# ABE for Unbounded Circuits from Circular LWE and Pairings

Shweta Agrawal* Supriyo Banerjee† Anuja Modi‡

Rajeev Raghunath§ Anshu Yadav¶ Shota Yamada‖

**Abstract**

Constructing attribute based encryption (ABE) for unbounded circuits, namely circuits with *both* depth and width being unbounded, and for Turing machines are perhaps the most important open questions in ABE research. Recent years saw some much-awaited progress on these long-standing problems – (i) the work of Hsieh, Lin and Luo [HLL23] (HLL23) provided the first construction of ABE for unbounded circuits from a new circular evasive LWE assumption, (ii) the follow-up work of Agrawal, Kumari and Yamada [AKY24b] and Cini and Wee [CW25] gave the first constructions of ABE for Turing machines.

Unfortunately, all these works relied on different versions of the "private-coin" Evasive Learning With Errors assumption which was invalidated in a sequence of attacks [BÜW25, BDJ+25, DJM+25, AMYY25, HJL25, HHY26]. Finding new constructions from better-understood assumptions[1] is therefore an important open problem.

In this work, we make progress towards this goal. Our contributions are:

1. *KP-ABE for Unbounded Circuits from Circular Small-Secret LWE and Bilinear GGM.* We provide a new candidate for key-policy ABE for unbounded circuits from circular small-secret Learning With Errors (csLWE) and the Bilinear Generic Group Model (BGGM). Since the BGGM has a long history of study, we believe that this provides strong evidence for the security of our scheme.

   In the standard model, we argue security from a novel, falsifiable "doubly circular" assumption which combines csLWE with key dependent message security of a variant of ElGamal encryption in an inseparable manner. We show that in a restricted setting, it is implied by the standard SXDH assumption and csLWE.

2. *SK CP-ABE for Unbounded Circuits from Circular Small-Secret LWE.* In the symmetric key setting, we provide a new candidate for *ciphertext-policy* ABE for unbounded circuits from only the csLWE assumption.

3. *SK KP-ABE for Turing Machines from Circular Small-Secret LWE.* In the symmetric key setting, we provide the first key-policy ABE for Turing machines from the csLWE assumption.

Curiously, each of our constructions is *intrinsically* key-policy or ciphertext-policy and stubbornly resists adaptation to its dual variant. Thus, our work exhibits a surprising (to us) dichotomy between the key and ciphertext policy versions of ABE for unbounded circuits. Finding dual versions of the above constructions, removing pairings altogether or dispensing with the symmetric key setting are fascinating open problems.

*IIT Madras, India, shwetaag@cse.iitm.ac.in

†IIT Madras, India, supriyobanerjee537@gmail.com

‡IIT Madras, India, anujamodi97@gmail.com

§IIT Madras, India, rajeev.mrug08@gmail.com

¶IIT Kanpur, India, anshuy@cse.iitk.ac.in

‖AIST Tokyo, Japan, yamada-shota@aist.go.jp

[1] Without relying on compact functional encryption or obfuscation.

# Contents


**1 Introduction** **3**
- 1.1 Our Results . . . 3
- 1.2 Technical Overview . . . 4
- 1.3 The Curious Case of CP versus KP for Unbounded ABE . . . 16
- 1.4 Open Problems. . . . 16

**2 Preliminaries** **16**
- 2.1 Lattice Preliminaries . . . 17
- 2.2 Bilinear Map Preliminaries . . . 18
- 2.3 Cryptographic Primitives . . . 19

**3 KP-ABE for Unbounded Depth Circuits** **24**
- 3.1 Recall: Unbounded Homomorphic Evaluation Procedures . . . 24
- 3.2 Construction . . . 24
- 3.3 Security Proof . . . 27

**4 Secret-Key CP-ABE for Unbounded-Depth Circuits** **34**
- 4.1 Construction . . . 35
- 4.2 Security Proof . . . 36

**5 Secret-Key KP-ABE for Turing Machines** **37**
- 5.1 Preliminaries and Ingredients . . . 38
- 5.2 Construction . . . 40

**6 Doubly Circular Assumption and its cryptanalysis** **44**
- 6.1 Assumption . . . 45
- 6.2 Reduction for Constant $\{f_i\}_i$ . . . 46
- 6.3 Cryptanalysis . . . 47

# 1 Introduction

Attribute-based encryption (ABE) [SW05, GPSW06] generalizes public key encryption so that decryption rights are no longer granted on a per-user basis, but are instead determined by an access control policy associated with the encrypted data. ABE comes in two variants – *key-policy*, denoted by KP-ABE and *ciphertext-policy*, denoted by CP-ABE, depending on whether the access control policy is embedded in the key or the ciphertext. In more detail, in KP-ABE, the ciphertext encodes a secret message $\mu$ and a public *attribute* $\mathbf{x}$, the secret key encodes a (public) function $f$, and decryption outputs $\mu$ if and only if $f(\mathbf{x}) = 0$. The dual CP-ABE swaps the embedding of $\mathbf{x}$ and $f$ to the secret key and ciphertext respectively.

**Unbounded Circuits & Turing Machines.** ABE has received significant attention in the literature with elegant constructions from diverse assumptions supporting various families of circuits [GPSW06, GVW13a, BGG+14, AY20a, Wee22, HLL23]. A significant advance in this area was achieved by the celebrated work of Gorbunov, Vaikuntanathan and Wee [GVW15] who constructed KP-ABE for all polynomial sized circuits of bounded depth from the standard Learning With Errors (LWE) assumption. The question of generalizing this construction to support unbounded circuits, namely circuits with both width as well as depth being unbounded remained tantalizingly open for several years, until the recent work of Hsieh, Lin and Luo [HLL23] (HLL23) provided the first construction of KP-ABE from a new assumption called the *circular evasive* LWE assumption. The follow-up work of [AKY24a] gave a construction from a non-circular variant of evasive LWE.

Subsequent work [AKY24b, CW25] generalized these ideas to provide the first constructions of ABE for Turing machines, also from (some kind of) evasive LWE assumptions. Supporting uniform models of computation is important, since circuits, though powerful, force the size of the input to be fixed ahead of time and additionally incur the worst case running time on every input, which is clearly wasteful. Put together, these results provided the first feasibility results for long-standing open problems, even if the underlying assumption was relatively new. This marked encouraging progress, but unfortunately the happiness was short-lived – almost concurrently to these constructions, a wave of devastating attacks invalidated the underlying assumptions [BÜW25, BDJ+25, DJM+25, AMYY25, HJL25, HHY26]. Thus, at present, we have *no* constructions of ABE for unbounded circuits or Turing machines whose security reduces to a reasonably well-understood, or at least unbroken assumption[2] (without using compact functional encryption). The present work takes steps towards filling this gap.

## 1.1 Our Results

We state our results below.

1. *KP-ABE for Unbounded Circuits from Circular Small-Secret LWE and Bilinear GGM (Section 3).* We provide a new candidate for key policy ABE for unbounded circuits from circular small-secret Learning With Errors (csLWE) and the Bilinear Generic Group Model (BGGM). Since the BGGM has a long history of study, we believe this provides strong evidence for the security of our scheme. A similar approach has been taken in the past for the (then) long-standing open problems of broadcast encryption [AY20b] and adaptively secure ABE [LLL22] – our approach is philosophically similar (albeit with important differences in details).

   In the standard model, we reduce security to a novel, falsifiable "doubly circular" assumption which combines circular small-secret LWE with key dependent message security of a variant of ElGamal encryption in an inseparable manner. We perform a thorough cryptanalysis of this assumption and show that in a restricted setting, it is implied by the standard SXDH assumption and circular small-secret LWE.

[2] We note that all the aforementioned *constructions* remain unbroken.

2. *SK CP-ABE for Unbounded Circuits from Circular Small-Secret LWE (Section 4).* In the symmetric key setting (where the encryptor also needs a master key), we provide a new candidate for *ciphertext-policy* ABE for unbounded circuits from only the circular Learning With Errors assumption.

3. *SK KP-ABE for Turing Machines from Circular Small-Secret LWE (Section 5).* In the symmetric key setting, we provide the first key-policy ABE for Turing machines from the circular small-secret LWE assumption.

Curiously, each of our constructions is *intrinsically* key or ciphertext policy and stubbornly resists adaptation to its dual variant (see Section 1.3 for a more detailed discussion). Thus, our work exhibits a curious dichotomy between the key and ciphertext policy versions of ABE for unbounded circuits which was not known before, to the best of our knowledge. Finding dual versions of the above constructions, removing pairings altogether or dispensing with the symmetric key setting are fascinating open problems.

## 1.2 Technical Overview

We outline our core ideas below. We use the notation $\underset{\sim}{x}$ to denote the noisy version of $x$, where the exact noise is not important here.

### 1.2.1 Public-Key KP-ABE for Circuits of Unbounded Depth (Section 3).

Our starting point is the construction of the $\mathsf{ABE}$ scheme for unbounded-depth circuits by [HLL23] (HLL23). The ABE of HLL23 proceeds in two stages: (i) they first provide a *single key* secure ABE scheme for unbounded depth circuits from the circular small-secret LWE assumption, (ii) they then lift this to a multi-key scheme by additionally adding terms from the evasive LWE assumption to the circular small-secret LWE assumption – this yields the circular evasive LWE assumption, which was recently broken as discussed above.

However, fortunately, the first component, namely the single key secure ABE scheme relies on the much more standard circular small-secret LWE assumption and remains secure. Therefore, a natural approach is to find a different way to compile the single key scheme to a multi-key one. We outline the single key scheme next.

*Unbounded Homomorphic Evaluation procedures of HLL23.* Let $C$ denote a circuit of *unbounded* depth. HLL23 designed two algorithms, namely $\mathsf{UEvalC}$ and $\mathsf{UEvalCX}$, to support homomorphic evaluation of $C$ on the public key $\mathbf{A}_{\mathsf{att}}$ and the encoding $\mathbf{c}_{\mathsf{att}}$ to yield a well-formed encoding of $C(\mathbf{x})$ under the evaluated public key. At a very high level, they enabled unbounded homomorphic evaluation of an ABE encoding by using a modulus-reduction-inspired noise-removal procedure (developed in the context of leveled fully homomorphic encryption (FHE)) [BV11, BGV12]) and a bootstrapping technique, which together serve to replace large evaluation noise in a given encoding with smaller, more manageable noise.

Let $\mathbf{G}^{-1}(\cdot)$ denote a binary decomposition operation such that $\mathbf{G}\mathbf{G}^{-1}(\mathbf{P}) = \mathbf{P}$. Their algorithms require encodings constructed as follows:

1. Sample a low norm secret key $\mathbf{s}$ for the FHE scheme by Gentry, Sahai and Waters [GSW13] (GSW) and public key $\mathbf{A}_{\mathsf{fhe}}$.

2. Compute a circular GSW encoding $\mathbf{S} = \mathbf{A}_{\mathsf{fhe}}\mathbf{R} - \mathsf{bits}(\mathbf{s}) \otimes \mathbf{G}$ where $\mathbf{R}$ is a matrix sampled at random of appropriate dimensions.

3. Construct LWE encodings (exact structure not important here) with LWE secret $\mathbf{s}$ using the arithmetic ABE scheme by Boneh et al. [BGG+14] (denoted by $\mathsf{Bgg}^+$): one for attribute $\mathbf{x}$ under public key $\mathbf{A}_{\mathsf{att}}$ and one of $\mathbf{S}$ under public key $\mathbf{A}_{\mathsf{circ}}$, as:

$$\mathbf{c}_{\mathsf{att}} = \mathsf{Bgg}^+.\mathsf{Enc}_{\mathbf{A}_{\mathsf{att}},\mathbf{s}}(\mathbf{x}), \quad \mathbf{c}_{\mathsf{circ}} = \mathsf{Bgg}^+.\mathsf{Enc}_{\mathbf{A}_{\mathsf{circ}},\mathbf{s}}(\mathbf{S})$$

Then, $\mathsf{UEvalC}(\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, C) \to \mathbf{A}_C$, $\mathsf{UEvalCX}(\mathbf{A}_{\mathsf{att}}, \mathbf{c}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{circ}}, C, \mathbf{x}, \mathbf{S}) = \mathbf{c}_{C,\mathbf{x}}^\intercal$

such that, with all but negligible probability we have

$$\mathbf{c}_{C,\mathbf{x}}^\intercal = \mathbf{s}^\intercal(\mathbf{A}_C - C(\mathbf{x})\mathbf{G})$$

*Single Key ABE from Circular Small-Secret LWE.* The one-key secure ABE scheme of HLL23 is as follows:

$\mathsf{Setup}(1^L)$. Samples matrices $\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, (\mathbf{B}, \mathbf{B}^{-1})$ and vectors $\mathbf{u}, \mathbf{u}'$ of appropriate dimensions uniformly at random. Outputs $\mathsf{mpk} = (\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{u}, \mathbf{u}', \mathbf{B})$ and $\mathsf{msk} = \mathbf{B}^{-1}$. Here, $\mathbf{B}^{-1}$ is trapdoor of $\mathbf{B}$ that allows to compute short pre-image $\mathbf{B}^{-1}(\mathbf{U})$ for any target matrix $\mathbf{U}$.

$\mathsf{Enc}(\mathsf{mpk}, \mathbf{x}, \mu)$. Samples a GSW FHE scheme with public key $\mathbf{A}_{\mathsf{fhe}}$ and low norm secret key $\mathbf{s}$. Constructs $\mathbf{S}$ as a circular GSW FHE encoding, $\mathbf{c}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{att}}$ as ABE encodings as described above, $\mathbf{c}_{\mathbf{B}} = \mathbf{s}^\intercal\mathbf{B}$, and $c_{\mathsf{msg}} = \mathbf{s}^\intercal\mathbf{u} + \mu\lfloor q/2 \rceil$. Outputs $\mathsf{ct} = (\mathbf{S}, \mathbf{c}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{att}}, \mathbf{c}_{\mathbf{B}}, c_{\mathsf{msg}})$.

$\mathsf{KeyGen}(\mathsf{msk}, C)$. Computes $\mathbf{A}_C \leftarrow \mathsf{UEvalC}(\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, C)$. Uses the trapdoor to sample $\mathbf{k} \leftarrow \mathbf{B}^{-1}(\mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u}') + \mathbf{u})$ and outputs $\mathsf{sk} = \mathbf{k}$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$. Computes $\mathbf{c}_{C,\mathbf{x}}^\intercal = \mathsf{UEvalCX}(\mathbf{A}_{\mathsf{att}}, \mathbf{c}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{circ}}, C, \mathbf{x}, \mathbf{S})$ and $c' = c_{\mathsf{msg}} + \mathbf{c}_{C,\mathbf{x}}^\intercal\mathbf{G}^{-1}(\mathbf{u}') - \mathbf{c}_{\mathbf{B}}^\intercal\mathbf{k}$. It outputs $\mu = 0$ if $c' \in [-q/4, q/4)$, else outputs $\mu = 1$.

To see correctness, note that for $C(\mathbf{x}) = 0$,[3] we have $\mathbf{c}_{C,\mathbf{x}}^\intercal = \mathbf{s}^\intercal(\mathbf{A}_C)$. Since $\mathbf{c}_{\mathbf{B}}^\intercal\mathbf{k} = \mathbf{s}^\intercal(\mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u}') + \mathbf{u})$, we have that $c' = c_{\mathsf{msg}} + \mathbf{c}_{C,\mathbf{x}}^\intercal\mathbf{G}^{-1}(\mathbf{u}') - \mathbf{c}_{\mathbf{B}}^\intercal\mathbf{k} = \mu\lfloor q/2 \rceil$.

Security is based on the circular small-secrets LWE assumption, which informally states that the distribution $(\mathbf{A}_{\mathsf{fhe}}, \mathbf{S}, \mathbf{A}, \mathbf{s}^\intercal\mathbf{A})$ is indistinguishable from random, where $\mathbf{A}_{\mathsf{fhe}}, \mathbf{S}, \mathbf{s}$ are as above and $\mathbf{A}$ is a random matrix of appropriate dimensions.

The proof works so long as the vector $\mathbf{p} = \mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u}') + \mathbf{u}$ is uniform, which is indeed the case here, since $\mathbf{u}$ is uniform. To see this, consider an intermediate hybrid where, instead of first sampling $\mathbf{B}$, computing $\mathbf{p} = \mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u}') + \mathbf{u}$, and then computing $\mathbf{k}$ using a trapdoor, we first sample $\mathbf{B}, \mathbf{k}$ from the appropriate distributions and then set $\mathbf{p} = \mathbf{B} \cdot \mathbf{k}$. This hybrid is indistinguishable from the previous one, as these two sampling methods are statistically indistinguishable by the property of the trapdoor generation algorithm [GPV08]. We then use the circular small-secret LWE assumption (csLWE) to move to a hybrid where $\mathsf{ct}$ is sampled uniformly at random, thereby hiding $\mu$.

*Obtaining Collusion Resistance.* The above scheme fails to satisfy multi-key security. To see this, suppose the adversary queries keys for $C_1$ and $C_2$ such that $C_i(\mathbf{x}) = 1$ and receives $\mathbf{k}_i$ for $i \in [2]$. It computes $c_i' = c_{\mathsf{msg}} + \mathbf{c}_{C_i,\mathbf{x}}^\intercal\mathbf{G}^{-1}(\mathbf{u}') - \mathbf{c}_{\mathbf{B}}^\intercal\mathbf{k}_i = \mu\lfloor q/2 \rceil - \mathbf{s}^\intercal\mathbf{u}' + e_i$, where $\mathbf{s}^\intercal\mathbf{u}'$ is a masking term that should hide $\mu$ for a non-authorized key. Since the vector $\mathbf{u}'$ is shared by all the keys, it can easily cancel out the mask as $c_1' - c_2' = e_1 - e_2$, thereby getting problematic leakage.

[3] Throughout the paper, we follow the convention in the lattice-based cryptography where the decryption is possible when $C(\mathbf{x}) = 0$.

A natural approach to collusion resistance, would be to use different $\mathbf{u}'_i$ for each $i \in [Q]$. In detail, for the two key setting, if we would have $\mathbf{k}_1 \leftarrow \mathbf{B}^{-1}(\mathbf{A}_{C_1}\mathbf{G}^{-1}(\mathbf{u}'_1) + \mathbf{u})$ and $\mathbf{k}_2 \leftarrow \mathbf{B}^{-1}(\mathbf{A}_{C_2}\mathbf{G}^{-1}(\mathbf{u}'_2) + \mathbf{u})$, then the above attack could be avoided. However, unlike the bounded depth setting [BGG+14], we do not have the necessary algebraic structure[4] which would allow us to simulate user secret keys. To overcome this hurdle, HLL23 introduced the circular evasive LWE assumption, where it is essentially *assumed* that giving the above preimages $\mathbf{k}_i$ is safe, in certain situations, which include that of the construction.

In more detail, let $\mathbf{K} = \mathbf{B}^{-1}(\mathbf{P})$ be a concatenation of the preimages $\mathbf{k}_i$ described above. Then, the assumption states that for an efficiently sampleable joint distribution of $\mathbf{A}_{\mathsf{circ}}, \overline{\mathbf{A}}, \mathbf{P}, \mathsf{aux}$,

$$\text{If} \quad (\mathbf{B}, \mathbf{P}, \overline{\mathbf{A}}, \mathbf{A}_{\mathsf{fhe}}, \underbrace{\mathbf{s}^\intercal \mathbf{B}}, \underbrace{\mathbf{s}^\intercal \overline{\mathbf{A}}}, \underbrace{\mathbf{s}^\intercal \mathbf{P}}, \mathbf{S}, \mathbf{c}_{\mathsf{circ}}, \mathsf{aux}) \approx (\mathbf{B}, \mathbf{P}, \overline{\mathbf{A}}, \$, \$, \$, \$, \$, \$, \mathsf{aux})$$

$$\text{Then} \quad (\mathbf{B}, \mathbf{P}, \overline{\mathbf{A}}, \mathbf{A}_{\mathsf{fhe}}, \underbrace{\mathbf{s}^\intercal \mathbf{B}}, \underbrace{\mathbf{s}^\intercal \overline{\mathbf{A}}}, \mathbf{K}, \mathbf{S}, \mathbf{c}_{\mathsf{circ}}, \mathsf{aux}) \approx (\mathbf{B}, \mathbf{P}, \overline{\mathbf{A}}, \$, \$, \$, \mathbf{K}, \$, \$, \mathsf{aux})$$

There are two heuristics embedded in the above assumption:

- It assumes that the preimage $\mathbf{K} = \mathbf{B}^{-1}(\mathbf{P})$ can only be used in the honest way by multiplying this to the ciphertext component $\mathbf{c}_{\mathbf{B}} = \underbrace{\mathbf{s}^\intercal \mathbf{B}}$ to obtain $\underbrace{\mathbf{s}^\intercal \mathbf{P}}$ so long as the "if" condition is true. Note that this is a heuristic – in more detail, $\mathbf{K}$ *cannot* be simulated in the proof and it is simply assumed that giving this to the adversary is safe. This heuristic is based on the observation that so long as the "if" condition holds, there are no nontrivial algebraic relations between the vectors in $\mathbf{K}$ that the adversary can exploit and it would be forced to use them in the expected honest way.

- It assumes that the correlated noise obtained in sample $\mathbf{c}_{\mathbf{B}}\mathbf{K}$ is safe. In more detail, for $\mathbf{s}^\intercal\mathbf{B} + \mathbf{e}_{\mathbf{B}}$, $\mathbf{c}_{\mathbf{B}}\mathbf{K} = \mathbf{s}^\intercal\mathbf{P} + \mathbf{e}_{\mathbf{B}}\mathbf{K}$. However the noise obtained in the resulting sample, i.e. $\mathbf{e}_{\mathbf{B}}\mathbf{K}$, is not i.i.d as in the corresponding sample in the pre-condition – it is correlated. The assumption is that this correlated noise cannot be exploited.

Unfortunately, in a recent work [AMYY25], the second heuristic above was shown to be false. In more detail, [AMYY25] showed that combining the circular encodings $(\mathbf{S}, \mathbf{c}_{\mathsf{circ}})$ with the preimage $\mathbf{k} \leftarrow \mathbf{B}^{-1}(\mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u}') + \mathbf{u})$ gives too much power to the adversary. Moreover, this vulnerability is fundamental and does not appear amenable to natural fix attempts.

*Digging Deeper into the Vulnerability.* Let us take stock of the two ingredients that the above template combines: the circular encoding $(\mathbf{S}, \mathbf{c}_{\mathsf{circ}})$ of the GSW secret under its own public key, and the preimage $\mathbf{k} \leftarrow \mathbf{B}^{-1}(\mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u}') + \mathbf{u})$ released by the key generator.

Of these, circularity appears difficult to dispense with. Its role is to enable the noise-removal step inside $\mathsf{UEvalCX}$: to evaluate a circuit whose depth is not known at setup time, one must periodically refresh the accumulated evaluation noise, and the only technique we know for refreshing noise without an a-priori depth bound is bootstrapping, which by its very nature needs an encoding of the secret key under its own public key. We observe that this is not special to [HLL23] – the same phenomenon is visible in fully homomorphic encryption, where the parameters of leveled schemes [BV11, BGV12] grow with the depth of the evaluated circuit, and the only known route to unbounded ("pure") homomorphism is via circular security. The situation in ABE is exactly parallel in that the parameters of [BGG+14] grow with the depth bound, and circularity is what enables independence from it. We therefore believe that circularity is inherent (at least by known techniques).

The preimage, by contrast, could be negotiable. Observe that neither ingredient by itself is known to be problematic: circular small-secret LWE is a well-studied assumption which remains unbroken, and the controlled release of preimages by itself in the *public-coin* evasive LWE assumption has so far resisted attack in natural parameter regimes. It is, unfortunately, the precise *combination* of the two that the recent

[4] For the informed reader, what we would need is trapdoor homomorphism.

attacks exploit – [AMYY25] show that once the circular encodings $(\mathbf{S}, \mathbf{c}_{\mathsf{circ}})$ are available alongside $\mathbf{k}$, the correlated noise in $\mathbf{c}_{\mathbf{B}}^{\intercal}\mathbf{k}$ can be turned into a distinguisher, and this vulnerability appears fundamental rather than incidental. Since circularity seems to be forced upon us, the natural question is whether we can keep it and obtain collusion resistance by some route that does not release preimages at all.

To find such a route it is instructive to recall where preimages came from in the first place. Evasive LWE, which is essentially LWE together with a controlled release of preimages, was introduced [Wee22] as a lattice-only substitute for the combination of LWE and pairings used by Agrawal and Yamada [AY20b] in the context of broadcast encryption. In that context, collusion resistance was obtained by lifting the relevant LWE encodings into the exponent of a bilinear group and re-randomizing each key with a fresh scalar $\gamma_i$, so that components belonging to different keys cannot be combined. Evasive LWE replaced this pairing-based joining by a lattice analog. We observe that, using lattices, it seems technically difficult to dispense with preimages – most if not all advanced encryption schemes, whether identity based [GPV08] or attribute based [BGG+14] or broadcast encryption [Wee22] schemes from lattices use preimages for the secret key. If the lattice route is blocked by its interaction with circularity, we ask:

*Can we go back in time and revive the pairing-based route which has been successful previously?*

Observe that prior work that took this pathway [AY20b, LLL22, ARYY23] proved security from LWE and Bilinear GGM, and this was considered a strong justification for security since the BGGM, similar to the ROM has a long history of study and admits only contrived counter-examples.

*Too Good to be True?* Unfortunately, there is an immediate barrier. In [AY20b], decryption enters the exponent at its very first step: the ciphertext and key components that must be combined are group elements, and this is performed in the very first step by applying the pairing, thereby "using it up". Everything that follows is now restricted to linear operations in the exponent of the target group. This was affordable in [AY20b] because the homomorphic computation performed during decryption was linear by a useful property of [BGG+14] evaluation. In our setting, sadly, it is not, and moreover, this is fundamental – $\mathsf{UEvalC}$ and $\mathsf{UEvalCX}$ perform bit decomposition and, crucially, the modulus-reduction based noise-removal step, and these are far outside what bilinear groups support.

For CP-ABE we do not know how to get around this. There, the policy circuit is held by the encryptor while the attribute is held by the key generator, so the encodings needed to run $\mathsf{UEvalCX}$ are not all produced by the same party; the decryptor's only way of bringing the two sides together is the pairing itself, which pushes the entire non-linear evaluation into the exponent. For KP-ABE, however, we are able to salvage this approach by restructuring the computation so that the non-linear part is performed downstairs and only linear computation is performed in the exponent. We now describe this in more detail.

*Going back in time: Pairings to the Rescue?* Let us revisit the two heuristics discussed above:

- The first heuristic says that the preimages $\mathbf{k}_i$ can only be used in the honest way to combine with ciphertext – we *enforce* this honest behavior by relying on inner product functional encryption (IPFE), described next.
- The second heuristic says that the correlated noise cannot be exploited – we enforce this by forcing the noise to live in the exponent of a group in a way that admits a security proof in the BGGM while permitting the desired functionality.

