{Q} \in \mathbb{N}$ is its *number of states*, $\mathbf{\Gamma}_0, \mathbf{\Gamma}_1 \in \{0,1\}^{\mathfrak{Q}\times\mathfrak{Q}}$ are its *transition matrices*, $\boldsymbol{\xi} \in \{0,1\}^{\mathfrak{Q}}$ is its *rejection state vector*, and each column of $\mathbf{\Gamma}_0, \mathbf{\Gamma}_1$ contains exactly one 1. The DFA *accepts* an input $\mathbf{x} \in \{0,1\}^L$ if and only if

$$\boldsymbol{\xi}^{\mathsf{T}} \cdot \prod_{\ell=L}^{1} \mathbf{\Gamma}_{\mathbf{x}[\ell]} \cdot \boldsymbol{\iota}_1 = 0.$$

Implicitly, for Γ, the set of states is $[\mathfrak{Q}]$, the initial state is 1, and the set of accept states is $\{\, \mathfrak{q} \in [\mathfrak{Q}] \mid \boldsymbol{\xi}[\mathfrak{q}] = 0 \,\}$. After reading $x \in \{0,1\}$, the machine transitions from $\mathfrak{q}$ to $\mathfrak{q}'$ if and only if $\mathbf{\Gamma}_x[\mathfrak{q}', \mathfrak{q}] = 1$, which is equivalent to $\mathbf{\Gamma}_x \boldsymbol{\iota}_{\mathfrak{q}} = \boldsymbol{\iota}_{\mathfrak{q}'}$ (recall that $\boldsymbol{\iota}_{\mathfrak{q}} \in \{0,1\}^{\mathfrak{Q}}$ is the $\mathfrak{q}^{\text{th}}$ standard basis vector). The following lemma, readily verified, will be handy:

Lemma 22. *For all DFA* $(1^{\mathfrak{Q}}, \mathbf{\Gamma}_0, \mathbf{\Gamma}_1, \boldsymbol{\xi})$ *and input* $\mathbf{x} \in \{0,1\}^L$*, it holds that*

$$\left\| \prod_{\ell=1}^{L} \mathbf{\Gamma}_{\mathbf{x}[\ell]}^{\mathsf{T}} \right\| \le 1, \qquad \boldsymbol{\xi}^{\mathsf{T}} \cdot \prod_{\ell=L}^{1} \mathbf{\Gamma}_{\mathbf{x}[\ell]} \cdot \boldsymbol{\iota}_1 \in \{0,1\}.$$

7.1 Noisy Linear Garbling for DFA

DFA is defined for inputs of arbitrary length, yet our notion of garbling is only for functions with fixed-length input. Following the paradigm in [LL20a], we consider the garbling scheme for each possible input length. The existing [Wat12,LL20a] linear secret sharing scheme for DFA can be cast into a noisy linear garbling (Definition 7).

Construction 6 (DFA garbling)**.** Let[20]

$$\mathsf{Params} = \left\{ (q, \overline{L}) \,\middle|\, q, \overline{L} \in \mathbb{N},\ q \geq 2 \right\}, \quad F = \{F_{\mathsf{param}}\}_{\mathsf{param} \in \mathsf{Params}},$$
$$F_{\mathsf{param}} = \left\{ f_{\Gamma,L} : \mathbb{Z}^L \to \{0,1,\bot\} \,\middle|\, \Gamma \text{ is a DFA and } L \in [0, \overline{L}] \right\},$$
$$f_{\Gamma,L}(\mathbf{x}) = \begin{cases} 1, & \text{if } \mathbf{x} \in \{0,1\}^L \text{ and } \Gamma \text{ accepts } \mathbf{x}; \\ 0, & \text{if } \mathbf{x} \in \{0,1\}^L \text{ and } \Gamma \text{ rejects } \mathbf{x}; \\ \bot, & \text{if } \mathbf{x} \notin \{0,1\}^L. \end{cases}$$

The function $f_{\Gamma,L}$ is represented by $(\Gamma, 1^L)$. The noisy linear garbling scheme for F works as follows.

- $\mathsf{Setup}(\mathsf{param})$ outputs $\mathsf{pp} = \mathsf{param} = (q, \overline{L})$.
- $\mathsf{GenF}(\mathsf{pp}, f_{\Gamma,L})$ parses $\Gamma = (1^{\mathfrak{Q}}, \mathbf{\Gamma}_0, \mathbf{\Gamma}_1, \boldsymbol{\xi})$, sets

$$n_f = (L+1)\mathfrak{Q} + 1, \qquad \mathbf{w}_{\mathsf{out}}^\intercal = (-\boldsymbol{\iota}_1^\intercal, \mathbf{0}_{1 \times L\mathfrak{Q}}, 0), \qquad \mathbf{R} = \bot,$$
$$\mathbf{W}_{\ell,0} = \begin{pmatrix} \mathbf{0}_{(\ell-1)\mathfrak{Q} \times \mathfrak{Q}} \\ -\mathbf{I}_{\mathfrak{Q}} \\ \mathbf{\Gamma}_0 \\ \mathbf{0}_{(L-\ell)\mathfrak{Q} \times \mathfrak{Q}} \\ \mathbf{0}_{1 \times \mathfrak{Q}} \end{pmatrix}, \qquad \mathbf{W}_{\ell,\times} = \begin{pmatrix} \mathbf{0}_{(\ell-1)\mathfrak{Q} \times \mathfrak{Q}} \\ \mathbf{0}_{\mathfrak{Q} \times \mathfrak{Q}} \\ \mathbf{\Gamma}_1 - \mathbf{\Gamma}_0 \\ \mathbf{0}_{(L-\ell)\mathfrak{Q} \times \mathfrak{Q}} \\ \mathbf{0}_{1 \times \mathfrak{Q}} \end{pmatrix}, \qquad \mathbf{T} = \begin{pmatrix} \mathbf{0}_{L\mathfrak{Q} \times \mathfrak{Q}} \\ \\ -\mathbf{I}_{\mathfrak{Q}} \\ \boldsymbol{\xi}^\intercal \end{pmatrix},$$

  and outputs $(1^{n_f}, \mathbf{w}_{\mathsf{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell \in [L]}, \mathbf{T}, \mathbf{R})$. Here, $\boldsymbol{\iota}_1 \in \{0,1\}^{\mathfrak{Q}}$.
- $\mathsf{Eval}(\mathsf{pp}, f_{\Gamma,L}, \mathbf{R}, \mathbf{x}, \{\mathbf{w}_\ell^\intercal\}_{\ell \in [L]}, \mathbf{t}^\intercal)$ computes and outputs

$$\mathbf{t}^\intercal \cdot \prod_{\ell'=L}^{1} \mathbf{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 + \sum_{\ell=L}^{1} \left(\mathbf{w}_\ell^\intercal \cdot \prod_{\ell'=\ell-1}^{1} \mathbf{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 \right).$$

Theorem 23 (¶)**.** *Construction 6 is* $(B_{\mathsf{in}}, B_{\mathsf{out}})$*-correct (Definition 7) if* $B_{\mathsf{out}} \geq (\overline{L}+2)B_{\mathsf{in}}$*,* 2*-short (Definition 8), and* GenNoise*-secure for all* GenNoise *(Definition 10).*

Proof (Theorem 23). The proof is basically that in [LL20a] plus handling the noise. It suffices to consider $\mathbf{x} \in \{0,1\}^L$. Note that (the second by Lemma 22)

$$\begin{aligned} \mathbf{w}_{\mathsf{out}}^\intercal = (-\boldsymbol{\iota}_1^\intercal, \mathbf{0}) &\implies \|\mathbf{w}_{\mathsf{out}}^\intercal\| \leq 1 \qquad \leq 2, \\ \mathbf{W}_{\ell,0}^\intercal + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times}^\intercal = (\mathbf{0}, -\mathbf{I}_{\mathfrak{Q}}, \mathbf{\Gamma}_{\mathbf{x}[\ell]}^\intercal, \mathbf{0}) &\implies \|\mathbf{W}_{\ell,0}^\intercal + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times}^\intercal\| \leq 1+1 \leq 2, \\ \mathbf{T}^\intercal = (\mathbf{0}, -\mathbf{I}_{\mathfrak{Q}}, \boldsymbol{\xi}) &\implies \|\mathbf{T}^\intercal\| \leq 1+1 \leq 2, \end{aligned}$$

so the scheme is 2-short.

Let the garbling randomness/noise be

$$\mathbf{s} = (\mathbf{s}_0^\intercal, \mathbf{s}_1^\intercal, \ldots, \mathbf{s}_L^\intercal, s_{L+1})^\intercal, \qquad \mathbf{e} = (e_0, \mathbf{e}_1^\intercal, \ldots, \mathbf{e}_L^\intercal, \mathbf{e}_{L+1}^\intercal)^\intercal,$$

where each $\mathbf{s}_\ell, \mathbf{e}_\ell$ is $\mathfrak{Q}$-dimensional, then

$$w_{\mathsf{out}} = -\mathbf{s}_0^\intercal \boldsymbol{\iota}_1 + e_0, \qquad \mathbf{w}_\ell^\intercal = \mathbf{s}_\ell^\intercal \mathbf{\Gamma}_{\mathbf{x}[\ell]} - \mathbf{s}_{\ell-1}^\intercal + \mathbf{e}_\ell^\intercal, \qquad \mathbf{t}^\intercal = s_{L+1}\boldsymbol{\xi}^\intercal - \mathbf{s}_L^\intercal + \mathbf{e}_{L+1}^\intercal.$$

Since evaluation is linear in $\mathbf{w}$'s and $\mathbf{t}$, we first consider the non-noisy part,

$$(\mathbf{t} - \mathbf{e}_{L+1})^\intercal \cdot \prod_{\ell'=L}^{1} \mathbf{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 + \sum_{\ell=L}^{1} \left((\mathbf{w}_\ell - \mathbf{e}_\ell)^\intercal \cdot \prod_{\ell'=\ell-1}^{1} \mathbf{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 \right)$$

[20] Here, the DFA input length upper bound $\overline{L}$ is needed due to perfect correctness requirement in the presence of noise accumulation. For our garbling and ABE constructions, we can simply set $\overline{L} = 2^\lambda$ to support arbitrary polynomial-size computations.

$$
\begin{aligned}
&= (s_{L+1}\boldsymbol{\xi}^\intercal - \mathbf{s}_L^\intercal) \cdot \prod_{\ell'=L}^{1} \boldsymbol{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 + \sum_{\ell=L}^{1} \left((\mathbf{s}_\ell^\intercal \boldsymbol{\Gamma}_{\mathbf{x}[\ell]} - \mathbf{s}_{\ell-1}^\intercal) \cdot \prod_{\ell'=\ell-1}^{1} \boldsymbol{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 \right) \\
&= s_{L+1}\boldsymbol{\xi}^\intercal \cdot \prod_{\ell'=L}^{1} \boldsymbol{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 - \mathbf{s}_L^\intercal \cdot \prod_{\ell'=L}^{1} \boldsymbol{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 \\
&\quad \text{(telescoping)} \quad + \sum_{\ell=L}^{1} \left(\mathbf{s}_\ell^\intercal \cdot \prod_{\ell'=\ell}^{1} \boldsymbol{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 - \mathbf{s}_{\ell-1}^\intercal \cdot \prod_{\ell'=\ell-1}^{1} \boldsymbol{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 \right) \\
&= s_{L+1}\boldsymbol{\xi}^\intercal \cdot \prod_{\ell'=L}^{1} \boldsymbol{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 - \mathbf{s}_0^\intercal \boldsymbol{\iota}_1 = (w_{\mathsf{out}} - e_0) + \begin{cases} 0, & \text{if } \Gamma \text{ accepts } \mathbf{x}; \\ s_{L+1}, & \text{otherwise.} \end{cases}
\end{aligned}
$$

Writing

$$
e_{\mathsf{Eval}} = \mathbf{e}_{L+1}^\intercal \cdot \prod_{\ell'=L}^{1} \boldsymbol{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 + \sum_{\ell=L}^{1} \left(\mathbf{e}_\ell^\intercal \cdot \prod_{\ell'=\ell-1}^{1} \boldsymbol{\Gamma}_{\mathbf{x}[\ell']} \cdot \boldsymbol{\iota}_1 \right) - e_0,
$$

we have

$$
\mathsf{Eval}(\mathsf{pp}, f_{\Gamma,L}, \mathbf{R}, \mathbf{x}, \{\mathbf{w}_\ell^\intercal\}_{\ell\in[L]}, \mathbf{t}^\intercal) = w_{\mathsf{out}} + e_{\mathsf{Eval}} + \begin{cases} 0, & \text{if } \Gamma \text{ accepts } \mathbf{x}; \\ s_{L+1}, & \text{otherwise.} \end{cases}
$$

Transposing e_{Eval} and applying Lemma 22, we find

$$
\begin{aligned}
|e_{\mathsf{Eval}}| &\le \|\boldsymbol{\iota}_1^\intercal\| \cdot \underbrace{\prod_{\ell'=1}^{L} \|\boldsymbol{\Gamma}_{\mathbf{x}[\ell']}^\intercal\|}_{\le 1} \cdot \|\mathbf{e}_{L+1}\| + \sum_{\ell=L}^{1} \left(\|\boldsymbol{\iota}_1^\intercal\| \cdot \underbrace{\prod_{\ell'=1}^{\ell-1} \|\boldsymbol{\Gamma}_{\mathbf{x}[\ell']}^\intercal\|}_{\le 1} \cdot \|\mathbf{e}_\ell\| \right) + |e_0| \\
&\le (L+2)B_{\mathsf{in}} \le (\overline{L}+2)B_{\mathsf{in}} \le B_{\mathsf{out}}.
\end{aligned}
$$

This proves $(B_{\mathsf{in}}, B_{\mathsf{out}})$-correctness.

Perfect security without noise is proven in [LL20a], which implies security with any noise. To recap, observe that with param, pp, $f_{\Gamma,L}$, output of GenF, and $\mathbf{x} \in \{0,1\}^L$ fixed, the values $(\mathbf{w}_1, \ldots, \mathbf{w}_L, s_{L+1})$ are jointly uniformly random — thanks to the subtraction of $\mathbf{s}_\ell$ in each $\mathbf{w}_\ell$. When $\mathbf{x}$ is rejected by Γ,

$$
w_{\mathsf{out}} = \underbrace{\mathsf{Eval}(\mathsf{pp}, f_{\Gamma,L}, \mathbf{R}, \mathbf{x}, \{\mathbf{w}_\ell^\intercal\}_{\ell\in[L]}, \mathbf{t}^\intercal) - e_{\mathsf{Eval}}}_{\text{independent of } s_{L+1}} - s_{L+1},
$$

so $(w_{\mathsf{out}}, \mathbf{w}_1, \ldots, \mathbf{w}_L)$ is again jointly uniformly random. □

7.2 Construction of KP-ABE for DFA

We present our KP-ABE for DFA, utilizing the telescoping-sum structure of the DFA garbling. The idea is similar to [LL20a] — an ABE key contains roughly $\mathfrak{Q}$ IPFE keys and an ABE ciphertext, L IPFE ciphertexts, so that decryption yields $\Theta(L\mathfrak{Q})$ inner products corresponding to a garbling generated using pseudorandomness. The proof, yet, is much simpler than that of [LL20a] (as well as those of all previous pairing-based ABE for DFA), thanks to the easy-to-use security of evasive IPFE (Definition 5).

Ingredients of Construction 7. We rely on

- Construction 6, a specific noisy linear garbling NLG for DFA, and
- an identity-based evasive IPFE scheme IPFE for $\mathcal{I} \supseteq \{0,1,2\}$ that is B_{IPFE}-correct and restricted-σ_{IPFE}-secure (Definitions 4 and 6).