*Inner Product Functional Encryption.* Inner Product Functional Encryption (IPFE) is a primitive that allows inner product computation over a *private* vector, in contrast to ABE where computation can only be performed over a public vector. In more detail, in an IPFE scheme, the ciphertext encrypts a secret vector $\mathbf{x}$, the secret key corresponds to a vector $\mathbf{y}$ and decryption recovers the inner product $\mathbf{x}^{\intercal}\mathbf{y}$.

To enforce honest combinations, we use IPFE as follows. According to the template above, we would like the decryptor to (approximately) compute $\mathbf{s}^\intercal(\mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u}') + \mathbf{u})$. Above, this was enabled by providing $\mathbf{k}$. Instead, we can provide an IPFE key for $\mathbf{y} = \mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u}') + \mathbf{u}$. On the encryptor's side, instead of providing $\mathbf{c_B}$, we provide IPFE ciphertext for $\mathbf{s}$. By correctness of IPFE, decryption computes $\mathbf{s}^\intercal(\mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u}') + \mathbf{u})$ as desired.

IPFE has been constructed from a wide variety of assumptions, including DDH, DCR and LWE [ABDP15, ALS16]. It would make sense to use IPFE from LWE, however there is an issue: we want to use the IPFE inside an ABE scheme, which supports an unbounded number of key queries. On the other hand, IPFE as a primitive is inherently bounded key, in the sense that given $|\mathbf{x}|$ keys for linearly independent vectors, the decryptor can recover $\mathbf{x}$ by functionality and no further security is required. The LWE based IPFE is designed so that each independent key request leaks some part of the master secret, so that after $|\mathbf{x}|$ keys, the entire master secret can be revealed. While this is permissible for the IPFE primitive, it is problematic if this IPFE is used within an ABE scheme.

*Pairing based IPFE.* To get around the above issue, we use pairing based IPFE [BJK15]. While the bounded key nature of IPFE is also true for pairing based IPFE, the secret keys in this setting are "stronger" than in lattice based IPFE in the sense that decryption does not leak the master secret key. We place the IPFE key and ciphertext vectors in the exponent and obtain the following simplified scheme (we skip describing $\mathsf{Setup}$ below for brevity):

$\mathsf{Enc}(\mathsf{mpk}, \mathbf{x}, \mu)$. Samples a GSW FHE scheme with public key $\mathbf{A}_{\mathsf{fhe}}$ and low norm secret key $\mathbf{s}$. Constructs $\mathbf{S}$ as a circular GSW FHE encoding, $\mathbf{c}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{att}}$ as ABE encodings as described previously, and IPFE ciphertext for $[\mathbf{s}, \mu\lfloor p/2\rceil]_1$. Outputs $\mathsf{ct} = (\mathbf{S}, \mathbf{c}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{att}}, \mathsf{IPFE.ct})$.

$\mathsf{KeyGen}(\mathsf{msk}, C)$. Computes $\mathbf{A}_C \leftarrow \mathsf{UEvalC}(\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, C)$. Compute IPFE key for $\mathbf{y} = [\mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u}), 1]_2$ and output this.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$. Runs IPFE decryption to obtain $[\mathbf{s}^\intercal(\mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u})) + \mu\lfloor p/2\rceil]_T$. Computes

$$\mathbf{c}^\intercal_{C,\mathbf{x}} = \mathsf{UEvalCX}(\mathbf{A}_{\mathsf{att}}, \mathbf{c}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{circ}}, C, \mathbf{x}, \mathbf{S}) = \mathbf{s}^\intercal(\mathbf{A}_C - C(\mathbf{x})\mathbf{G}) = \mathbf{s}^\intercal\mathbf{A}_C \text{ when } C(\mathbf{x}) = 0$$

Compute $\mathbf{c}^\intercal_{C,\mathbf{x}}\mathbf{G}^{-1}(\mathbf{u})$ and subtract it from IPFE decryption value to get $[\mu\lfloor p/2\rceil]_T$. If there exists a polynomially bounded $\eta$ such that $[1]^\eta_T = [\mu\lfloor p/2\rceil]_T$, output 0, else output 1.

In this scheme, even if leakage about the master key is revealed information theoretically, it remains computationally hidden due to the hardness of discrete log.

*Randomizing in the Exponent.* Unfortunately, the above scheme is still not collusion resistant. To see this, let the adversary query the secret key for $C_1$ and $C_2$, getting $\mathsf{sk}_1$ and $\mathsf{sk}_2$ respectively. The adversary can now compute the decryption equation, getting $[c_i]_T = [\mathbf{s}^\intercal(\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})) + \mu\lfloor p/2\rceil - \mathbf{s}^\intercal\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u}) + \mathbf{s}^\intercal\mathbf{u}]_T = [\mu\lfloor p/2\rceil + \mathbf{s}^\intercal\mathbf{u}]_T = [\mu\lfloor p/2\rceil + \mathbf{s}^\intercal\mathbf{u} + e_i]_T$ for $i \in \{1, 2\}$. It then computes $[c_1 - c_2]_T = [e_1 - e_2]_T$, thereby getting leakage on error, for unauthorized keys.

To prevent the combination of different keys, we borrow a trick from [AY20b] and re-randomize each $\mathbf{y}_i$ in the exponent using a fresh and random $\gamma_i$ for each key. In more detail, we compute the secret key for $[\mathbf{y}_i]_2 = [\gamma_i(\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u}), 1)]_2$ for each $i \in [Q]$. By doing so, the aforementioned attack disappears, because the key specific randomness disallows the cancellation from before.

*Are we done?* The above high level strategy looks promising but must deal with some technical hurdles in execution:

1. When we use the security of IPFE in the ABE security proof, we are required to embed the IPFE decryption value into the vector $\mathbf{y}$ encoded by the IPFE key. To embed this, we provide extra "space" in the vector associated with the key i.e., $\mathbf{y} = [\gamma_i(\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u}), 1, 0)]_2$ and match it by setting $(\mathbf{x} = [\mathbf{s}, \mu \lfloor p/2 \rceil, 0]_1)$. To "program" the decryption result into the last component of $\mathbf{y}$, a natural idea is to rely on a function hiding IPFE but this is only known in the symmetric key setting and is therefore unsuitable for our purposes.

   To overcome this issue, we use *slotted* IPFE, similar to [AWY20] which is a hybrid between public-key IPFE and secret-key function hiding IPFE. Here, a vector $\tilde{\mathbf{u}} \in \mathbb{Z}_q^n$ has two parts- public and private $\tilde{\mathbf{u}} = (\tilde{\mathbf{u}}_{\mathsf{pub}}, \tilde{\mathbf{u}}_{\mathsf{priv}})$. Given the master secret key, one can compute the encryption of $\tilde{\mathbf{u}}$, but given the public key, one can compute the encryption of the public part $\tilde{\mathbf{u}}_{\mathsf{pub}}$ – thus, the real-world algorithm can be public key while the secret "slot" can be used in the proof. Slotted IPFE only guarantees function hiding with respect to the private slot, but this is sufficient for the programming we need to perform.

2. For correctness, we require the decryption error to be polynomially bounded. However, error $e_{C_i,\mathbf{x}}$ in $[\mu \lfloor p/2 \rceil]_T = [\mu \lfloor p/2 \rceil + e_{C_i,\mathbf{x}}]_T$ would not be polynomially bounded in general.

   To solve this problem, we use the modulus reduction technique, i.e., we choose $M \in \mathbb{Z}^+$ such that $\|e_{C_i,\mathbf{x}}\| \ll M$. Now the decryptor will compute $\left\lfloor \frac{\mathbf{c}_{C_i,\mathbf{x}}}{M} \right\rceil$, which ensures that the accumulated error is polynomially bounded and the key generator will generate $\mathsf{sk}$ for vector $\left[\left(\gamma_i \left(\left\lfloor \frac{\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rceil, 1\right), 0\right)\right]_2$ to ensure correctness.

3. In the idea outlined above, we computed $\mathbf{A}_{C_i}$ and $\mathbf{c}_{C_i,\mathbf{x}}$ in the exponent, which is not possible. Specifically, the algorithms $\mathsf{UEvalC}$ and $\mathsf{UEvalCX}$ cannot be implemented solely with quadratic operations, since they involve modulus reduction. To get around this issue, we simply observe that all these algorithms' inputs are available in plain and the required non-quadratic terms can be computed "downstairs" and then raised to the exponent to proceed with the computation.

We note that the last two issues also arose in the work of [LLL22], in a different context.

*Final Construction.* We now provide our final sketch of ABE below.

$\mathsf{Setup}(1^L)$. It samples $\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{u}$ uniformly at random and generates the slotted IPFE master public key $\mathsf{impk}$ and master secret key $\mathsf{imsk}$ using the slotted $\mathsf{IPFE}$ setup algorithm. Fix $M$ such that $q/M = p$. It outputs $\mathsf{mpk} = (\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{u}, \mathsf{impk}, M)$ and $\mathsf{msk} = \mathsf{imsk}$.

$\mathsf{Enc}(\mathsf{mpk}, \mathbf{x}, \mu)$. Samples a GSW FHE scheme with public key $\mathbf{A}_{\mathsf{fhe}}$ and low norm secret key $\mathbf{s}$. Constructs $\mathbf{S}$ as a circular GSW FHE encoding, $\mathbf{c}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{att}}$ as ABE encodings as described previously, and IPFE ciphertext for $[\mathbf{s}, \mu \lfloor p/2 \rceil]_1$. Outputs $\mathsf{ct} = (\mathbf{S}, \mathbf{c}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{att}}, \mathsf{IPFE.ct})$.

$\mathsf{KeyGen}(\mathsf{msk}, C)$. It computes $\mathbf{A}_C \leftarrow \mathsf{UEvalC}(\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, C)$, samples $\gamma \leftarrow \mathbb{Z}_p$. It then runs the slotted $\mathsf{IPFE}$ key generation algorithm on input the vector $\left[\left(\gamma \left(\left\lfloor \frac{\mathbf{A}_C\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rceil, 1\right), 0\right)\right]_2$ to get secret key $\mathsf{IPFE.sk}$ and outputs $\mathsf{sk} = (\mathsf{IPFE.sk}, [\gamma]_2)$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$. It computes $\mathbf{c}_{C,\mathbf{x}}^{\mathsf{T}} = \mathsf{UEvalCX}(\mathbf{A}_{\mathsf{att}}, \mathbf{c}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{circ}}, C, \mathbf{x}, \mathbf{S})$, $[\gamma]_T = e([1]_1, [\gamma]_2)$, $[z]_T = \mathsf{iDec}(\mathsf{IPFE.sk}, \mathsf{IPFE.ct}) \cdot [\gamma]_T^{-\tilde{z}}$ where $\tilde{z} = \left\lfloor \frac{\mathbf{c}_{C,\mathbf{x}}^{\mathsf{T}}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rceil$ and $\mathsf{iDec}$ is the slotted $\mathsf{IPFE}$ decryption. If $[\gamma]_T^{\eta} = [z]_T$ for $\eta \in [-\mathrm{poly}(\lambda), \mathrm{poly}(\lambda)]$, set $\mu = 0$. Else set $\mu = 1$.

*Correctness.* First, we note that from the correctness of slotted IPFE, we have $\left[\gamma \mathbf{s}^\intercal \left\lfloor \frac{\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor + \gamma\mu \lfloor p/2 \rceil\right]_T = \mathsf{iDec}(\mathsf{IPFE.sk}, \mathsf{IPFE.ct})$. Next, we have

$$\tilde{z} = \left\lfloor \frac{\mathbf{c}_{C,\mathbf{x}}^\intercal \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor = \left\lfloor \frac{\mathbf{s}^\intercal \mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u}) - C(\mathbf{x})\mathbf{s}^\intercal \mathbf{u}}{M} \right\rfloor = \mathbf{s}^\intercal \left\lfloor \frac{\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor$$

where the third equality follows from $C(\mathbf{x}) = 0$ and $\lfloor \mathbf{s}^\intercal \mathbf{a} \rfloor = \mathbf{s}^\intercal \lfloor \mathbf{a} \rfloor$ for short $\mathbf{s}$ and (possibly non-integer) vector $\mathbf{a}$.Therefore, we have

$$z = \gamma \mathbf{s}^\intercal \left\lfloor \frac{\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor + \gamma\mu \lfloor p/2 \rceil - \gamma \mathbf{s}^\intercal \left\lfloor \frac{\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor = \gamma\mu \lfloor p/2 \rceil$$

By carefully setting the parameters, we will have polynomial-bounded accumulated error, thereby ensuring correctness and ensuring decryption runs in polynomial time. Note that for an unauthorized key i.e., when $C(\mathbf{x}) = 1$, we will have $z = \gamma\mu \lfloor p/2 \rceil + \gamma \lfloor \mathbf{s}^\intercal \mathbf{u}/M \rfloor$. Please see Section 3.2 for more details.

*Security.* We now provide a high-level security proof sketch of the above ABE in Shoup's/RR model [Zha22] in the bilinear setting. In Shoup's/RR Bilinear Generic Group Model (BGGM), the group elements are randomly encoded as bit strings (called labels), with the underlying group isomorphic to the additive group $\mathbb{Z}_p$, and group operations (including pairings) are performed via oracle queries. While proving the security, we will be needing to switch to the symbolic group model, which essentially gives the same interface as the corresponding BGGM, except that now the labels given to the adversary correspond to a ring polynomial $\mathbb{Z}_p[X_1, \ldots, X_n]$ for some $n \in \mathbb{N}$ and all the element gates compute the same operation over the ring.

The high-level proof sketch is as follows.

**Hybrid 0.** This is the real world where we encrypt the message $b \leftarrow \{0,1\}$. The adversary gets $\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{u}, M, \mathbf{S}, \mathbf{c}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{att}}$ and the labels $\mathsf{impk}, \mathsf{IPFE.ct}, \mathsf{IPFE.sk}_i, \gamma_i$ for all $i \in [Q]$[5].

**Hybrid 1.** In this hybrid, for all $i \in [Q]$, we change the last component of the key vector from 0 to IPFE decryption value, i.e., to

$$\gamma_i \mathbf{s}^\intercal \left\lfloor \frac{\mathbf{A}_{C_i} \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor + \gamma_i b \lfloor p/2 \rceil \tag{1}$$

We change the ciphertext vector accordingly - the public slot is changed from $(\mathbf{s}, b \lfloor q/2 \rceil)$ to $\mathbf{0}$ and the private slot is changed from 0 to 1, so that the inner product remains unchanged. The indistinguishability from the previous hybrid follows from the function-hiding property of the slotted IPFE scheme [6].

**Hybrid 2.** In this hybrid, for all $i \in [Q]$, we change the last component of the key vector as in Equation (1) to Equation (2)

$$\Delta_i := \gamma_i \left( \left\lfloor \frac{\mathbf{c}_{C_i,\mathbf{x}}^\intercal \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor - \mathsf{err}_i + \left\lfloor \frac{\mathbf{s}^\intercal \mathbf{u} + e}{M} \right\rfloor \right) + \gamma_i b \lfloor p/2 \rceil \tag{2}$$

for appropriately chosen $\mathsf{err}_i$ (whose exact structure is not important for high-level sketch) that depends on $\mathbf{s}, C_i, \mathbf{u}, \mathbf{c}_{C_i,\mathbf{x}}$. We note that Hybrid 1 and Hybrid 2 are indistinguishable, since

[5] By labels corresponding to $\mathsf{impk}, \mathsf{IPFE.ct}, \mathsf{IPFE.sk}_i$ we mean, we output the labels corresponding to group elements present in it and output the non-group elements in plain.

[6] We use slotted IPFE in a black-box way in order to keep the construction and security proof clean.

$\mathbf{s}^\intercal \left\lfloor \frac{\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rceil$ equal to the red term in Equation (2) by algebraic restructuring, and by the choice of LWE noise $e$ such that $\|e\|/M \in \mathsf{negl}(\lambda)$.

Note that the indistinguishability argument in Hybrids 1 and 2 holds regardless of whether we provide the actual group elements or just their labels.

**Hybrid 3.** In this hybrid, for all $i \in [Q]$, we change the last component of the key vector to $\Delta_i := \gamma_i \left\lfloor \frac{\mathbf{c}^\intercal_{C_i,\mathbf{x}}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rceil + t_i + \gamma_i b \lfloor p/2 \rceil$ where $t_i \leftarrow \mathbb{Z}_p$. In Lemma 1.1, we show that the labels corresponding to $\gamma_i$ and $t_i = \gamma_i(-\mathsf{err}_i + \lfloor \frac{\mathbf{s}^\intercal\mathbf{u}+e}{M} \rceil)$ in Hybrid 2 are indistinguishable from the labels corresponding to $\gamma_i, t_i$ in Hybrid 3, even in the presence of the circular LWE terms. Since the lemma establishes the indistinguishability of the inputs to the KeyGen algorithm in the presence of these additional terms, the corresponding outputs of KeyGen are indistinguishable as well by a standard reduction.

**Hybrid 4.** In this hybrid, for all $i \in [Q]$, we sample $\Delta_i \leftarrow \mathbb{Z}_p$. We have Hybrid 3 equivalent to Hybrid 4, since anything added to a random value remains random. At this point, $b$ is information-theoretically hidden. Hence, the proof.

**Lemma 1.1 (Informal).** Assuming the circular small-secret LWE assumption, labels corresponding to $\gamma_i, t_i$ in Hybrid 2 are indistinguishable from the labels corresponding to $\gamma_i, t_i$ in Hybrid 3, even in the presence of the circular small-secret LWE terms in BGGM.

*Proof.* We prove this lemma at a high level using hybrids. Let $\mathcal{A}$ be the distinguisher. In particular, we start with the structure in Hybrid 3 and use a sequence of hybrids to prove that it is indistinguishable from the structure in Hybrid 2. We note that the term involving $\mathbf{c}^\intercal_{C_i,\mathbf{x}}$ can be kept aside for the indistinguishability argument, since it can be computed from publicly known terms and does not contribute to the security analysis.

**Hybrid** I**.** This hybrid is the same as hybrid 3.

**Hybrid** II**.** In this hybrid, we switch to the symbolic group model and replace $\gamma_i$ and $t_i$ with formal variables $\Gamma_i$ and $T_i$, respectively, for all $i$. Hence, labels given to the adversary will now correspond to a polynomial that can be represented as

$$g_{\mathbf{x}}(\Gamma_1, \ldots, \Gamma_Q, T_1, \ldots, T_Q) := x_0 + \sum_{i\in[Q]} x_i T_i + \sum_{i\in[Q]} \tilde{x}_i \Gamma_i$$

where $g_{\mathbf{x}} \in \mathbb{Z}_p[\Gamma_1, \ldots, \Gamma_Q, T_1, \ldots, T_Q]$ and challenger can extract $\mathbf{x} = (x_0, x_1, \ldots, \tilde{x}_1, \ldots, \tilde{x}_Q)$ by keeping track of group operations performed by $\mathcal{A}$. The only way $\mathcal{A}$ can notice this change is when it can obtain the same labels for two polynomials $g_{\mathbf{x}}$ and $g_{\mathbf{x}'}$ such that $\mathbf{x} \neq \mathbf{x}'$. Since $\gamma_i, t_i$ are sampled uniformly at random, by the Schwartz-Zippel lemma and the union bound, the probability of this occurring is at most $\frac{(\text{total number of labels given to } \mathcal{A})^2}{p}$. This establishes the desired indistinguishability.

**Hybrid** III**.** In this hybrid, we change the polynomial representations as

$$\hat{g}_{\mathbf{x}}(\Gamma_1, \ldots, \Gamma_Q) := x_0 + \sum_{i\in[Q]} x_i \left( -\mathsf{err}_i + \left\lfloor \frac{\mathbf{s}^\intercal\mathbf{u} + e}{M} \right\rceil \right) \Gamma_i + \sum_{i\in[Q]} \tilde{x}_i \Gamma_i$$

We now prove that Hybrid II $\approx$ Hybrid III, assuming circular small-secret LWE. At first, note that the only way $\mathcal{A}$ can distinguish between Hybrid II and Hybrid III is if it finds a $\mathbf{z} =$

$(z_0, z_1, \ldots, z_Q, \tilde{z}_1, \ldots, \tilde{z}_Q) \in \mathbb{Z}^{2Q+1}$, such that $\mathbf{z} \neq \mathbf{0}$, $g_{\mathbf{z}} \neq 0$ but $\hat{g}_{\mathbf{z}} = 0$. This implies that there exists $i \in [Q]$ such that

$$(z_i, \tilde{z}_i) \neq (0,0) \quad \wedge \quad z_i \left( -\mathsf{err}_i + \left\lfloor \frac{\mathbf{s}^\intercal \mathbf{u} + e}{M} \right\rceil \right) + \tilde{z}_i = 0$$

Now we prove that finding such a vector $\mathbf{z}$ satisfying the above condition (called "bad" vector) is hard under csLWE. We prove this in two steps:

- In the first step, we prove that there exists a PPT adversary $\mathcal{B}$ that outputs $z_i, \tilde{z}_i$ satisfying the above condition with probability at least the distinguishing advantage of $\mathcal{A}$, up to a polynomial guessing loss. Here, $\mathcal{B}$ runs $\mathcal{A}$ and outputs $z_i, \tilde{z}_i$, by guessing which queries and which coordinate $i$ carry a bad vector. This guessing costs a loss of a polynomial factor. Note that $\mathcal{B}$ can simulate the symbolic group model for $\mathcal{A}$, and this simulation does not require the knowledge of $\mathbf{s}^\intercal \mathbf{u} + e$.
- In the second step, we prove that the probability of such a $\mathcal{B}$ existing is negligible via csLWE. At a high level, this is because, using csLWE, we can replace LWE instance $\mathbf{s}^\intercal \mathbf{u} + e$ with a random element $v \leftarrow \mathbb{Z}_q$, which will then be a random element over $\mathbb{Z}_p$ when divided by $M$. Hence, the probability with which the above condition is satisfied is at most $(2\delta + 1)/p$ where $\|\mathsf{err}_i\| \in [-\delta, \delta]$ as $v$ is sampled uniformly and independently from $z_i, \tilde{z}_i$. In the construction, $\mathsf{err}_i$ can be bounded by some polynomial $\mathrm{poly}(\lambda)$. Thus, by setting $\delta = \mathrm{poly}(\lambda)$ and setting $p$ appropriately, this probability becomes negligible. After this, we again use csLWE to replace the random $v$ with $\mathbf{s}^\intercal \mathbf{u} + e$.

**Hybrid** IV**.** In this hybrid, we switch back to the BGGM model. This argument is the same as the one in Hybrid II. □

For more details, please refer to Section 3.3.

**Doubly Circular Assumption.** The main complexity of the ABE security proof lies in proving Lemma 1.1, the only place where we need BGGM. We observe that this argument can be cleanly abstracted into a new assumption, the *Doubly Circular* assumption, which combines circular, small-secret LWE with key dependent message (KDM) security of a variant of ElGamal encryption in an inseparable manner. Plugging this into the proof, we obtain the security of our ABE scheme in the standard model from our Doubly Circular assumption. Please see Section 3.3 for a detailed description.

### 1.2.2 Secret-Key CP-ABE for Unbounded Circuits (Section 4).

As discussed above, the previous construction idea fails for the ciphertext-policy setting. In order to make progress, we turn to the weaker symmetric key setting, previously considered by Agrawal and Yamada [AY20a] in the context of CP-ABE for *bounded depth* circuits.

*Starting Point: Secret-Key CP-ABE for Bounded Depth Circuits.* We begin with a recap of the approach of Agrawal and Yamada [AY20a]. Let us first recall the context: KP-ABE for circuits had already been constructed from LWE [GVW13b, BGG+14] and ciphertext-policy was open. The folklore approach via universal circuits provided an immediate solution: to encrypt under a policy $C$, place the description $\tilde{C}$ as the attribute of a KP-ABE ciphertext, and let the CP-ABE key for an attribute $\mathbf{x}$ be a KP-ABE secret key for the universal circuit $\mathcal{U}_{s_{\mathsf{max}}}[\mathbf{x}](\cdot) := \mathcal{U}_{s_{\mathsf{max}}}(\cdot, \mathbf{x})$. Then, decryption evaluates $\mathcal{U}_{s_{\mathsf{max}}}[\mathbf{x}](\tilde{C}) = C(\mathbf{x})$ as desired. The limitation of this template is that it only supports circuits of a-priori bounded size $s_{\mathsf{max}}$ (say), corresponding to the input size that circuit $U_{s_{\mathsf{max}}}$ is required to support (thus, the parameterized notation).

To handle circuits of unbounded size (but bounded depth), the insight of [AY20a] is to make use of the time "budget" available to each algorithm by re-distributing computation between them appropriately. In particular, in the CP-ABE setting, the (unbounded) circuit size is known to the encryptor and not to the key generator, whose input is of fixed length. So, only the encryptor has the "budget" to run in time proportional to the circuit size $s_{\mathsf{max}}$. To leverage this, [AY20a] distribute the computation of $\mathsf{ABE.KeyGen}(\mathsf{ABE.msk}, \mathcal{U}_{s_{\mathsf{max}}}[\mathbf{x}])$[7] between the two algorithms using a functional encryption (FE) scheme (which must be chosen carefully) as follows. The CP-ABE key for $\mathbf{x}$ is an FE encryption of $\mathbf{x}$ (of fixed length $L$) along with $\mathsf{ABE.msk}$, and the CP-ABE ciphertext carries an FE secret key for the circuit $\mathcal{H}_{s_{\mathsf{max}},L}$, which, on input $(\mathbf{x}, \mathsf{ABE.msk})$ outputs $\mathsf{ABE.KeyGen}(\mathsf{ABE.msk}, \mathcal{U}_{s_{\mathsf{max}}}[\mathbf{x}])$. The CP-ABE ciphertext additionally carries the KP-ABE ciphertext for the attribute $\tilde{C}$ as before. FE decryption provides the decryptor with the KP-ABE key which is then used to decrypt the KP-ABE ciphertext.

The above template seems deceptively simple but, in fact, there are many elements that are necessary to make it work. In particular, instantiating the template imposes properties from the ingredient schemes that are highly specific and it is a-priori unclear that these would be satisfied. For example, consider the limitation placed by the FE – the only one available from LWE [GKP+13] has a ciphertext size that depends on the output length of the circuit. Since we want to use this FE scheme to output a KP-ABE secret key, this imposes the condition that the ingredient KP-ABE should have a secret-key size which is compact, namely which depends only on the supported circuit depth, not on circuit size. Fortunately, the KP-ABE by [BGG+14] achieves exactly this, and for a fixed depth bound, the output length of $\mathcal{H}_{s_{\mathsf{max}},L}$ can be bounded independently of $s_{\mathsf{max}}$.

Another property being used crucially is the succinctness of the FE by [GKP+13]: since FE encryption time is independent of the size of the supported circuit, keeping the FE input and output lengths bounded allows CP-ABE key generation to avoid a cost that grows with $s_{\mathsf{max}}$. Instead, the computation that depends on $s_{\mathsf{max}}$ is deferred to the encryptor, who knows $|C|$ and can choose an appropriate size bound. However, a hurdle remains: [GKP+13] supports only single-key security, which allows only one circuit $\mathcal{H}_{s_{\mathsf{max}},L}$ per FE instance. Hence, the same instance cannot accommodate different choices of $s_{\mathsf{max}}$ across encryptions.

To support different size bounds, [AY20a] (in turn building upon [AMY19]), uses $\lambda$ separate FE instances for progressively larger circuit sizes via PRF. Namely, slot $j \in [\lambda]$ serves policies with $2^{j-1} < |C| \leq 2^j$. The key generator produces, for *every* slot $j$, one FE ciphertext of message $(\mathbf{x}, r_j)$ under the slot-$j$ FE instance, where (possibly large) ABE master secret key $\mathsf{ABE.msk}_j$ is replaced with a short PRF seed $r_j$ from which the former is generated. The encryptor computes a slot $j$ from $|C|$ and works entirely within slot $j$: it re-derives the same FE instance, issues the (single) FE key for $\mathcal{H}_{2^j,L}$ and encrypts $\tilde{C}$ under a slot-$j$ KP-ABE instance derived from the same randomness $r_j$. Crucially, $2^j = O(|C|)$, so generating the FE key for $\mathcal{H}_{2^j,L}$ takes time polynomial in $\lambda$ and $|C|$. Thus, this construction supports unbounded circuit size while retaining a depth bound fixed at setup. It remains in the secret-key setting because the encryptor needs the FE master secret key to issue the key for $\mathcal{H}_{2^j,L}$.

*Handling Unbounded Depth.* To extend the construction of [AY20a] to unbounded depth, our first idea is to retain the same size slots but remove the common depth bound. In particular, we now want slot $j$ to support all policies with $2^{j-1} < |C| \leq 2^j$, regardless of their depth. Due to this, the depth of $\mathcal{H}_{2^j,L}$ grows with $j$. Since CP-ABE key generation must prepare FE ciphertexts for every slot, the FE setup and encryption algorithms must remain efficient even for slots supporting very deep circuits. The succinct FE instantiation based on [GKP+13] does not meet this requirement.

A promising replacement of [GKP+13] is the recent single-key succinct FE of [HLL23] (HLL23): unlike [GKP+13], it supports circuits of *unbounded depth* under the circular small-secret LWE assumption, with setup and encryption time depending only on the security parameter and the input and output lengths. Using this scheme takes us further but is still not enough. The next challenge is that the FE output length

[7] where, $\mathsf{ABE}$ is the underlying KP-ABE scheme with master secret key $\mathsf{msk}$

must still be fixed at setup. In detail, the output of $\mathcal{H}_{2^j,L}$ is a KP-ABE secret key for $\mathcal{U}_{2^j,L}[\mathbf{x}]$, so this key must have bounded size independently of $j$. In particular, its size must be independent of both the supported depth and the attribute length.

This places a stringent requirement on the underlying KP-ABE – it rules out [BGG+14], whose secret-key size grows with the depth bound. Another alternative is the single-key KP-ABE for unbounded circuits by [HLL23], but this gets ruled out due to its insufficient security. Since multiple CP-ABE decryptions need multiple KP-ABE keys under the same instance, the underlying KP-ABE must be collusion resistant, which [HLL23] is not.

*The Beautiful Serendipity.* Amazingly enough, the very recent KP-ABE for circuits by Cini and Wee [CW23] provides us exactly the piece of the puzzle we are missing! Specifically, we can obtain the required keys from [CW23], whose secret-key size is of fixed polynomial $\mathrm{poly}(\lambda)$, independent of depth and attribute length and which relies on the plain LWE assumption.

We observe that although this scheme supports only *bounded-depth* circuits, it suffices for our purpose because it supports *unbounded key requests* and it's depth limitation can be overcome when combined with FE of [HLL23] which supports *unbounded depth* but only a *single key* request. Additionally, the size restriction can be overcome for free by using the slotting technique described previously.

Using [CW23], we complete the scheme as follows: the CP-ABE encryptor knows the slot $j$ and can regenerate the slot-$j$ KP-ABE instance from $r_j$, using a depth bound $\hat{d} = O(2^j)$ sufficient for the corresponding universal circuit. Since $2^j = O(|C|)$, the encryptor can afford the cost of setting up this instance and generating the FE key and the corresponding KP-ABE ciphertext. The depth dependence of these computations is therefore harmless; what must remain independent of depth is only the KP-ABE key length, since that determines the FE output length, which is satisfied by [CW23].

Thus, our key contribution is simply to *observe* that everything needed was already provided by recent, beautiful works that were developed without our compiler in mind! By virtue of this observation we obtain not only the first unbounded CP-ABE with complete collusion resistance, but also the first ABE for Turing machines from the standard circular small-secret LWE assumption, albeit in the secret key setting (see Section 1.2.3).

We also draw attention of the reader to a subtle but crucial difference between the bounded depth setting of [AY20a] and the unbounded setting here. In [AY20a], the necessary requirements of the ingredients, namely succinctness of FE ciphertexts and compactness of KP-ABE keys, are both derived from the algebraic properties of a single KP-ABE scheme by Boneh et al. [BGG+14] (the $\mathsf{Bgg}^+$ we saw above). The $\mathsf{Bgg}^+$ scheme boasts of both compact keys as well as succinct ciphertexts, and the succinct FE scheme of [GKP+13] which is used in [AY20a], lifts this to the FE setting via a clever compilation with FHE which does not affect the original ciphertext succinctness. Collusion resistance is also derived from $\mathsf{Bgg}^+$. On the other hand, in our setting, the ingredient schemes, namely the FE by Hsieh, Lin and Luo [HLL23] and the KP-ABE by Cini and Wee [CW23] are fundamentally different in design and each satisfies exactly one of the required properties while strongly resisting the other. The former [HLL23] has depth-independent ciphertext but no collusion resistance, whereas the latter [CW23] has depth independent key and collusion resistance but large ciphertexts. It is therefore especially fortuitous that these two very different, highly non-trivial schemes, can work together to give (so much) more than a sum of parts, that too via such a simple compiler.

*Security in brief.* We wrap up by describing security. For each slot $j$, the FE key for the fixed circuit $\mathcal{H}_{2^j,L}$ is reused across all encryption queries, by determinism of FE key generation. Thus, single-key FE security suffices even when the adversary has access to an encryption oracle. Consider any adversary $\mathcal{A}$ that interacts with the very selective security experiment for the above constructed scheme. Recall that, $\mathcal{A}$ first commits to the challenge $C^*$ and all key-queries $\{\mathbf{x}^{(t)}\}_{t\in[Q]}$ with $C^*(\mathbf{x}^{(t)}) = 1$ for every $t \in [Q]$,

and then the master keys are sampled. Let $j^*$ be such that $2^{j^*-1} < |C^*| \leq 2^{j^*}$, and let $\widetilde{C}^*$ denote the padded description of $C^*$.The proof is a short hybrid argument.

First, using single-key very-selective simulation security of FE, we jointly simulate the FE public key, the functional key for $\mathcal{H}_{2^{j^*},L}$, and the FE ciphertexts in the key-query responses at slot $j^*$, given the circuit and its outputs $\{\mathcal{H}_{2^{j^*},L}(\mathbf{x}^{(t)}, r_{j^*})\}_{t\in[Q]}$. In contrast, a simulator that attempts to simulate all $\lambda$ FE keys/ciphertexts will no longer be efficient – since at slot $\lambda$ (for example), the circuit $\mathcal{H}_{2^\lambda,L}$ has exponential size; however we show that simulating only for slot $j^*$ is sufficient to argue security!

At this point the adversary's view contains only KP-ABE keys for the circuits $\mathcal{U}_{2^{j^*},L}[\mathbf{x}^{(t)}]$ and KP-ABE encryptions of policy descriptions. Finally, since admissibility guarantees $\mathcal{U}_{2^{j^*},L}[\mathbf{x}^{(t)}](\widetilde{C}^*) = C^*(\mathbf{x}^{(t)}) = 1$ for every queried $\mathbf{x}^{(t)}$, i.e. all released keys are non-decrypting for the challenge attribute $\widetilde{C}^*$, a direct reduction to the Sel-IND security of the (collusion resistant) KP-ABE concludes the proof. Please see Section 4 for details.

**KP-ABE Analog?** At first glance, one would expect that the dual question – a secret key KP-ABE for unbounded depth circuits from circular small-secret LWE alone should follow by symmetry, but this is far from true. In the key policy setting, the key generator now holds the circuit $C$ and can afford to work with objects of size $\mathsf{poly}(|C|)$, while the encryptor now holds only the short attribute $\mathbf{x}$ and does not know the size or the depth of the policies. Redistributing the computation appropriately, the natural compiler would now need a bounded-depth KP-ABE whose *ciphertext* size is independent of the depth bound with no constraint at all on the sizes of the master public key or of the secret keys. This is precisely the "Dual of [CW23]": there the secret key is compact, $|\mathsf{sk}| = \mathsf{poly}(\lambda)$, while the ciphertext grows with the depth bound and here, the depth independence must be moved from the secret key to the ciphertext. Unfortunately, building such a scheme is an interesting open problem. Of course, for the other ingredient, HLL23 [HLL23] do achieve depth independence of not only the ciphertexts but also keys whereas what would suffice here is a weakening in which the master public key and the secret keys are allowed to be large. However, we do not currently know how to leverage this weakening even in the SK setting.

### 1.2.3 Secret-Key KP-ABE for Turing Machines (Section 5).

We now sketch how to lift the previous CP-ABE to KP-ABE for Turing machines from the same assumption. Thus, we obtain a secret-key KP-ABE in which the ciphertext encrypts an attribute $(\mathbf{x}, 1^t)$, the key encodes a machine $M$, and decryption succeeds when $M$ accepts $\mathbf{x}$ within $t$ steps. The compiler follows the blueprint of [AKY24b] (in turn building on [AMVY21]) and is black-box in its two ingredients: the secret-key CP-ABE for unbounded depth circuits constructed above, and any bounded-depth KP-ABE, e.g. [BGG+14].

The difficulty specific to Turing machines is that there are *two* independent, a-priori unbounded quantities – the attribute length $L_x := |(\mathbf{x}, 1^t)|$ and the machine length $L_M := |M|$ – and each party sees only its own, while any circuit emulation of the computation must fix both. The resolution, as in [AKY24b], is to compare the two sizes and let the party holding the larger object prepare one slot for every possible length of the smaller one. Accordingly, we build two schemes, each correct on one side of the comparison, and run them in parallel.

*Case* $L_M \leq L_x$. The encryptor computes, for every $i \in [L_x]$, a ciphertext of the secret-key CP-ABE under instance $i$, encrypting the policy $\hat{U}_{i,\mathbf{x},t}$ which takes an $i$-bit machine description as input and emulates it on $\mathbf{x}$ for $t$ steps. The key for $M$ is a single CP-ABE attribute key for $\tilde{M}$ under instance $L_M$; decryption selects the matching ciphertext component and performs the CP-ABE decryption. We can bound $|\hat{U}_{i,\mathbf{x},t}| = \mathsf{poly}(L_x, t)$ with depth $O(t\lambda)$. Since the time bound $t$ is chosen per ciphertext, the depth of the policy is unbounded from the key generator's viewpoint, which is what necessitates using CP-ABE for unbounded depth circuits.

*Case* $L_M > L_x$. In this case, the key generator computes, for every $i \in [L_M]$, a bounded-depth KP-ABE key under instance $i$ for the circuit $U_{i,M}$, which takes an $i$-bit attribute $(\mathbf{x}, 1^t)$ as input and emulates $M$ on $\mathbf{x}$ for $t$ steps. The encryptor produces a single KP-ABE ciphertext of $(\mathbf{x}\|1^t)$ under instance $L_x$. Here the depth causes no trouble: since $1^t$ is part of the $i$-bit attribute, we have $t \leq i$, so the depth of $U_{i,M}$ is at most $i\lambda$ – known at key generation time – and a bounded-depth KP-ABE with per-slot depth bound $i\lambda$ suffices. In both cases, the exponentially many potential instances are kept consistent between $\mathsf{KeyGen}$ and $\mathsf{Enc}$ by deriving the randomness of instance $i$ from $\mathsf{PRF.Eval}(K, i)$.

The compiler itself is similar to [AKY24b] (and previously [AMVY21]), but can be simplified since we are in the symmetric key setting: for instance, unlike [AKY24b], we do not require IBE or garbled circuits. Please see Section 5 for details.

### 1.3 The Curious Case of CP versus KP for Unbounded ABE

All of our constructions are either ciphertext-policy (CP) or key-policy (KP) and refuse adaptation to the dual counterpart[8]. The curious KP-CP asymmetry is visible already in the simplest possible setting, that of a single key unbounded ABE. For ciphertext policy, a single key ABE for unbounded circuits is immediate from the garbled circuit based construction of Sahai and Seyalioglu [SS10] which uses only public key encryption. In the key policy setting, the only single key KP-ABE for unbounded depth circuits that we know is the sophisticated construction of HLL23 [HLL23], from circular small-secret LWE.

Now consider the constructions presented in this work. As discussed above, the public key KP-ABE of Section 3 and the secret key CP-ABE of Section 4 encounter barriers when adapted to their dual settings. The TM-ABE of Section 5 then inherits the second obstruction, since its ciphertext policy analogue would call for exactly the unbounded depth secret key KP-ABE that we are missing.

We find this surprising, and we do not have a satisfying explanation for it. Understanding the reason for this phenomenon and hopefully overcoming it is left to future work.

### 1.4 Open Problems.

This leaves us with several open problems. The most immediate ones are to find dual versions of each of our three constructions, namely, a public key CP-ABE for unbounded circuits from circular LWE and pairings, a secret key KP-ABE for unbounded circuits from circular LWE, and a CP-ABE for Turing machines from circular LWE. Beyond these, it would be very interesting to remove pairings from the construction of Section 3, or to dispense with the symmetric key setting in Sections 4 and 5. Finally, our doubly circular assumption is currently reduced to SXDH and circular LWE only for constant $\{f_i\}_i$ (Section 3.3); extending this reduction to the general case or making the assumption instance-independent would considerably strengthen the standard model security of our scheme. We leave these as fascinating directions for future work.

## 2 Preliminaries

In this section, we present the notation and the required preliminaries, some of which are recalled or adapted from [AY20b, AWY20, AMYY25, Zha22, AKY24b].

*Notation.* We use bold capital letters to denote a matrix and bold small letters to represent vectors. By default, a vector $\mathbf{v}$ is a column vector and $\mathbf{v}^\intercal$ is a row vector. Throughout the paper, we use $\lambda$ to denote the security parameter. Let $q, n \in \mathbb{N}$. For any element $v \in \mathbb{Z}_q$, we denote $(b_1, \ldots, b_{\lceil \log q \rceil}) = \mathsf{bits}(v)$ to be the bit representation of $v$ such that $b_i \in \{0,1\}$ and $b_1$ is LSB. The notation is easily extended to

[8] Observe that the folklore universal circuit transformation fails for unbounded circuits since this requires the circuit size bound $s$ must be fixed at setup, which is not available in the unbounded setting.

the vector $\mathbf{v} \in \mathbb{Z}_q^n$. For any vector $\mathbf{v}$, $\mathbf{v}[i]$ denotes the $i$th element. For any matrix $\mathbf{V}$, $\mathbf{V}[i,j]$ denotes the element at $i$-th row and $j$-th column. We consider the infinity norm $\|v\| = v$, $\|\mathbf{v}\| = \max_i |\mathbf{v}[i]|$, where we identify an element $x \in \mathbb{Z}_q$ with its centered representative in $[-\lfloor q/2 \rfloor, \lfloor q/2 \rfloor]$ when we consider a norm of it. For a set $S$, $|S|$ denotes its cardinality, and for a string $x$, $|x|$ denotes its length. For integers $a \leq b$, let $[a,b] = \{k \in \mathbb{N} : a \leq k \leq b\}$, $[n] = [1,n]$.. The strings of the form $(x, 1^t)$ are encoded over the alphabet $\{0,1,,\}$ so that the delimiter is explicit; thus, $(x,1^t)$ is distinguished from $(x1, 1^{t-1})$.

A function $f(n)$ is *negligible* if $f(n) = O(n^{-c})$ for every constant $c > 0$, and we write $\mathsf{negl}(n)$ for a negligible function. It is *polynomial* if $f(n) = O(n^c)$ for some constant $c > 0$, denoted by $\mathrm{poly}(n)$. We use PPT to denote probabilistic polynomial-time algorithms. An event occurs with *overwhelming probability* if it holds with probability $1 - \mathsf{negl}(\lambda)$. For distribution ensembles $\{X_\lambda\}$ and $\{Y_\lambda\}$, $X_\lambda \approx_c Y_\lambda$ denotes computational indistinguishability.

We prove the following lemma, which will be needed later.

**Lemma 2.1 (Rounding error).** For $a, b \in \mathbb{R}$, we have $\| \lfloor a+b \rfloor - \lfloor a \rfloor - \lfloor b \rfloor \| < 2$.

*Proof.* We have $\| \lfloor a+b \rfloor - \lfloor a \rfloor - \lfloor b \rfloor \| = \| \lfloor a - \lfloor a \rfloor + b - \lfloor b \rfloor \rfloor \| < 2$. □

## 2.1 Lattice Preliminaries

Here, we recall some facts on lattices that are needed for the technical part. Throughout this section, $n$, $m$, and $q$ are integers such that $n = \mathrm{poly}(\lambda)$ and $m \geq n\lceil \log q \rceil$. In the following, we define a function $\rho_\sigma$ as $\rho_\sigma = \mathsf{Exp}(-\pi \|\mathbf{x}/\sigma\|_2^2)$, where $\|\cdot\|_2$ is the Euclidean norm. We will overload the notation to denote by $\rho_\sigma(S)$ the summation of $\rho_\sigma(\mathbf{x})$ over all $\mathbf{x} \in S$ for a countable set $X$. For a countable set $S$, $\mathcal{D}_{S,\gamma}$ refers to a distribution that outputs $\mathbf{x} \in S$ with probability $\rho_\sigma(\mathbf{x})/\rho(S)$.

Let

$$\mathbf{g} = (2^0, 2^1, \ldots, 2^{\frac{m}{n+1}-1})^\intercal, \quad \mathbf{G} = \mathbf{I}_{n+1} \otimes \mathbf{g}^\intercal$$

be the gadget vector and the gadget matrix. For $\mathbf{p} \in \mathbb{Z}_q^{n+1}$, we write $\mathbf{G}^{-1}(\mathbf{p})$ for the $m$-bit vector $(\mathsf{bits}(\mathbf{p}[1]), \ldots, \mathsf{bits}(\mathbf{p}[n+1]))^\intercal$, where $\mathsf{bits}(\mathbf{p}[i])$ are $m/(n+1)$ bits for each $i \in [n+1]$. The notation extends column-wise to matrices and it holds that $\mathbf{G}\mathbf{G}^{-1}(\mathbf{P}) = \mathbf{P}$.

**Lemma 2.2 ([Lyu12, Lemma 4.4]).** We have $\Pr\left[\|z\| > \sqrt{\lambda}\sigma; z \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}\right] \leq 2\mathsf{Exp}(-\lambda/2)$.

*Assumption* 2.3 ((Circular Small Secret) LWE Assumption [HLL23]). Let $n, m, m', q, \sigma, \sigma_{\mathbf{s}}, \sigma'$ be functions of $\lambda$ and

$$\begin{aligned}
&\overline{\mathbf{A}}_{\mathsf{fhe}} \leftarrow \mathbb{Z}_q^{n \times m}, \quad \overline{\mathbf{A}}_0 \leftarrow \mathbb{Z}_q^{n \times m'}, \quad \mathbf{r} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma_{\mathbf{s}}}^n, \quad \mathbf{s} \leftarrow (\mathbf{r}^\intercal, -1)^\intercal, \\
&\mathbf{e}_{\mathsf{fhe}} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^m, \quad \mathbf{e}_0 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{m'}, \quad \mathbf{R} \leftarrow \{0,1\}^{m \times (n+1)\lceil \log q \rceil m}
\end{aligned}$$

The circular small-secret LWE assumption $\mathsf{csLWE}_{n,m,m',q,\sigma,\sigma_{\mathbf{s}},\sigma'}$ states that

$$\begin{aligned}
&\left\{ \left(1^\lambda, \quad \mathbf{A}_{\mathsf{fhe}} := \begin{pmatrix} \overline{\mathbf{A}}_{\mathsf{fhe}} \\ \mathbf{c}_{\mathsf{fhe}}^\intercal = \mathbf{r}^\intercal \overline{\mathbf{A}}_{\mathsf{fhe}} + \mathbf{e}_{\mathsf{fhe}}^\intercal \end{pmatrix}, \quad \mathbf{S} = \mathbf{A}_{\mathsf{fhe}}\mathbf{R} - \mathsf{bits}(\mathbf{s}) \otimes \mathbf{G}, \quad \overline{\mathbf{A}}_0, \quad \mathbf{c}_0^\intercal = \mathbf{r}^\intercal \overline{\mathbf{A}}_0 + \mathbf{e}_0^\intercal \right) \right\}_{\lambda \in \mathbb{N}} \\
&\approx_c \left\{ \left(1^\lambda, \quad \underbrace{\begin{pmatrix} \overline{\mathbf{A}}_{\mathsf{fhe}} \\ \mathbf{c}_{\mathsf{fhe}}^\intercal \leftarrow \mathbb{Z}_q^{1 \times m} \end{pmatrix}, \quad \mathbf{S} \leftarrow \mathbb{Z}_q^{(n+1) \times (n+1)\lceil \log q \rceil m}}_{\text{circular terms}}, \quad \overline{\mathbf{A}}_0, \quad \mathbf{c}_0^\intercal \leftarrow \mathbb{Z}_q^{1 \times m'} \right) \right\}_{\lambda \in \mathbb{N}}
\end{aligned}$$

For the LWE assumption $\mathrm{LWE}_{n,m',q,\sigma'}$, the distribution of $\mathbf{r}$ is changed to $\mathbf{r} \leftarrow \mathbb{Z}_q^n$ and the circular terms are removed. When we refer simply to LWE, we require the assumption to hold for every polynomially bounded sample count $m' = m'(\lambda)$. For circular small-secret LWE, we additionally set $m \geq n\lceil \log q \rceil$. We refer to LWE with $q/\sigma' = 2^{n^\delta}$ for some constant $\delta \in (0,1)$ as LWE with a subexponential modulus-to-error ratio.

*Remark* 2.4 (Equivalent formulation of circular small-secret LWE). We also use an equivalent formulation in which the non-circular terms are

$$\mathbf{A}_0 \leftarrow \mathbb{Z}_q^{(n+1)\times m'}, \qquad \mathbf{c}_0^\intercal = \mathbf{s}^\intercal \mathbf{A}_0 + \mathbf{e}_0^\intercal,$$

and $\mathbf{c}_0^\intercal$ is replaced by an independent uniform vector in $\mathbb{Z}_q^{1\times m'}$ in the ideal distribution. All circular terms remain unchanged. It is straightforward to see that this variant is equivalent to the above, by considering a simulation that appends a random vector to $\overline{\mathbf{A}}_0$ and modify $\mathbf{c}_0$ accordingly.

## 2.2 Bilinear Map Preliminaries

**Notation.** Let a group generator $\mathsf{GroupGen}$ takes as input $1^\lambda$ and outputs a group description $\mathcal{G} = (p, \mathbb{G}, g)$, where $p$ is a prime of $\Theta(\lambda)$ bits, $\mathbb{G}$ is a cyclic group of order $p$, and $g$ is the generator of $\mathbb{G}$. Let a bilinear group generator $\mathsf{BLGroupGen}$ takes as input $1^\lambda$ and outputs a group description $\mathcal{G}_{\mathsf{BL}} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, [1]_2)$, where $p$ is a prime of $\Theta(\lambda)$ bits, $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$, are cyclic groups of order $p$, $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a non-degenerate bilinear map, and $[1]_1, [1]_2$ is the generator of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. We employ the implicit representation of group elements: for a matrix $\mathbf{A}$ over $\mathbb{Z}_p$, we define for $s \in \{1, 2, T\}$, $[\mathbf{A}]_s := g_s^{\mathbf{A}}$, where exponentiation is carried out component-wise. We will use similar notation for vectors.

*Assumption* 2.5 (Symmetric External Diffie-Hellman). Let $\mathcal{G}_{\mathsf{BL}} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, [1]_2) \leftarrow \mathsf{BLGroupGen}(1^\lambda)$. The symmetric external Diffie-Hellman assumption (SXDH) holds with respect to $\mathsf{BLGroupGen}$ if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ and all $j \in \{1, 2\}$,

$$|\Pr[\mathcal{A}(\mathcal{G}_{\mathsf{BL}}, [x]_j, [y]_j, [xy]_j) = 1] - \Pr[\mathcal{A}(\mathcal{G}_{\mathsf{BL}}, [x]_j, [y]_j, [r]_j) = 1]| = \mathsf{negl}(\lambda)$$

where $x, y, r \leftarrow \mathbb{Z}_p$. In other words, the SXDH assumption corresponds to DDH holding in both $\mathbb{G}_1$ and $\mathbb{G}_2$. We use a generalisation of this assumption where instead of providing one $[y]_j$ and $[xy]_j$ to $\mathcal{A}$, we provide $[y_1]_j, \ldots, [y_Q]_j$ and $[xy_1]_j, \ldots, [xy_Q]_j$ for some $Q \in \mathbb{N}$. The latter follows directly from the former by a standard hybrid argument.

### 2.2.1 Random Representation (RR)/ Shoup Model for Bilinear Group Setting

In this section, we extend the Shoup Model [Sho97] to the Bilinear Group Setting. We use the explicit definition of this model from [Zha22].

Let $\mathcal{G}_{\mathsf{BL}} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, [1]_2) \leftarrow \mathsf{BLGroupGen}(1^\lambda)$. Let $S_1, S_2, S_T \subseteq \{0,1\}^*$ be a set of strings of cardinality at least $p$. We will assume an upper bound is known on the length of strings in $S_1, S_2, S_T$. A random injection $L_j : \mathbb{Z}_p \to S_j$ for $j \in \{1, 2, T\}$ is chosen, which we will call the labeling functions. We will think of $L_j(x)$ as corresponding to $[x]_j$. The notation is extended component-wise to vectors. All parties, including the adversary, the cryptosystem, and the challenger, are able to make the following queries:

- **Labeling queries.** The party submits $(j, x) \in \{1, 2, T\} \times \mathbb{Z}_p$, and receives $L_j(x)$.
- **Group operations.** The party submits $(j, \ell_1, \ell_2, a_1, a_2) \in \{1, 2, T\} \times S_j^2 \times \mathbb{Z}_p^2$. If there exists $x_1, x_2 \in \mathbb{Z}_p$ such that $L_j(x_1) = \ell_1$ and $L_j(x_2) = \ell_2$, then the party receives $L_j(a_1 x_1 + a_2 x_2)$. Otherwise, the party receives $\perp$.
- **Mapping Queries.** The party submits $(\ell_1, \ell_2) \in S_1 \times S_2$. If there exists $x_1, x_2 \in \mathbb{Z}_p$ such that $L_1(x_1) = \ell_1$ and $L_2(x_2) = \ell_2$, then party receives $L_T(x_1 \cdot x_2)$. Otherwise, the party receives $\perp$.

For an algorithm $A$ in the random representation model, we write $A^{\mathcal{G}_{\mathsf{RR}}}$.