Construction 7 (KP-ABE for DFA)**.** Let

$$\mathsf{Params} = \left\{ \overline{L} \,\middle|\, \overline{L} \in \mathbb{N} \right\}, \qquad X_{\overline{L}} = \left\{ \text{DFA } \Gamma \right\}, \qquad Y_{\overline{L}} = \left\{ \mathbf{x} \in \{0,1\}^L \,\middle|\, L \le \overline{L} \right\},$$
$$P_{\overline{L}}(\Gamma, \mathbf{x}) = \begin{cases} 1, & \text{if } \Gamma \text{ accepts } \mathbf{x}; \\ 0, & \text{if } \Gamma \text{ rejects } \mathbf{x}. \end{cases}$$

Our KP-ABE for DFA works as follows.

- $\mathsf{Setup}(\overline{L})$ picks suitable q, n in a straight-forward way.[21] The algorithm sets $\mathsf{pp} = (q, \overline{L})$ and $K = 0, Z = 3n$. It runs $(\mathsf{impk}, \mathsf{imsk}) \xleftarrow{\$} \mathsf{IPFE.Setup}(q, 1^K, 1^Z)$ and outputs

$$\mathsf{mpk} = (\mathsf{pp}, 1^n, 1^K, 1^Z, \mathsf{impk}), \qquad \mathsf{msk} = (\mathsf{mpk}, \mathsf{imsk}).$$

- $\mathsf{KeyGen}(\mathsf{msk}, \Gamma)$ parses $\Gamma = (1^{\Omega}, \mathbf{\Gamma}_0, \mathbf{\Gamma}_1, \boldsymbol{\xi})$. It samples as sets

$$\mathbf{R} \xleftarrow{\$} \mathbb{Z}_q^{n \times \Omega}, \qquad \mathbf{r} \xleftarrow{\$} \mathbb{Z}_q^n,$$
$$\mathbf{v}_{\mathrm{msg}} = \begin{pmatrix} -\mathbf{R}\boldsymbol{\iota}_1 \\ \mathbf{0}_n \\ 1 \\ \mathbf{0}_{n-1} \end{pmatrix}, \qquad \mathbf{V}_{\mathrm{t}} = \begin{pmatrix} -\mathbf{R} \\ \mathbf{r}\boldsymbol{\xi}^{\intercal} \\ \mathbf{0}_{n \times \Omega} \end{pmatrix}, \qquad \mathbf{V}_{\mathrm{in}} = \begin{pmatrix} \mathbf{R}\mathbf{\Gamma}_0 \\ \mathbf{R}\mathbf{\Gamma}_1 \\ -\mathbf{R} \end{pmatrix},$$

  where $\boldsymbol{\iota}_1 = (1, 0, \ldots, 0)^{\intercal} \in \{0,1\}^{\Omega}$ is the first standard basis vector. The algorithm generates IPFE secret keys

$$\mathsf{isk}_{\mathrm{msg}} \xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{imsk}, 0, \mathbf{v}_{\mathrm{msg}}, \bot),$$
$$\mathsf{isk}_{\mathrm{t}} \xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{imsk}, 1, \mathbf{V}_{\mathrm{t}}, \bot),$$
$$\mathsf{isk}_{\mathrm{in}} \xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{imsk}, 2, \mathbf{V}_{\mathrm{in}}, \bot).$$

  It outputs $\mathsf{sk} = (\mathbf{R}, \mathbf{r}, \mathsf{isk}_{\mathrm{msg}}, \mathsf{isk}_{\mathrm{t}}, \mathsf{isk}_{\mathrm{in}})$.

- $\mathsf{Enc}(\mathsf{mpk}, \mathbf{x}, \mu)$ samples

$$\widetilde{\mathbf{s}}_0, \widetilde{\mathbf{s}}_1, \ldots, \widetilde{\mathbf{s}}_L, \widetilde{\mathbf{s}}_{L+1} \xleftarrow{\$} \mathbb{Z}_q^n, \qquad s_{\mathrm{msg}} \xleftarrow{\$} \mathbb{Z}_q \setminus [-2 \cdot 2^{-\kappa} q, 2 \cdot 2^{-\kappa} q].$$

  It sets

$$\begin{aligned} \mathbf{u}_{\mathrm{msg}}^{\intercal} &= (\quad \widetilde{\mathbf{s}}_0^{\intercal}, \quad \mathbf{0}_{1\times n}, \quad \mu s_{\mathrm{msg}}, \mathbf{0}_{1\times(n-1)} \quad), \\ \mathbf{u}_{\mathrm{t}}^{\intercal} &= (\quad \widetilde{\mathbf{s}}_L^{\intercal}, \quad \widetilde{\mathbf{s}}_{L+1}^{\intercal}, \quad \mathbf{0}_{1\times n} \quad), \\ \mathbf{u}_{\ell}^{\intercal} &= (\quad (1 - \mathbf{x}[\ell])\widetilde{\mathbf{s}}_{\ell}^{\intercal}, \quad \mathbf{x}[\ell]\widetilde{\mathbf{s}}_{\ell}^{\intercal}, \quad \widetilde{\mathbf{s}}_{\ell-1}^{\intercal} \quad) \qquad \text{for all } \ell \in [L], \end{aligned}$$

[21] Precisely speaking, the randomness used to sample q, n must be separate from that to run IPFE.Setup and efficiently sampleable conditioned on and given q, n.

generates IPFE ciphertexts

$$\begin{aligned}\mathsf{ict}_{\mathrm{msg}} &\xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, 0, \mathfrak{o}, \mathbf{u}_{\mathrm{msg}}^{\intercal}),\\ \mathsf{ict}_{\mathrm{t}} &\xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, 1, \mathfrak{o}, \mathbf{u}_{\mathrm{t}}^{\intercal}),\\ \mathsf{ict}_{\ell} &\xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, 2, \mathfrak{o}, \mathbf{u}_{\ell}^{\intercal}) \qquad \text{for all } \ell \in [L],\end{aligned}$$

and outputs $\mathsf{ct} = (\mathsf{ict}_{\mathrm{msg}}, \mathsf{ict}_{\mathrm{t}}, \mathsf{ict}_1, \ldots, \mathsf{ict}_L)$.

- $\mathsf{Dec}(\mathsf{mpk}, \Gamma, \mathsf{sk}, \mathbf{x}, \mathsf{ct})$ outputs $\perp$ and terminates if Γ rejects $\mathbf{x}$. Otherwise, it parses sk, ct as in $\mathsf{KeyGen}, \mathsf{Enc}$. Suppose $\mathbf{x} \in \{0,1\}^L$, the algorithm recomputes $\mathbf{v}_{\mathrm{msg}}, \mathbf{V}_{\mathrm{t}}, \mathbf{V}_{\mathrm{in}}$ as in KeyGen and performs IPFE decryptions

$$\begin{aligned}w_{\mathrm{msg}} &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, 0, \mathbf{v}_{\mathrm{msg}}, \perp, \mathsf{isk}_{\mathrm{msg}}, \mathfrak{o}, \mathsf{ict}_{\mathrm{msg}}),\\ \mathbf{t}^{\intercal} &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, 1, \mathbf{V}_{\mathrm{t}}, \perp, \mathsf{isk}_{\mathrm{t}}, \mathfrak{o}, \mathsf{ict}_{\mathrm{t}}),\\ \mathbf{w}_{\ell}^{\intercal} &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, 2, \mathbf{V}_{\mathrm{in}}, \perp, \mathsf{isk}_{\mathrm{in}}, \mathfrak{o}, \mathsf{ict}_{\ell}) \qquad \text{for all } \ell \in [L].\end{aligned}$$

It evaluates the DFA garbling

$$w_{\mathrm{out}} \leftarrow \mathsf{NLG.Eval}(\mathsf{pp}, (\Gamma, 1^L), \perp, \mathbf{x}, \{\mathbf{w}_{\ell}^{\intercal}\}_{\ell\in[L]}, \mathbf{t}^{\intercal})$$

and outputs

$$\mu' \leftarrow \begin{cases} 0, & \text{if } w_{\mathrm{msg}} - w_{\mathrm{out}} \in [-2^{-\kappa}q, 2^{-\kappa}q];\\ 1, & \text{otherwise.}\end{cases}$$

Theorem 24 (¶). *Construction 7 is correct (Definition 1) if* $(\overline{L}+2)B_{\mathsf{IPFE}}(q,0,3n) \le 2^{-\kappa}q$.

Proof (Theorem 24). The IPFE decryption results in Dec are

$$\begin{aligned}w_{\mathrm{msg}} &= -\widetilde{\mathbf{s}}_0^{\intercal}\mathbf{R}\boldsymbol{\iota}_1 + \mu s_{\mathrm{msg}} + e_{\mathrm{out}} = \widetilde{\mathbf{s}}_0^{\intercal}\mathbf{R}\cdot(-\boldsymbol{\iota}_1) + \mu s_{\mathrm{msg}} + e_{\mathrm{out}},\\ \mathbf{t}^{\intercal} &= \mathbf{u}_{\mathrm{t}}^{\intercal}\mathbf{V}_{\mathrm{t}} + \mathbf{e}_{\mathrm{t}}^{\intercal} = -\widetilde{\mathbf{s}}_L^{\intercal}\mathbf{R} + \widetilde{\mathbf{s}}_{L+1}^{\intercal}\mathbf{r}\cdot\boldsymbol{\xi}^{\intercal} + \mathbf{e}_{\mathrm{t}}^{\intercal},\\ \mathbf{w}_{\ell}^{\intercal} &= \mathbf{u}_{\ell}^{\intercal}\mathbf{V}_{\mathrm{in}} + \mathbf{e}_{\ell}^{\intercal} = (1-\mathbf{x}[\ell])\widetilde{\mathbf{s}}_{\ell}^{\intercal}\mathbf{R}\boldsymbol{\Gamma}_0 + \mathbf{x}[\ell]\widetilde{\mathbf{s}}_{\ell}^{\intercal}\mathbf{R}\boldsymbol{\Gamma}_1 - \widetilde{\mathbf{s}}_{\ell-1}^{\intercal}\mathbf{R} + \mathbf{e}_{\ell}^{\intercal}\\ &\qquad = \widetilde{\mathbf{s}}_{\ell}^{\intercal}\mathbf{R}\cdot\boldsymbol{\Gamma}_{\mathbf{x}[\ell]} - \widetilde{\mathbf{s}}_{\ell-1}^{\intercal}\mathbf{R} + \mathbf{e}_{\ell}^{\intercal} \qquad \text{for all } \ell \in [L],\end{aligned}$$

where the errors $e_{\mathrm{out}}, \mathbf{e}_{\mathrm{t}}, \{\mathbf{e}_{\ell}\}_{\ell\in[L]}$ are B_{IPFE}-bounded by the correctness of IPFE. They form a DFA garbling with randomness

$$\mathbf{s} = (\widetilde{\mathbf{s}}_0^{\intercal}\mathbf{R},\ \widetilde{\mathbf{s}}_1^{\intercal}\mathbf{R},\ \ldots,\ \widetilde{\mathbf{s}}_L^{\intercal}\mathbf{R},\ \widetilde{\mathbf{s}}_{L+1}^{\intercal}\mathbf{r})^{\intercal}.$$

Therefore, by the $(B_{\mathsf{IPFE}}, (\overline{L}+2)B_{\mathsf{IPFE}})$-correctness of NLG (Theorem 23), we have

$$\begin{aligned}w_{\mathrm{out}} - (\widetilde{\mathbf{s}}_0^{\intercal}\mathbf{R}\cdot(-\boldsymbol{\iota}_1) + e_{\mathrm{out}}) &\in [-(\overline{L}+2)B_{\mathsf{IPFE}}(q,0,3n), (\overline{L}+2)B_{\mathsf{IPFE}}(q,0,3n)]\\ &\subseteq [-2^{-\kappa}q, 2^{\kappa}q].\end{aligned}$$

If $\mu = 0$, it always holds that $w_{\mathrm{msg}} - w_{\mathrm{out}} \in [-2^{-\kappa}q, 2^{-\kappa}q]$, which never holds when $\mu = 1$ by our choice of s_{msg}. We conclude that Construction 7 is correct. □

7.3 Security of KP-ABE for DFA

Theorem 25 (¶). *Construction 7 is very selectively secure (Definition 2) if* $\mathsf{LWE}_{n,\mathrm{poly}(\lambda),q,\sigma_\mathrm{r}}$ *(Assumption 1) holds for some* $\sigma_\mathrm{r} \le \frac{2^{-\kappa-6}\sigma_{\mathsf{IPFE}}}{2\sqrt{\kappa}}$.[22]

Proof (Theorem 25). The proof is similar to that of Theorem 19. Let $\mathcal{A}$ be an efficient adversary and assume that it always chooses challenges such that Γ_j rejects $\mathbf{x}$ for all $j \in [J]$. Consider the following evasive IPFE sampler $\mathcal{S}^\beta = (\mathcal{S}_\mathrm{v}, \mathcal{S}^\beta_\mathrm{u})$ per Definition 5.

- $\mathcal{S}_\mathrm{v}(r_\mathrm{pub})$ parses $r_\mathrm{pub} = (r_\mathcal{A}, r_\mathrm{qn}, r_\mathrm{r})$. It runs $\mathcal{A}(r_\mathcal{A})$ to obtain

$$\mathsf{param} = \overline{L}, \qquad \{\Gamma_j\}_{j\in[J]} \quad (\Gamma_j = (1^{\mathfrak{Q}_j}, \mathbf{\Gamma}_{j,0}, \mathbf{\Gamma}_{j,1}, \xi_j)), \qquad \mathbf{x} \in \{0,1\}^{<\overline{L}}.$$

  The algorithm uses r_qn to sample q, n as in Setup of Construction 7. It uses r_r to sample $\mathbf{R}_j \in \mathbb{Z}_q^{n\times\mathfrak{Q}_j}$ and $\mathbf{r}_j \in \mathbb{Z}_q^n$ uniformly at random in a straight-forward way[23] for all $j \in [J]$. It computes

$$\mathbf{v}_{\mathrm{msg},j} = \begin{pmatrix} -\mathbf{R}_j \boldsymbol{\iota}_1 \\ \mathbf{0}_n \\ 1 \\ \mathbf{0}_{n-1} \end{pmatrix}, \qquad \mathbf{V}_{\mathrm{t},j} = \begin{pmatrix} -\mathbf{R}_j \\ \mathbf{r}_j \boldsymbol{\xi}_j^\intercal \\ \mathbf{0}_{n\times\mathfrak{Q}_j} \end{pmatrix}, \qquad \mathbf{V}_{\mathrm{in},j} = \begin{pmatrix} \mathbf{R}_j\mathbf{\Gamma}_{j,0} \\ \mathbf{R}_j\mathbf{\Gamma}_{j,1} \\ -\mathbf{R}_j \end{pmatrix},$$

  and sets

$$K = 0, \quad Z = 3n, \quad \mathsf{ID} = \{0,1,2\}, \quad J_0 = J, \quad J_1 = J_2 = \sum_{j\in[J]} \mathfrak{Q}_j,$$
$$I_{0,\mathfrak{o}} = I_{1,\mathfrak{o}} = 1, \quad I_{2,\mathfrak{o}} = L, \quad I_{0,\mathfrak{g}} = I_{1,\mathfrak{g}} = I_{2,\mathfrak{g}} = 0, \quad \sigma_\mathrm{pre} = \sigma_{\mathsf{IPFE}},$$
$$\mathbf{V}_{0,\mathfrak{o}} = (\mathbf{v}_{\mathrm{msg},1}, \ldots, \mathbf{v}_{\mathrm{msg},J}), \quad \mathbf{V}_{1,\mathfrak{o}} = (\mathbf{v}_{\mathrm{t},1}, \ldots, \mathbf{v}_{\mathrm{t},J}),$$
$$\mathbf{V}_{2,\mathfrak{o}} = (\mathbf{v}_{\mathrm{in},1}, \ldots, \mathbf{v}_{\mathrm{in},J}), \quad \mathbf{V}_{0,\mathfrak{g}} = \mathbf{V}_{1,\mathfrak{g}} = \mathbf{V}_{2,\mathfrak{g}} = \bot.$$

  Lastly, the algorithm outputs the required components.