**Symbolic Group Model.** The symbolic group model for a bilinear group setting $\mathcal{G}_{\mathsf{BL}}$ gives to the adversary the same interface as the corresponding generic group model, except that internally the challenger stores lists of elements in the ring $\mathbb{Z}_p[X_1, \ldots, X_n]$ instead of lists of group elements. All the element gates compute the same operation over the ring.

## 2.3 Cryptographic Primitives

### 2.3.1 Attribute-Based Encryption for Circuits.

Here, we introduce attribute-based encryption (ABE) both in the public key and secret key settings. We consider the public key setting by default and will explicitly say so when we consider the secret key setting.

**Public-Key ABE.** Let $R_{\mathsf{prm}} : \mathcal{X}_{\mathsf{prm}} \times \mathcal{Y}_{\mathsf{prm}} \to \{0,1\}$ be a relation family. An ABE scheme $\mathsf{ABE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ for a relation family $R = \{R_{\mathsf{prm}}\}_{\mathsf{prm}}$ consists of the following probabilistic polynomial-time algorithms.

$\mathsf{Setup}(1^\lambda, \mathsf{prm}) \to (\mathsf{mpk}, \mathsf{msk})$: The setup algorithm takes as input the unary representation of the security parameter $\lambda$ and the parameters $\mathsf{prm}$, and outputs a master public key $\mathsf{mpk}$, and a master secret key $\mathsf{msk}$.

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, Y) \to \mathsf{sk}_Y$: The key generation algorithm takes as input the master public key $\mathsf{mpk}$, the master secret key $\mathsf{msk}$, and $Y \in \mathcal{Y}_{\mathsf{prm}}$ and outputs a secret key $\mathsf{sk}_Y$.

$\mathsf{Enc}(\mathsf{mpk}, X, \mu) \to \mathsf{ct}_X$: The encryption algorithm takes as input a master public key $\mathsf{mpk}$, $X \in \mathcal{X}_{\mathsf{prm}}$, and a message $\mu \in \{0,1\}$, and outputs a ciphertext $\mathsf{ct}_X$.

$\mathsf{Dec}(\mathsf{mpk}, \mathsf{ct}_X, X, \mathsf{sk}_Y, Y) \to \mu$ or $\perp$: The decryption algorithm takes as input the master public key $\mathsf{mpk}$, a ciphertext $\mathsf{ct}$, $X$, a private key $\mathsf{sk}_Y$, and $Y$. It outputs the message $\mu$ or $\perp$.

**Definition 2.6 (Correctness).** The $\mathsf{ABE}$ scheme for a relation family $R$ is correct if, for every $\mathsf{prm}$, $X \in \mathcal{X}_{\mathsf{prm}}$, $Y \in \mathcal{Y}_{\mathsf{prm}}$ such that $R_{\mathsf{prm}}(X,Y) = 0$,[9] and every message $\mu \in \{0,1\}$,

$$\Pr\left[\mathsf{Dec}(\mathsf{mpk}, \mathsf{ct}_X, X, \mathsf{sk}_Y, Y) \neq \mu : \begin{array}{l}(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{prm}),\\ \mathsf{sk}_Y \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, Y),\\ \mathsf{ct}_X \leftarrow \mathsf{Enc}(\mathsf{mpk}, X, \mu)\end{array}\right] \leq \mathsf{negl}(\lambda).$$

**Definition 2.7 (Sel-IND Security for ABE).** For $\mathsf{ABE}$ scheme for a relation family $R$ and message space $\{0,1\}$, let us define the Sel-IND experiment $\mathsf{Expt}_{\mathcal{A}}^{\mathsf{ABE}}(1^\lambda, b)$ as follows:

1. **Choosing the Attribute:** The adversary commits to $(\mathsf{prm}, X^\star)$.
2. **Setup Phase and Challenge Phase:** The challenger samples $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, \mathsf{prm})$. It computes $\mathsf{ct}_X \leftarrow \mathsf{Enc}(\mathsf{mpk}, X^*, b)$. It gives $(\mathsf{mpk}, \mathsf{ct}_X)$ to $\mathcal{A}$.
3. **Query Phase:** For the $i$-th key query, $\mathcal{A}$ chooses $Y_i \in \mathcal{Y}_{\mathsf{prm}}$ and sends it to the challenger. If $R_{\mathsf{prm}}(X^*, Y_i) = 0$, the challenger aborts the game. Otherwise it replies with $\mathsf{sk}_{Y_i} \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, Y_i)$.
4. **Output Phase:** $\mathcal{A}$ outputs a guess bit $b'$ as the output of the experiment.

[9] For circuit ABE, we follow the convention common in lattice-based constructions that decryption is authorized when $C(\mathbf{x}) = 0$ rather than $C(\mathbf{x}) = 1$. Here, $C$ is associated with the ciphertext and $\mathbf{x}$ with the secret key in CP-ABE, and vice versa in KP-ABE. Accordingly, our general definition takes $R_{\mathsf{prm}}(X,Y) = 0$ to indicate that decryption is authorized.

The ABE scheme for a relation family $R$ and message space $\{0,1\}$ is Sel-IND secure if, for every PPT adversary $\mathcal{A}$,

$$\mathsf{Adv}^{\mathsf{Sel\text{-}IND}}_{\mathsf{ABE},\mathcal{A}} := \left|\Pr\left[\mathsf{Expt}^{\mathsf{ABE}}_{\mathcal{A}}(1^\lambda,0)=1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{ABE}}_{\mathcal{A}}(1^\lambda,1)=1\right]\right| \leq \mathsf{negl}(\lambda).$$

**Definition 2.8 (VerSel-IND Security for ABE).** The VerSel-IND security game is defined similar to Sel-IND security game except that the adversary $\mathcal{A}$ chooses the key queries $\{Y_i \in \mathcal{Y}_{\mathsf{prm}}\}_{i\in[Q]}$, where $Q$ is the number of key queries made by $\mathcal{A}$, along with the challenge $X^*$ in the beginning of the security game. We define the advantage $\mathsf{Adv}^{\mathsf{VerSel\text{-}IND}}_{\mathsf{ABE},\mathcal{A}}$ of the adversary $\mathcal{A}$ accordingly and say that the scheme satisfies VerSel-IND security if the quantity is negligible.

**Secret-Key ABE.** The syntax and correctness of secret-key ABE are defined analogously to those of public-key ABE, except that Enc takes msk instead of mpk, and mpk is omitted from the inputs to all algorithms. Moreover, the adversary is given oracle access to the encryption algorithm.

**Definition 2.9 (VerSel-IND Security for secret-key ABE).** For ABE scheme for a relation family $R$ and message space $\{0,1\}$, let us define the very selective IND-CPA experiment $\mathsf{Expt}^{\mathsf{ABE}}_{\mathcal{A}}(1^\lambda,b)$ as follows:

1. **Choosing the Attribute:** The adversary commits to prm, $X^\star \in \mathcal{X}_{\mathsf{prm}}$ and $\{Y_i \in \mathcal{Y}_{\mathsf{prm}}\}_{i\in[Q]}$.
2. **Setup and Challenge Phase:** The challenger samples $\mathsf{msk} \leftarrow \mathsf{Setup}(1^\lambda,\mathsf{prm})$. It computes $\mathsf{ct}_{X^\star} \leftarrow \mathsf{Enc}(\mathsf{msk},X^*,b)$. If $R_{\mathsf{prm}}(X^*,Y_i)=0$ for any $i$, challenger aborts the game. Otherwise, it computes $\mathsf{sk}_{Y_i} \leftarrow \mathsf{KeyGen}(\mathsf{msk},Y_i)$. It gives $\mathsf{ct}_{X^\star}$ and $\{\mathsf{sk}_{Y_i}\}_{i\in[Q]}$ to $\mathcal{A}$.
3. **Query Phase:** $\mathcal{A}$ then makes encryption queries in an arbitrary order.
   - As the $i$-th encryption query, $\mathcal{A}$ submits $X_i \in \mathcal{X}_{\mathsf{prm}}$ and a message $b_i \in \{0,1\}$ to the challenger. $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{msk},X_i,b_i)$. It returns $\mathsf{ct}_i$ to $\mathcal{A}$.
4. **Output Phase:** $\mathcal{A}$ outputs a guess bit $b'$ as the output of the experiment.

The ABE scheme for a relation family $R$ and message space $\{0,1\}$ is VerSel-IND secure if, for every PPT adversary $\mathcal{A}$,

$$\left|\Pr\left[\mathsf{Expt}^{\mathsf{ABE}}_{\mathcal{A}}(1^\lambda,0)=1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{ABE}}_{\mathcal{A}}(1^\lambda,1)=1\right]\right| \leq \mathsf{negl}(\lambda).$$

In the following, we recall definitions of various ABEs by specifying the relation, both in the public-key and secret-key settings.

**Key-Policy ABE for circuits.** To define KP-ABE for circuits, we set $\mathcal{X} = \{0,1\}^*$ and $\mathcal{Y}$ as the set of all circuits and define $R(\mathbf{x},C) = C(\mathbf{x})$. In this paper, we consider circuit class $\mathcal{C}_{\mathsf{inp},d}$ that consists of circuits with input length $\mathsf{inp} := \mathsf{inp}(\lambda)$ and depth $d := d(\lambda)$. To do so, we set $\mathsf{prm} = (1^{\mathsf{inp}},1^d)$, $\mathcal{X}_{\mathsf{prm}} = \{0,1\}^{\mathsf{inp}}$, and $\mathcal{Y}_{\mathsf{prm}} = \mathcal{C}_{\mathsf{inp},d}$.

**Ciphertext-Policy ABE for circuits.** To define CP-ABE for circuits, we set $\mathcal{Y} = \{0,1\}^*$ and $\mathcal{X}$ as the set of all circuits and define $R(C,\mathbf{x}) = C(\mathbf{x})$. In this paper, we consider circuit class $\mathcal{C}_{\mathsf{inp},d}$ that consists of circuits with input length $\mathsf{inp} := \mathsf{inp}(\lambda)$ and depth $d := d(\lambda)$. To do so, we set $\mathsf{prm} = (1^{\mathsf{inp}},1^d)$, $\mathcal{X}_{\mathsf{prm}} = \mathcal{C}_{\mathsf{inp},d}$, and $\mathcal{Y}_{\mathsf{prm}} = \{0,1\}^{\mathsf{inp}}$.

For KP-ABE and CP-ABE for circuits of unbounded depth, we will drop $1^d$ from the prm and from the subscript and fix $\mathsf{inp} = L$.

**Lemma 2.10.** (**Adapted from [CW23]**.) Fix a constant $\delta \in (0,1)$. Assuming the hardness of LWE with modulus-to-error ratio $2^{n^\delta}$, where $n$ is the LWE dimension, there exists a selectively secure KP-ABE scheme (as per Definition 2.7) supporting for circuits of depth $d$ over $L$-bit inputs, with

$$|\mathsf{mpk}| = L \cdot d^{2+2/\delta} \cdot \mathrm{poly}(\lambda), \qquad |\mathsf{ct}| = L \cdot d^{2+1/\delta} \cdot \mathrm{poly}(\lambda), \qquad |\mathsf{sk}| = \mathrm{poly}(\lambda).$$

### 2.3.2 Functional Encryption for Circuits

Let $\mathcal{C} = \{\mathcal{C}_{\mathsf{prm}}\}_{\mathsf{prm}}$ be a family of circuit classes, where each $C \in \mathcal{C}_{\mathsf{prm}}$ maps an input space $\mathcal{X}_{\mathsf{prm}}$ to an output space $\mathcal{Y}_{\mathsf{prm}}$ and $\mathsf{prm}$ specifies the circuit class. A functional encryption scheme $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.KeyGen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ for $\mathcal{C}$ consists of the following algorithms:

$\mathsf{FE.Setup}(1^\lambda, \mathsf{prm}) \to (\mathsf{mpk}, \mathsf{msk})$: A PPT algorithm that takes the security parameter $\lambda$ and parameters $\mathsf{prm}$, and outputs a master public key $\mathsf{mpk}$ and a master secret key $\mathsf{msk}$.

$\mathsf{FE.KeyGen}(\mathsf{mpk}, \mathsf{msk}, C) \to \mathsf{sk}_C$: A PPT algorithm that takes the master public key $\mathsf{mpk}$, master secret key $\mathsf{msk}$, and a circuit $C \in \mathcal{C}_{\mathsf{prm}}$, and outputs a functional secret key $\mathsf{sk}_C$.

$\mathsf{FE.Enc}(\mathsf{mpk}, x) \to \mathsf{ct}$: A PPT algorithm that takes the master public key $\mathsf{mpk}$ and a message $x \in \mathcal{X}_{\mathsf{prm}}$, and outputs a ciphertext $\mathsf{ct}$.

$\mathsf{FE.Dec}(\mathsf{mpk}, C, \mathsf{sk}_C, \mathsf{ct}) \to y$: A deterministic polynomial-time algorithm that takes the master public key $\mathsf{mpk}$, circuit $C$, functional secret key $\mathsf{sk}_C$, and ciphertext $\mathsf{ct}$, and outputs $y \in \mathcal{Y}_{\mathsf{prm}} \cup \{\perp\}$.

**Definition 2.11 (Correctness).** The scheme $\mathsf{FE}$ is correct if, for every supported choice of $\mathsf{prm}$, $C \in \mathcal{C}_{\mathsf{prm}}$, and $x \in \mathcal{X}_{\mathsf{prm}}$ fixed independently of the setup randomness,

$$\Pr\left[\begin{array}{l}(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{FE.Setup}(1^\lambda, \mathsf{prm});\\ \mathsf{ct} \leftarrow \mathsf{FE.Enc}(\mathsf{mpk}, x);\\ \mathsf{sk}_C \leftarrow \mathsf{FE.KeyGen}(\mathsf{mpk}, \mathsf{msk}, C) :\\ \mathsf{FE.Dec}(\mathsf{mpk}, C, \mathsf{sk}_C, \mathsf{ct}) \neq C(x)\end{array}\right] \leq \mathsf{negl}(\lambda),$$

where the probability is over the random coins of $\mathsf{FE.Setup}$, $\mathsf{FE.KeyGen}$, and $\mathsf{FE.Enc}$.

**Definition 2.12 (One-Key Very Selective-SIM Security).** For a stateful PPT adversary $\mathcal{A}$ and a PPT simulator $\mathsf{Sim}$, consider the following experiments. The adversary commits to $\mathsf{prm}$, a circuit $C \in \mathcal{C}_{\mathsf{prm}}$, and a polynomial-length sequence $\{x_i\}_{i\in[Q]}$ with $x_i \in \mathcal{X}_{\mathsf{prm}}$ before setup, and retains its state between its two invocations.

| **Real experiment** $\mathsf{Expt}^{\mathsf{real}}_{\mathsf{FE},\mathcal{A}}(1^\lambda)$ | **Ideal experiment** $\mathsf{Expt}^{\mathsf{ideal}}_{\mathsf{FE},\mathcal{A},\mathsf{Sim}}(1^\lambda)$ |
|---|---|
| • $(\mathsf{prm}, C, \{x_i\}_{i\in[Q]}) \leftarrow \mathcal{A}(1^\lambda)$.<br>• $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{FE.Setup}(1^\lambda, \mathsf{prm})$.<br>• $\mathsf{sk}_C \leftarrow \mathsf{FE.KeyGen}(\mathsf{mpk}, \mathsf{msk}, C)$.<br>• $\mathsf{ct}_i \leftarrow \mathsf{FE.Enc}(\mathsf{mpk}, x_i)$ for each $i \in [Q]$.<br>• Output $\mathcal{A}(\mathsf{mpk}, \mathsf{sk}_C, \{\mathsf{ct}_i\}_{i\in[Q]})$. | • $(\mathsf{prm}, C, (x_i)_{i\in[Q]}) \leftarrow \mathcal{A}(1^\lambda)$.<br>• $(\mathsf{mpk}, \mathsf{sk}_C, \{\mathsf{ct}_i\}_{i\in[Q]})$<br>$\leftarrow \mathsf{Sim}(1^\lambda, \mathsf{prm}, C, \{C(x_i)\}_{i\in[Q]})$.<br>• Output $\mathcal{A}(\mathsf{mpk}, \mathsf{sk}_C, \{\mathsf{ct}_i\}_{i\in[Q]})$. |

The scheme $\mathsf{FE}$ is *one-key very selective-SIM secure* if there exists a PPT simulator $\mathsf{Sim}$ such that, for every stateful PPT adversary $\mathcal{A}$,

$$\left|\Pr\left[\mathsf{Expt}^{\mathsf{real}}_{\mathsf{FE},\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathsf{ideal}}_{\mathsf{FE},\mathcal{A},\mathsf{Sim}}(1^\lambda) = 1\right]\right| \leq \mathsf{negl}(\lambda).$$

The simulator receives the circuit and its evaluations, but not the messages themselves.

**Unbounded-depth circuits.** In this paper, we set $\mathsf{prm} = (1^{L_{in}}, 1^{L_{\mathsf{out}}})$, $\mathcal{X}_{\mathsf{prm}} = \{0,1\}^{L_{in}}$, and $\mathcal{Y}_{\mathsf{prm}} = \{0,1\}^{L_{\mathsf{out}}}$, and let $\mathcal{C}_{\mathsf{prm}} = \mathcal{C}_{L_{in},L_{\mathsf{out}}}$ consist of all Boolean circuits with input length $L_{in}$ and output length $L_{\mathsf{out}}$ (without specifying a circuit-size or depth bound at setup). Thus, setup and encryption run in time polynomial in $(\lambda, L_{in}, L_{\mathsf{out}})$, and the public key and ciphertext sizes satisfy the same bound, independently of circuit size and depth. Thus, we sometimes call such an FE *succinct* following the terminology of [GKP+13] meaning that the running time of the encryption algorithm is independent of the size of the circuits supported. Here, the dependency on the depth is removed as well. We write $\mathsf{FE.Setup}(1^\lambda, 1^{L_{in}}, 1^{L_{\mathsf{out}}})$ for $\mathsf{FE.Setup}(1^\lambda, (1^{L_{in}}, 1^{L_{\mathsf{out}}}))$ and use the same convention for the simulator.

The construction of [HLL23] provides such an FE scheme under the circular small-secret LWE assumption (Assumption 2.3), with the additional compact-key guarantee stated below.

*Remark* 2.13. The security definition above allows polynomially many ciphertexts under a single functional key. For the instantiation of [HLL23], the single-ciphertext simulation extends to this setting by a standard hybrid argument, using its ciphertext simulator under the same simulated public and the functional key [GVW12].

**Lemma 2.14 (Adapted from [HLL23, Corollary 18 and Section 3.4]).** Under the circular small-secret LWE assumption with subexponential modulus-to-error ratio, there exists a one-key very selective-SIM secure FE scheme for circuits of unbounded depth and size, with input and output lengths $L_{\mathsf{in}}$ and $L_{\mathsf{out}}$ specified at setup, such that

$$|\mathsf{FE.mpk}|,\ |\mathsf{FE.ct}| = (L_{\mathsf{in}} + L_{\mathsf{out}}) \cdot \mathrm{poly}(\lambda), \qquad |\mathsf{FE.sk}_C| = L_{\mathsf{out}} \cdot \mathrm{poly}(\lambda).$$

### 2.3.3 Slotted Inner Product Functional Encryption

We need slotted Inner Product Functional Encryption (IPFE) due to Lin and Vaikuntanathan [LV16, Lin17, BW19]. Slotted IPFE is a hybrid between a secret-key function-hiding IPFE and a public-key IPFE. In this scheme, a vector $\mathbf{u} \in \mathbb{Z}_p^n$ is divided into a public and private part respectively $\mathbf{u} = (\mathbf{u}_{\mathsf{pub}}, \mathbf{u}_{\mathsf{priv}})$ such that given the master secret key, the encryption algorithm can encrypt any vector $\mathbf{u}$ of its choice, but given only the public key, it can encrypt only to the public slot, i.e. $\mathbf{u}_{\mathsf{priv}} = 0$. Slotted IPFE can guarantee function hiding only with respect to the private slot. We provide the definitions from [BW19].

Let $\mathsf{BLGroupGen}$ be a group generator that outputs bilinear group $\mathcal{G}_{\mathsf{BL}} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, [1]_2)$. A slotted inner-product functional encryption (IPFE) scheme based on $\mathbb{G}$ consists of 5 efficient algorithms:

$\mathsf{iSetup}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{priv}}) \to (\mathsf{impk}, \mathsf{imsk})$: The setup algorithm takes as input two disjoint index sets, the public slot $\mathfrak{s}_{\mathsf{pub}}$ and the private slot $\mathfrak{s}_{\mathsf{priv}}$, and outputs a pair of master public key and master secret key $(\mathsf{impk}, \mathsf{imsk})$. The whole index set $\mathfrak{s}$ is $\mathfrak{s}_{\mathsf{pub}} \cup \mathfrak{s}_{\mathsf{priv}}$.

$\mathsf{iKeyGen}(\mathsf{imsk}, [\mathbf{v}]_2) \to \mathsf{isk}$: The key generation algorithm takes as input the master secret key and an encoding of a function vector $[\mathbf{v}]_2$, and outputs a secret key $\mathsf{isk}$ for $\mathbf{v} \in \mathbb{Z}_p^{\mathfrak{s}}$.

$\mathsf{iEnc}(\mathsf{imsk}, [\mathbf{u}]_1) \to \mathsf{ict}$: The encrypt algorithm takes as input the master secret key and an encoding of a message vector $[\mathbf{u}]_1$ and outputs a ciphertext $\mathsf{ict}$ for $\mathbf{u} \in \mathbb{Z}_p^{\mathfrak{s}}$.

$\mathsf{iDec}(\mathsf{isk}, \mathsf{ict}) \to T \vee \perp$: The decrypt algorithm takes as input a secret key $\mathsf{isk}$ and a ciphertext $\mathsf{ict}$, and outputs an element $T \in \mathbb{G}_T$ or $\perp$.

$\mathsf{iSlotEnc}(\mathsf{impk}, [\mathbf{u}_{\mathsf{pub}}]_1) \to \mathsf{ict}$: The slot encryption algorithm takes as input the master public key and a vector $\mathbf{u}_{\mathsf{pub}} \in \mathbb{Z}_p^{\mathfrak{s}_{\mathsf{pub}}}$ and outputs a ciphertext $\mathsf{ict}$.

**Definition 2.15 (Correctness).** We say the slotted inner-product functional encryption scheme satisfies decryption correctness if for all $\lambda \in \mathbb{N}$, all index sets $\mathfrak{s}$ and all vectors $\mathbf{u}, \mathbf{v} \in \mathbb{Z}_p^{\mathfrak{s}}$,

$$\Pr\left[\mathsf{iDec}(\mathsf{isk},\mathsf{ict}) = [\langle \mathbf{u},\ \mathbf{v}\rangle]_T \middle| \begin{array}{l}(\mathsf{impk},\mathsf{imsk}) \leftarrow \mathsf{iSetup}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{priv}}) \\ \mathsf{isk} \leftarrow \mathsf{iKeyGen}(\mathsf{imsk}, [\mathbf{v}]_2) \\ \mathsf{ict} \leftarrow \mathsf{iEnc}(\mathsf{imsk}, [\mathbf{u}]_1)\end{array}\right] = 1\ .$$

We say the slotted inner-product functional encryption scheme satisfies slot-mode correctness if for all $\lambda \in \mathbb{N}$, all disjoint index sets $\mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{priv}}$ and all vectors $\mathbf{u}_{\mathsf{pub}} \in \mathbb{Z}_p^{\mathfrak{s}_{\mathsf{pub}}}$, the following two distributions should be identical:

$$\left\{(\mathsf{impk},\mathsf{imsk},\mathsf{ict}) \middle| \begin{array}{l}(\mathsf{impk},\mathsf{imsk}) \leftarrow \mathsf{iSetup}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{priv}}) \\ \mathsf{ict} \leftarrow \mathsf{iEnc}(\mathsf{imsk}, [\mathbf{u}]_1)\end{array}\right\}$$

and

$$\left\{(\mathsf{impk},\mathsf{imsk},\mathsf{ict}) \middle| \begin{array}{l}(\mathsf{impk},\mathsf{imsk}) \leftarrow \mathsf{iSetup}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{priv}}) \\ \mathsf{ict} \leftarrow \mathsf{iSlotEnc}(\mathsf{impk}, [\mathbf{u}_{\mathsf{pub}}]_1)\end{array}\right\}$$

where $\mathbf{u}^\intercal = (\mathbf{u}_{\mathsf{pub}}^\intercal, \mathbf{0}^\intercal) \in \mathbb{Z}_p^{\mathfrak{s}}$. Slotted IPFE generalizes both secret-key and public-key IPFEs: we may obtain the former by setting $\mathfrak{s} = \mathfrak{s}_{\mathsf{priv}}$ and the latter by setting $\mathfrak{s} = \mathfrak{s}_{\mathsf{pub}}$.

Next, we define the adaptive function hiding property.

**Definition 2.16 (Function Hiding Slotted IPFE).** Let $(\mathsf{iSetup}, \mathsf{iKeyGen}, \mathsf{iEnc}, \mathsf{iDec}, \mathsf{iSlotEnc})$ be a slotted IPFE scheme as defined above. The scheme is function hiding if $\mathsf{Exp}^0_{\mathsf{FH}}$ is indistinguishable from $\mathsf{Exp}^1_{\mathsf{FH}}$ for all efficient adversary $\mathcal{A}$ where $\mathsf{Exp}^b_{\mathsf{FH}}$ for $b \in \{0,1\}$ is defined as follows:

1. **Setup**: Run the adversary $\mathcal{A}$ and obtain the disjoint index sets $\mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{priv}}$ from $\mathcal{A}$. Let $\mathfrak{s} = \mathfrak{s}_{\mathsf{pub}} \cup \mathfrak{s}_{\mathsf{priv}}$. Let $(\mathsf{impk}, \mathsf{imsk}) \leftarrow \mathsf{iSetup}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{priv}})$ and return impk to $\mathcal{A}$.

2. **Challenge**: Repeat the following for arbitrarily many rounds determined by $\mathcal{A}$: In each round, $\mathcal{A}$ has 2 options:

   - $\mathcal{A}$ chooses $\mathbf{v}_j^0, \mathbf{v}_j^1 \in \mathbb{Z}_p^{\mathfrak{s}}$ and submits $[\mathbf{v}_j^0]_2, [\mathbf{v}_j^1]_2$ for a secret key. Upon receiving this, compute $\mathsf{isk}_j \leftarrow \mathsf{iKeyGen}(\mathsf{imsk}, [\mathbf{v}_j^b]_2)$ and return this to $\mathcal{A}$.
   - $\mathcal{A}$ chooses $\mathbf{u}_i^0, \mathbf{u}_i^1 \in \mathbb{Z}_p^{\mathfrak{s}}$ and submits $[\mathbf{u}_i^0]_1, [\mathbf{u}_i^1]_1$ for a ciphertext. Upon receiving this, compute $\mathsf{ict}_i \leftarrow \mathsf{iEnc}(\mathsf{imsk}, [\mathbf{u}_i^b]_1)$ and return this to $\mathcal{A}$.