- $\mathcal{S}^\beta_\mathrm{u}(r_\mathrm{pub}; r_\mathrm{priv})$ first runs the deterministic subprocedure $\mathcal{S}^\beta_{\mathrm{A}0}(r_\mathrm{pub})$, which does the following. It first reruns $\mathcal{S}_\mathrm{v}(r_\mathrm{pub})$ to obtain $q, n, \mathbf{x}, L$. The algorithm finds matrices $\mathbf{A}_\mathrm{msg}, \mathbf{A}_\mathrm{t}, \{\mathbf{A}_\ell\}_{\ell\in[L]}$ such that for all $\widetilde{\mathbf{s}}_0, \ldots, \widetilde{\mathbf{s}}_{L+1} \in \mathbb{Z}_q^n$ and $s_\mathrm{msg} \in \mathbb{Z}_q$,

$$\mathbf{d}_0^\intercal \mathbf{A}_\mathrm{msg} = \mathbf{u}_\mathrm{msg}^\intercal = (\widetilde{\mathbf{s}}_0^\intercal, \mathbf{0}_{1\times n}, \beta s_\mathrm{msg}, \mathbf{0}_{1\times(n-1)}), \qquad \mathbf{d}_0^\intercal \mathbf{A}_\mathrm{t} = \mathbf{u}_\mathrm{t}^\intercal = (\widetilde{\mathbf{s}}_L^\intercal, \widetilde{\mathbf{s}}_{L+1}^\intercal, \mathbf{0}_{1\times n}),$$
$$\mathbf{d}_0^\intercal \mathbf{A}_\ell = \mathbf{u}_\ell^\intercal = ((1-\mathbf{x}[\ell])\widetilde{\mathbf{s}}_\ell^\intercal, \mathbf{x}[\ell]\widetilde{\mathbf{s}}_\ell^\intercal, \widetilde{\mathbf{s}}_{\ell-1}^\intercal) \qquad (\text{for all } \ell \in [L]),$$

  where $\mathbf{d}_0^\intercal = (s_\mathrm{msg}, \widetilde{\mathbf{s}}_0^\intercal, \ldots, \widetilde{\mathbf{s}}_{L+1}^\intercal)$. The algorithm $\mathcal{S}_{\mathrm{A}0}$ outputs

$$1^{n'} = 1^{n(L+2)+1}, \quad \mathbf{A}_{0,\mathfrak{o},0,1} = \mathbf{A}_\mathrm{msg}, \quad \mathbf{A}_{1,\mathfrak{o},0,1} = \mathbf{A}_\mathrm{t}, \quad \mathbf{A}_{2,\mathfrak{o},0,\ell} = \mathbf{A}_\ell \ \ (\text{for all } \ell \in [L]).$$

  Completing $\mathcal{S}_{\mathrm{A}0}$, the algorithm $\mathcal{S}_\mathrm{u}$ samples $\mathbf{d}_0 \xleftarrow{\$} \mathbb{Z}_q^{n'}$ using r_priv and outputs

$$\mathbf{U}_{\mathsf{id},\mathfrak{o}}^\intercal = \begin{pmatrix} \mathbf{d}_0^\intercal \mathbf{A}_{\mathsf{id},\mathfrak{o},0,1} \\ \vdots \\ \mathbf{d}_0^\intercal \mathbf{A}_{\mathsf{id},\mathfrak{o},0,I_{\mathsf{id},\mathfrak{o}}} \end{pmatrix} \qquad \text{for all } \mathsf{id} \in \mathsf{ID}.$$

[22] This condition is the same as in Theorem 19, instantiated with $B_\mathrm{short} = 2$ and $B_\mathsf{NLG} = 0$.
[23] Precisely speaking, r_r must be efficiently sampleable conditioned on and given $\mathbf{R}_1, \mathbf{r}_1, \ldots, \mathbf{R}_J, \mathbf{r}_J$.

By definition, $\mathcal{S}^\beta$ is a restricted sampler (Definition 6). We have the following:

Claim 26 (¶). *$\mathcal{S}^\beta$ has pseudorandom noisy inner products, i.e., $\mathsf{IPFEsec}^{\mathcal{S}^\beta}_{\mathrm{pre}}$ (Definition 5) holds for both $\beta \in \{0,1\}$.*

The very selective security of Construction 7 follows from Claim 26. Consider the following hybrids.

- H^β_0 is $\mathsf{Exp}^\beta_{\mathrm{ABE}}$ for Construction 7. Note that s_{msg} in the challenge ciphertext is uniformly random over $\mathbb{Z}_q \setminus [-2\cdot 2^{-\kappa}q, 2\cdot 2^{-\kappa}q]$.
- In H^β_1, the scalar s_{msg} is changed to be uniformly random over $\mathbb{Z}_q$. Clearly, $\mathsf{H}^\beta_0 \approx_{\mathrm{s}} \mathsf{H}^\beta_1$. Moreover, H^β_1 corresponds to the left distribution of $\mathsf{IPFEsec}^{\mathcal{S}^\beta}_{\mathrm{post}}$.
- In H^β_2, the vectors inside IPFE ciphertexts (components of the ABE challenge ciphertext) are replaced by random, which corresponds to the right distribution of $\mathsf{IPFEsec}^{\mathcal{S}^\beta}_{\mathrm{post}}$. By Claim 26 and the restricted-σ_{IPFE}-security of IPFE, we have $\mathsf{H}^\beta_1 \approx \mathsf{H}^\beta_2$.

Lastly, $\mathsf{H}^0_2 \equiv \mathsf{H}^1_2$. By hybrid argument, we conclude $\mathsf{Exp}^0_{\mathrm{ABE}} \approx \mathsf{Exp}^1_{\mathrm{ABE}}$, completing the proof. □

Proof (Claim 26). Fix $\beta \in \{0,1\}$ and consider the following hybrids.

- H_0 is the left distribution of $\mathsf{IPFEsec}^{\mathcal{S}^\beta}_{\mathrm{pre}}$. Recall that the public randomness is $r_{\mathrm{pub}} = (r_{\mathcal{A}}, r_{\mathrm{qn}}, r_{\mathrm{r}})$. Adopting the notations of the ABE decryption algorithm, the noisy inner products are

$$\begin{aligned} w_{\mathrm{msg},j} &= \mathbf{u}^\top_{\mathrm{msg}}\mathbf{v}_{\mathrm{msg},j} + e_{\mathrm{out},j} = \widetilde{\mathbf{s}}_0^\top \mathbf{R}_j \cdot (-\boldsymbol{\iota}_1) + \beta s_{\mathrm{msg}} + e_{\mathrm{out},j}, \\ \mathbf{t}^\top_j &= \mathbf{u}^\top_{\mathrm{t}}\mathbf{V}_{\mathrm{t},j} + \mathbf{e}^\top_{\mathrm{t},j} = -\widetilde{\mathbf{s}}_L^\top \mathbf{R}_j + \widetilde{\mathbf{s}}_{L+1}^\top \mathbf{r}_j \cdot \boldsymbol{\xi}^\top_j + \mathbf{e}^\top_{\mathrm{t},j}, \\ \mathbf{w}^\top_{j,\ell} &= \mathbf{u}^\top_\ell \mathbf{V}_\ell + \mathbf{e}^\top_{j,\ell} = \widetilde{\mathbf{s}}_\ell^\top \mathbf{R}_j \cdot \boldsymbol{\Gamma}_{\mathbf{x}[\ell]} - \widetilde{\mathbf{s}}_{\ell-1}^\top \mathbf{R}_j + \mathbf{e}^\top_{j,\ell} \qquad (\text{for all } \ell \in [L]), \end{aligned}$$

  where $j \in [J]$ and the entries of $e, \mathbf{e}$'s are independent $\mathcal{D}_{\mathbb{Z},\sigma_{\mathsf{IPFE}}}$.
- In H_1, additional small noises are attached to $\widetilde{\mathbf{s}}^\top\mathbf{R}, \widetilde{\mathbf{s}}^\top\mathbf{r}$'s. For all $j \in [J]$, set the inner products to

$$\begin{aligned} w_{\mathrm{msg},j} &= (\widetilde{\mathbf{s}}_0^\top \mathbf{R}_j + \mathbf{e}^\top_{\mathrm{r},j,0})(-\boldsymbol{\iota}_1) + \beta s_{\mathrm{msg}} + e_{\mathrm{out},j}, \\ \mathbf{t}^\top_j &= -(\widetilde{\mathbf{s}}_L^\top \mathbf{R}_j + \mathbf{e}^\top_{\mathrm{r},j,L}) + (\widetilde{\mathbf{s}}_{L+1}^\top \mathbf{r}_j + e_{\mathrm{r},j,L+1})\boldsymbol{\xi}^\top_j + \mathbf{e}^\top_{\mathrm{t},j}, \\ \mathbf{w}^\top_{j,\ell} &= (\widetilde{\mathbf{s}}_\ell^\top \mathbf{R}_j + \mathbf{e}^\top_{\mathrm{r},j,\ell})\boldsymbol{\Gamma}_{\mathbf{x}[\ell]} - (\widetilde{\mathbf{s}}_{\ell-1}^\top \mathbf{R}_j + \mathbf{e}^\top_{\mathrm{r},j,\ell-1}) + \mathbf{e}^\top_{j,\ell} \qquad (\text{for all } \ell \in [L]), \end{aligned}$$

  where $\mathbf{e}_{\mathrm{r},j,\ell} \xleftarrow{\$} \mathcal{D}^{\Omega_j}_{\mathbb{Z},\sigma_{\mathrm{r}},\le\sigma_{\mathrm{r}}\sqrt{\kappa}}$ for all $\ell \in \{0,\dots,L\}$ and $e_{\mathrm{r},j,L+1} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{r}},\le\sigma_{\mathrm{r}}\sqrt{\kappa}}$. The errors introduced into the inner products are bounded by

$$2\sigma_{\mathrm{r}}\sqrt{\kappa} \le 2^{-\kappa-6}\sigma_{\mathsf{IPFE}},$$

  so they are flooded (Lemma 2) by $\{e_{\mathrm{out},j}, \mathbf{e}_{\mathrm{t},j}, \{\mathbf{e}_{j,\ell}\}_{\ell\in[L]}\}_{j\in[J]}$, and $\mathsf{H}_0 \approx_{\mathrm{s}} \mathsf{H}_1$.
- In H_2, the $\mathbf{e}_{\mathrm{r}}$'s are no longer truncated, i.e., for all $j \in [J]$, sample $\mathbf{e}_{\mathrm{r},j,\ell} \xleftarrow{\$} \mathcal{D}^{\Omega_j}_{\mathbb{Z},\sigma_{\mathrm{r}}}$ for all $\ell \in \{0,\dots,L\}$ and $e_{\mathrm{r},j,L+1} \xleftarrow{\$} \mathcal{D}_{\mathbb{Z},\sigma_{\mathrm{r}}}$. We have $\mathsf{H}_1 \approx_{\mathrm{s}} \mathsf{H}_2$ by Lemma 1.

- In H_3, the garblings use uniform randomness. For all $j \in [J]$, sample $\mathbf{s}_{j,\ell} \xleftarrow{\$} \mathbb{Z}_q^n$ for all $\ell \in \{0,\dots,L\}$ and $s_{j,L+1} \xleftarrow{\$} \mathbb{Z}_q$, and set the inner products to

$$\begin{aligned} w_{\mathrm{msg},j} &= \mathbf{s}_{j,0}^\top \cdot (-\boldsymbol{\iota}_1) + \beta s_{\mathrm{msg}} + e_{\mathrm{out},j}, \\ \mathbf{t}_j^\top &= -\mathbf{s}_{j,L}^\top + s_{j,L+1}\boldsymbol{\xi}_j^\top + \mathbf{e}_{\mathrm{t},j}^\top, \\ \mathbf{w}_{j,\ell}^\top &= \mathbf{s}_{j,\ell}^\top \boldsymbol{\Gamma}_{\mathbf{x}[\ell]} - \mathbf{s}_{j,\ell-1}^\top + \mathbf{e}_{j,\ell}^\top \qquad (\text{for all } \ell \in [L]), \end{aligned}$$

  By $(L+1)$ hybrids of applying $\mathsf{LWE}_{n,J_2,q,\sigma_\mathrm{r}}$ with public matrix $(\mathbf{R}_1,\dots,\mathbf{R}_J)$ and secrets $\widetilde{\mathbf{s}}_0,\dots,\widetilde{\mathbf{s}}_L$, then $\mathsf{LWE}_{n,J,q,\sigma_\mathrm{r}}$ with public matrix $(\mathbf{r}_1,\dots,\mathbf{r}_J)$ and secret $\widetilde{\mathbf{s}}_{L+1}$, we have

$$\left\{ \{\widetilde{\mathbf{s}}_\ell^\top \mathbf{R}_j + \mathbf{e}_{\mathrm{r},j,\ell}^\top\}_{\ell\in\{0,\dots,L\}}, \widetilde{\mathbf{s}}_{L+1}^\top \mathbf{r}_j + e_{\mathrm{r},j,L+1} \right\}_{j\in[J]} \approx \left\{ \{\mathbf{s}_{j,\ell}^\top\}_{\ell\in\{0,\dots,L\}}, s_{j,L+1} \right\}_{j\in[J]}$$

  which implies $\mathsf{H}_2 \approx \mathsf{H}_3$. In this step, we rely on the "straight-forwardness" of sampling $\mathbf{R},\mathbf{r}$'s from r_r in $\mathcal{S}^\beta$.

- In H_4, all the inner products are replaced by random, i.e., the right distribution of $\mathsf{IPFEsec}_{\mathrm{pre}}^{\mathcal{S}^\beta}$. Note that the inner products in H_3 are well-randomized garblings (by $\mathbf{s}$'s) plus independent noises ($e,\mathbf{e}$'s). Therefore, by J hybrids of applying the security of NLG, we conclude $\mathsf{H}_3 \equiv \mathsf{H}_4$.

By hybrid argument, $\mathsf{H}_0 \approx \mathsf{H}_4$, i.e., $\mathsf{IPFEsec}_{\mathrm{pre}}^{\mathcal{S}^\beta}$ holds. □

7.4 CP-ABE for DFA and Summary

Our CP-ABE for DFA is similar to KP-ABE for DFA and general CP-ABE and features a mixture of how components are set in both schemes.

Ingredients of Construction 8. We rely on

- Construction 6, a specific noisy linear garbling NLG for DFA, and
- an identity-based evasive IPFE scheme IPFE for $\mathcal{I} \supseteq \{0,1,2\}$ that is B_{IPFE}-correct and restricted-σ_{IPFE}-secure (Definitions 4 and 6).

Construction 8 (CP-ABE for DFA)**.** Let

$$\mathsf{Params} = \left\{ \overline{L} \,\middle|\, \overline{L} \in \mathbb{N} \right\}, \qquad X_{\overline{L}} = \left\{ \mathbf{x} \in \{0,1\}^L \,\middle|\, L \le \overline{L} \right\}, \qquad Y_{\overline{L}} = \left\{ \text{DFA } \Gamma \right\},$$
$$P_{\overline{L}}(\mathbf{x},\Gamma) = \begin{cases} 1, & \text{if } \Gamma \text{ accepts } \mathbf{x}; \\ 0, & \text{if } \Gamma \text{ rejects } \mathbf{x}. \end{cases}$$

Our CP-ABE for DFA works as follows.

- $\mathsf{Setup}(\overline{L})$ picks suitable q,n in a straight-forward way. It sets $\mathsf{pp} = (q,\overline{L}), K = 0$, $Z = 3n$. The algorithm runs $(\mathsf{impk},\mathsf{imsk}) \xleftarrow{\$} \mathsf{IPFE.Setup}(q,1^K,1^Z)$ and outputs

$$\mathsf{mpk} = (\mathsf{pp}, 1^n, 1^K, 1^Z, \mathsf{impk}), \qquad \mathsf{msk} = (\mathsf{mpk},\mathsf{imsk}).$$

- $\mathsf{KeyGen}(\mathsf{msk},\mathbf{x})$, given $\mathbf{x} \in \{0,1\}^L$, samples and sets

$$\mathbf{r}_0,\dots,\mathbf{r}_L \xleftarrow{\$} \mathbb{Z}_q^n, \qquad r_{L+1} \xleftarrow{\$} \mathbb{Z}_q,$$

$$\mathbf{v}_{\mathsf{msg}} = \begin{pmatrix} \mathbf{r}_0 \\ 1 \\ \mathbf{0}_{2n-1} \end{pmatrix}, \qquad \mathbf{v}_{\mathsf{t}} = \begin{pmatrix} \mathbf{r}_L \\ r_{L+1} \\ \mathbf{0}_{2n-1} \end{pmatrix}, \qquad \mathbf{v}_\ell = \begin{pmatrix} (1-\mathbf{x}[\ell])\mathbf{r}_\ell \\ \mathbf{x}[\ell]\mathbf{r}_\ell \\ \mathbf{r}_{\ell-1} \end{pmatrix} \quad \text{for all } \ell \in [L].$$

It generates IPFE secret keys

$$\begin{aligned} \mathsf{isk}_{\mathsf{msg}} &\xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{imsk}, 0, \mathbf{v}_{\mathsf{msg}}, \bot), \\ \mathsf{isk}_{\mathsf{t}} &\xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{imsk}, 1, \mathbf{v}_{\mathsf{t}}, \bot), \\ \mathsf{isk}_\ell &\xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{imsk}, 2, \mathbf{v}_\ell, \bot) \qquad \text{for all } \ell \in [L]. \end{aligned}$$

The algorithm outputs $\mathsf{sk} = (\mathbf{r}_0, \ldots, \mathbf{r}_L, r_{L+1}, \mathsf{isk}_{\mathsf{msg}}, \mathsf{isk}_{\mathsf{t}}, \mathsf{isk}_1, \ldots, \mathsf{isk}_L)$.

- $\mathsf{Enc}(\mathsf{mpk}, \Gamma, \mu)$ parses $\Gamma = (1^{\mathfrak{Q}}, \boldsymbol{\Gamma}_0, \boldsymbol{\Gamma}_1, \boldsymbol{\xi})$. It samples

$$\mathbf{S} \xleftarrow{\$} \mathbb{Z}_q^{\mathfrak{Q} \times n}, \qquad s_{-1} \xleftarrow{\$} \mathbb{Z}_q, \qquad s_{\mathsf{msg}} \xleftarrow{\$} \mathbb{Z}_q \setminus [-2 \cdot 2^{-\kappa} q, 2 \cdot 2^{-\kappa} q],$$

and sets

$$\mathbf{u}_{\mathsf{msg}}^\intercal = (-\boldsymbol{\iota}_1^\intercal \mathbf{S}, \mu s_{\mathsf{msg}}, \mathbf{0}_{1\times(2n-1)}), \quad \mathbf{U}_{\mathsf{t}}^\intercal = (-\mathbf{S}, s_{-1}\boldsymbol{\xi}, \mathbf{0}_{\mathfrak{Q}\times(2n-1)}), \quad \mathbf{U}_{\mathsf{in}}^\intercal = (\boldsymbol{\Gamma}_0^\intercal \mathbf{S}, \boldsymbol{\Gamma}_1^\intercal \mathbf{S}, -\mathbf{S}).$$

The algorithm generates IPFE ciphertexts

$$\begin{aligned} \mathsf{ict}_{\mathsf{msg}} &\xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, 0, \mathfrak{o}, \mathbf{u}_{\mathsf{msg}}^\intercal), \\ \mathsf{ict}_{\mathsf{t}} &\xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, 1, \mathfrak{o}, \mathbf{U}_{\mathsf{t}}^\intercal), \\ \mathsf{ict}_{\mathsf{in}} &\xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, 2, \mathfrak{o}, \mathbf{U}_{\mathsf{in}}^\intercal), \end{aligned}$$

and outputs $\mathsf{ct} = (\mathsf{ict}_{\mathsf{msg}}, \mathsf{ict}_{\mathsf{t}}, \mathsf{ict}_{\mathsf{in}})$.

- $\mathsf{Dec}(\mathsf{mpk}, \mathbf{x}, \mathsf{sk}, \Gamma, \mathsf{ct})$ outputs $\bot$ and terminates if Γ rejects $\mathbf{x}$. Otherwise, it parses sk, ct as in $\mathsf{KeyGen}, \mathsf{Enc}$. Suppose $\mathbf{x} \in \{0,1\}^L$, the algorithm recomputes $\mathbf{v}_{\mathsf{msg}}, \mathbf{v}_{\mathsf{t}}, \mathbf{v}_1, \ldots, \mathbf{v}_L$ as in KeyGen and performs IPFE decryptions

$$\begin{aligned} w_{\mathsf{msg}} &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, 0, \mathbf{v}_{\mathsf{msg}}, \bot, \mathsf{isk}_{\mathsf{msg}}, \mathfrak{o}, \mathsf{ict}_{\mathsf{msg}}), \\ \mathbf{t} &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, 1, \mathbf{v}_{\mathsf{t}}, \bot, \mathsf{isk}_{\mathsf{t}}, \mathfrak{o}, \mathsf{ict}_{\mathsf{t}}), \\ \mathbf{w}_\ell &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, 2, \mathbf{v}_\ell, \bot, \mathsf{isk}_\ell, \mathfrak{o}, \mathsf{ict}_{\mathsf{in}}) \qquad \text{for all } \ell \in [L]. \end{aligned}$$

It evaluates the DFA garbling

$$w_{\mathsf{out}} \leftarrow \mathsf{NLG.Eval}(\mathsf{pp}, (\Gamma, 1^L), \bot, \mathbf{x}, \{\mathbf{w}_\ell^\intercal\}_{\ell\in[L]}, \mathbf{t}^\intercal)$$

and outputs

$$\mu' \leftarrow \begin{cases} 0, & \text{if } w_{\mathsf{msg}} - w_{\mathsf{out}} \in [-2^{-\kappa} q, 2^{-\kappa} q]; \\ 1, & \text{otherwise.} \end{cases}$$

Theorem 27. *Construction 8 is correct (Definition 1) if* $(\overline{L}+2) B_{\mathsf{IPFE}}(q, 0, 3n) \le 2^{-\kappa} q$.

Theorem 28. *Construction 8 is very selectively secure (Definition 2) if* $\mathsf{LWE}_{n,\mathrm{poly}(\lambda),q,\sigma_{\mathrm{r}}}$ *(Assumption 1) holds for some* $\sigma_{\mathrm{r}} \le \frac{2^{-\kappa-6}\sigma_{\mathsf{IPFE}}}{2\sqrt{\kappa}}$.

Summary. By instantiating our ABE schemes with the IPFE scheme of Construction 2, we obtain KP- and CP-ABE for DFA.

Corollary 29. *Under LWE (Assumption 1) and evasive LWE (Assumption 4), there exists KP-ABE for DFA with*

$$|\mathsf{mpk}| = \mathrm{poly}(\lambda), \qquad |\mathsf{sk}_\Gamma| = (|\Gamma|+1)\,\mathrm{poly}(\lambda), \qquad |\mathsf{ct}_{\mathbf{x}}| = (|\mathbf{x}|+1)\,\mathrm{poly}(\lambda).$$

and CP-ABE for DFA with

$$|\mathsf{mpk}| = \mathrm{poly}(\lambda), \qquad |\mathsf{sk}_{\mathbf{x}}| = (|\mathbf{x}|+1)\,\mathrm{poly}(\lambda), \qquad |\mathsf{ct}_\Gamma| = (|\Gamma|+1)\,\mathrm{poly}(\lambda).$$

We remark that, starting from the linear secret sharing scheme [LL20a] for L, we also obtain ABE for L via the same construction. However, ABE for NFA or NL cannot be obtained so. The schemes for DFA and L crucially rely on the fact that, when noises are added to the LSSS shares, the output noise grows *linearly* with the computation size. On the other hand, the secret sharing schemes for NFA and NL, once cast as noisy linear garbling, incur exponential noise growth during reconstruction. The latter fact makes the resultant ABE bounded.

8 CP-ABE with Succinct Ciphertexts

In this section, we show that with the help of evasive IPFE with structured noises, we can construct CP-ABE from general noisy linear garbling scheme (not necessarily short). When instantiated with the noisy linear garbling implicit in [BGG+14], this yields a CP-ABE scheme with ciphertext size dependent only on circuit depth instead of circuit size.

8.1 Succinct Noisy Linear Garbling for Circuits

The work of [BGG+14] presented a KP-ABE scheme with succinct keys for circuits based on lattices. At the core of their construction are a pair of deterministic homomorphic evaluation algorithms $\mathsf{EvalC}, \mathsf{EvalCX}$, which, for circuit $C : \{0,1\}^L \to \{0,1\}$, input $x \in \{0,1\}^L$, and public matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m(L+1)}$, satisfies the relation

$$\mathsf{EvalC}(\mathbf{A}, C) = \mathbf{A}_C \in \mathbb{Z}_q^{n \times m},$$
$$\mathsf{EvalCX}(\mathbf{s}^\intercal(\mathbf{A} - (1, \mathbf{x}^\intercal) \otimes \mathbf{G}) + \mathbf{e}^\intercal, f, \mathbf{x}) = \mathbf{s}^\intercal(\mathbf{A}_C - C(\mathbf{x}) \cdot \mathbf{G}) + \mathbf{e}_C^\intercal,$$

where $\|\mathbf{e}_C\|$, relative to $\|\mathbf{e}\|$, grows exponentially with the depth of C. (In [BGG+14], it also works for bounded-arithmetic circuits, but we only present the Boolean version.) It naturally yields a noisy linear garbling scheme.

Construction 9 (garbling adapted from [BGG+14])**.** Let

$$\mathsf{Params} = \left\{ 1^d \,\middle|\, d \in \mathbb{N} \right\},$$
$$F_{1^d} = \left\{ f_C \,\middle|\, C \text{ is a Boolean circuit of depth no more than } d \right\},$$
$$f_C(\mathbf{x}) = \begin{cases} \bot, & \text{if } \mathbf{x} \notin \{0,1\}^L; \\ \neg C(\mathbf{x}), & \text{if } \mathbf{x} \in \{0,1\}^L; \end{cases} \qquad \text{for } C : \{0,1\}^L \to \{0,1\} \text{ and } \mathbf{x} \in \mathbb{Z}^L.$$

The noisy linear garbling scheme works as follows.

- $\mathsf{Setup}(1^d)$ picks suitable q, n, m (with $m \geq n\lceil \log_2 q \rceil$) and outputs $\mathsf{pp} = (q, 1^n, 1^m)$.
- $\mathsf{GenF}(\mathsf{pp}, C)$ samples $\mathbf{A}_0, \mathbf{A}_1, \ldots, \mathbf{A}_L \xleftarrow{\$} \mathbb{Z}_q^{n \times m(L+1)}$ and $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n$. It computes $\mathbf{A}_C \leftarrow \mathsf{EvalC}((\mathbf{A}_0, \mathbf{A}_1, \ldots, \mathbf{A}_L), C)$. The algorithm outputs

$$1^{n_C} = 1^n, \quad \mathbf{w}_{\mathrm{out}} = \mathbf{A}_C \mathbf{G}^{-1}(\mathbf{a}), \quad \mathbf{W}_{\ell,0} = \mathbf{A}_\ell, \qquad \mathbf{W}_{\ell,\times} = -\mathbf{G}, \quad (\text{for all } \ell \in [L])$$
$$\mathbf{T} = \mathbf{A}_0 - \mathbf{G}, \qquad \mathbf{R} = \mathbf{a}.$$

  All the samplings are done in a straight-forward way.
- $\mathsf{Eval}(\mathsf{pp}, C, \mathbf{R}, \mathbf{x}, \{\mathbf{w}_\ell^\intercal\}_{\ell \in [L]}, \mathbf{t}^\intercal)$ computes and outputs

$$\mathsf{EvalCX}((\mathbf{t}^\intercal, \mathbf{w}_1^\intercal, \ldots, \mathbf{w}_L^\intercal), C, \mathbf{x}) \cdot \mathbf{G}^{-1}(\mathbf{a}).$$

Lemma 30 ([BGG+14])**.** *Construction 9 is correct (Definition 7) if*

$$(m+2)^{d+1} B_{\mathrm{in}}(d) \leq B_{\mathrm{out}}(d).$$

It has fixed randomness dimension (Definition 9). Suppose

$$\mathsf{GenGaussian}'(1^d, (q, n, m), C, \mathbf{w}_{\mathrm{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell \in [L]}, \mathbf{T}, \mathbf{R}, \mathbf{x})$$

samples (truncated) Gaussian noises with appropriate width σ of suitable shape

$$(e_{\mathrm{out}}, \mathbf{e}_1, \ldots, \mathbf{e}_L, \mathbf{e}_{\mathrm{t}}) \xleftarrow{\$} \mathcal{D}_{\mathbb{Z}, \sigma, \leq \sigma\sqrt{\kappa}}^{1+(L+1)m},$$

then Construction 9 is $\mathsf{GenGaussian}'$-secure (Definition 10) if $\mathsf{LWE}_{n, \mathrm{poly}(\lambda), q, \sigma'}$ (Assumption 2) holds for some $\sigma' \leq \frac{\sigma}{2^{\kappa+6}(m+2)^{d+1}\sqrt{\kappa}}$.

8.2 CP-ABE from General Noisy Linear Garbling

We present a construction of CP-ABE from noisy linear garbling with fixed randomness dimension that is not necessarily short. It has a lot in common with Construction 5. Both compute rerandomized garbling using IPFE. In Construction 5, security relies on flooding to introduce noises to argue that the garblings use good pseudorandomness. Here, we use structured noises, together with some algebraic tweaks ("noisy secret sharing", see Section 1.2), to attach the same noise to the garbling pseudorandomness.

Ingredients of Construction 10. We rely on

- a noisy linear garbling scheme NLG supporting $F = \{F_{\mathsf{param}'}\}_{\mathsf{param}' \in \mathsf{Params}'}$ that is $(B_{\mathrm{in}}, B_{\mathrm{out}})$-correct, has fixed randomness dimension, and is $\mathsf{GenNoise}$-secure such that the output of $\mathsf{GenNoise}$ is in $[-B_{\mathsf{NLG}}, B_{\mathsf{NLG}}]^\star$ (Definitions 7, 9, and 10), and
- an identity-based evasive IPFE with structured noises scheme IPFE for identity space $\mathcal{I} \supseteq \{-1, 0, 1, \ldots, L\}$ that is B_{IPFE}-correct and restricted-σ_{IPFE}-secure (Definitions 4 and 6).