3. **Guess**: $\mathcal{A}$ outputs its guess $b'$.

The outcome of the experiment is defined as $b'$ if all the public components of the key queries are equal, i.e. $\mathbf{v}_j^0|_{\mathfrak{s}_{\mathsf{pub}}} = \mathbf{v}_j^1|_{\mathfrak{s}_{\mathsf{pub}}}$ for all $j$ and $\langle \mathbf{u}_i^0,\ \mathbf{v}_j^0\rangle = \langle \mathbf{u}_i^1,\ \mathbf{v}_j^1\rangle$ for all $i, j$.

We will also require the following lemma by [ALS16, Wee17, LV16, Lin17, BW19]:

**Lemma 2.17.** Let $\mathsf{BLGroupGen}$ be a group generator that outputs bilinear group $\mathcal{G}_{\mathsf{BL}} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, [1]_2)$ and $k \geq 1$ an integer constant. If $\mathsf{MDDH}_k$ holds in both $\mathbb{G}_1$ and $\mathbb{G}_2$, then there is an (adaptively) function-hiding slotted IPFE scheme on $\mathsf{BLGroupGen}$.

Note that the $\mathsf{MDDH}_k$ assumption on $\mathbb{G}_s$ ($s \in \{1,2\}$) says that a random group element $[\mathbf{r}]_s$ is indistinguishable from $[\mathbf{sA}]_s$ given $[\mathbf{A}]$, where $\mathbf{A} \leftarrow \mathbb{Z}_p^{k\times(k+1)}$, $\mathbf{r} \leftarrow \mathbb{Z}_p^{k+1}$, and $\mathbf{s} \leftarrow \mathbb{Z}_p^k$. The assumption is implied by the standard $k$-$\mathsf{LIN}$ assumption, which becomes progressively weaker as $k$ becomes larger.

# 3 KP-ABE for Unbounded Depth Circuits

## 3.1 Recall: Unbounded Homomorphic Evaluation Procedures

In this section, we recall the two procedures $\mathsf{UEvalC}$ and $\mathsf{UEvalCX}$ used in unbounded homomorphic evaluation as in [HLL23, Construction 2].

**Theorem 3.1.** Let $\lambda$ be sufficiently large and $q, \beta \in \mathbb{N}$ satisfy the constraints mentioned below. Let $C : \{0,1\}^L \to \{0,1\}$ and $\mathbf{x} \in \{0,1\}^L$. Suppose

$$\begin{array}{lll}
\mathbf{r} \leftarrow \mathcal{D}^n_{\mathbb{Z},\sigma_s}, & \mathbf{e}_{\mathsf{fhe}} \leftarrow \mathcal{D}^m_{\mathbb{Z},\sigma_{\mathsf{fhe}}}, & \mathbf{e}_{\mathsf{att}} \leftarrow \mathcal{D}^{m(L+1)}_{\mathbb{Z},\sigma_{\mathsf{att}}} \\
\mathrm{L_S} = m(n+1)^2 \lceil \log q \rceil^2, & \mathbf{e}_{\mathsf{circ}} \leftarrow \mathcal{D}^{m(\mathrm{L_S}+1)}_{\mathbb{Z},\sigma_{\mathsf{circ}}}, & \mathbf{s} = (\mathbf{r}^\intercal, -1)^\intercal, \\
\mathbf{R} \in \{0,1\}^{m \times m(n+1)\lceil \log q \rceil}, & \bar{\mathbf{A}}_{\mathsf{fhe}} \in \mathbb{Z}_q^{n \times m}, & \mathbf{A}_{\mathsf{att}} \leftarrow \mathbb{Z}_q^{(n+1)\times m(L+1)}, \\
\mathbf{A}_{\mathsf{circ}} \leftarrow \mathbb{Z}_q^{(n+1)\times m(\mathrm{L_S}+1)}, & \mathbf{S} = \begin{pmatrix} \bar{\mathbf{A}}_{\mathsf{fhe}} \\ \mathbf{r}^\intercal \bar{\mathbf{A}}_{\mathsf{fhe}} + \mathbf{e}^\intercal_{\mathsf{fhe}} \end{pmatrix} \mathbf{R} - \mathsf{bits}(\mathbf{s}) \otimes \mathbf{G} & \\
\mathbf{c}^\intercal_{\mathsf{att}} = \mathbf{s}^\intercal(\mathbf{A}_{\mathsf{att}} - (1, \mathbf{x}^\intercal) \otimes \mathbf{G}) + \mathbf{e}^\intercal_{\mathsf{att}}, & \mathbf{c}^\intercal_{\mathsf{circ}} = \mathbf{s}^\intercal(\mathbf{A}_{\mathsf{circ}} - (1, \mathsf{bits}(\mathbf{S})) \otimes \mathbf{G}) + \mathbf{e}^\intercal_{\mathsf{circ}} &
\end{array}$$

where $\|\mathbf{s}\| \le \beta, \|\mathbf{e}_{\mathsf{att}}\|, \|\mathbf{e}_{\mathsf{fhe}}\|, \|\mathbf{e}_{\mathsf{circ}}\| \le 2^{-\lambda}\beta, \beta/q = \mathsf{negl}(\lambda)$. Note that all the parameters except for $\mathbf{c}_{\mathsf{att}}$ are sampled independent of $C$ and $\mathbf{x}$. There are two efficient algorithms,

$$\mathsf{UEvalC}(\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, C) \to \mathbf{A}_C, \qquad \mathsf{UEvalCX}(\mathbf{A}_{\mathsf{att}}, \mathbf{c}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{circ}}, C, \mathbf{x}, \mathbf{S}) = \mathbf{c}^\intercal_{C,\mathbf{x}}$$

satisfying, with all but negligible probability $\mathbf{c}^\intercal_{C,\mathbf{x}} = \mathbf{s}^\intercal(\mathbf{A}_C - C(\mathbf{x})\mathbf{G}) + \mathbf{e}^\intercal_{C,\mathbf{x}}$ and $\|\mathbf{e}^\intercal_{C,\mathbf{x}}\| \le \beta$.

## 3.2 Construction

In this section, we construct our KP-ABE scheme $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ for a circuit family $\mathcal{C}_{\lambda, L=L(\lambda)} = \{C : \{0,1\}^L \to \{0,1\}\}$ of unbounded depth. For the construction, we use the bilinear group generator $\mathsf{BLGroupGen}$ (Section 2.2) and a slotted IPFE scheme $(\mathsf{iSetup}, \mathsf{iKeyGen}, \mathsf{iSlotEnc}, \mathsf{iDec})$ (Section 2.3.3), and the unbounded homomorphic evaluation procedures $\mathsf{UEvalC}$ and $\mathsf{UEvalCX}$ introduced in Theorem 3.1.

$\mathsf{Setup}(1^\lambda, 1^L)$. The setup algorithm does the following.

1. $\mathcal{G}_{\mathsf{BL}} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, [1]_2) \leftarrow \mathsf{BLGroupGen}(1^\lambda)$.
2. Fix a positive integer $M \in \mathbb{Z}$ such that $q = Mp$.
3. Sample $\mathbf{A}_{\mathsf{att}} \leftarrow \mathbb{Z}_q^{(n+1)\times m(L+1)}$, $\mathbf{A}_{\mathsf{circ}} \leftarrow \mathbb{Z}_q^{(n+1)\times m(\mathrm{L_S}+1)}$, $\mathbf{u} \leftarrow \mathbb{Z}_q^{(n+1)}$ where $\mathrm{L_S} = m(n+1)^2\lceil \log q \rceil^2$.
4. $(\mathsf{impk}, \mathsf{imsk}) \leftarrow \mathsf{iSetup}(1^\lambda, \{1, \ldots, n+2\}, \{n+3\})$.
5. Set and output $\mathsf{mpk} = (\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{u}, \mathcal{G}_{\mathsf{BL}}, \mathsf{impk}, M)$ and $\mathsf{msk} = \mathsf{imsk}$.

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, C)$. The key generation algorithm does the following.

1. Compute $\mathbf{A}_C = \mathsf{UEvalC}(\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, C) \in \mathbb{Z}_q^{(n+1)\times m}$.
2. Sample $\gamma \leftarrow \mathbb{Z}_p$ and compute

$$\mathsf{isk}_C = \mathsf{iKeyGen}\left(\mathsf{imsk}, \left[\left(\gamma \left\lfloor \frac{\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor^\intercal, \gamma, 0\right)^\intercal\right]_2\right).$$

Here, the expression $\left\lfloor \mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u})/M \right\rfloor$ is interpreted as follows. The numerator is an element of $\mathbb{Z}_q$, and since $M \mid q$, evaluating it using different integer representatives differs by a

multiple of $q/M = p$. Such differences only affect the exponent by a multiple of $p$. Since the values are used as exponents of group elements of order $p$, this difference has no effect. Hence, we omit writing $\bmod q$ explicitly in the numerator. We will use a similar abuse of notation throughout this section whenever no confusion arises.

3. Output $\mathsf{sk}_C = (\mathsf{isk}_C, [\gamma]_2)$.

$\mathsf{Enc}(\mathsf{mpk}, \mathbf{x} \in \{0,1\}^L, \mu \in \{0,1\})$. The encryption algorithm does the following.

1. Sample $\mathbf{r} \leftarrow \mathcal{D}^n_{\mathbb{Z},\sigma_\mathbf{s}}$ and set $\mathbf{s} = (\mathbf{r}^\intercal, -1)^\intercal \in \mathbb{Z}^{n+1}$.
2. Sample $\bar{\mathbf{A}}_{\mathsf{fhe}} \leftarrow \mathbb{Z}_q^{n\times m}$, $\mathbf{e}_{\mathsf{fhe}} \leftarrow \mathcal{D}^m_{\mathbb{Z},\sigma_{\mathsf{fhe}}}$, $\mathbf{R}_i \leftarrow \{0,1\}^{m\times m}$ for all $i \in [(n+1)\lceil \log q \rceil]$ and compute [GSW13] FHE encryption as follows.

$$\mathbf{A}_{\mathsf{fhe}} := \begin{pmatrix} \bar{\mathbf{A}}_{\mathsf{fhe}} \\ \mathbf{r}^\intercal \bar{\mathbf{A}}_{\mathsf{fhe}} + \mathbf{e}_{\mathsf{fhe}}^\intercal \end{pmatrix}, \quad \mathbf{S} = \mathbf{A}_{\mathsf{fhe}}\mathbf{R} - \mathsf{bits}(\mathbf{s}) \otimes \mathbf{G} \in \mathbb{Z}_q^{(n+1)\times m(n+1)\lceil \log q \rceil}$$

where $\mathbf{R} = (\mathbf{R}_1, \ldots, \mathbf{R}_{(n+1)\lceil \log q \rceil})$. Let $\mathrm{L_S} = m(n+1)^2 \lceil \log q \rceil^2$ be the bit length of $\mathbf{S}$.

3. Compute [BGG+14] ABE encoding as follows.

$$\mathbf{e}_{\mathsf{att}} \leftarrow \mathcal{D}^{m(L+1)}_{\mathbb{Z},\sigma_{\mathsf{att}}}, \quad \mathbf{c}_{\mathsf{att}}^\intercal := \mathbf{s}^\intercal(\mathbf{A}_{\mathsf{att}} - (1, \mathbf{x}^\intercal) \otimes \mathbf{G}) + \mathbf{e}_{\mathsf{att}}^\intercal \in \mathbb{Z}_q^{1\times m(L+1)}$$

$$\mathbf{e}_{\mathsf{circ}} \leftarrow \mathcal{D}^{m(\mathrm{L_S}+1)}_{\mathbb{Z},\sigma_{\mathsf{circ}}}, \quad \mathbf{c}_{\mathsf{circ}}^\intercal := \mathbf{s}^\intercal(\mathbf{A}_{\mathsf{circ}} - (1, \mathsf{bits}(\mathbf{S})) \otimes \mathbf{G}) + \mathbf{e}_{\mathsf{circ}}^\intercal \in \mathbb{Z}_q^{1\times m(\mathrm{L_S}+1)}$$

4. Compute $\mathsf{ict} \leftarrow \mathsf{iSlotEnc}(\mathsf{impk}, [(\mathbf{s}^\intercal, \mu \lfloor p/2 \rceil)^\intercal]_1)$.
5. Output $\mathsf{ct} = (\mathbf{S}, \mathbf{c}_{\mathsf{att}}, \mathbf{c}_{\mathsf{circ}}, \mathsf{ict})$.

$\mathsf{Dec}(\mathsf{mpk}, \mathsf{ct}, \mathbf{x}, \mathsf{sk}_C, C)$. The decryption algorithm does the following.

1. Compute $\mathbf{A}_C = \mathsf{UEvalC}(\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, C)$ and $\mathbf{c}_{C,\mathbf{x}}^\intercal = \mathsf{UEvalCX}(\mathbf{A}_{\mathsf{att}}, \mathbf{c}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{c}_{\mathsf{circ}}, C, \mathbf{x}, \mathbf{S}) \in \mathbb{Z}_q^{1\times m}$.
2. Compute $\tilde{z} = \left\lfloor \frac{\mathbf{c}_{C,\mathbf{x}}^\intercal \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor \bmod p$ and $[\gamma]_T = e([1]_1, [\gamma]_2)$.
3. Compute $[y]_T = \mathsf{iDec}(\mathsf{isk}_C, \mathsf{ict})$ and $[z]_T = [y]_T \cdot [\gamma]_T^{-\tilde{z}}$.
4. If $[\gamma]_T^\eta = [z]_T$ for $\eta \in [-B, B]$, set $\mu = 0$. Else set $\mu = 1$.

**Constraints on Parameters.** We first list the constraints on the parameters.

1. $q = Mp$.
2. From Theorem 3.1, $\|\mathbf{s}\| \le \beta$; $\|\mathbf{e}_{\mathsf{att}}\|, \|\mathbf{e}_{\mathsf{fhe}}\|, \|\mathbf{e}_{\mathsf{circ}}\| \le 2^{-\lambda}\beta$; $\beta/q = \mathsf{negl}(\lambda)$ and $\|\mathbf{e}_{C,\mathbf{x}}^\intercal\| \le \beta$.
3. For correctness, $m\beta < M$, $(n+1)\sqrt{\lambda}\|\sigma_\mathbf{s}\| + 4 \le B$ and $B = \mathrm{poly}(\lambda)$.

**Parameters.** We now set the concrete parameters.
$M = 2^{6\lambda}$, $p \in (2^{2\lambda}, 2^{2\lambda+1}]$, $q = Mp$, $n = \mathrm{poly}(\lambda)$, $m = 3(n+1)\lceil \log q \rceil$,
$\sigma_\mathbf{s} = \sigma_{\mathsf{fhe}} = \sigma_{\mathsf{att}} = \sigma_{\mathsf{circ}} = \mathrm{poly}(\lambda)$, $\beta = 2^{4\lambda}$, $B = \mathrm{poly}(\lambda)$

**Correctness.** We now prove the correctness of our scheme below as per Definition 2.6.

- First note that from Theorem 3.1, we have $\mathbf{c}_{C,\mathbf{x}}^\intercal = \mathbf{s}^\intercal(\mathbf{A}_C - C(\mathbf{x})\mathbf{G}) + \mathbf{e}_{C,\mathbf{x}}^\intercal$.

- Now,

$$\begin{aligned}
\tilde{z} &= \left\lfloor \frac{\mathbf{c}_{C,\mathbf{x}}^{\mathsf{T}} \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor \bmod p \\
&= \left\lfloor \frac{(\mathbf{s}^{\mathsf{T}}(\mathbf{A}_C - C(\mathbf{x})\mathbf{G}) + \mathbf{e}_{C,\mathbf{x}}^{\mathsf{T}})\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor \bmod p \\
&= \left( \left\lfloor \frac{\mathbf{s}^{\mathsf{T}}(\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u}) - C(\mathbf{x})\mathbf{u})}{M} \right\rfloor + \left\lfloor \frac{\mathbf{e}_{C,\mathbf{x}}^{\mathsf{T}} \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor + e \right) \bmod p \\
&= \left( \left\lfloor \frac{\mathbf{s}^{\mathsf{T}}(\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u}) - C(\mathbf{x})\mathbf{u})}{M} \right\rfloor + e' + e \right) \bmod p \qquad (3) \\
&= \left( \mathbf{s}^{\mathsf{T}} \left\lfloor \frac{\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u}) - C(\mathbf{x})\mathbf{u}}{M} \right\rfloor + e' + e + e_{\mathbf{s}} \right) \bmod p
\end{aligned}$$

  holds, where $e$, $e'$, and $e_{\mathbf{s}}$ are defined and bounded as

$$e \stackrel{\text{def}}{=} \left( \left\lfloor \frac{(\mathbf{s}^{\mathsf{T}}(\mathbf{A}_C - C(\mathbf{x})\mathbf{G}) + \mathbf{e}_{C,\mathbf{x}}^{\mathsf{T}})\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor - \left\lfloor \frac{\mathbf{s}^{\mathsf{T}}(\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u}) - C(\mathbf{x})\mathbf{u})}{M} \right\rfloor - \left\lfloor \frac{\mathbf{e}_{C,\mathbf{x}}^{\mathsf{T}} \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor \right) \bmod p,$$

  where we have $\|e\| < 2$ by Lemma 2.1, $e' \stackrel{\text{def}}{=} \left\lfloor \frac{\mathbf{e}_{C,\mathbf{x}}^{\mathsf{T}} \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor \bmod p$, where we have $\|e'\| < 2$ by $\|\mathbf{e}_{C,\mathbf{x}}^{\mathsf{T}} \mathbf{G}^{-1}(\mathbf{u})\| \leq m\|\mathbf{e}_{C,\mathbf{x}}\| \leq m\beta < M$. Here, the second inequality follows from Theorem 3.1, since the input and the circuits are chosen independently of $\mathsf{mpk}$ and the requirement of the theorem that all the parameters except $\mathbf{c}_{\mathsf{att}}$ be independent of $C$ and $\mathbf{x}$ is satisfied.

$$\begin{aligned}
e_{\mathbf{s}} &\stackrel{\text{def}}{=} \left( \left\lfloor \frac{\mathbf{s}^{\mathsf{T}}(\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u}) - C(\mathbf{x})\mathbf{u})}{M} \right\rfloor - \mathbf{s}^{\mathsf{T}} \left\lfloor \frac{(\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u}) - C(\mathbf{x})\mathbf{u})}{M} \right\rfloor \right) \bmod p \\
&= \left\lfloor \mathbf{s}^{\mathsf{T}} \underbrace{\left( \frac{\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u}) - C(\mathbf{x})\mathbf{u}}{M} - \left\lfloor \frac{\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u}) - C(\mathbf{x})\mathbf{u}}{M} \right\rfloor \right)}_{<1} \right\rfloor \bmod p \qquad (4)
\end{aligned}$$

  $\implies \|e_{\mathbf{s}}\| \leq (n+1)\|\mathbf{s}\|$.

- From the correctness of the slotted IPFE, we have

$$[y]_T = \left[ \gamma \mathbf{s}^{\mathsf{T}} \left\lfloor \frac{\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor + \gamma\mu \lfloor p/2 \rceil \right]_T.$$

- Now, noting that $C(\mathbf{x}) = 0$,[10]

$$\begin{aligned}
[z]_T &= [y]_T \cdot [\gamma]_T^{-\tilde{z}} = [y - \gamma\tilde{z}]_T \\
&= \left[ \gamma \mathbf{s}^{\mathsf{T}} \left\lfloor \frac{\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor + \gamma\mu \lfloor p/2 \rceil - \gamma \mathbf{s}^{\mathsf{T}} \left\lfloor \frac{\mathbf{A}_C \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor - \gamma(e' + e + e_{\mathbf{s}}) \right]_T \\
&= \left[ \gamma\mu \lfloor p/2 \rceil - \gamma(e' + e + e_{\mathbf{s}}) \right]_T
\end{aligned}$$

- From our parameter setting, we have $\| - e' - e - e_{\mathbf{s}}\| \leq \sqrt{\lambda}(n+1)\sigma_{\mathbf{s}} + 4 \leq \mathrm{poly}(\lambda) = B$. The correctness, therefore, follows. Note that the decryption runs in polynomial time as $B = \mathrm{poly}(\lambda)$.

[10] Since the above equations will be used for the case $C(\mathbf{x}) = 1$, we intentionally defer substituting $C(\mathbf{x}) = 0$ until this point.

### 3.3 Security Proof

In this section, we state the security of our ABE scheme.

**Theorem 3.2.** If $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma_{\mathbf{s}},\sigma}$ assumption (Assumption 2.3) holds and Slotted IPFE satisfies slot-mode correctness (Definition 2.15) and function hiding (Definition 2.16), the construction as in Section 3.2 is VerSel-IND-secure (Definition 2.8) in the RR model.

*Proof of Theorem 3.2.* Let us fix a PPT adversary $\mathcal{A}$. To prove security, we consider the following sequence of games. Let $\mathsf{E}_x$ denote the event that $\mathcal{A}$ outputs a correct guess for $b$ in $\mathsf{Game}_x$. All transitions are established in the standard model, except for the transition from $\mathsf{Game}_3$ to $\mathsf{Game}_4$, which is proved in the RR model for bilinear groups. Consequently, the theorem as a whole establishes security in the RR model.

$\mathsf{Game}_0$. This is the same as the real world, where we encrypt the message $b$ sampled as $b \leftarrow \{0,1\}$. By definition, the advantage of $\mathcal{A}$ against the scheme is $|\Pr[\mathsf{E}_0] - \frac{1}{2}|$.

$\mathsf{Game}_1$. This is the same as the previous hybrid except that in this hybrid, we change the way ict is being computed. Instead of computing it as $\mathsf{ict} \leftarrow \mathsf{iSlotEnc}\left(\mathsf{impk}, [(\mathbf{s}^\intercal, b\lfloor p/2 \rceil)^\intercal]_1\right)$, it is computed as $\mathsf{ict} \leftarrow \mathsf{iEnc}(\mathsf{imsk}, [(\mathbf{s}^\intercal, b\lfloor p/2 \rceil, 0)^\intercal]_1)$. By the slot-mode correctness of slotted IPFE, the distribution of ict is unchanged. Therefore, we have $\Pr[\mathsf{E}_0] = \Pr[\mathsf{E}_1]$.

$\mathsf{Game}_2$. This is the same as the previous hybrid except that in this hybrid, we change the vector being encoded inside ict and $\mathsf{isk}_{C_i}$ i.e. for all $i \in [Q]$, we compute

$$\mathsf{ict} \leftarrow \mathsf{iEnc}\left(\mathsf{imsk}, \left[\left((0^{n+1})^\intercal, 0, 1\right)^\intercal\right]_1\right) \quad \&$$

$$\mathsf{isk}_{C_i} = \mathsf{iKeyGen}\left(\mathsf{imsk}, \left[\left(\gamma_i \left\lfloor \frac{\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor^\intercal, \gamma_i, \gamma_i \mathbf{s}^\intercal \left\lfloor \frac{\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor + \gamma_i b \lfloor p/2 \rceil\right)^\intercal\right]_2\right).$$

The indistinguishability of $\mathsf{Game}_1$ and $\mathsf{Game}_2$ follows from the function-hiding property of slotted IPFE, since the decryption outcome of the ciphertext under any secret key remains unchanged from the previous game, and the vectors encoded in the public slots of the secret keys are also unchanged. Hence, we have $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathsf{negl}(\lambda)$.

$\mathsf{Game}_3$. This is the same as the previous hybrid, except that in this hybrid, we change the computation of the third component of the vector encoded in $\mathsf{isk}_{C_i}$ for all $i \in [Q]$ from $\gamma_i \mathbf{s}^\intercal \left\lfloor \frac{\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor + \gamma_i b \lfloor p/2 \rceil$ to

$$\gamma_i \left( \left\lfloor \frac{\mathbf{c}^\intercal_{C_i,\mathbf{x}^*}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor - \mathsf{err}_i + \left\lfloor \frac{\mathbf{s}^\intercal \mathbf{u}}{M} \right\rfloor \right) + \gamma_i b \lfloor p/2 \rceil$$

where $\mathsf{err}_i$ is defined as below.

$$\begin{aligned}
\left\lfloor \frac{\mathbf{c}^\intercal_{C_i,\mathbf{x}^*}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor &= \left\lfloor \frac{\mathbf{s}^\intercal \mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u}) - \mathbf{s}^\intercal \mathbf{u}}{M} \right\rfloor + \mathsf{err}_{i,0} \bmod p \\
&= \left\lfloor \frac{\mathbf{s}^\intercal \mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor - \left\lfloor \frac{\mathbf{s}^\intercal \mathbf{u}}{M} \right\rfloor + \mathsf{err}_{i,0} + \mathsf{err}_{i,1} \quad \bmod p \\
&= \mathbf{s}^\intercal \left\lfloor \frac{\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor - \left\lfloor \frac{\mathbf{s}^\intercal \mathbf{u}}{M} \right\rfloor + \mathsf{err}_{i,0} + \mathsf{err}_{i,1} + \mathsf{err}_{i,2} \quad \bmod p.
\end{aligned}$$

where

$$\begin{aligned}
\mathsf{err}_i &= \mathsf{err}_{i,0} + \mathsf{err}_{i,1} + \mathsf{err}_{i,2} \qquad (5)\\
\mathsf{err}_{i,0} &= \left\lfloor \frac{\mathbf{c}^{\mathsf{T}}_{C_i,\mathbf{x}^*}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor - \left\lfloor \frac{\mathbf{s}^{\mathsf{T}}\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u}) - \mathbf{s}^{\mathsf{T}}\mathbf{u}}{M} \right\rfloor \mod p\\
\mathsf{err}_{i,1} &= \left\lfloor \frac{\mathbf{s}^{\mathsf{T}}\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u}) - \mathbf{s}^{\mathsf{T}}\mathbf{u}}{M} \right\rfloor - \left( \left\lfloor \frac{\mathbf{s}^{\mathsf{T}}\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor - \left\lfloor \frac{\mathbf{s}^{\mathsf{T}}\mathbf{u}}{M} \right\rfloor \right) \mod p\\
\mathsf{err}_{i,2} &= \left\lfloor \frac{\mathbf{s}^{\mathsf{T}}\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor - \mathbf{s}^{\mathsf{T}} \left\lfloor \frac{\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor \mod p.
\end{aligned}$$

This is only a conceptual change where we rearrange the terms. Hence, we have $\Pr[\mathsf{E}_2] = \Pr[\mathsf{E}_3]$.

$\mathsf{Game}_4$. This is the same as the previous hybrid, except that in this hybrid, we change the computation of the third component of the vector encoded in $\mathsf{isk}_{C_i}$ for all $i \in [Q]$ from

$$\gamma_i \left( \left\lfloor \frac{\mathbf{c}^{\mathsf{T}}_{C_i,\mathbf{x}^*}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor - \mathsf{err}_i + \left\lfloor \frac{\mathbf{s}^{\mathsf{T}}\mathbf{u}}{M} \right\rfloor \right) + \gamma_i b \lfloor p/2 \rceil$$

$$\text{to} \qquad \Delta_i := \gamma_i \left\lfloor \frac{\mathbf{c}^{\mathsf{T}}_{C_i,\mathbf{x}^*}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor + t_i + \gamma_i b \lfloor p/2 \rceil$$

where $t_i \leftarrow \mathbb{Z}_p$. We prove that $|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_4]| = \mathsf{negl}(\lambda)$ in Lemma 3.4.