Construction 10 (CP-ABE from general garbling)**.** Define

$$\mathsf{Params} = \left\{ \mathsf{param} = (\mathsf{param}', 1^L) \,\middle|\, \mathsf{param}' \in \mathsf{Params}', L \in \mathbb{N} \right\}, \quad X_{\mathsf{param}',1^L} = \{0,1\}^L,$$
$$Y_{\mathsf{param}',1^L} = \left\{ f : \mathbb{Z}^L \to \{0,1,\bot\} \,\middle|\, f \in F_{\mathsf{param}'} \right\}, \qquad P_{\mathsf{param}',1^L}(\mathbf{x}, f) = f(\mathbf{x}).$$

(Note that $\mathbf{x}$ is confined to Boolean.) Our CP-ABE scheme works as follows.

- $\mathsf{Setup}(\mathsf{param}', 1^L)$ runs

$$\mathsf{pp} = (q, 1^n, \dots) \xleftarrow{\$} \mathsf{NLG.Setup}(\mathsf{param}').$$

  The algorithm sets $K = \lceil \log_2 q \rceil$ to be the dimension of $\widetilde{\mathbf{g}}$ (for structured noises), and $Z = 2(n + n^2K)$. It runs $(\mathsf{impk}, \mathsf{imsk}) \xleftarrow{\$} \mathsf{IPFE.Setup}(q, 1^K, 1^Z)$ and outputs

$$\mathsf{mpk} = (\mathsf{pp}, 1^K, 1^Z, \mathsf{impk}), \qquad \mathsf{msk} = (\mathsf{mpk}, \mathsf{imsk}).$$

- $\mathsf{KeyGen}(\mathsf{msk}, \mathbf{x})$ samples $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_q^n$ and $\mathbf{Q} \xleftarrow{\$} \mathbb{Z}_q^{n \times nK}$. Recall that $\mathsf{col}(\mathbf{Q})$ is a column of dimension n^2K concatenating the columns of $\mathbf{Q}$. The algorithm sets

$$\mathbf{V}_{\mathrm{Q}} = \begin{pmatrix} \mathbf{Q} \\ \mathbf{0}_{(Z-n)\times nK} \end{pmatrix}, \quad \mathbf{v}_0 = \begin{pmatrix} \mathbf{r} \\ \mathsf{col}(\mathbf{Q}) \\ 1 \\ \mathbf{0}_{n-1+n^2K} \end{pmatrix}, \quad \mathbf{v}_\ell = \begin{pmatrix} (1-\mathbf{x}[\ell])\mathbf{r} \\ (1-\mathbf{x}[\ell])\mathsf{col}(\mathbf{Q}) \\ \mathbf{x}[\ell]\mathbf{r} \\ \mathbf{x}[\ell]\mathsf{col}(\mathbf{Q}) \end{pmatrix} \text{ for all } \ell \in [L].$$

  It generates IPFE secret keys

$$\begin{aligned} \mathsf{isk}_{-1} &\xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{imsk}, -1, \mathbf{0}, \mathbf{V}_{\mathrm{Q}}), \\ \mathsf{isk}_0 &\xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{imsk}, 0, \mathbf{v}_0, \mathbf{0}), \\ \mathsf{isk}_\ell &\xleftarrow{\$} \mathsf{IPFE.KeyGen}(\mathsf{imsk}, \ell, \mathbf{v}_\ell, \mathbf{0}) \qquad \text{for all } \ell \in [L], \end{aligned}$$

  and outputs $\mathsf{sk} = (\mathbf{r}, \mathbf{Q}, \mathsf{isk}_{-1}, \mathsf{isk}_0, \mathsf{isk}_1, \dots, \mathsf{isk}_L)$.

- $\mathsf{Enc}(\mathsf{mpk}, f, \mu)$ runs

$$(1^n, \mathbf{w}_{\mathrm{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}, \mathbf{T}, \mathbf{R}) \xleftarrow{\$} \mathsf{GenF}(\mathsf{pp}, f).$$

  It samples $\mathbf{S} \xleftarrow{\$} \mathbb{Z}_q^{n\times n}$, $\mathbf{s}_{\mathrm{Q}} \xleftarrow{\$} \mathbb{Z}_q^n$, $s_{\mathrm{msg}} \xleftarrow{\$} \mathbb{Z}_q \setminus [-2\cdot 2^{-\kappa}q, 2\cdot 2^{-\kappa}q]$, and sets

$$\begin{aligned} \mathbf{u}_{\mathrm{Q}}^\intercal &= \left(\mathbf{s}_{\mathrm{Q}}^\intercal, \mathbf{0}_{1\times(Z-n)}\right), \\ \mathbf{u}_{\mathrm{msg}}^\intercal &= \left(\mathbf{w}_{\mathrm{out}}^\intercal \mathbf{S}, -(\widetilde{\mathbf{G}}^{-1}(\mathbf{w}_{\mathrm{out}}) \otimes \mathbf{s}_{\mathrm{Q}})^\intercal, \mu s_{\mathrm{msg}}, \mathbf{0}_{1\times(n-1+n^2K)}\right), \\ \mathbf{U}_{\mathrm{t}}^\intercal &= \left(\mathbf{T}^\intercal \mathbf{S}, -(\widetilde{\mathbf{G}}^{-1}(\mathbf{T}) \otimes \mathbf{s}_{\mathrm{Q}})^\intercal, \mathbf{0}_{\star\times(n+n^2K)}\right), \\ \mathbf{U}_\ell^\intercal &= \left(\mathbf{W}_{\ell,0}^\intercal \mathbf{S}, -(\widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0}) \otimes \mathbf{s}_{\mathrm{Q}})^\intercal, (\mathbf{W}_{\ell,0}^\intercal + \mathbf{W}_{\ell,\times}^\intercal)\mathbf{S}, -(\widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0} + \mathbf{W}_{\ell,\times}) \otimes \mathbf{s}_{\mathrm{Q}})^\intercal\right), \end{aligned}$$

  where $\ell \in [L]$. The algorithm generates IPFE ciphertexts

$$\begin{aligned} \mathsf{ict}_{\mathrm{Q}} &\xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, -1, \mathfrak{g}, \mathbf{u}_{\mathrm{Q}}^\intercal), \\ \mathsf{ict}_{\mathrm{msg}} &\xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, 0, \mathfrak{o}, \mathbf{u}_{\mathrm{msg}}^\intercal), \qquad \mathsf{ict}_{\mathrm{t}} \xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, 0, \mathfrak{o}, \mathbf{U}_{\mathrm{t}}^\intercal), \\ \mathsf{ict}_\ell &\xleftarrow{\$} \mathsf{IPFE.Enc}(\mathsf{impk}, \ell, \mathfrak{o}, \mathbf{U}_\ell^\intercal) \qquad \text{for all } \ell \in [L], \end{aligned}$$

  and outputs $\mathsf{ct} = (\mathbf{R}, \mathsf{ict}_{\mathrm{Q}}, \mathsf{ict}_{\mathrm{msg}}, \mathsf{ict}_{\mathrm{t}}, \mathsf{ict}_1, \dots, \mathsf{ict}_L)$.

- $\mathsf{Dec}(\mathsf{mpk}, \mathbf{x}, \mathsf{sk}, f, \mathsf{ct})$ outputs $\perp$ and terminates if $f(\mathbf{x}) \neq 1$. Otherwise, it parses sk, ct as in $\mathsf{KeyGen}, \mathsf{Enc}$, recomputes $\mathbf{V}_Q, \mathbf{v}_0, \{\mathbf{v}_\ell\}_{\ell\in[L]}$, and performs IPFE decryptions for the shares

$$\begin{aligned}
\mathsf{sh}_Q^\intercal &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, -1, \mathbf{0}, \mathbf{V}_Q, \mathsf{isk}_{-1}, \mathfrak{g}, \mathsf{ict}_Q),\\
\mathsf{sh}_{\mathrm{msg}} &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, 0, \mathbf{v}_0, \mathbf{0}, \mathsf{isk}_0, \mathfrak{o}, \mathsf{ict}_{\mathrm{msg}}),\\
\mathsf{sh}_t &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, 0, \mathbf{v}_0, \mathbf{0}, \mathsf{isk}_0, \mathfrak{o}, \mathsf{ict}_t),\\
\mathsf{sh}_\ell &\leftarrow \mathsf{IPFE.Dec}(\mathsf{impk}, \ell, \mathbf{v}_\ell, \mathbf{0}, \mathsf{isk}_\ell, \mathfrak{o}, \mathsf{ict}_\ell) \qquad \text{for all } \ell \in [L].
\end{aligned}$$

The algorithm reconstructs the garbling from the shares as

$$\begin{aligned}
w_{\mathrm{msg}} &\leftarrow \mathsf{sh}_{\mathrm{msg}} + \mathsf{sh}_Q^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{w}_{\mathrm{out}}), & \mathbf{t}^\intercal &\leftarrow \mathsf{sh}_t^\intercal + \mathsf{sh}_Q^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{T}),\\
\mathbf{w}_\ell^\intercal &\leftarrow \mathsf{sh}_\ell^\intercal + \mathsf{sh}_Q^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0} + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times}) & &\text{for all } \ell \in [L].
\end{aligned}$$

It evaluates the garbling by

$$w_{\mathrm{out}} \leftarrow \mathsf{NLG.Eval}(\mathsf{pp}, f, \mathbf{R}, \mathbf{x}, \{\mathbf{w}_\ell^\intercal\}_{\ell\in[L]}, \mathbf{t}^\intercal)$$

and outputs

$$\mu' \leftarrow \begin{cases} 0, & \text{if } w_{\mathrm{msg}} - w_{\mathrm{out}} \in [-2^{-\kappa}q, 2^\kappa q];\\ 1, & \text{otherwise.}\end{cases}$$

Theorem 31 (¶). *Construction 5 is correct (Definition 1) if*

- *the modulus q output by* $\mathsf{NLG.Setup}(\mathsf{param}')$ *is always a prime and always satisfies* $B_{\mathrm{out}}(\mathsf{param}') \le 2^{-\kappa}q$, *and*
- *it holds that* $(nK+1)B_{\mathsf{IPFE}}(q, K, 2(n+n^2K)) \le B_{\mathrm{in}}(\mathsf{param}')$.

Proof (Theorem 31). By the correctness of IPFE and how we set the vectors, we have

$$\mathsf{sh}_Q^\intercal = \mathbf{s}_Q^\intercal\mathbf{Q} + \widetilde{\mathbf{g}}^\intercal \otimes \mathbf{e}_{\mathrm{sr}}^\intercal + \mathbf{e}_Q^\intercal = \mathbf{s}_Q^\intercal\mathbf{Q} + \mathbf{e}_{\mathrm{sr}}^\intercal\widetilde{\mathbf{G}} + \mathbf{e}_Q^\intercal$$

for some B_{IPFE}-bounded errors $\mathbf{e}_{\mathrm{sr}}, \mathbf{e}_Q$. For any matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times\star}$,

$$\mathsf{sh}_Q^\intercal\widetilde{\mathbf{G}}^{-1}(\mathbf{A}) = (\mathbf{s}_Q^\intercal\mathbf{Q} + \mathbf{e}_{\mathrm{sr}}^\intercal\widetilde{\mathbf{G}} + \mathbf{e}_Q^\intercal)\widetilde{\mathbf{G}}^{-1}(\mathbf{A}) = \underbrace{\mathbf{s}_Q^\intercal\mathbf{Q}\widetilde{\mathbf{G}}^{-1}(\mathbf{A})}_{\text{one-time pad}} + \underbrace{\mathbf{e}_{\mathrm{sr}}^\intercal\mathbf{A}}_{\text{“attached” error}} + \underbrace{\mathbf{e}_Q^\intercal\widetilde{\mathbf{G}}^{-1}(\mathbf{A})}_{\text{small error}}.$$

For the table shares, we have

$$\begin{aligned}
\mathsf{sh}_t = \mathbf{U}_t^\intercal\mathbf{v}_0 + \mathbf{e}_t &= \mathbf{T}^\intercal\mathbf{S}\mathbf{r} - ((\widetilde{\mathbf{G}}^{-1}(\mathbf{T}))^\intercal \otimes \mathbf{s}_Q^\intercal)\mathsf{col}(\mathbf{Q}) + \mathbf{e}_t\\
&= \mathbf{T}^\intercal\mathbf{S}\mathbf{r} - \mathsf{col}(\mathbf{s}_Q^\intercal\mathbf{Q}\widetilde{\mathbf{G}}^{-1}(\mathbf{T})) + \mathbf{e}_t = \big((\mathbf{S}\mathbf{r})^\intercal\mathbf{T} + \mathbf{e}_t^\intercal - \mathbf{s}_Q^\intercal\mathbf{Q}\widetilde{\mathbf{G}}^{-1}(\mathbf{T})\big)^\intercal,
\end{aligned}$$

where the third equality follows from $\mathsf{col}(\mathbf{ABC}) = (\mathbf{C}^\intercal \otimes \mathbf{A})\mathsf{col}(\mathbf{B})$ and the last equality holds because $\mathbf{s}_Q^\intercal\mathbf{Q}\widetilde{\mathbf{G}}^{-1}(\mathbf{T})$ is a row vector. The reconstructed garbled table is hence

$$\mathbf{t}^\intercal = \underbrace{(\mathbf{S}\mathbf{r})^\intercal\mathbf{T} + \mathbf{e}_t^\intercal - \mathbf{s}_Q^\intercal\mathbf{Q}\widetilde{\mathbf{G}}^{-1}(\mathbf{T})}_{\mathsf{sh}_t^\intercal} + \underbrace{\mathbf{s}_Q^\intercal\mathbf{Q}\widetilde{\mathbf{G}}^{-1}(\mathbf{T}) + \mathbf{e}_{\mathrm{sr}}^\intercal\mathbf{T} + \mathbf{e}_Q^\intercal\widetilde{\mathbf{G}}^{-1}(\mathbf{T})}_{\mathsf{sh}_Q^\intercal\widetilde{\mathbf{G}}^{-1}(\mathbf{T})} = (\mathbf{S}\mathbf{r} + \mathbf{e}_{\mathrm{sr}})^\intercal\mathbf{T} + \widetilde{\mathbf{e}}_t^\intercal,$$

where $\widetilde{\mathbf{e}}_{\mathsf{t}} = \mathbf{e}_{\mathsf{t}} + (\widetilde{\mathbf{G}}^{-1}(\mathbf{T}))^{\top}\mathbf{e}_{\mathsf{Q}}$ is $(nK+1)B_{\mathsf{IPFE}}$-bounded. Similarly,

$$\begin{aligned} w_{\mathsf{msg}} &= (\mathbf{S}\mathbf{r} + \mathbf{e}_{\mathsf{sr}})^{\top}\mathbf{w}_{\mathsf{out}} + \mu s_{\mathsf{msg}} + \widetilde{e}_{\mathsf{out}}, \\ \mathbf{w}_{\ell}^{\top} &= (\mathbf{S}\mathbf{r} + \mathbf{e}_{\mathsf{sr}})^{\top}(\mathbf{W}_{\ell,0} + \mathbf{x}[\ell]\mathbf{W}_{\ell,\times}) + \widetilde{\mathbf{e}}_{\ell}^{\top} \qquad \text{for all } \ell \in [L], \end{aligned}$$

with $\widetilde{e}_{\mathsf{out}}, \widetilde{\mathbf{e}}_{\ell}$'s bounded by $(nK+1)B_{\mathsf{IPFE}}$. The garbling uses secret $\mathbf{S}\mathbf{r} + \mathbf{e}_{\mathsf{sr}} \in \mathbb{Z}_q^n$ and its noises ($\widetilde{\mathbf{e}}$'s) are bounded by $(nK+1)B_{\mathsf{IPFE}}(q,K,Z) \le B_{\mathsf{in}}(\mathsf{param}')$. Therefore, by the correctness of NLG,

$$w_{\mathsf{out}} - ((\mathbf{S}\mathbf{r} + \mathbf{e}_{\mathsf{sr}})^{\top}\mathbf{w}_{\mathsf{out}} + e'_{\mathsf{out}}) \in [-B_{\mathsf{out}}(\mathsf{param}'), B_{\mathsf{out}}(\mathsf{param}')] \subseteq [-2^{-\kappa}q, 2^{-\kappa}q].$$

If $\mu = 0$, it always holds that $w_{\mathsf{msg}} - w_{\mathsf{out}} \in [-2^{-\kappa}q, 2^{-\kappa}q]$, which never holds when $\mu = 1$ by our choice of s_{msg}. We conclude that Construction 10 is correct. □

8.3 Security and Summary

Theorem 32 (¶). *Construction 10 is very selectively secure (Definition 2) if*

- $\mathsf{LWE}_{n,\mathrm{poly}(\lambda),q,\sigma}$ *(Assumption 1) holds for some* $\sigma \le \frac{2^{-\kappa-6}\sigma_{\mathsf{IPFE}}}{nK\sqrt{\kappa}}$*, and*
- *it holds that* $2^{\kappa+6}B_{\mathsf{NLG}}(\mathsf{param}') \le \sigma_{\mathsf{IPFE}}$.