$\mathsf{Game}_5$. This is the same as the previous hybrid, except that in this hybrid, we replace the computation of the third component of the vector encoded in $\mathsf{isk}_{C_i}$ for all $i \in [Q]$ from $\Delta_i$ as computed in $\mathsf{Game}_4$ to $\Delta_i \leftarrow \mathbb{Z}_p$. The hybrids $\mathsf{Game}_4$ and $\mathsf{Game}_5$ are equivalent because for $[t_i]_2$ where $t_i \leftarrow \mathbb{Z}_p$, $[\Delta_i]_2$ where $\Delta_i := \gamma_i \left\lfloor \frac{\mathbf{c}^{\mathsf{T}}_{C_i,\mathbf{x}^*}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor + t_i + \gamma_i b \lfloor p/2 \rceil$ is also distributed uniformly in $\mathbb{Z}_p$. Moreover, this $t_i$ is not used anywhere else. Hence, we have $\Pr[\mathsf{E}_4] = \Pr[\mathsf{E}_5]$.

In $\mathsf{Game}_5$, $b$ is information theoretically hidden. Hence, we have $\Pr[\mathsf{E}_5] = 1/2$. Therefore, we have $|\Pr[\mathsf{E}_0] - 1/2| \leq \mathsf{negl}(\lambda)$ by the triangle inequality. □

It remains to show that $|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_4]| = \mathsf{negl}(\lambda)$. To this end, we introduce the following assumption. We first prove that $\mathsf{Game}_3$ and $\mathsf{Game}_4$ are indistinguishable under the assumption, and then show that the assumption holds in the RR model.

### Doubly Circular Assumption

*Assumption* 3.3 (Doubly Circular assumption). Let $n, m, \delta, q, L', M, \sigma, \sigma_{\mathbf{s}}, \sigma_{\mathsf{fhe}} \in \mathbb{N}$ satisfy $q = Mp$, $\sigma \leq 2^{-\lambda}M$, and $\delta(\lambda) = \mathrm{poly}(\lambda)$. The Doubly Circular assumption states that for every PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ there exists a negligible function $\mathsf{negl}(\cdot)$ such that

$$\left| \Pr\left[ \mathcal{A}_1\left( \{[\gamma_i]_2, [t_{i,\beta}]_2\}_{i\in[Q]}, \mathsf{st} \right) = \beta \;\middle|\; (\mathsf{st}, \{f_i\}_{i\in[Q]}) \leftarrow \mathcal{A}_0(\mathsf{pp}, \mathbf{A}, \mathbf{u}, \mathbf{b}, \mathbf{S}) \right] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda),$$

where the variables are generated as follows. Let $\mathcal{G}_{\mathsf{BL}} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, [1]_2) \leftarrow \mathsf{BLGroupGen}(1^\lambda)$. The public parameters, challenge values, and auxiliary information are then sampled according to the distribution specified below.

$$\mathsf{pp} = (\mathcal{G}_{\mathsf{BL}}, n, m, \delta, q, L', M, \sigma, \sigma_{\mathbf{s}}, \sigma_{\mathsf{fhe}}) \quad \mathbf{A} \leftarrow \mathbb{Z}_q^{(n+1)\times L'}, \quad \bar{\mathbf{A}}_{\mathsf{fhe}} \leftarrow \mathbb{Z}_q^{n\times m},$$
$$\mathbf{R} \leftarrow \{0,1\}^{m\times m(n+1)\lceil \log q\rceil}, \quad \mathbf{u} \leftarrow \mathbb{Z}_q^{n+1}, \quad \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{L'},$$
$$\mathbf{e}_{\mathsf{fhe}} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma_{\mathsf{fhe}}}^{m}, \quad \mathbf{r} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma_{\mathbf{s}}}^{n}, \quad \mathbf{s} = (\mathbf{r}^\intercal, -1)^\intercal,$$
$$\mathbf{A}_{\mathsf{fhe}} = \begin{pmatrix} \bar{\mathbf{A}}_{\mathsf{fhe}} \\ \mathbf{r}^\intercal \bar{\mathbf{A}}_{\mathsf{fhe}} + \mathbf{e}_{\mathsf{fhe}}^\intercal \end{pmatrix}, \quad \{\gamma_i \leftarrow \mathbb{Z}_p\}_{i\in[Q]}, \quad v = \lfloor \mathbf{s}^\intercal \mathbf{u}/M \rfloor,$$
$$\beta \leftarrow \{0,1\} \quad \{t_{i,0} = \gamma_i(v + f_i(\mathbf{s}))\}_{i\in[Q]}, \quad \{t_{i,1} \leftarrow \mathbb{Z}_p\}_{i\in[Q]}$$
$$\mathbf{b}^\intercal = \mathbf{s}^\intercal \mathbf{A} + \mathbf{e}^\intercal, \quad \mathbf{S} = \underbrace{\mathbf{A}_{\mathsf{fhe}}\mathbf{R} - \mathsf{bits}(\mathbf{s}) \otimes \mathbf{G}}_{\texttt{circular encoding}},$$

and for every $i \in [Q]$, $f_i$ is a function such that $f_i : \mathbb{Z}_q^{n+1} \to [-\delta, \delta]$ and $f_i(\mathbf{s})$ means $f_i(\mathbf{s} \bmod q)$ in the above.

***Continuing the proof of Theorem 3.2:*** Assuming the above assumption, we can show that the adversary cannot distinguish $\mathsf{Game}_3$ from $\mathsf{Game}_4$ in the standard model.

**Lemma 3.4.** Assuming the doubly circular assumption (Assumption 3.3) holds for parameters defined as in Section 3.2 and $\delta > 2\sqrt{\lambda} n \sigma_{\mathbf{s}}$, we have $|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_4]| = \mathsf{negl}(\lambda)$.

*Proof.* To prove the lemma, we prove that if there exists an adversary $\mathcal{A}$ that can distinguish $\mathsf{Game}_3$ from $\mathsf{Game}_4$, then we can use $\mathcal{A}$ to create $\mathcal{B} = (\mathcal{B}_0, \mathcal{B}_1)$ that can break the doubly circular assumption. The reduction $\mathcal{B} = (\mathcal{B}_0, \mathcal{B}_1)$ does the following.

1. $\mathcal{B}_0$ gets input $1^\lambda, \mathbf{x}^*, \{C_i\}_{i\in[Q]}$ from $\mathcal{A}$.
2. $\mathcal{B}$ prepares response for $\mathcal{A}$ as follows.
   (a) **Setup Phase**. $\mathcal{B}_0$ on input $(\mathsf{pp}, \mathbf{A}, \mathbf{u}, \mathbf{b}, \mathbf{S})$ from its own challenger, does the following:
      i. Parse $\mathsf{pp} = (\mathcal{G}_{\mathsf{BL}}, n, m, \delta, q, L', M, \sigma, \sigma_{\mathbf{s}}, \sigma_{\mathsf{fhe}})$ and $\mathbf{A} = (\tilde{\mathbf{A}}_{\mathsf{att}}, \tilde{\mathbf{A}}_{\mathsf{circ}})$, where $\tilde{\mathbf{A}}_{\mathsf{att}} \in \mathbb{Z}_q^{(n+1)\times m(L+1)}$ and $\tilde{\mathbf{A}}_{\mathsf{circ}} \in \mathbb{Z}_q^{(n+1)\times m(\mathrm{L_S}+1)}$. Here $L' = m(L+1) + m(\mathrm{L_S}+1)$.
      ii. Set $\mathbf{A}_{\mathsf{att}} = \tilde{\mathbf{A}}_{\mathsf{att}} + (1, (\mathbf{x}^*)^\intercal) \otimes \mathbf{G}$ and $\mathbf{A}_{\mathsf{circ}} = \tilde{\mathbf{A}}_{\mathsf{circ}} + (1, \mathsf{bits}(\mathbf{S})) \otimes \mathbf{G}$.
      iii. Run $\mathsf{iSetup}(1^\lambda, \{1, \ldots, n+2\}, \{n+3\}) \to (\mathsf{impk}, \mathsf{imsk})$.
      iv. Set $\mathsf{mpk} = (\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{u}, \mathcal{G}_{\mathsf{BL}}, \mathsf{impk}, M)$ and $\mathsf{msk} = \mathsf{imsk}$.
   (b) **Challenge Phase**. To simulate the ciphertext for $\mathcal{A}$, $\mathcal{B}_0$ does the following:
      i. Parse $\mathbf{b}^\intercal = (\mathbf{b}_{\mathsf{att}}^\intercal \in \mathbb{Z}_q^{1\times m(L+1)}, \mathbf{b}_{\mathsf{circ}}^\intercal \in \mathbb{Z}_q^{1\times m(\mathrm{L_S}+1)})$.
      ii. Compute $\mathsf{ict} \leftarrow \mathsf{iEnc}\left(\mathsf{imsk}, \left[\left((0^{n+1})^\intercal, 0, 1\right)^\intercal\right]_1\right)$.
      iii. Set $\mathsf{ct} = (\mathbf{S}, \mathbf{b}_{\mathsf{att}}, \mathbf{b}_{\mathsf{circ}}, \mathsf{ict})$.
   (c) **Key Query Phase**.
      i. $\mathcal{B}_0$ does the following.
         A. For all $i \in [Q]$, compute $\mathbf{A}_{C_i} = \mathsf{UEvalC}(\mathbf{A}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, C_i)$.
         B. Compute $\mathbf{c}_{C_i,\mathbf{x}^*}^\intercal = \mathsf{UEvalCX}(\mathbf{A}_{\mathsf{att}}, \mathbf{b}_{\mathsf{att}}, \mathbf{A}_{\mathsf{circ}}, \mathbf{b}_{\mathsf{circ}}, C_i, \mathbf{x}^*, \mathbf{S}) \in \mathbb{Z}_q^{1\times m}$.
         C. For all $i \in [Q]$, set the functions $f_i : \mathbb{Z}_q^{n+1} \to [-\delta, \delta]$ as

$$f_i(\mathbf{w}) := \mathsf{Truncate}_\delta\left(\mathbf{w}^\intercal \left\lfloor \frac{\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor - \left\lfloor \frac{\mathbf{w}^\intercal \mathbf{u}}{M} \right\rfloor - \left\lfloor \frac{\mathbf{c}_{C_i,\mathbf{x}^*}^\intercal \mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor \bmod p\right),$$

   where $\mathsf{Truncate}_\delta : \mathbb{Z}_p \to [-\delta, \delta]$ is a function that takes as input an element of $\mathbb{Z}_p$ in its centered representation and outputs the same value if it lies in $[-\delta, \delta]$, and outputs 0 otherwise.

D. It then gives $(\{f_i\}_{i\in[Q]}, \mathsf{st})$ where $\mathsf{st}$ consists of $\mathsf{imsk}$ and all the public information available to the adversary. Clearly, each of $f_i$ is legitimate query with the range being in $[-\delta, \delta]$.

ii. $\mathcal{B}_1$ on input $\left(\{[\gamma_i]_2, [t_i]_2\}_{i\in[Q]}, \mathsf{st}\right)$ from its own challenger, it first samples $b \leftarrow \{0,1\}$. It then does the following for all $i \in [Q]$:

A. Compute $[\gamma_i]_2^{\left\lfloor \frac{\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor}$, $[\gamma_i]_2^{\left\lfloor \frac{\mathbf{c}_{C_i,\mathbf{x}^*}^\mathsf{T}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor}$, and

$$[\Delta_i]_2 := [\gamma_i]_2^{\left\lfloor \frac{\mathbf{c}_{C_i,\mathbf{x}^*}^\mathsf{T}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor} \times [t_i]_2 \times [\gamma_i]_2^{b\lfloor p/2 \rceil} = \left[\gamma_i \left\lfloor \frac{\mathbf{c}_{C_i,\mathbf{x}^*}^\mathsf{T}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor + t_i + \gamma_i b \lfloor p/2 \rceil\right]_2.$$

B. Compute $\mathsf{isk}_{C_i} = \mathsf{iKeyGen}\left(\mathsf{imsk}, \left[\left(\gamma_i \left\lfloor \frac{\mathbf{A}_{C_i}\mathbf{G}^{-1}(\mathbf{u})}{M} \right\rfloor^\mathsf{T}, \gamma_i, \Delta_i\right)^\mathsf{T}\right]_2\right)$.

3. $\mathcal{B}$ gives $(\mathsf{mpk}, \mathsf{ct}, \{\mathsf{isk}_{C_i}, [\gamma_i]_2\}_{i\in[Q]})$ to $\mathcal{A}$.

4. Finally, $\mathcal{A}$ outputs the guess $b'$ for $b$. $\mathcal{B}$ outputs 1 if $b = b'$ and outputs 0 otherwise.

We then argue that that if $\mathcal{B}$'s challenger used $t_i = \gamma_i(\lfloor \mathbf{s}^\mathsf{T}\mathbf{u}/M \rfloor + f_i(\mathbf{s}))$, then $\mathcal{B}$ simulated $\mathsf{Game}_3$, else it simulated $\mathsf{Game}_4$. To see this, it suffices to show that w.o.p.,

$$f_i(\mathbf{s}) = -\mathsf{err}_i$$

hold, where $\mathsf{err}_i$ is as defined in equation (5). One can easily see that $f_i(\mathbf{s}) = \mathsf{Truncate}_\delta(-\mathsf{err}_i)$. Therefore, it suffices to show that $\mathsf{err}_i \in [-\delta, \delta]$ so that the truncation has no effect. Recall that $\mathsf{err}_i = \mathsf{err}_{i,0} + \mathsf{err}_{i,1} + \mathsf{err}_{i,2}$. By Equation (3) with $C(\mathbf{x}) = 1$, we have $\|\mathsf{err}_{i,0}\| < 4$ except for negligible probability. Note that Equation (3) requires $C$ and $\mathbf{x}$ to be chosen independently of the setup randomness. This condition holds here because the very selective adversary commits to both before setup. By Lemma 2.1, we have $\|\mathsf{err}_{i,1}\| < 2$. Furthermore, Equation (4) with $C(\mathbf{x}) = 1$ implies $\|\mathsf{err}_{i,2}\| \le \sqrt{\lambda}(n+1)\sigma_\mathbf{s}$. Combining these bounds, we obtain $\|\mathsf{err}_i\| \le \|\mathsf{err}_{i,0}\| + \|\mathsf{err}_{i,1}\| + \|\mathsf{err}_{i,2}\| < 6 + \sqrt{\lambda}(n+1)\sigma_\mathbf{s} \le \delta$. This completes the proof of Lemma 3.4. □

It remains to prove Lemma 3.5, which asserts that the doubly circular assumption holds in the RR model. Note that this is the only part in the proof of Theorem 3.2 where we need the RR model. Before stating the lemma, we recall the notation for the RR model.

**Notation.** We recall here the notation used for the RR model. Let $L_j : \mathbb{Z}_p \to S_j$ for $j \in \{1, 2, T\}$ be a random injection. We will think of $L_j(x)$ as corresponding to $[x]_j$. Here, $S_j \subseteq \{0,1\}^*$ for $j \in \{1,2,T\}$, is a set of strings of cardinality at least $p$. We also recall that an algorithm $A$ that has an oracle access to the group oracles is denoted by $A^{\mathcal{G}_{\mathsf{RR}}}$. Please refer to Section 2.2.1 for further notational details.

**Lemma 3.5.** Assuming $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma_\mathbf{s},\sigma}$ assumption (Assumption 2.3), for any PPT adversary $\mathcal{A}^{\mathcal{G}_{\mathsf{RR}}} = (\mathcal{A}_0^{\mathcal{G}_{\mathsf{RR}}}, \mathcal{A}_1^{\mathcal{G}_{\mathsf{RR}}})$ and for any polynomial $\delta(\lambda) = \mathrm{poly}(\lambda)$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that:

$$\left|\Pr\left[\mathcal{A}_1^{\mathcal{G}_{\mathsf{RR}}}\left(\{L_2(\gamma_i), L_2(t_{i,\beta})\}_{i\in[Q]}, \mathsf{st}\right) = \beta \;\middle|\; (\mathsf{st}, \{f_i^{\mathcal{G}_{\mathsf{RR}}}\}_{i\in[Q]}) \leftarrow \mathcal{A}_0^{\mathcal{G}_{\mathsf{RR}}}(\mathsf{pp}, \mathbf{A}, \mathbf{u}, \mathbf{b}, \mathbf{S})\right] - \frac{1}{2}\right| \le \mathsf{negl}(\lambda),$$

where $\mathsf{pp}, \mathbf{A}, \mathbf{u}, \mathbf{b}, \mathbf{S}, \gamma_i, \beta, t_{i,\beta}$ are chosen as in Assumption 3.3 and for all $i \in [Q]$, $f_i^{\mathcal{G}_{\mathsf{RR}}}$ is a function that has an oracle access to $\mathcal{G}_{\mathsf{RR}}$ such that $f_i^{\mathcal{G}_{\mathsf{RR}}} : \mathbb{Z}_q^{n+1} \to [-\delta, \delta]$.

**Discussion on the formulation.** We formulate the Doubly Circular assumption in the RR model so that the functions $f_i$ chosen by the adversary may access the group oracles. One could alternatively consider a weaker formulation where the functions $f_i$ do not have such access. The weaker formulation already suffices for proving Theorem 3.2, since the reduction in Lemma 3.4 only uses $f_i$ that does not have access to the group oracles. However, our motivation is not merely to complete the proof of Theorem 3.2. Rather, we view the Doubly Circular assumption as an independent assumption of potential interest, and therefore seek to provide the strongest possible evidence for its plausibility. By allowing oracle access, we capture a significantly broader class of generic attacks, including attacks that explicitly exploit group encodings. Consequently, proving security in this stronger formulation provides stronger evidence for the assumption itself.

*Proof of Lemma 3.5.* Let $\mathcal{A}^{\mathcal{G}_{\mathsf{RR}}} = (\mathcal{A}_0^{\mathcal{G}_{\mathsf{RR}}}, \mathcal{A}_1^{\mathcal{G}_{\mathsf{RR}}})$ be the PPT adversary distinguishing the two distributions. Let us denote the event that $\mathcal{A}_1^{\mathcal{G}_{\mathsf{RR}}}$ outputs 1 at the end of $\mathsf{Game}_{\mathsf{x}}$ as $\mathsf{E}_{\mathsf{x}}$. We denote the number of all the labels given to either $\mathcal{A}^{\mathcal{G}_{\mathsf{RR}}}$ or $\{f_i^{\mathcal{G}_{\mathsf{RR}}}\}_i$ by $Q_{\mathsf{lab}}$. From here on, we drop the superscript $\mathcal{G}_{\mathsf{RR}}$ from $\mathcal{A}$ and $f_i$ for notational simplicity.

$\mathsf{Game}_0$**.** This is the same as the ideal world with $\beta = 1$. Namely, the labels $L_2(\gamma_i)$ and $L_2(t_i)$ corresponding to $\gamma_i \leftarrow \mathbb{Z}_p$ and $t_i \leftarrow \mathbb{Z}_p$ for all $i \in [Q]$ are given to the adversary. Here, we drop the challenge bit $\beta = 1$ from $t_{i,\beta}$ for the simplicity of the notation.

$\mathsf{Game}_1$**.** This is the same as $\mathsf{Game}_0$ except that in this game, we switch to the symbolic group model and replace $\gamma_i$ and $t_i$ with formal variables $\Gamma_i$ and $T_i$ for all $i \in [Q]$. Note that all the labels given to $\mathcal{A}_1$ during the game correspond to a polynomial that is represented as

$$g_{\mathbf{x}}(\Gamma_1, \ldots, \Gamma_Q, T_1, \ldots, T_Q) := x_0 + \sum_{i\in[Q]} x_i T_i + \sum_{i\in[Q]} \tilde{x}_i \Gamma_i \in \mathbb{Z}_p[\Gamma_1, \ldots, \Gamma_Q, T_1, \ldots, T_Q] \quad (6)$$

where challenger can extract the vector $\mathbf{x} = (x_0, x_1, \ldots, \tilde{x}_1, \ldots, \tilde{x}_Q) \in \mathbb{Z}_p^{2Q+1}$ by keeping track of the group operations performed by $\mathcal{A}_1$.

This game differs from $\mathsf{Game}_0$ only when $\mathcal{A}_1$ obtains two labels $L_j(g_{\mathbf{x}})$ and $L_j(g_{\mathbf{x}'})$ for polynomials $g_{\mathbf{x}}, g_{\mathbf{x}'} \in \mathbb{Z}_p[\Gamma_1, \ldots, \Gamma_Q, T_1, \ldots, T_Q]$ and $j \in \{1, 2, T\}$ such that $\mathbf{x} \neq \mathbf{x}'$, but $g_{\mathbf{x}}(\gamma_1, \ldots, \gamma_Q, t_1, \ldots, t_Q) = g_{\mathbf{x}'}(\gamma_1, \ldots, \gamma_Q, t_1, \ldots, t_Q)$. By the union bound considering all the combinations of the labels obtained by $\mathcal{A}$ and Schwartz-Zippel lemma, we have the probability of this occurring at most $Q_{\mathsf{lab}}^2/p$ since $\gamma_i$ and $t_i$ for all $i \in [Q]$ are chosen uniformly at random independently from anything else. Hence, we have $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_0]| \leq \mathsf{negl}(\lambda)$.

$\mathsf{Game}_2$**.** This is the same as $\mathsf{Game}_1$ except that in this game, we replace $T_i$ with $(\lfloor \frac{\mathbf{s}^\intercal \mathbf{u} + e}{M} \rfloor + f_i(\mathbf{s}))\Gamma_i$ in polynomial as in Equation (6). In more detail, now all the labels given to $\mathcal{A}_1$ during the game correspond to a polynomial that is represented as

$$\hat{g}_{\mathbf{x}}(\Gamma_1, \ldots, \Gamma_Q) := x_0 + \sum_{i\in[Q]} x_i \left( \left\lfloor \frac{\mathbf{s}^\intercal \mathbf{u} + e}{M} \right\rfloor + f_i(\mathbf{s}) \right) \Gamma_i + \sum_{i\in[Q]} \tilde{x}_i \Gamma_i \in \mathbb{Z}_p[\Gamma_1, \ldots, \Gamma_Q] \quad (7)$$

Note that here, each of $f_i(\mathbf{s})$ can get labels corresponding to elements in $\mathbb{Z}_p$ via oracle queries, but it does not have access to the labels corresponding to elements in $\mathbb{Z}_p[\Gamma_1, \ldots, \Gamma_Q] \backslash \mathbb{Z}_p$. In Lemma 3.6, we show $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| \leq \mathsf{negl}(\lambda)$ assuming $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma_{\mathbf{s}},\sigma}$.

$\mathsf{Game}_3$**.** This is the same as $\mathsf{Game}_2$ except that in this game, we replace $\lfloor \frac{\mathbf{s}^\intercal \mathbf{u} + e}{M} \rfloor$ with $\lfloor \frac{\mathbf{s}^\intercal \mathbf{u}}{M} \rfloor$ where $e \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}$. Since $\mathbf{s}^\intercal \mathbf{u}$ is uniform over $\mathbb{Z}_q$, the two values can differ only if $\mathbf{s}^\intercal \mathbf{u}$ lies within distance $\|e\|$ of a rounding boundary. This occurs with probability at most $2\|e\|/M$, which is negligible since $\sigma/M = \mathsf{negl}(\lambda)$. Therefore, we have $|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_2]| \leq \mathsf{negl}(\lambda)$.

$\mathsf{Game}_4$. This is the same as $\mathsf{Game}_3$ except that in this game, we switch back to the RR model. Namely, we replace the formal variable $\Gamma_i$ with $\gamma_i$ for all $i \in [Q]$ again. Here we have $|\Pr[\mathsf{E}_4] - \Pr[\mathsf{E}_3]| \leq Q_{\mathsf{lab}}^2/p = \mathsf{negl}(\lambda)$ similarly to the transition from $\mathsf{Game}_0$ to $\mathsf{Game}_1$, where here, we take into account the labels obtained by the functions $\{f_i\}_i$ in addition to those obtained by $\mathcal{A}_1$.

One can see that $\mathsf{Game}_4$ equals to the real world distribution with $\beta = 0$. From the discussion above, we have $|\Pr[\mathsf{E}_0] - \Pr[\mathsf{E}_4]| \leq \mathsf{negl}(\lambda)$. This implies that the distribution in the case of $\beta = 0$ is indistinguishable from those of $\beta = 1$. □

To complete the proof of Lemma 3.5, it suffices to show the following lemma.

**Lemma 3.6.** Assuming $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma_{\mathbf{s}},\sigma}$ holds, $|\Pr[\mathsf{E}_2] - \Pr[\mathsf{E}_1]| \leq \mathsf{negl}(\lambda)$.

*Proof.* Observe that in RR model, the oracle returns a label to $\mathcal{A}$ for any query made by it. In this proof, we do not differentiate the type of queries (labeling, operation, and mapping queries), since it does not matter. Instead, we focus on the labels given to $\mathcal{A}$ and corresponding polynomials.

Recall that in both games, labels given to $\mathcal{A}$ during the game correspond to some polynomial $g$. We say that a label $L_j(g)$ obtained by an oracle query is *new* if $\mathcal{A}$ has not previously received $L_j(g)$. It is clear that all new labels are distributed uniformly at random over the space for labels $S_j$ (subject to the constraint that there is no repeated labels) in $\mathsf{Game}_1$. The same statement holds for $\mathsf{Game}_2$, although some care is required since each $f_i$ has oracle access and sees labels which is not seen by $\mathcal{A}$, which may potentially affect their distribution. We resolve this subtlety by considering two cases. If $\hat{g} \in \mathbb{Z}_p$, then $f_i(\mathbf{s})$ does not appear in the expression, and the above concern is irrelevant. On the other hand, if $\hat{g} \in \mathbb{Z}_p[\Gamma_1, \ldots, \Gamma_Q] \setminus \mathbb{Z}_p$, then $f_i(\mathbf{s})$ may appear in the expression. However, this does not cause a problem, since the computation of $f_i(\mathbf{s})$ only has oracle access to labels corresponding to elements in $\mathbb{Z}_p$, and not to those corresponding to elements in $\mathbb{Z}_p[\Gamma_1, \ldots, \Gamma_Q] \setminus \mathbb{Z}_p$. Therefore, regardless of how $f_i(\mathbf{s})$ is correlated with the distribution of labels corresponding to elements in $\mathbb{Z}_p$, it remains uncorrelated with labels corresponding to elements outside of $\mathbb{Z}_p$.

From the above discussion, we see that each new label follows the same distribution in the two games. Therefore, $\mathsf{Game}_2$ can differ from $\mathsf{Game}_1$ only if $\mathcal{A}$ receives a new label $L_j(g_{\mathbf{x}})$ as in Equation (6), while the corresponding label $L_j(\hat{g}_{\mathbf{x}})$ as in Equation (7) (with the same $\mathbf{x}$) is not new. This occurs if there exists a label $L_j(g_{\mathbf{x}'})$ already given to the adversary such that $g_{\mathbf{x}} \neq g_{\mathbf{x}'}$, but $\hat{g}_{\mathbf{x}} = \hat{g}_{\mathbf{x}'}$. We call such a vector $\mathbf{x}$ a *bad* vector and $\mathbf{x}'$ its *associating* vector for $\mathbf{x}$. Letting $\mathbf{z} := \mathbf{x} - \mathbf{x}'$, we obtain $g_{\mathbf{z}} \neq 0$ and $\hat{g}_{\mathbf{z}} = 0$. This implies, in particular, that there exists $i \in [Q]$ such that $(z_i, \tilde{z}_i) \neq (0,0)$, but $z_i(v + f_i(\mathbf{s})) + \tilde{z}_i = 0$, where $v = \lfloor (\mathbf{s}^\intercal \mathbf{u} + e)/M \rfloor$. By setting $(c, \tilde{c}) = (z_i, \tilde{z}_i)$ for such $i$, we have

$$(c, \tilde{c}) \neq (0,0) \ \wedge \ c\left(\left\lfloor \frac{\mathbf{s}^\intercal \mathbf{u} + e}{M} \right\rceil + y\right) + \tilde{c} = 0 \quad \text{for some } y \in [-\delta, \delta], \tag{8}$$

where $\delta$ bounds $|f_i(\mathbf{s})|$ for all $i \in [Q]$. However, we show that finding such a vector is hard under the $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma}$ assumption, thereb ruling out the possibility of getting differing label. Formally, we show the indistinguishability of $\mathsf{Game}_2$ and $\mathsf{Game}_1$ via the following two claims:

*Claim* 3.7. There exists a PPT adversary $\mathcal{B}$ (in the standard model) such that

$$\Pr\left[\mathcal{B}(\mathbf{A}, \mathbf{u}, \mathbf{b}, \mathbf{S}) \to (c, \tilde{c}) : \ (c, \tilde{c}) \text{ satisfies Condition (8)} \right] \ \geq \ \nu / Q Q_{\mathsf{lab}}^2.$$

where $\nu = |\Pr[\mathsf{E}_2] - \Pr[\mathsf{E}_1]|$ and $\mathbf{A}, \mathbf{u}, \mathbf{b}, \mathbf{S}$ are sampled/computed as in Lemma 3.5.

*Claim* 3.8. Assuming $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma_{\mathbf{s}},\sigma}$ holds, then for all PPT adversary $\mathcal{B}$,

$$\Pr\left[\mathcal{B}(\mathbf{A}, \mathbf{u}, \mathbf{b}, \mathbf{S}) \to (c, \tilde{c}) : \ (c, \tilde{c}) \text{ satisfies Condition (8)} \right] \ \leq \ (2\delta + 1)/p + \mathsf{negl}(\lambda)$$

where $\mathbf{A}, \mathbf{b}, \mathbf{S}, \mathbf{u}$ are sampled/computed as defined in Lemma 3.5.

Combining the two claims, we get $\nu / Q Q_{\mathsf{lab}}^2 \leq 2\delta/p + \mathsf{negl}(\lambda)$ which implies $\nu \leq \mathsf{negl}(\lambda)$, as the lemma states. We begin with proving Claim 3.8.

**Proof of Claim 3.8.**

*Proof.* To prove the claim, we first move to a hybrid in which $\mathbf{b} \leftarrow \mathbb{Z}_q^{L'}$, $\mathbf{S} \leftarrow \mathbb{Z}_q^{(n+1)\times(n+1)m\lceil \log q \rceil}$, and $\mathbf{s}^\intercal \mathbf{u} + e$ is replaced by a uniformly random value $v' \leftarrow \mathbb{Z}_q$, relying on the hardness of csLWE, where we use the equivalent formulation of the assumption in Remark 2.4, with $m' = L' + 1$. The probability that the adversary outputs a pair $(c, \tilde{c})$ satisfying Condition (8) changes only negligibly, since otherwise one could distinguish the corresponding csLWE distributions by checking whether the condition holds. In this hybrid, the values $c$ and $\tilde{c}$ must satisfy $c(v+y) + \tilde{c} = 0$ for some $y \in [-\delta, \delta]$, where $v = \lfloor v'/M \rfloor$ is uniformly distributed over $\mathbb{Z}_p$, since $v' \leftarrow \mathbb{Z}_q$ and $M$ divides $q$. However, this event occurs with probability at most $(2\delta+1)/p$. To see this, observe that for any fixed choice of $(c, \tilde{c}) \in \mathbb{Z}_p^2 \setminus \{(0,0)\}$, and for each $y \in [-\delta, \delta]$, there is at most one value of $v$ satisfying $c(v+y) + \tilde{c} = 0$. Thus, there are at most $2\delta + 1$ values of $v$ for which the condition holds for some $y \in [-\delta, \delta]$. Since $v$ is chosen uniformly at random from $\mathbb{Z}_p$ and independently of $c$ and $\tilde{c}$, the probability of this event is at most $(2\delta+1)/p$. (Note that $\mathcal{B}$ is not given $v$.) □

**Proof of Claim 3.7.**

*Proof.* Recall that the two hybrids differ in $\mathcal{A}$'s view only when $\mathcal{A}$ receives a label $L_j(g_{\mathbf{x}})$ corresponding to a bad vector $\mathbf{x}$. We call the query for which the label is returned a *bad* query. Likewise, we define the query that is responded by the associating vector $\mathbf{x}'$ for $\mathbf{x}$ an *associating* query. We know that $\mathcal{A}$ makes a bad query with probability at least $\nu := |\Pr[\mathsf{F}_1] - \Pr[\mathsf{F}_2]|$ in $\mathsf{Game}_2$. We recall that for such a bad query and associating query, there exists $i$ such that $(c, \tilde{c}) = (z_i, \tilde{z}_i)$ satisfies Condition (8), where we set $\mathbf{z} = \mathbf{x} - \mathbf{x}'$. To prove the theorem, we further consider the following sequence of games. In the following, let us denote $\mathsf{F}_{\mathrm{xx}}$ the event that $\mathcal{A}$ wins the game in $\mathsf{Game}_{2,\mathrm{xx}}$ defined below.

$\mathsf{Game}_{2,1}$: This is the same as $\mathsf{Game}_2$. We assume that the challenger simulates the symbolic group model. We say that $\mathcal{A}$ wins the game if it makes a bad query. By definition, we have $\Pr[\mathsf{F}_1] \geq \nu$.

$\mathsf{Game}_{2,2}$: In this game, we change the previous game so that the challenger picks a random guess $k^*$ and $k'$ for the *first* bad query and associating query as $k^*, k' \leftarrow [Q_{\mathsf{lab}}]$ at the beginning of the game. In addition, it also picks a random guess $i^*$ for which $(c, \tilde{c}) = (z_{i^*}, \tilde{z}_{i^*})$ defined by $\mathbf{x}$ and $\mathbf{x}'$ satisfies the Condition (8) as $i^* \leftarrow [Q]$. Furthermore, we modify the game so that $\mathcal{A}$ is deemed to lose if any of its guesses is incorrect. Since $k^*$, $k'$, and $i^*$ are chosen uniformly at random and independent from the view of $\mathcal{A}$, the guess is correct with probability at least $1/QQ_{\mathsf{lab}}^2$ conditioned on $\mathsf{F}_1$. Therefore, we have $\Pr[\mathsf{F}_2] \geq \Pr[\mathsf{F}_1]/QQ_{\mathsf{lab}}^2$.

$\mathsf{Game}_{2,3}$: This game is the same as the previous game except that the challenger aborts the game and stops immediately after $\mathcal{A}$ makes the $k^*$-th query. Since whether $\mathsf{F}_2$ occurs or not is irrelevant to how the game proceeds after the $k^*$-th query is made by $\mathcal{A}$, we clearly have $\Pr[\mathsf{F}_3] = \Pr[\mathsf{F}_2]$.

$\mathsf{Game}_{2,4}$: This game is the same as the previous game except that the symbolic group model is simulated as in $\mathsf{Game}_1$ rather than $\mathsf{Game}_2$. Since the way oracles are answered is the same in $\mathsf{Game}_1$ and $\mathsf{Game}_2$ until the first bad query, we have $\Pr[\mathsf{F}_4] = \Pr[\mathsf{F}_3]$.

We then construct $\mathcal{B}$, which acts as the challenger in $\mathsf{Game}_{2,4}$ as follows.

$\mathcal{B}$ on input $(\mathbf{A}, \mathbf{u}, \mathbf{b}, \mathbf{S})$ chooses random $k^*, k' \leftarrow [Q_{\mathsf{lab}}]$ and $i^* \leftarrow [Q]$. It then runs $\mathcal{A}$ on input $\mathbf{A}, \mathbf{u}, \mathbf{b}, \mathbf{S}$ and simulates $\mathsf{Game}_{2,4}$ for $\mathcal{A}$. In particular, $\mathcal{B}$ can simulate the symbolic group model for $\mathcal{A}$, as in $\mathsf{Game}_1$, and this simulation does not require knowledge of $\mathbf{s}^\intercal \mathbf{u} + e$. It simulates the game honestly until the $k^*$-th query made by $\mathcal{A}$. When $\mathcal{A}$ makes the $k^*$-th query, $\mathcal{B}$ extracts $\mathbf{x}$ from the $k^*$-th query. This is possible by keeping track of $\mathcal{A}$'s group operations. It also extracts the vector $\mathbf{x}'$ corresponding to the $k'$-th query, with the hope that it is an associating vector of $\mathbf{x}$. It then defines $(c, \tilde{c}) = (z_{i^*}, \tilde{z}_{i^*})$ and outputs it.

Since $\mathcal{B}$ perfectly simulates $\mathsf{Game}_{2,4}$, $\mathcal{B}$ outputs $c, \tilde{c}$ satisfying Condition (8) with probability $\Pr[\mathsf{F}_4] \geq \nu/QQ_{\mathsf{lab}}^2$. This completes the proof of Claim 3.7. □

This completes the proof of Theorem 3.2. □

The above proof also shows the following theorem, which asserts the security of the construction in the standard model assuming the doubly circular assumption.

**Theorem 3.9.** If the doubly circular assumption (Assumption 3.3) holds with respect to the parameters as in Section 3.2 and Slotted IPFE satisfies slot-mode correctness (Definition 2.15) and function hiding (Definition 2.16), the construction as in Section 3.2 is VerSel-IND-secure (Definition 2.8).

# 4 Secret-Key CP-ABE for Unbounded-Depth Circuits

In this section, we construct a secret-key CP-ABE scheme for circuits of unbounded depth from circular small secret LWE assumption.

**Ingredients.** We use bounded-depth KP-ABE with compact keys and succinct one-key FE for unbounded circuits. For both these schemes, we assume that setup uses a $\lambda$-bit random seed and that key generation is deterministic. These conventions can be achieved using standard PRF-based transformations to generate the required randomness, preserving correctness and the required security.

1. A family of universal circuits $\{\mathcal{U}_{s,L}\}$ such that each $\mathcal{U}_{s,L}$, parameterized by a description-length bound $s$ and input length $L$, takes as input a circuit description padded to length $s$ and a vector $\mathbf{x} \in \{0,1\}^L$, and outputs $C(\mathbf{x})$ for the described circuit $C$. Namely, we have $\mathcal{U}_{s,L}(\tilde{C}, \mathbf{x}) = C(\mathbf{x})$, where $\tilde{C}$ is the description of the circuit $C$.
   We denote by $\mathcal{U}_{s,L}[\mathbf{x}](\cdot) := \mathcal{U}_{s,L}(\cdot, \mathbf{x})$ the circuit with the second input vector $\mathbf{x}$ fixed/hardwired. The depth-preserving construction of the universal circuit by [CH85] gives $d(\mathcal{U}_{s,L}) = O(s)$, where $d(C)$ for a circuit $C$ denotes the depth of $C$.
2. A bounded-depth public-key KP-ABE scheme $\mathsf{KP} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with secret keys of fixed polynomial size $\mathrm{poly}_1(\lambda)$, independent of the circuit depth and input length. We require selective security for the KP-ABE. For example, the construction of [CW23] (see Lemma 2.10) satisfies this. Looking ahead, we will use it in our construction as follows:
   - A string representation of a circuit $C$, padded to length $2^j$, will be used as an attribute to encrypt a message within a KP-ABE ciphertext.
   - A key for the circuit $\mathcal{U}_{2^j,L}[\mathbf{x}]$ (described in Item 1 below) will be given out. Note that the size of this key is crucially independent of both the circuit depth and input size.
   - To enable the above operations, we define the relation for the KP-ABE as follows. For fixed $L$, $j$, and a depth bound $d$ for $\mathcal{U}_{2^j,L}$: $\mathsf{prm} := (1^{2^j}, 1^d)$, $\mathcal{X}_{\mathsf{prm}} := \{0,1\}^{2^j}$ (corresponding to padded circuit descriptions), $\mathcal{Y}_{\mathsf{prm}} := \{\mathcal{U}_{2^j,L}[\mathbf{x}] : \mathbf{x} \in \{0,1\}^L\}$ (see Item 1 above), and $R_{\mathsf{prm}}(\tilde{C}, \mathcal{U}_{2^j,L}[\mathbf{x}]) = \mathcal{U}_{2^j,L}(\tilde{C}, \mathbf{x})$.
3. A succinct one-key FE scheme $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.KeyGen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ for circuits of unbounded depth with bounded input and output lengths. We require one-key very selective simulation security (as per Definition 2.12). We can instantiate this with [HLL23] (see Lemma 2.14).
   - Looking ahead, for $j \in [\lambda]$, we will give out a single FE key for the circuit $\mathcal{H}_{2^j,L}$ described in Figure 1. The scheme of [HLL23] supports circuits with bounded input and output lengths that need to be specified during setup. For $\mathcal{H}_{2^j,L}$, the input is a tuple $(\mathbf{x}, r)$ of length $L + \lambda$, and the output is a KP-ABE secret key of size bounded by $\mathrm{poly}_1(\lambda)$ (see Item 2 above).
   - Since the scheme only supports one-key security, we reuse the same FE key for $\mathcal{H}_{2^j,L}$ in every encryption using slot $j$. This follows from the determinism of FE.KeyGen assumed above.

**Circuit $\mathcal{H}_{2^j,L}(\mathbf{x}, r)$:**

- ⋆ Input: $\mathbf{x} \in \{0,1\}^L$, $r \in \{0,1\}^\lambda$.
- Build $\mathcal{U}_{2^j,L}[\mathbf{x}]$ with $\mathbf{x}$ hardwired; set $\widehat{d_j} = d(\mathcal{U}_{2^j,L}) = O(2^j)$.
- $(\mathsf{KP.mpk}_j, \mathsf{KP.msk}_j) \leftarrow \mathsf{KP.Setup}(1^\lambda, 1^{2^j}, 1^{\widehat{d_j}}; r)$.
- $\mathsf{KP.sk}_{j,\mathbf{x}} \leftarrow \mathsf{KP.KeyGen}(\mathsf{KP.mpk}_j, \mathsf{KP.msk}_j, \mathcal{U}_{2^j,L}[\mathbf{x}])$.
- ⋆ Output $\mathsf{KP.sk}_{j,\mathbf{x}}$.

Figure 1: Description of the circuit $\mathcal{H}_{2^j,L}$.

Note that $\mathcal{H}_{2^j,L}$ is deterministic: its input seed $r$ determines the KP-ABE setup, and KP.KeyGen is deterministic. Although the size of each $\mathcal{H}_{2^j,L}$ may be unbounded, its input and output lengths are bounded by $L_{\text{in}} = L + \lambda$ and $L_{\text{out}} = \text{poly}_1(\lambda)$, respectively. Both bounds are independent of the depth of the policy circuit.

## 4.1 Construction

The secret-key CP-ABE scheme $\Pi_{\text{CP}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with fixed input length $L = L(\lambda)$ is defined as follows.

$\mathsf{Setup}(1^\lambda, 1^L)$. The setup algorithm does the following.

1. For every $i \in [\lambda]$, sample an independent seed $r_i \leftarrow \{0,1\}^\lambda$ uniformly at random.
2. Set $L_{\text{in}} = L + \lambda$ and $L_{\text{out}} = \text{poly}_1(\lambda)$.
3. For every $i \in [\lambda]$, compute:
$$(\mathsf{FE.mpk}_i, \mathsf{FE.msk}_i) \leftarrow \mathsf{FE.Setup}(1^\lambda, 1^{L_{\text{in}}}, 1^{L_{\text{out}}}).$$
4. Set and output $\mathsf{msk} := \left(\{r_i\}_{i\in[\lambda]}, L, \{\mathsf{FE.mpk}_i, \mathsf{FE.msk}_i\}_{i\in[\lambda]}\right)$.

$\mathsf{KeyGen}(\mathsf{msk}, \mathbf{x})$. The key generation algorithm does the following.

1. Parse $\mathsf{msk}$ as $\left(\{r_i\}_{i\in[\lambda]}, L, \{\mathsf{FE.mpk}_i, \mathsf{FE.msk}_i\}_{i\in[\lambda]}\right)$.
2. For every $i \in [\lambda]$, compute:
   - $\mathsf{FE.ct}_i \leftarrow \mathsf{FE.Enc}(\mathsf{FE.mpk}_i, (\mathbf{x}, r_i))$.
3. Output $\mathsf{sk}_{\mathbf{x}} = (\{\mathsf{FE.ct}_i\}_{i\in[\lambda]})$.

$\mathsf{Enc}(\mathsf{msk}, C, \mu)$. The encryption algorithm does the following.

1. Parse $\mathsf{msk}$ as $\left(\{r_i\}_{i\in[\lambda]}, L, \{\mathsf{FE.mpk}_i, \mathsf{FE.msk}_i\}_{i\in[\lambda]}\right)$ and find $j$ such that $2^{j-1} < |C| \le 2^j$.
2. Let $\widetilde{C} \in \{0,1\}^{2^j}$ be the circuit description padded with zeros to length $2^j$. Set $\widehat{d_j} = d(\mathcal{U}_{2^j,L})$.
3. Compute $\mathsf{FE.sk}_j \leftarrow \mathsf{FE.KeyGen}(\mathsf{FE.mpk}_j, \mathsf{FE.msk}_j, \mathcal{H}_{2^j,L})$.
4. Compute $(\mathsf{KP.mpk}_j, \mathsf{KP.msk}_j) \leftarrow \mathsf{KP.Setup}(1^\lambda, 1^{2^j}, 1^{\widehat{d_j}}; r_j)$.
5. Compute $\mathsf{KP.ct}_j \leftarrow \mathsf{KP.Enc}(\mathsf{KP.mpk}_j, \widetilde{C}, \mu)$.
6. Output $\mathsf{ct}_C = (\mathsf{FE.mpk}_j, \mathsf{FE.sk}_j, \mathsf{KP.mpk}_j, \mathsf{KP.ct}_j, \widetilde{C}, j)$.

$\mathsf{Dec}(\mathsf{ct}_C, C, \mathsf{sk}_{\mathbf{x}}, \mathbf{x})$. The decryption algorithm does the following.

1. Parse $\mathsf{sk}_{\mathbf{x}} = (\{\mathsf{FE.ct}_i\}_{i\in[\lambda]})$ and $\mathsf{ct}_C = (\mathsf{FE.mpk}_j, \mathsf{FE.sk}_j, \mathsf{KP.mpk}_j, \mathsf{KP.ct}_j, \widetilde{C}, j)$.

2. Compute
$$\mathsf{KP.sk}_{j,\mathbf{x}} \leftarrow \mathsf{FE.Dec}(\mathsf{FE.mpk}_j, \mathcal{H}_{2^j,L}, \mathsf{FE.sk}_j, \mathsf{FE.ct}_j).$$
3. Output
$$\mathsf{KP.Dec}(\mathsf{KP.mpk}_j, \mathsf{KP.ct}_j, \widetilde{C}, \mathsf{KP.sk}_{j,\mathbf{x}}, \mathcal{U}_{2^j,L}[\mathbf{x}]).$$

**Lemma 4.1.** Assume that FE satisfies the correctness (Definition 2.11) and the efficiency guarantees of Lemma 2.14, and that the bounded-depth KP-ABE scheme KP satisfies the correctness (Definition 2.6) and the efficiency guarantee of Lemma 2.10. Then $\Pi_{\mathrm{CP}}$ is correct.

*Proof.* Fix $L = \mathrm{poly}(\lambda)$, $\mu \in \{0,1\}$, $\mathbf{x} \in \{0,1\}^L$, and $C \in \mathcal{C}_{L(\lambda)}$ such that $C(\mathbf{x}) = 0$. Let $\mathsf{sk}_{\mathbf{x}}$ and $\mathsf{ct}_C$ be honestly generated under the same master secret key, and let $j$ satisfy $2^{j-1} < |C| \le 2^j$.

The circuit $\mathcal{H}_{2^j,L}$ takes an input of length $L + \lambda$ and outputs a KP-ABE key. By compactness, this key has length at most $\mathrm{poly}_1(\lambda)$, independently of the circuit depth and input length. Hence, $\mathcal{H}_{2^j,L}$ fits within the FE input and output bounds and is a valid circuit for the FE scheme despite its potentially large size and depth. FE correctness therefore gives

$$\mathsf{FE.Dec}(\mathsf{FE.mpk}_j, \mathcal{H}_{2^j,L}, \mathsf{FE.sk}_j, \mathsf{FE.ct}_j) = \mathcal{H}_{2^j,L}(\mathbf{x}, r_j)$$

except with negligible probability.

It remains to show that the resulting KP-ABE key decrypts $\mathsf{KP.ct}_j$. Since encryption and $\mathcal{H}_{2^j,L}$ use the same seed $r_j$, they generate the same KP-ABE instance. Thus, $\mathcal{H}_{2^j,L}(\mathbf{x}, r_j)$ is a key for $\mathcal{U}_{2^j,L}[\mathbf{x}]$ under the instance used in $\mathsf{KP.ct}_j$. Since $\mathcal{U}_{2^j,L}[\mathbf{x}](\widetilde{C}) = C(\mathbf{x}) = 0$, KP-ABE correctness guarantees recovery of $\mu$ using this key. A union bound over the FE and KP-ABE correctness errors proves the claim.

Finally, FE succinctness makes setup and key generation polynomial in $\lambda$ and $L$, independently of the policy size and depth. Encryption and decryption are also efficient, since they only process the selected slot, where $2^j = O(|C|)$. □

## 4.2 Security Proof

We prove security of the CP-ABE below.

**Theorem 4.2.** Assume that the bounded-depth KP-ABE scheme KP is selectively secure (as per Definition 2.7) and that FE is one-key very selective simulation secure (as per Definition 2.12). Then $\Pi_{\mathrm{CP}}$ is very selectively secure, satisfying (as per Definition 2.9).

*Proof.* Let $\mathcal{A}$ be an admissible PPT adversary that commits, before setup, to the challenge policy $(C^*, L)$ and the entire sequence of key-query attributes $\{\mathbf{x}^{(t)}\}_{t\in[Q]}$, where $C^*(\mathbf{x}^{(t)}) = 1$ for every $t \in [Q]$. Let $j^*$ satisfy $2^{j^*-1} < |C^*| \le 2^{j^*}$, and let $\widetilde{C}^*$ be the description of $C^*$ padded to length $2^{j^*}$. Set $L_{\mathrm{in}} = L + \lambda$, $L_{\mathrm{out}} = \mathrm{poly}_1(\lambda)$, and $\mathcal{H}^* = \mathcal{H}_{2^{j^*},L}$. For $b \in \{0,1\}$, consider the following experiments, each outputting the final bit of $\mathcal{A}$.

$\mathsf{Game}_{1,b}$. This is the real very selective security experiment for $\Pi_{\mathrm{CP}}$ with challenge bit $b$.

$\mathsf{Game}_{2,b}$. This is the same as $\mathsf{Game}_{1,b}$, except that the FE instance in slot $j^*$ is simulated. Using the seed $r_{j^*}$ sampled for this slot, compute

$$y_t = \mathcal{H}^*(\mathbf{x}^{(t)}, r_{j^*}) \qquad \text{for all } t \in [Q],$$

and run

$$\big(\mathsf{FE.mpk}_{j^*}, \mathsf{FE.sk}_{j^*}, \{\mathsf{FE.ct}_{j^*}^{(t)}\}_{t\in[Q]}\big) \leftarrow \mathsf{FE.Sim}\big(1^\lambda, 1^{L_{\mathrm{in}}}, 1^{L_{\mathrm{out}}}, \mathcal{H}^*, \{y_t\}_{t\in[Q]}\big).$$

Use $\mathsf{FE.ct}_{j^*}^{(t)}$ as the slot-$j^*$ component of the response to the $t$-th key query. Use the returned $\mathsf{FE.mpk}_{j^*}$ and $\mathsf{FE.sk}_{j^*}$ in the challenge ciphertext and every response to encryption queries using this slot. All other FE instances and all KP-ABE components are generated honestly.

$\mathsf{Game}_{1,b} \approx_c \mathsf{Game}_{2,b}$: This follows from the one-key very selective simulation security of FE. After receiving $\mathcal{A}$'s commitment, the reduction samples $r_{j^*}$ and commits to $\mathcal{H}^*$ and the entire message sequence $\{(\mathbf{x}^{(t)}, r_{j^*})\}_{t\in[Q]}$ before receiving the FE public key. It embeds the returned public key, functional key, and ciphertext sequence in slot $j^*$ and generates everything else honestly. Knowing all seeds $r_i$, it can generate every KP-ABE component and answer all encryption queries. Determinism of $\mathsf{FE.KeyGen}$ ensures that reusing the returned functional key agrees with the real encryption algorithm. Thus, the real and ideal FE experiments yield $\mathsf{Game}_{1,b}$ and $\mathsf{Game}_{2,b}$, respectively. The reduction is efficient since $2^{j^*} = O(|C^*|)$ and $Q$ is polynomially bounded.

$\mathsf{Game}_{2,0} \approx_c \mathsf{Game}_{2,1}$: We reduce this transition to selective security of KP-ABE. The reduction commits to the challenge attribute $\widetilde{C}^*$ and parameters $(1^{2^{j^*}}, 1^{\widehat{d}_{j^*}})$, and uses the received public key as $\mathsf{KP.mpk}_{j^*}$. Before running the FE simulator, it requests a KP-ABE key $k_t$ for each circuit $\mathcal{U}_{2^{j^*},L}[\mathbf{x}^{(t)}]$. All these queries are admissible because

$$\mathcal{U}_{2^{j^*},L}[\mathbf{x}^{(t)}](\widetilde{C}^*) = C^*(\mathbf{x}^{(t)}) = 1.$$

By the definition of $\mathcal{H}^*$ and the setup and deterministic key-generation conventions, $(k_t)_{t\in[Q]}$ has exactly the joint distribution of $(y_t)_{t\in[Q]}$ in $\mathsf{Game}_{2,b}$, including for repeated attributes. The reduction therefore runs the simulator above with $(k_t)_{t\in[Q]}$ and answers key queries using the resulting ciphertext sequence.