Proof (Theorem 32). Let $\mathcal{A}$ be an efficient adversary and assume that it always chooses challenges such that $f(\mathbf{x}_j) = 0$ for all $j \in [J]$. Consider the following evasive IPFE sampler $\mathcal{S}^{\beta} = (\mathcal{S}_{\mathsf{V}}, \mathcal{S}_{\mathsf{U}}^{\beta})$ per Definition 5.

- $\mathcal{S}_{\mathsf{V}}(r_{\mathsf{pub}})$ parses $r_{\mathsf{pub}} = (r_{\mathcal{A}}, r_{\mathsf{NLG.Setup}}, r_{\mathsf{rq}}, r_{\mathsf{NLG.GenF}})$. It runs $\mathcal{A}(r_{\mathcal{A}})$ to obtain

  $$\mathsf{param} = (\mathsf{param}', 1^L), \qquad \{\mathbf{x}_j\}_{j\in[J]} \quad (\mathbf{x}_j \in \{0,1\}^L), \qquad f : \mathbb{Z}^L \to \{0,1,\perp\}.$$

  The algorithm next sets up the noisy linear garbling by

  $$\mathsf{pp} = (q, 1^n, \dots) \leftarrow \mathsf{NLG.Setup}(\mathsf{param}'; r_{\mathsf{NLG.Setup}}),$$

  and computes K, Z as in Setup of Construction 10. It then uses r_{rq} to sample $\mathbf{r}_1, \dots, \mathbf{r}_J \in \mathbb{Z}_q^n$, $\mathbf{Q}_1, \dots, \mathbf{Q}_J \in \mathbb{Z}_q^{n\times nK}$ uniformly at random in a straight-forward way.[24] The algorithm sets for all $j \in [J]$,

  $$\mathbf{V}_{\mathsf{Q},j} = \begin{pmatrix} \mathbf{Q}_j \\ \mathbf{0}_{(Z-n)\times nK} \end{pmatrix}, \quad \mathbf{v}_{j,0} = \begin{pmatrix} \mathbf{r}_j \\ \mathsf{col}(\mathbf{Q}_j) \\ 1 \\ \mathbf{0}_{n-1+n^2K} \end{pmatrix}, \quad \mathbf{v}_{j,\ell} = \begin{pmatrix} (1-\mathbf{x}_j[\ell])\mathbf{r}_j \\ (1-\mathbf{x}_j[\ell])\mathsf{col}(\mathbf{Q}_j) \\ \mathbf{x}_j[\ell]\mathbf{r}_j \\ \mathbf{x}_j[\ell]\mathsf{col}(\mathbf{Q}_j) \end{pmatrix} \text{ for all } \ell \in [L].$$

  It runs

  $$(1^n, \mathbf{w}_{\mathsf{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}, \mathbf{T}, \mathbf{R}) \leftarrow \mathsf{NLG.GenF}(\mathsf{pp}, f; r_{\mathsf{NLG.GenF}}).$$

  Suppose $\mathbf{W}_{\ell,0} \in \mathbb{Z}_q^{n\times m_\ell}$ and $\mathbf{T} \in \mathbb{Z}_q^{n\times m_{\mathsf{t}}}$, the algorithm outputs

  $$\begin{aligned} &q, 1^K, 1^Z, && \mathsf{ID} = \{-1, 0, 1, \dots, L\}, && 1^{J_{\mathsf{id}}} = 1^J, \\ &1^{I_{-1,\mathfrak{o}}} = 1^0, && 1^{I_{0,\mathfrak{o}}} = 1^{m_{\mathsf{t}}+1}, && 1^{I_{\ell,\mathfrak{o}}} = 1^{m_\ell}, \\ &1^{I_{-1,\mathfrak{g}}} = 1^1, && 1^{I_{0,\mathfrak{g}}} = 1^{I_{\ell,\mathfrak{g}}} = 1^0, && \sigma_{\mathsf{pre}} = \sigma_{\mathsf{IPFE}}, \end{aligned}$$

[24] Precisely speaking, r_{rq} must be efficiently sampleable conditioned on and given $\mathbf{r}_1, \dots, \mathbf{r}_J$ and $\mathbf{Q}_1, \dots, \mathbf{Q}_J$.

$$\mathbf{v}_{-1,\mathfrak{g}} = (\mathbf{v}_{\mathrm{Q},1}, \ldots, \mathbf{v}_{\mathrm{Q},J}), \qquad \mathbf{v}_{-1,\mathfrak{o}} = \mathbf{0},$$
$$\mathbf{V}_{0,\mathfrak{o}} = (\mathbf{v}_{0,0}, \ldots, \mathbf{v}_{J,0}), \qquad \mathbf{V}_{0,\mathfrak{g}} = \mathbf{0},$$
$$\mathbf{V}_{\ell,\mathfrak{o}} = (\mathbf{v}_{1,\ell}, \ldots, \mathbf{v}_{J,\ell}), \qquad \mathbf{V}_{\ell,\mathfrak{g}} = \mathbf{0},$$

where the indices are $\mathsf{id} \in \mathsf{ID}$ and $\ell \in [L]$.

- $\mathcal{S}^{\beta}_{\mathrm{U}}(r_{\mathrm{pub}}; r_{\mathrm{priv}})$ first runs the deterministic subprocedure $\mathcal{S}^{\beta}_{\mathrm{A0}}(r_{\mathrm{pub}})$, which does the following. It first reruns $\mathcal{S}_{\mathrm{V}}(r_{\mathrm{pub}})$ to obtain $\mathbf{w}_{\mathrm{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}, \mathbf{T}$. The algorithm then rearranges them into $\mathbf{A}_{\mathrm{Q}}, \mathbf{A}_{\mathrm{msg}}, \{\mathbf{A}_{\mathrm{t},i}\}_{i\in[m_{\mathrm{t}}]}, \{\mathbf{A}_{\ell,i}\}_{\ell\in[L],i\in[m_\ell]}$ such that for all $\mathbf{s}_{\mathrm{Q}}, \widetilde{\mathbf{s}}_1, \ldots, \widetilde{\mathbf{s}}_n \in \mathbb{Z}_q^n$ and $s_{\mathrm{msg}} \in \mathbb{Z}_q$,

$$\begin{aligned}
\mathbf{d}_0^{\mathsf{T}}\mathbf{A}_{\mathrm{Q}} &= \mathbf{u}_{\mathrm{Q}}^{\mathsf{T}} = \left(\mathbf{s}_{\mathrm{Q}}^{\mathsf{T}}, \mathbf{0}_{1\times(Z-n)}\right),\\
\mathbf{d}_0^{\mathsf{T}}\mathbf{A}_{\mathrm{msg}} &= \mathbf{u}_{\mathrm{msg}}^{\mathsf{T}} = \left(\mathbf{w}_{\mathrm{out}}^{\mathsf{T}}\mathbf{S}, -(\widetilde{\mathbf{G}}^{-1}(\mathbf{w}_{\mathrm{out}}) \otimes \mathbf{s}_{\mathrm{Q}})^{\mathsf{T}}, \beta s_{\mathrm{msg}}, \mathbf{0}_{1\times(n-1+n^2K)}\right),\\
(\mathbf{A}_{\mathrm{t},1}^{\mathsf{T}}\mathbf{d}_0, \ldots, \mathbf{A}_{\mathrm{t},m_{\mathrm{t}}}^{\mathsf{T}}\mathbf{d}_0)^{\mathsf{T}} &= \mathbf{U}_{\mathrm{t}}^{\mathsf{T}} = \left(\mathbf{T}^{\mathsf{T}}\mathbf{S}, -(\widetilde{\mathbf{G}}^{-1}(\mathbf{T}) \otimes \mathbf{s}_{\mathrm{Q}})^{\mathsf{T}}, \mathbf{0}_{m_{\mathrm{t}}\times(n+n^2K)}\right),\\
(\mathbf{A}_{\ell,1}^{\mathsf{T}}\mathbf{d}_0, \ldots, \mathbf{A}_{\ell,m_\ell}^{\mathsf{T}}\mathbf{d}_0)^{\mathsf{T}} &= \mathbf{U}_{\ell}^{\mathsf{T}}\\
&= \left(\mathbf{W}_{\ell,0}^{\mathsf{T}}\mathbf{S}, -(\widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0}) \otimes \mathbf{s}_{\mathrm{Q}})^{\mathsf{T}}, (\mathbf{W}_{\ell,0}^{\mathsf{T}} + \mathbf{W}_{\ell,\times}^{\mathsf{T}})\mathbf{S}, -(\widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0} + \mathbf{W}_{\ell,\times}) \otimes \mathbf{s}_{\mathrm{Q}})^{\mathsf{T}}\right),
\end{aligned}$$

where $\ell \in [L]$ and $\mathbf{d}_0^{\mathsf{T}} = (s_{\mathrm{msg}}, \mathbf{s}_{\mathrm{Q}}^{\mathsf{T}}, \widetilde{\mathbf{s}}_1^{\mathsf{T}}, \ldots, \widetilde{\mathbf{s}}_n^{\mathsf{T}})$ and $\mathbf{S} = (\widetilde{\mathbf{s}}_1, \ldots, \widetilde{\mathbf{s}}_n)$. This is possible since every entry on the right-hand side is linear in $\mathbf{d}_0$. The algorithm $\mathcal{S}_{\mathrm{A0}}$ outputs

$$\begin{aligned}
1^{n'} &= 1^{n(n+1)+1}, & \mathbf{A}_{-1,\mathfrak{g},0,1} &= \mathbf{A}_{\mathrm{Q}},\\
\mathbf{A}_{0,\mathfrak{o},0,m_{\mathrm{t}}+1} &= \mathbf{A}_{\mathrm{msg}}, & \mathbf{A}_{0,\mathfrak{o},0,i} &= \mathbf{A}_{\mathrm{t},i} && \text{(for all } i \in [m_{\mathrm{t}}]),\\
& & \mathbf{A}_{\ell,\mathfrak{o},0,i} &= \mathbf{A}_{\ell,i} && \text{(for all } \ell \in [L],\ i \in [m_\ell]).
\end{aligned}$$

Completing $\mathcal{S}_{\mathrm{A0}}$, the algorithm $\mathcal{S}_{\mathrm{U}}$ samples $\mathbf{d}_0 \xleftarrow{\$} \mathbb{Z}_q^{n'}$ using r_{priv} and outputs

$$\mathbf{U}_{\mathsf{id},\mathfrak{s}}^{\mathsf{T}} = \begin{pmatrix} \mathbf{d}_0^{\mathsf{T}}\mathbf{A}_{\mathsf{id},\mathfrak{s},0,1} \\ \vdots \\ \mathbf{d}_0^{\mathsf{T}}\mathbf{A}_{\mathsf{id},\mathfrak{s},0,I_{\mathsf{id},\mathfrak{s}}} \end{pmatrix} \qquad \text{for all } \mathsf{id} \in \mathsf{ID}, \mathfrak{s} \in \{\mathfrak{o}, \mathfrak{g}\}.$$

By definition, $\mathcal{S}^{\beta}$ is a restricted sampler (Definition 6). We have the following:

Claim 33 (¶). $\mathcal{S}^{\beta}$ has pseudorandom structurally noisy inner products, i.e., $\mathsf{IPFEsec}^{\mathcal{S}^{\beta}}_{\mathrm{pre}}$ (Definition 5) holds for both $\beta \in \{0,1\}$.

The very selective security of Construction 10 follows from Claim 33. Consider the following hybrids.

- H_0^{β} is $\mathsf{Exp}^{\beta}_{\mathrm{ABE}}$ for Construction 10. Note that s_{msg} in the challenge ciphertext is uniformly random over $\mathbb{Z}_q \setminus [-2\cdot 2^{-\kappa}q, 2\cdot 2^{-\kappa}q]$.
- In H_1^{β}, the scalar s_{msg} is changed to be uniformly random over $\mathbb{Z}_q$. Clearly, $\mathsf{H}_0^{\beta} \approx_{\mathrm{s}} \mathsf{H}_1^{\beta}$. Moreover, H_1^{β} corresponds to the left distribution of $\mathsf{IPFEsec}^{\mathcal{S}^{\beta}}_{\mathrm{post}}$.
- In H_2^{β}, the vectors inside IPFE ciphertexts (components of the ABE challenge ciphertext) are replaced by random, which corresponds to the right distribution of $\mathsf{IPFEsec}^{\mathcal{S}^{\beta}}_{\mathrm{post}}$. By Claim 33 and the restricted-σ_{IPFE}-security of IPFE, we have $\mathsf{H}_1^{\beta} \approx \mathsf{H}_2^{\beta}$.

Lastly, $\mathsf{H}_2^0 \equiv \mathsf{H}_2^1$. By hybrid argument, we conclude $\mathsf{Exp}^0_{\mathrm{ABE}} \approx \mathsf{Exp}^1_{\mathrm{ABE}}$, completing the proof. □

Proof (Claim 33). Fix $\beta \in \{0,1\}$ and consider the following hybrids.

- H_0 is the left distribution of $\mathsf{IPFEsec}^{\mathcal{S}^\beta}_{\mathrm{pre}}$. Recall that the public randomness is $r_{\mathrm{pub}} = (r_{\mathcal{A}}, r_{\mathsf{NLG.Setup}}, r_{\mathrm{rq}}, r_{\mathsf{NLG.GenF}})$. Adopting the notations of the ABE decryption algorithm, the noisy inner products are

$$\begin{aligned}
\mathsf{sh}^\intercal_{\mathrm{Q},j} &= \mathbf{s}_{\mathrm{Q}}^\intercal \mathbf{Q}_j + \mathbf{e}^\intercal_{\mathrm{sr},j}\widetilde{\mathbf{G}} + \mathbf{e}^\intercal_{\mathrm{Q},j},\\
\mathsf{sh}_{\mathrm{msg},j} &= (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{w}_{\mathrm{out}} + e_{\mathrm{out},j} + \beta s_{\mathrm{msg}} - \mathbf{s}_{\mathrm{Q}}^\intercal \mathbf{Q}_j \widetilde{\mathbf{G}}^{-1}(\mathbf{w}_{\mathrm{out}}),\\
\mathsf{sh}^\intercal_{\mathrm{t},j} &= (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{T} + \mathbf{e}^\intercal_{\mathrm{t},j} - \mathbf{s}_{\mathrm{Q}}^\intercal \mathbf{Q}_j \widetilde{\mathbf{G}}^{-1}(\mathbf{T}),\\
\mathsf{sh}^\intercal_{j,\ell} &= (\mathbf{S}\mathbf{r}_j)^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}^\intercal_{j,\ell} - \mathbf{s}_{\mathrm{Q}}^\intercal \mathbf{Q}_j \widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) \quad (\ell \in [L]),
\end{aligned}$$

  where $j \in [J]$ and the entries of $\mathbf{e}$'s are independent $\mathcal{D}_{\mathbb{Z},\sigma_{\mathsf{IPFE}}}$.

- In H_1, additional small noises are attached. For all $j \in [J]$, the inner products become

$$\begin{aligned}
\mathsf{sh}^\intercal_{\mathrm{Q},j} &= \mathbf{s}_{\mathrm{Q}}^\intercal \mathbf{Q}_j + \widetilde{\mathbf{e}}^\intercal_{\mathrm{Q},j} + \widetilde{\mathbf{e}}^\intercal_{\mathrm{sr},j}\widetilde{\mathbf{G}} + \mathbf{e}^\intercal_{\mathrm{sr},j}\widetilde{\mathbf{G}} + \mathbf{e}^\intercal_{\mathrm{Q},j},\\
\mathsf{sh}_{\mathrm{msg},j} &= (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{w}_{\mathrm{out}} + e_{\mathrm{out},j} + \beta s_{\mathrm{msg}} - (\mathbf{s}_{\mathrm{Q}}^\intercal \mathbf{Q}_j + \widetilde{\mathbf{e}}^\intercal_{\mathrm{Q},j})\widetilde{\mathbf{G}}^{-1}(\mathbf{w}_{\mathrm{out}}),\\
\mathsf{sh}^\intercal_{\mathrm{t},j} &= (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{T} + \mathbf{e}^\intercal_{\mathrm{t},j} - (\mathbf{s}_{\mathrm{Q}}^\intercal \mathbf{Q}_j + \widetilde{\mathbf{e}}^\intercal_{\mathrm{Q},j})\widetilde{\mathbf{G}}^{-1}(\mathbf{T}),\\
\mathsf{sh}^\intercal_{j,\ell} &= (\mathbf{S}\mathbf{r}_j)^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}^\intercal_{j,\ell} - (\mathbf{s}_{\mathrm{Q}}^\intercal \mathbf{Q}_j + \widetilde{\mathbf{e}}^\intercal_{\mathrm{Q},j})\widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}),
\end{aligned}$$

  where $\ell \in [L]$ and the entries of $\widetilde{\mathbf{e}}$ are independent $\mathcal{D}_{\mathbb{Z},\sigma,\le\sigma\sqrt{\kappa}}$. In $\mathsf{sh}_{\mathrm{Q},j}$'s, the $\widetilde{\mathbf{e}}_{\mathrm{Q},j}$'s and $\widetilde{\mathbf{e}}_{\mathrm{sr},j}$'s are flooded by $\mathbf{e}_{\mathrm{Q},j}$'s and $\mathbf{e}_{\mathrm{sr},j}$'s, respectively. In the other inner products, each $\widetilde{\mathbf{e}}^\intercal_{\mathrm{Q},j}\widetilde{\mathbf{G}}^{-1}(\cdots)$ is flooded by the e or $\mathbf{e}$ in that term. Since

$$\begin{aligned}
\text{(for } \mathsf{sh}_{\mathrm{Q},j}\text{'s)} \qquad & \sigma\sqrt{\kappa} \le \frac{2^{-\kappa-6}\sigma_{\mathsf{IPFE}}}{nK} \le 2^{-\kappa-6}\sigma_{\mathsf{IPFE}},\\
\text{(for the rest)} \qquad & nK \cdot \sigma\sqrt{\kappa} \le nK \cdot \frac{2^{-\kappa-6}\sigma_{\mathsf{IPFE}}}{nK} = 2^{-\kappa-6}\sigma_{\mathsf{IPFE}},
\end{aligned}$$

  Lemma 2 applies and $\mathsf{H}_0 \approx_{\mathrm{s}} \mathsf{H}_1$.

- In H_2, the $\widetilde{\mathbf{e}}$'s are no longer truncated, i.e., their entries follow $\mathcal{D}_{\mathbb{Z},\sigma}$. We have $\mathsf{H}_1 \approx_{\mathrm{s}} \mathsf{H}_2$ by Lemma 1.

- In H_3, the LWE samples $(\mathbf{s}_{\mathrm{Q}}^\intercal \mathbf{Q}_j + \widetilde{\mathbf{e}}^\intercal_{\mathrm{Q},j})$ are replaced by random. For all $j \in [J]$, sample $\mathbf{q}_j \xleftarrow{\$} \mathbb{Z}_q^n$ and the inner products are

$$\begin{aligned}
\mathsf{sh}^\intercal_{\mathrm{Q},j} &= \mathbf{q}_j^\intercal + \widetilde{\mathbf{e}}^\intercal_{\mathrm{sr},j}\widetilde{\mathbf{G}} + \mathbf{e}^\intercal_{\mathrm{sr},j}\widetilde{\mathbf{G}} + \mathbf{e}^\intercal_{\mathrm{Q},j},\\
\mathsf{sh}_{\mathrm{msg},j} &= (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{w}_{\mathrm{out}} + e_{\mathrm{out},j} + \beta s_{\mathrm{msg}} - \mathbf{q}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{w}_{\mathrm{out}}),\\
\mathsf{sh}^\intercal_{\mathrm{t},j} &= (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{T} + \mathbf{e}^\intercal_{\mathrm{t},j} - \mathbf{q}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{T}),\\
\mathsf{sh}^\intercal_{j,\ell} &= (\mathbf{S}\mathbf{r}_j)^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}^\intercal_{j,\ell} - \mathbf{q}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) \quad (\ell \in [L]).
\end{aligned}$$

  By $\mathsf{LWE}_{n,nJ,q,\sigma}$, we have $\mathsf{H}_2 \approx \mathsf{H}_3$. In this step, we rely on $\mathbf{Q}$'s sampling being "straight-forward" (for reduction to sample r_{rq} conditioned on and given $\mathbf{Q}$'s).

- In H_4, we perform a change of variable. For all $j \in [J]$, sample $\widetilde{\mathbf{q}}_j \xleftarrow{\$} \mathbb{Z}_q^n$ and set $\mathbf{q}_j^\intercal = \widetilde{\mathbf{q}}_j^\intercal - \widetilde{\mathbf{e}}_{\mathsf{Q},j}^\intercal \widetilde{\mathbf{G}}$, then the inner products are

$$\begin{aligned}
\mathsf{sh}_{\mathsf{Q},j}^\intercal &= \widetilde{\mathbf{q}}_j^\intercal + \mathbf{e}_{\mathsf{sr},j}^\intercal \widetilde{\mathbf{G}} + \mathbf{e}_{\mathsf{Q},j}^\intercal, \\
\mathsf{sh}_{\mathsf{msg},j} &= (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{w}_{\mathsf{out}} + e_{\mathsf{out},j} + \beta s_{\mathsf{msg}} - (\widetilde{\mathbf{q}}_j^\intercal - \widetilde{\mathbf{e}}_{\mathsf{Q},j}^\intercal \widetilde{\mathbf{G}}) \widetilde{\mathbf{G}}^{-1}(\mathbf{w}_{\mathsf{out}}) \\
&= (\mathbf{S}\mathbf{r}_j + \widetilde{\mathbf{e}}_{\mathsf{Q},j})^\intercal \mathbf{w}_{\mathsf{out}} + e_{\mathsf{out},j} + \beta s_{\mathsf{msg}} - \widetilde{\mathbf{q}}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{w}_{\mathsf{out}}), \\
\mathsf{sh}_{\mathsf{t},j}^\intercal &= (\mathbf{S}\mathbf{r}_j)^\intercal \mathbf{T} + \mathbf{e}_{\mathsf{t},j}^\intercal - (\widetilde{\mathbf{q}}_j^\intercal - \widetilde{\mathbf{e}}_{\mathsf{Q},j}^\intercal \widetilde{\mathbf{G}}) \widetilde{\mathbf{G}}^{-1}(\mathbf{T}) \\
&= (\mathbf{S}\mathbf{r}_j + \widetilde{\mathbf{e}}_{\mathsf{Q},j})^\intercal \mathbf{T} + \mathbf{e}_{\mathsf{t},j}^\intercal - \widetilde{\mathbf{q}}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{T}), \\
\mathsf{sh}_{j,\ell}^\intercal &= (\mathbf{S}\mathbf{r}_j)^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_{j,\ell}^\intercal - (\widetilde{\mathbf{q}}_j^\intercal - \widetilde{\mathbf{e}}_{\mathsf{Q},j}^\intercal \widetilde{\mathbf{G}}) \widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) \\
&= (\mathbf{S}\mathbf{r}_j + \widetilde{\mathbf{e}}_{\mathsf{Q},j})^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_{j,\ell}^\intercal - \widetilde{\mathbf{q}}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}),
\end{aligned}$$

  where $\ell \in [L]$. Clearly, $\mathsf{H}_3 \equiv \mathsf{H}_4$.

- In H_5, the LWE samples $(\mathbf{S}\mathbf{r}_j + \widetilde{\mathbf{e}}_{\mathsf{Q},j})$ are replaced by random. For all $j \in [J]$, sample $\mathbf{s}_j \xleftarrow{\$} \mathbb{Z}_q^n$ and the inner products are

$$\begin{aligned}
\mathsf{sh}_{\mathsf{Q},j}^\intercal &= \widetilde{\mathbf{q}}_j^\intercal + \mathbf{e}_{\mathsf{sr},j}^\intercal \widetilde{\mathbf{G}} + \mathbf{e}_{\mathsf{Q},j}^\intercal, \\
\mathsf{sh}_{\mathsf{msg},j} &= \mathbf{s}_j^\intercal \mathbf{w}_{\mathsf{out}} + e_{\mathsf{out},j} + \beta s_{\mathsf{msg}} - \widetilde{\mathbf{q}}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{w}_{\mathsf{out}}), \\
\mathsf{sh}_{\mathsf{t},j}^\intercal &= \mathbf{s}_j^\intercal \mathbf{T} + \mathbf{e}_{\mathsf{t},j}^\intercal - \widetilde{\mathbf{q}}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{T}), \\
\mathsf{sh}_{j,\ell}^\intercal &= \mathbf{s}_j^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \mathbf{e}_{j,\ell}^\intercal - \widetilde{\mathbf{q}}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) \qquad (\ell \in [L]).
\end{aligned}$$

  By n hybrids of $\mathsf{LWE}_{n,J,q,\sigma}$ over the rows of $\mathbf{S}$ with public matrix $(\mathbf{r}_1, \ldots, \mathbf{r}_J)$, we have $\mathsf{H}_4 \approx \mathsf{H}_5$.

- In H_6, additional noises generated by GenNoise are attached to the garblings. For all $j \in [J]$, run

$$(\overline{e}_{\mathsf{out},j}, \{\overline{\mathbf{e}}_{j,\ell}\}_{\ell\in[L]}, \overline{\mathbf{e}}_{\mathsf{t},j}) \xleftarrow{\$} \mathsf{GenNoise}(\mathsf{param}', \mathsf{pp}, f, \mathbf{w}_{\mathsf{out}}, \{\mathbf{W}_{\ell,0}, \mathbf{W}_{\ell,\times}\}_{\ell\in[L]}\mathbf{T}, \mathbf{R}, \mathbf{x}_j),$$

  and set the inner products to

$$\begin{aligned}
\mathsf{sh}_{\mathsf{Q},j}^\intercal &= \widetilde{\mathbf{q}}_j^\intercal + \mathbf{e}_{\mathsf{sr},j}^\intercal \widetilde{\mathbf{G}} + \mathbf{e}_{\mathsf{Q},j}^\intercal, \\
\mathsf{sh}_{\mathsf{msg},j} &= \mathbf{s}_j^\intercal \mathbf{w}_{\mathsf{out}} + \overline{e}_{\mathsf{out},j} + e_{\mathsf{out},j} + \beta s_{\mathsf{msg}} - \widetilde{\mathbf{q}}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{w}_{\mathsf{out}}), \\
\mathsf{sh}_{\mathsf{t},j}^\intercal &= \mathbf{s}_j^\intercal \mathbf{T} + \overline{\mathbf{e}}_{\mathsf{t},j}^\intercal + \mathbf{e}_{\mathsf{t},j}^\intercal - \widetilde{\mathbf{q}}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{T}), \\
\mathsf{sh}_{j,\ell}^\intercal &= \mathbf{s}_j^\intercal (\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) + \overline{\mathbf{e}}_{j,\ell}^\intercal + \mathbf{e}_{j,\ell}^\intercal - \widetilde{\mathbf{q}}_j^\intercal \widetilde{\mathbf{G}}^{-1}(\mathbf{W}_{\ell,0} + \mathbf{x}_j[\ell]\mathbf{W}_{\ell,\times}) \quad (\ell \in [L]).
\end{aligned}$$

  The $\overline{e}, \overline{\mathbf{e}}$'s introduced into the inner products are bounded by $B_{\mathsf{NLG}} \le 2^{-\kappa-6}\sigma_{\mathsf{IPFE}}$, so they are flooded (Lemma 2) by $e, \mathbf{e}$'s. We have $\mathsf{H}_5 \approx_s \mathsf{H}_6$.

- In H_7, all the inner products are replaced by random, i.e., the right distribution of $\mathsf{IPFEsec}_{\mathsf{pre}}^{\mathcal{S}^\beta}$. Note that in H_6, non-$\mathsf{sh}_{\mathsf{Q},j}$'s are well-randomized garblings (by $\mathbf{s}, \overline{e}, \overline{\mathbf{e}}$'s) plus independent noises and values derived from $\mathsf{sh}_{\mathsf{Q},j}$. Therefore, by J hybrids of applying GenNoise-security of NLG, the non-$\mathsf{sh}_{\mathsf{Q},j}$'s can be replaced by random, after which $\widetilde{\mathbf{q}}_j$'s ensure $\mathsf{sh}_{\mathsf{Q},j}$'s are independent random. Therefore, $\mathsf{H}_6 \approx \mathsf{H}_7$.

By hybrid argument, $\mathsf{H}_0 \approx \mathsf{H}_7$, i.e., $\mathsf{IPFEsec}_{\mathsf{pre}}^{\mathcal{S}^\beta}$ holds. □

Summary. Combining Corollary 14, Construction 9, and Construction 10, we obtain a CP-ABE scheme for Boolean circuits with succinct ciphertexts.

Corollary 34. *Under LWE (Assumption 1) and evasive learning with structured errors (Assumption 3), there exists a CP-ABE scheme for Boolean circuits with*

$$|\mathsf{mpk}| = \mathrm{poly}(\lambda, d), \qquad |\mathsf{sk}_{\mathbf{x}}| = (L+1)\,\mathrm{poly}(\lambda, d), \qquad |\mathsf{ct}_C| = (L+1)\,\mathrm{poly}(\lambda, d),$$

where $L = |\mathbf{x}|$ is the attribute length and d is the maximum depth.

Acknowledgments. The authors were supported by NSF grants CNS-1936825 (CAREER), CNS-2026774, a JP Morgan AI Research Award, a Cisco Research Award, and a Simons Collaboration on the Theory of Algorithmic Fairness. The views expressed are those of the authors and do not reflect the official policy or position of the funding agencies. The authors thank the anonymous reviewers of Eurocrypt 2024 for their valuable comments.

References

[ABB10] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 553–572. Springer, Heidelberg, May / June 2010.

[ABDP15] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 733–751. Springer, Heidelberg, March / April 2015.

[ACF+18] Michel Abdalla, Dario Catalano, Dario Fiore, Romain Gay, and Bogdan Ursu. Multi-input functional encryption for inner products: Function-hiding realizations and constructions without pairings. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 597–627. Springer, Heidelberg, August 2018.

[Agr19] Shweta Agrawal. Indistinguishability obfuscation without multilinear maps: New methods for bootstrapping and instantiation. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 191–225. Springer, Heidelberg, May 2019.

[AIK11] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. How to garble arithmetic circuits. In Rafail Ostrovsky, editor, *52nd FOCS*, pages 120–129. IEEE Computer Society Press, October 2011.

[ALMT20] Shweta Agrawal, Benoît Libert, Monosij Maitra, and Radu Titiu. Adaptive simulation security for inner product functional encryption. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part I*, volume 12110 of *LNCS*, pages 34–64. Springer, Heidelberg, May 2020.

[ALS16] Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 333–362. Springer, Heidelberg, August 2016.

[AMY19] Shweta Agrawal, Monosij Maitra, and Shota Yamada. Attribute based encryption (and more) for nondeterministic finite automata from LWE. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 765–797. Springer, Heidelberg, August 2019.

[AP20] Shweta Agrawal and Alice Pellet-Mary. Indistinguishability obfuscation without maps: Attacks and fixes for noisy linear FE. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 110–140. Springer, Heidelberg, May 2020.

[AS17] Shweta Agrawal and Ishaan Preet Singh. Reusable garbled deterministic finite automata from learning with errors. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *ICALP 2017*, volume 80 of *LIPIcs*, pages 36:1–36:13. Schloss Dagstuhl, July 2017.

[AWY20] Shweta Agrawal, Daniel Wichs, and Shota Yamada. Optimal broadcast encryption from LWE and pairings in the standard model. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 149–178. Springer, Heidelberg, November 2020.

[AY20] Shweta Agrawal and Shota Yamada. Optimal broadcast encryption from pairings and LWE. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 13–43. Springer, Heidelberg, May 2020.

[BDHM92] Gerhard Buntrock, Carsten Damm, Ulrich Hertrampf, and Christoph Meinel. Structure and importance of logspace-MOD class. *Mathematical Systems Theory*, 25(3):223–237, 1992.

[Ber84] Stuart J. Berkowitz. On computing the determinant in small parallel time using a small number of processors. *Information Processing Letters*, 18(3):147–150, 1984.

[BGG+14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 533–556. Springer, Heidelberg, May 2014.

[CHL+15] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 3–12. Springer, Heidelberg, April 2015.

[CVW18] Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. GGH15 beyond permutation branching programs: Proofs, attacks, and candidates. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 577–607. Springer, Heidelberg, August 2018.

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013.

[GKP+13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. How to run turing machines on encrypted data. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 536–553. Springer, Heidelberg, August 2013.

[GKW16] Rishab Goyal, Venkata Koppula, and Brent Waters. Semi-adaptive security and bundling functionalities made generic and easy. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 361–388. Springer, Heidelberg, October / November 2016.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 545–554. ACM Press, June 2013.

[GW20] Junqing Gong and Hoeteck Wee. Adaptively secure ABE for DFA from k-Lin and more. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 278–308. Springer, Heidelberg, May 2020.

[GWW19] Junqing Gong, Brent Waters, and Hoeteck Wee. ABE for DFA from k-Lin. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 732–764. Springer, Heidelberg, August 2019.

[HJL21] Samuel B. Hopkins, Aayush Jain, and Huijia Lin. Counterexamples to new circular security assumptions underlying iO. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part II*, volume 12826 of *LNCS*, pages 673–700, Virtual Event, August 2021. Springer, Heidelberg.

[HLL23a] Yao-Ching Hsieh, Huijia Lin, and Ji Luo. Attribute-based encryption for circuits of unbounded depth from lattices. In *64th FOCS*, pages 415–434. IEEE Computer Society Press, November 2023.

[HLL23b] Yao-Ching Hsieh, Huijia Lin, and Ji Luo. Attribute-based encryption for circuits of unbounded depth from lattices: Garbled circuits of optimal size, laconic functional evaluation, and more. Cryptology ePrint Archive, Report 2023/1716, 2023. https://eprint.iacr.org/2023/1716.

[HLL24] Yao-Ching Hsieh, Huijia Lin, and Ji Luo. A general framework for lattice-based ABE using evasive inner-product functional encryption. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part II*, volume 14652 of *LNCS*, pages 433–464. Springer, Cham, May 2024.

[IW14] Yuval Ishai and Hoeteck Wee. Partial garbling schemes and their applications. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *ICALP 2014, Part I*, volume 8572 of *LNCS*, pages 650–662. Springer, Heidelberg, July 2014.

[JLL23] Aayush Jain, Huijia Lin, and Ji Luo. On the optimal succinctness and efficiency of functional encryption and attribute-based encryption. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part III*, volume 14006 of *LNCS*, pages 479–510. Springer, Heidelberg, April 2023.

[JLLS23] Aayush Jain, Huijia Lin, Paul Lou, and Amit Sahai. Polynomial-time cryptanalysis of the subspace flooding assumption for post-quantum $i\mathcal{O}$. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part I*, volume 14004 of *LNCS*, pages 205–235. Springer, Heidelberg, April 2023.

[KW93] Mauricio Karchmer and Avi Wigderson. On span programs. In *SCT 1993*, pages 102–111. IEEE, May 1993.

[LL20a] Huijia Lin and Ji Luo. Compact adaptively secure ABE from k-Lin: Beyond NC^1 and towards NL. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 247–277. Springer, Heidelberg, May 2020.

[LL20b] Huijia Lin and Ji Luo. Succinct and adaptively secure ABE for ABP from k-lin. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 437–466. Springer, Heidelberg, December 2020.

[LLL22] Hanjun Li, Huijia Lin, and Ji Luo. ABE for circuits with constant-size secret keys and adaptive security. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 680–710. Springer, Heidelberg, November 2022.

[MP11] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. Cryptology ePrint Archive, Report 2011/501, 2011. https://eprint.iacr.org/2011/501.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, April 2012.

[Mul87] Ketan Mulmuley. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Combinatorica*, 7(1):101–104, 1987.

[QWW18] Willy Quach, Hoeteck Wee, and Daniel Wichs. Laconic function evaluation and applications. In Mikkel Thorup, editor, *59th FOCS*, pages 859–870. IEEE Computer Society Press, October 2018.

[QWW21] Willy Quach, Brent Waters, and Daniel Wichs. Targeted lossy functions and applications. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 424–453, Virtual Event, August 2021. Springer, Heidelberg.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[RW15] Yannis Rouselakis and Brent Waters. Efficient statically-secure large-universe multi-authority attribute-based encryption. In Rainer Böhme and Tatsuaki Okamoto, editors, *FC 2015*, volume 8975 of *LNCS*, pages 315–332. Springer, Heidelberg, January 2015.

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Heidelberg, May 2005.

[TCH12] Terence Tao, Ernest Croot, and Harald Helfgott. Deterministic methods to find primes. *Mathematics of Computation*, 81(278):1233–1246, 2012.

[Tsa22] Rotem Tsabary. Candidate witness encryption from lattice techniques. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part I*, volume 13507 of *LNCS*, pages 535–559. Springer, Heidelberg, August 2022.

[VWW22] Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. Witness encryption and null-IO from evasive LWE. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 195–221. Springer, Heidelberg, December 2022.

[Wat12] Brent Waters. Functional encryption for regular languages. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 218–235. Springer, Heidelberg, August 2012.

[Wee17] Hoeteck Wee. Attribute-hiding predicate encryption in bilinear groups, revisited. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 206–233. Springer, Heidelberg, November 2017.

[Wee21] Hoeteck Wee. ABE for DFA from LWE against bounded collusions, revisited. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part II*, volume 13043 of *LNCS*, pages 288–309. Springer, Heidelberg, November 2021.

[Wee22] Hoeteck Wee. Optimal broadcast encryption and CP-ABE from evasive lattice assumptions. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 217–241. Springer, Heidelberg, May / June 2022.

[WWW22] Brent Waters, Hoeteck Wee, and David J. Wu. Multi-authority ABE from lattices without random oracles. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 651–679. Springer, Heidelberg, November 2022.

A Noisy Linear Garbling for Boolean Circuits

In this section, we present an alternative construction of noisy linear garbling. It is closer to the Yao's garbling scheme and supports bounded depth boolean circuits.

Construction 11 (noisy linear garbling for Boolean circuits)**.** Let

$$\begin{aligned}\mathsf{Params} &= \left\{ 1^d \;\middle|\; d \in \mathbb{N} \right\},\\ F_{1^d} &= \left\{ f_C \;\middle|\; C \text{ is a circuit (using only NAND gates) of depth no more than } d \right\},\\ f_C(\mathbf{x}) &= \begin{cases} \bot, & \text{if } \mathbf{x} \notin \{0,1\}^L;\\ \neg C(\mathbf{x}), & \text{if } \mathbf{x} \in \{0,1\}^L; \end{cases} \qquad \text{for } C : \{0,1\}^L \to \{0,1\} \text{ and } \mathbf{x} \in \mathbb{Z}^L. \end{aligned}$$

The scheme works as follows.

- $\mathsf{Setup}(1^d)$ picks and outputs suitable $\mathsf{pp} = (q, 1^N)$.
- $\mathsf{GenF}(\mathsf{pp}, C)$ first parses C into L input gates $1, \ldots, L$ and $(|C| - L)$ NAND gates $L+1, \ldots, |C|$, sorted topologically. Let $\mathsf{gin}[i, \gamma] < i$ be the index of the γ^{th} input to gate i, where $\gamma \in \{1, 2\}$ and $L < i \le |C|$. The algorithm samples the shrunken labels and sets the label functions

  $$\mathbf{W}_{i,0}, \mathbf{W}_{i,1} \xleftarrow{\$} \{0,1\}^{N \times N} \quad \text{for all } i \in [|C|], \qquad \mathbf{W}_{\ell,\times} = \mathbf{W}_{\ell,1} - \mathbf{W}_{\ell,0} \quad \text{for all } \ell \in [L].$$

  For each NAND gate $i > L$, it samples the expanded labels

  $$\widetilde{\mathbf{W}}^{(i,\gamma,x'')}_{\mathsf{gin}[i,\gamma],x'} \xleftarrow{\$} \{0,1\}^{N \times N} \qquad \text{for all } \gamma \in \{1,2\},\ x', x'' \in \{0,1\},$$

  and prepares four garbled table entries

  $$\mathbf{T}_{i,x_1,x_2} = \mathbf{W}_{i,\neg(x_1 \wedge x_2)} + \widetilde{\mathbf{W}}^{(i,1,x_2)}_{\mathsf{gin}[i,1],x_1} + \widetilde{\mathbf{W}}^{(i,2,x_1)}_{\mathsf{gin}[i,2],x_2} \qquad \text{for all } x_1, x_2 \in \{0,1\}.$$

  The subscript of $\widetilde{\mathbf{W}}$ indicates the gate and its purported value. Its superscript indicates why the block is added — because the gate is the γ^{th} input to gate i and the other input to gate i evaluates to a particular value. The table entry $\mathbf{T}_{i,x_1,x_2}$ enables conversion from the expanded labels of gates $\mathsf{gin}[i,1], \mathsf{gin}[i,2]$ to the shrunken label of gate i if the two inputs to gate i are x_1, x_2, respectively. For the output gate, the algorithm samples the expanded labels and sets

  $$\widetilde{\mathbf{W}}_{|C|,0}, \widetilde{\mathbf{W}}_{|C|,1} \xleftarrow{\$} \{0,1\}^N, \qquad \mathbf{w}_{\text{out}} = \widetilde{\mathbf{W}}_{|C|,0}.$$

  It prepares garbled table entries for converting shrunken labels to expanded labels. For each tuple (i', i, γ) with $\mathsf{gin}[i, \gamma] = i'$, the algorithm samples and sets

  $$\mathbf{R}^{(i,\gamma,x'')}_{i',x'} \xleftarrow{\$} \{0,1\}^{N \times N}, \quad \mathbf{T}^{(i,\gamma,x'')}_{i',x'} = \mathbf{W}_{i',x'}\mathbf{R}^{(i,\gamma,x'')}_{i',x'} + \widetilde{\mathbf{W}}^{(i,\gamma,x'')}_{i',x'} \qquad \text{for all } x', x'' \in \{0,1\}.$$

  It samples and sets

  $$\mathbf{R}_{|C|,x} \xleftarrow{\$} \{0,1\}^N, \quad \mathbf{T}_{|C|,x} = \mathbf{W}_{|C|,x}\mathbf{R}_{|C|,x} + \widetilde{\mathbf{W}}_{|C|,x} \qquad \text{for all } x \in \{0,1\}.$$

  The algorithm collects $\mathbf{T}, \mathbf{R}$ and outputs the components in the required format, with $n_C = n$.

- $\mathsf{Eval}(\mathsf{pp}, C, \mathbf{R}, \mathbf{x}, \{\mathbf{w}_\ell^\mathsf{T}\}_{\ell\in[L]}, \mathbf{t}^\mathsf{T})$ parses C as in GenF and evaluates $C(\mathbf{x})$ for the wire values $\{x_i\}_{i\in[|C|]}$ by

$$x_i \leftarrow \begin{cases} \mathbf{x}[\ell], & \text{if gate } i \text{ is an input gate } (i = \ell \in [L]); \\ \neg(x_{\mathsf{gin}[i,1]} \wedge x_{\mathsf{gin}[i,2]}), & \text{if gate } i \text{ is NAND.} \end{cases}$$

  It computes the label of each non-input gate in increasing order by its index i.

  - The algorithm decrypts for the expanded labels using

$$(\widetilde{\mathbf{w}}_{\mathsf{gin}[i,1]}^{(i,1,x_{\mathsf{gin}[i,2]})})^\mathsf{T} \leftarrow (\mathbf{t}_{\mathsf{gin}[i,1],x_{\mathsf{gin}[i,1]}}^{(i,1,x_{\mathsf{gin}[i,2]})})^\mathsf{T} - \mathbf{w}_{\mathsf{gin}[i,1]}^\mathsf{T}\mathbf{R}_{\mathsf{gin}[i,1],x_{\mathsf{gin}[i,1]}}^{(i,1,x_{\mathsf{gin}[i,2]})}.$$

    and similarly for $\widetilde{\mathbf{w}}_{\mathsf{gin}[i,2]}^{(i,2,x_{\mathsf{gin}[i,1]})}$.
  - The algorithm recovers the shrunken label using

$$\mathbf{w}_i \leftarrow \mathbf{t}_{i,x_{\mathsf{gin}[i,1]},x_{\mathsf{gin}[i,2]}} - \widetilde{\mathbf{w}}_{\mathsf{gin}[i,1]}^{(i,1,x_{\mathsf{gin}[i,2]})} - \widetilde{\mathbf{w}}_{\mathsf{gin}[i,2]}^{(i,2,x_{\mathsf{gin}[i,1]})}.$$

  Lastly, the algorithm computes and outputs the secret

$$w_{\text{out}} \leftarrow \mathbf{t}_{|C|,x_{|C|}}^\mathsf{T} - \mathbf{w}_{|C|}^\mathsf{T}\mathbf{R}_{|C|,x_{|C|}}.$$