All other slots are generated honestly. Encryption queries in slot $j^*$ are answered using the public algorithm $\mathsf{KP.Enc}$ and the simulated FE master public key $\mathsf{FE.mpk}_{j^*}$ and functional key $\mathsf{FE.sk}_{j^*}$. Given the KP-ABE challenge ciphertext $\mathsf{KP.ct}^*$, the reduction returns

$$\mathsf{ct}_{C^*} = (\mathsf{FE.mpk}_{j^*}, \mathsf{FE.sk}_{j^*}, \mathsf{KP.mpk}_{j^*}, \mathsf{KP.ct}^*, \widetilde{C}^*, j^*).$$

This reproduces $\mathsf{Game}_{2,b}$ for the KP-ABE challenger's bit $b$. Hence, distinguishing $\mathsf{Game}_{2,0}$ from $\mathsf{Game}_{2,1}$ would break selective KP-ABE security.

Combining the transitions gives

$$\mathsf{Game}_{1,0} \approx_c \mathsf{Game}_{2,0} \approx_c \mathsf{Game}_{2,1} \approx_c \mathsf{Game}_{1,1},$$

which proves the theorem. □

Instantiating the ingredients using Lemmata 2.10 and 2.14, we have the following corollary.

**Corollary 4.3.** If csLWE holds (Assumption 2.3) with the parameter settings required for the instantiations in Lemmata 2.10 and 2.14, then there exists a very selectively secure secret-key CP-ABE scheme for polynomial-size circuits of unbounded depth.

# 5 Secret-Key KP-ABE for Turing Machines

In this section, we construct secret-key KP-ABE for Turing machines from CP-ABE for circuits of unbounded depth and size and KP-ABE for circuits of bounded depth and unbounded size. Instantiating the former with our construction in Section 4 and the latter with the construction of [BGG+14], we obtain secret-key KP-ABE for Turing machines under the circular small-secret LWE assumption.

## 5.1 Preliminaries and Ingredients

Here, we recall the definition of a Turing machine (TM) following [LL20]. The definition considers Turing machines with two tapes, namely, the input tape and the working tape.

**Definition 5.1 (Turing Machine).** A (deterministic) TM $M$ is represented by the tuple $M = (Q, \delta, F)$ where $Q$ is the number of states (we use $[Q]$ as the set of states and 1 as the initial state), $F \subset [Q]$ is the set of accepting state and

$$\begin{aligned} \delta : [Q] \times \{0,1\} \times \{0,1\} &\to [Q] \times \{0,1\} \times \{0, \pm 1\} \times \{0, \pm 1\} \\ (q, b_1, b_2) &\mapsto (q', b_2', \Delta i, \Delta j) \end{aligned}$$

is the state transition function, which, given the current state $q$, the symbol $b_1$ on the input tape under scan, and the symbol $b_2$ on the work tape under scan, specifies the new state $q'$, the symbol $b_2'$, overwriting $b_2$, the direction $\Delta i$ to which the input tape pointers move, and the direction $\Delta j$ to which the work tape pointer moves. The machine is required to hang (instead of halting) once it reaches an accepting state. Thus, for all $q \in F$ and $b_1, b_2 \in \{0,1\}$, it holds that $\delta(q, b_1, b_2) = (q, b_2, 0, 0)$.

For input length $n \geq 1$ and space complexity bound $s \geq 1$, the set of internal configurations of $M$ is

$$\mathcal{Q}_{M,n,s} = [n] \times [s] \times \{0,1\}^s \times [Q]$$

where $(i, j, W, q) \in \mathcal{Q}_{M,n,s}$ specifies the input tape pointer $i \in [n]$, the work tape pointer $j \in [s]$, the content of the work tape $W \in \{0,1\}^s$ and the machine state $q \in [Q]$.

For any bit-string $\mathbf{x} \in \{0,1\}^n$ for $n \geq 1$ and time/space complexity bounds $t, s \geq 1$, the machine $M$ accepts $\mathbf{x}$ within time $t$ and space $s$ if there exists a sequence of internal configurations (computation path of $t$ steps) $c_0, \ldots, c_t \in \mathcal{Q}_{M,n,s}$ with $c_k = (i_k, j_k, W_k, q_k)$ such that $(i_0, j_0, W_0, q_0) = (1, 1, 0^s, 1)$ (initial configuration),

$$\text{for all } 0 \leq k < t\text{:} \quad \begin{cases} \delta(q_k, \mathbf{x}[i_k], W_k[j_k]) = (q_{k+1}, W_{k+1}[j_k], i_{k+1} - i_k, j_{k+1} - j_k) \\ W_{k+1}[j] = W_k[j] \quad \text{for all } j \neq j_k \qquad \text{(valid transitions);} \end{cases}$$

where $\mathbf{x}[i]$ denotes the $i$ the bit of the string $\mathbf{x}$ and $W_k[j]$ denotes the $j$ the bit of the string $W_k$, and $q_t \in F$ (accepting). We also say $M$ accepts $\mathbf{x}$ within time $t$ (without the space bound) if $M$ accepts $\mathbf{x}$ within time $t$ and space $s = t$.

The following lemma can be obtained by a simple argument on emulating a Turing machine on Boolean circuits. While we have many other clever methods of simulating Turing machines on circuits (e.g., [PF79]), we use the following simple one because it evaluates the depth of the circuit as a function on the size of Turing machine, which is usually regarded as a constant and ignored.

**Lemma 5.2 (Emulating a Turing Machine on Circuit).** Consider a circuit that takes as input a description of a (deterministic) Turing machine $M = (Q, \delta, F)$, input $\mathbf{x}$ to $M$, a configuration $(i, j, W, q) \in \mathcal{Q}_{M,|\mathbf{x}|,|W|}$ and outputs the next configuration $(i', j', W', q')$. We can implement such a circuit with depth $\mathrm{poly}(\log |\mathbf{x}|, \log |W|, \log |M|)$ and size $\mathrm{poly}(|\mathbf{x}|, |W|, |M|)$.

**ABE for Turing Machines.** To define ABE for Turing machines, we set $\mathcal{X} = \{0,1\}^*$, let $\mathcal{Y}$ be the set of all Turing machines, and define $R_{\mathrm{TM}} : \mathcal{X} \times \mathcal{Y} \to \{0,1\} \cup \{\bot\}$ as

$$R_{\mathrm{TM}}((\mathbf{x}, 1^t), M) = \begin{cases} 0 & \text{if } M \text{ accepts } \mathbf{x} \text{ in } t \text{ steps} \\ 1 & \text{otherwise.} \end{cases}$$

**Ingredients.**

For the construction, use PRF, secret-key KP-ABE and secret-key CP-ABE. For KP-ABE and CP-ABE schemes, we assume that setup uses a $\lambda$-bit random seed. The conventions can be achieved using standard PRF-based transformations to generate the required randomness, preserving correctness and the required security.

1. A pseudorandom function scheme $\mathsf{PRF} = \mathsf{PRF}.(\mathsf{Gen}, \mathsf{Eval})$ with the input space $[2^\lambda]$ and the output space $\{0,1\}^\lambda$.
2. A secret-key CP-ABE scheme $\mathsf{CP} = \mathsf{CP}.(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ supporting unbounded depth and size. This can be instantiated by our scheme in Section 4.1.
3. A secret-key KP-ABE scheme $\mathsf{KP} = \mathsf{KP}.(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ supporting bounded depth and unbounded size. A public-key KP-ABE scheme supporting the same circuit class (e.g., [BGG+14] and [CW23]) implies this by storing its master public key in the master secret key and private key.
4. Families of circuits $\mathcal{U}_1 := \{\mathcal{U}_{i,M}\}$ and $\mathcal{U}_2 := \{\hat{\mathcal{U}}_{i,\mathbf{x},t}\}$ as described below Figure 2. We first note that, while the family $\mathcal{U}_1$ contains unbounded-size circuits, their depths are bounded (by the parameters). In particular, each circuit $\mathcal{U}_{i,M} \in \mathcal{U}_1$ is built as follows:
   - Let $M$ be the Turing machine $(Q, \delta, F)$. For each time step $j \in [i]$, build a circuit $C_j$ that takes as input a configuration of $M$, input $\mathbf{x}$, and outputs a configuration corresponding to the transition of $M$. This is built via Lemma 5.2. The size of input is $\log|\mathbf{x}|$, size of the work tape is $\log i$ and size of state configuration is $\log|M|$. Thus, the depth of each $C_j$ is bounded by $\mathrm{poly}(\log|\mathbf{x}|, \log i, \log|M|) \le \lambda$.
   - Sequentially compose the $i$ circuits $C_1, \ldots, C_i$ defined above to obtain the final circuit $\mathcal{U}_{i,M}$. The overall depth is therefore bounded by $i\lambda$.

**Circuit $\mathcal{U}_{i,M}(\vec{a})$:**

- ⋆ Input: string $\vec{a} \in \{0,1\}^i$
- – parse $\vec{a}$ as $(\mathbf{x}, 1^t)$
- – simulate $M(\mathbf{x})$ for $t$ steps
- ⋆ Output 0 if $M$ is in an accept state; otherwise, output 1.

Depth $d(\mathcal{U}_{i,M}) = O(i\lambda)$

**Circuit $\hat{\mathcal{U}}_{i,\mathbf{x},t}(\widetilde{M})$:**

- ⋆ Input: an $i$-bit description $\widetilde{M}$ of a TM $M$.
- – simulate $M(\mathbf{x})$ for $t$ steps
- ⋆ Output 0 if $M$ is in an accept state; otherwise, output 1.

Figure 2: Description of universal circuits.

*Remark* 5.3. Write $\widetilde{M}$ for the bit-string description of $M$, and use $|M| = |\widetilde{M}|$ for its description length. For malformed inputs or invalid machine descriptions, the relation and the corresponding universal circuits output 1. For well-formed $(\mathbf{x}, 1^t)$, setting $L_\mathbf{x} = |(\mathbf{x}, 1^t)|$ and $L_M = |M|$ gives

$$\mathcal{U}_{L_\mathbf{x},M}((\mathbf{x}, 1^t)) = \hat{\mathcal{U}}_{L_M,\mathbf{x},t}(\widetilde{M}) = R_{\mathrm{TM}}((\mathbf{x}, 1^t), M).$$

## 5.2 Construction

As in [AKY24b], we first construct two schemes with complementary correctness ranges. Write

$$L_{\mathbf{x}} = |(\mathbf{x}, 1^t)| \qquad \text{and} \qquad L_M = |M|.$$

The first scheme in Section 5.2.1 is correct when $L_M \leq L_{\mathbf{x}}$. The second scheme in Section 5.2.2 is correct when $L_M \geq L_{\mathbf{x}}$. Both schemes are correct in the equality case. Then, in Section 5.2.3, we show how to combine the two schemes to get full correctness. We primarily focus on the case of the very-selective security here. However, the construction works also for the selective security and adaptive security.

### 5.2.1 The Case $L_M \leq L_{\mathbf{x}}$

Define $\Pi^{\leq} = (\mathsf{Setup}^{\leq}, \mathsf{KeyGen}^{\leq}, \mathsf{Enc}^{\leq}, \mathsf{Dec}^{\leq})$ as follows.

$\mathsf{Setup}^{\leq}(1^\lambda)$. The setup algorithm does the following.

1. Sample $K^{\leq} \leftarrow \mathsf{PRF.Gen}(1^\lambda)$.
2. Output $\mathsf{msk}^{\leq} := K^{\leq}$.

$\mathsf{KeyGen}^{\leq}(\mathsf{msk}^{\leq}, M)$. The key generation algorithm does the following.

1. Parse $\mathsf{msk}^{\leq}$ as $K^{\leq}$, set $L_M = |M|$, and compute $r_{L_M} = \mathsf{PRF.Eval}(K^{\leq}, L_M)$.
2. Compute
$$\mathsf{CP.msk}_{L_M} \leftarrow \mathsf{CP.Setup}(1^\lambda, 1^{L_M}; r_{L_M}).$$
3. Compute $\mathsf{CP.sk}_M \leftarrow \mathsf{CP.KeyGen}(\mathsf{CP.msk}_{L_M}, \widetilde{M})$, where $\widetilde{M}$ is a string representation of $M$ and is considered an attribute.
4. Output $\mathsf{sk}_M^{\leq} = \mathsf{CP.sk}_M$.

$\mathsf{Enc}^{\leq}(\mathsf{msk}^{\leq}, (\mathbf{x}, 1^t), \mu)$. The encryption algorithm does the following.

1. Parse $\mathsf{msk}^{\leq}$ as $K^{\leq}$ and set $L_{\mathbf{x}} = |(\mathbf{x}, 1^t)|$.
2. For every $i \in [L_{\mathbf{x}}]$, compute
$$r_i = \mathsf{PRF.Eval}(K^{\leq}, i), \qquad \mathsf{CP.msk}_i \leftarrow \mathsf{CP.Setup}(1^\lambda, 1^i; r_i),$$
and
$$\mathsf{CP.ct}_i \leftarrow \mathsf{CP.Enc}(\mathsf{CP.msk}_i, \widehat{\mathcal{U}}_{i,\mathbf{x},t}, \mu).$$
3. Output $\mathsf{ct}_{\mathbf{x},t}^{\leq} = (\{\mathsf{CP.ct}_i\}_{i\in[L_{\mathbf{x}}]})$.

$\mathsf{Dec}^{\leq}(\mathsf{ct}_{\mathbf{x},t}^{\leq}, (\mathbf{x}, 1^t), \mathsf{sk}_M^{\leq}, M)$. The decryption algorithm does the following.

1. Set $L_{\mathbf{x}} = |(\mathbf{x}, 1^t)|$ and $L_M = |M|$.
2. Parse $\mathsf{sk}_M^{\leq} = \mathsf{CP.sk}_M$ and $\mathsf{ct}_{\mathbf{x},t}^{\leq} = (\{\mathsf{CP.ct}_i\}_{i\in[L_{\mathbf{x}}]})$.
3. If $L_M > L_{\mathbf{x}}$, output $\bot$. Otherwise, output $\mathsf{CP.Dec}(\mathsf{CP.ct}_{L_M}, \widehat{\mathcal{U}}_{L_M,\mathbf{x},t}, \mathsf{CP.sk}_M, \widetilde{M})$.

**Lemma 5.4 (Correctness).** If CP is correct, then $\Pi^{\leq}$ in Section 5.2.1 is correct when $L_M \leq L_{\mathbf{x}}$.

*Proof.* Fix $\mu \in \{0,1\}$, $(\mathbf{x}, 1^t)$, and $M$ such that $L_M \leq L_{\mathbf{x}}$ and $M(\mathbf{x})$ halts at an accepting state within $t$ steps, where $L_{\mathbf{x}} = |(\mathbf{x}, 1^t)|$ and $L_M = |M|$. Both key generation and encryption derive the same $\mathsf{CP.msk}_{L_M}$ from $r_{L_M} = \mathsf{PRF.Eval}(K^{\leq}, L_M)$. The secret key is generated for attribute $\widetilde{M}$, and the ciphertext contains component $L_M$ with policy $\widehat{\mathcal{U}}_{L_M,\mathbf{x},t}$. Since

$$\widehat{\mathcal{U}}_{L_M,\mathbf{x},t}(\widetilde{M}) = R_{\mathrm{TM}}((\mathbf{x}, 1^t), M) = 0,$$

correctness of CP implies that decryption outputs $\mu$ with overwhelming probability. □

**Theorem 5.5 (Very Selective Security).** If PRF is a secure PRF and CP is very-selectively secure (as per Definition 2.9), then the $\Pi^{\leq}$ scheme in Section 5.2.1 is very-selectively secure (as per Definition 2.9).

*Proof.* Let $\mathcal{A}$ be a PPT very selective adversary. Before setup, it commits to $(\mathbf{x}^\star, 1^{t^\star})$ and to all its key queries $M_1, \ldots, M_Q$ such that $R_{\mathrm{TM}}((\mathbf{x}^\star, 1^{t^\star}), M_i) = 1$ for all $i \in [Q]$. For $b \in \{0,1\}$, consider the following games.

$\mathsf{Game}_{0,b}$. This is the real very selective security experiment with challenge bit $b$.

$\mathsf{Game}_{1,b}$. This game replaces $\{\mathsf{PRF.Eval}(K^{\leq}, i)\}_{i \in [L_{\mathsf{max}}]}$ with independent uniform strings, where $L_{\mathsf{max}}$ denotes the maximum length of $(\mathbf{x}, 1^t)$ or $M$ that appears in the encryption queries, key-generation queries, and the challenge query. PRF security gives $\mathsf{Game}_{0,b} \approx_c \mathsf{Game}_{1,b}$.

$\mathsf{Game}_{2,h}$. For $h \in \{0, \ldots, L_{\mathbf{x}^\star}\}$, this game is identical to $\mathsf{Game}_{1,0}$, except that the challenge ciphertext is generated as $\mathsf{ct}^{\leq}_{\mathbf{x}^\star,t^\star} = \{\mathsf{CP.ct}_i\}_{i \in [L_{\mathbf{x}^\star}]}$, where, for each $i \in [L_{\mathbf{x}^\star}]$,

$$\mathsf{CP.ct}_i \leftarrow \begin{cases} \mathsf{CP.Enc}\left(\mathsf{CP.msk}_i, \widehat{\mathcal{U}}_{i,\mathbf{x}^\star,t^\star}, \mu_1\right), & \text{if } i \leq h, \\ \mathsf{CP.Enc}\left(\mathsf{CP.msk}_i, \widehat{\mathcal{U}}_{i,\mathbf{x}^\star,t^\star}, \mu_0\right), & \text{if } i > h, \end{cases}$$

where $\mathsf{CP.msk}_i \leftarrow \mathsf{CP.Setup}(1^\lambda, 1^i; r_i)$, and the $r_i$'s are the independent uniform strings generated in $\mathsf{Game}_{1,b}$. Thus,

$$\mathsf{Game}_{2,0} \equiv \mathsf{Game}_{1,0} \qquad \text{and} \qquad \mathsf{Game}_{2,L_{\mathbf{x}^\star}} \equiv \mathsf{Game}_{1,1}.$$

Now, for every $h \in [L_{\mathbf{x}^\star}]$, $\mathsf{Game}_{2,h-1} \approx_c \mathsf{Game}_{2,h}$, which follows directly from a reduction to the security of the underlying secret-key CP-ABE scheme CP. In more detail, the reduction commits to the size parameter $1^h$. It makes the challenge, key, and encryption queries for simulating the $h$-th instance of CP for $\mathcal{A}$, while simulating other instances honestly. It makes key queries to $\widetilde{M}$ only when $|M| = h$ and It embeds the challenge ciphertext received from its challenger in the $h$-th component of the outer challenge ciphertext. Since any key query $M$ satisfies $R_{\mathrm{TM}}((\mathbf{x}^\star, 1^{t^\star}), M) = 1$, we have $\widehat{\mathcal{U}}_{h,\mathbf{x}^\star,t^\star}(M) = 1$ and the reduction never violates the admissibility. In addition, the reduction can simulate the answer to an encryption query that $\mathcal{A}$ makes by using its encryption oracle for the $h$-th instance and computing other components honestly.

By the hybrid argument and the very selective security of the underlying CP-ABE scheme, we obtain

$$\mathsf{Game}_{0,0} \approx_c \mathsf{Game}_{1,0} \equiv \mathsf{Game}_{2,0} \approx_c \cdots \approx_c \mathsf{Game}_{2,L_{\mathbf{x}^\star}} \equiv \mathsf{Game}_{1,1} \approx_c \mathsf{Game}_{0,1},$$

which proves very selective security of $\Pi^{\leq}$. □

### 5.2.2 The Case $L_M \geq L_{\mathbf{x}}$

Define $\Pi^{\geq} = (\mathsf{Setup}^{\geq}, \mathsf{KeyGen}^{\geq}, \mathsf{Enc}^{\geq}, \mathsf{Dec}^{\geq})$ as follows.

$\mathsf{Setup}^{\geq}(1^{\lambda})$. The setup algorithm does the following.

1. Sample $K^{\geq} \leftarrow \mathsf{PRF.Gen}(1^{\lambda})$.
2. Output $\mathsf{msk}^{\geq} := K^{\geq}$.

$\mathsf{KeyGen}^{\geq}(\mathsf{msk}^{\geq}, M)$. The key generation algorithm does the following.

1. Parse $\mathsf{msk}^{\geq}$ as $K^{\geq}$ and set $L_M = |M|$.
2. For every $i \in [L_M]$, compute $r_i = \mathsf{PRF.Eval}(K^{\geq}, i)$ and
$$\mathsf{KP.msk}_i \leftarrow \mathsf{KP.Setup}(1^{\lambda}, 1^i, 1^{i\lambda}; r_i).$$
3. For every $i \in [L_M]$, compute
$$\mathsf{KP.sk}_i \leftarrow \mathsf{KP.KeyGen}(\mathsf{KP.msk}_i, \mathcal{U}_{i,M}).$$
4. Output $\mathsf{sk}^{\geq}_M = (\{\mathsf{KP.sk}_i\}_{i\in[L_M]})$.

$\mathsf{Enc}^{\geq}(\mathsf{msk}^{\geq}, (\mathbf{x}, 1^t), \mu)$. The encryption algorithm does the following.

1. Parse $\mathsf{msk}^{\geq}$ as $K^{\geq}$, set $L_{\mathbf{x}} = |(\mathbf{x}, 1^t)|$, and compute $r_{L_{\mathbf{x}}} = \mathsf{PRF.Eval}(K^{\geq}, L_{\mathbf{x}})$.
2. Compute
$$\mathsf{KP.msk}_{\mathbf{x}} \leftarrow \mathsf{KP.Setup}(1^{\lambda}, 1^{L_{\mathbf{x}}}, 1^{L_{\mathbf{x}}\lambda}; r_{L_{\mathbf{x}}}).$$
3. Compute
$$\mathsf{KP.ct}_{\mathbf{x}} \leftarrow \mathsf{KP.Enc}(\mathsf{KP.msk}_{\mathbf{x}}, (\mathbf{x}, 1^t), \mu).$$
4. Output $\mathsf{ct}^{\geq}_{\mathbf{x},t} = \mathsf{KP.ct}_{\mathbf{x}}$.

$\mathsf{Dec}^{\geq}(\mathsf{ct}^{\geq}_{\mathbf{x},t}, (\mathbf{x}, 1^t), \mathsf{sk}^{\geq}_M, M)$. The decryption algorithm does the following.

1. Set $L_{\mathbf{x}} = |(\mathbf{x}, 1^t)|$ and $L_M = |M|$.
2. Parse $\mathsf{sk}^{\geq}_M = (\{\mathsf{KP.sk}_i\}_{i\in[L_M]})$ and $\mathsf{ct}^{\geq}_{\mathbf{x},t} = \mathsf{KP.ct}_{\mathbf{x}}$.
3. If $L_M < L_{\mathbf{x}}$, output $\perp$. Otherwise, output $\mathsf{KP.Dec}(\mathsf{KP.ct}_{\mathbf{x}}, (\mathbf{x}, 1^t), \mathsf{KP.sk}_{L_{\mathbf{x}}}, \mathcal{U}_{L_{\mathbf{x}},M})$.

**Lemma 5.6 (Correctness).** If KP is correct, then $\Pi^{\geq}$ in Section 5.2.2 is correct when $L_M \geq L_{\mathbf{x}}$.

*Proof.* Fix $\mu$, $(\mathbf{x}, 1^t)$, and $M$ such that $L_M \geq L_{\mathbf{x}}$ and $R_{\mathrm{TM}}((\mathbf{x}, 1^t), M) = 0$. Key generation and encryption derive the same master key $\mathsf{KP.msk}_{L_{\mathbf{x}}}$ from $r_{L_{\mathbf{x}}} = \mathsf{PRF.Eval}(K^{\geq}, L_{\mathbf{x}})$. Since $L_M \geq L_{\mathbf{x}}$, the secret key contains a component for $\mathcal{U}_{L_{\mathbf{x}},M}$. Moreover,

$$\mathcal{U}_{L_{\mathbf{x}},M}((\mathbf{x}, 1^t)) = R_{\mathrm{TM}}((\mathbf{x}, 1^t), M) = 0.$$

The depth of this circuit is within the supported bound $L_{\mathbf{x}}\lambda$, as described in Figure 2. Correctness of KP therefore implies that decryption outputs $\mu$ with overwhelming probability.

□

**Theorem 5.7 (Very Selective Security).** If PRF is secure and KP is very-selectively secure (as per Definition 2.9), $\Pi^{\geq}$ in Section 5.2.2 is very selectively secure (as per Definition 2.9).

*Proof.* Let $\mathcal{A}$ be a PPT very selective adversary that commits, before setup, to $(\mathbf{x}^\star, 1^{t^\star})$ and to all its machine-key queries $M_1, \ldots, M_Q$ such that $R_{\mathrm{TM}}((\mathbf{x}^\star, 1^{t^\star}), M_i) = 1$ for all $i \in [Q]$. Set $N = L_{\mathbf{x}^\star} = |(\mathbf{x}^\star, 1^{t^\star})|$. For $b \in \{0,1\}$, define the following games.

$\mathsf{Game}_{0,b}$. This is the real very selective security experiment for $\Pi^{\geq}$ with challenge bit $b \in \{0,1\}$. In particular, the setup randomness for the underlying KP-ABE scheme at each index $i$ is generated as $r_i = \mathsf{PRF.Eval}(K^{\geq}, i)$.

$\mathsf{Game}_{1,b}$. This game is identical to $\mathsf{Game}_{0,b}$, except that for every $i \in [L_{\mathsf{max}}]$, the values $r_i$ are independent uniform strings instead of $\mathsf{PRF.Eval}(K^{\geq}, i)$. Here, $L_{\mathsf{max}}$ denotes the maximum length of $M$ or $(\mathbf{x}, 1^t)$ that appears in any key-generation or encryption query, including the challenge oracle. By PRF security, $\mathsf{Game}_{0,b} \approx_c \mathsf{Game}_{1,b}$.

We prove that $|\Pr[\mathsf{Game}_{0,0} = 1] - \Pr[\mathsf{Game}_{0,1} = 1]|$ is negligible by the following sequence of hybrid games:

$$\mathsf{Game}_{0,0} \approx_c \mathsf{Game}_{1,0} \approx_c \mathsf{Game}_{1,1} \approx_c \mathsf{Game}_{0,1}.$$

It remains to show that $\mathsf{Game}_{1,0} \approx_c \mathsf{Game}_{1,1}$, which follows directly from a reduction to the security of the underlying KP-ABE scheme $\mathsf{KP}$. In more detail, the reduction commits to the size parameter $(1^N, 1^{N\lambda})$, makes the challenge query for $(\mathbf{x}^\star, 1^{t^\star})$, and key queries for $\mathcal{U}_{N,M_i}$ for $i \in [Q]$ such that $L_{M_i} \geq N$. Since any key query $M$ satisfies $R_{\mathrm{TM}}((\mathbf{x}^\star, 1^{t^\star}), M) = 1$, we have $\mathcal{U}_{N,M}(\mathbf{x}^\star, 1^{t^\star}) = 1$ and the reduction never violates the admissibility. Furthermore, the reduction can simulate the answer to encryption query $\mathcal{A}$ makes by using its encryption query for the $N$-th component and computing the answer honestly for other instances. Therefore, $\Pi^{\geq}$ is very selectively secure. □

### 5.2.3 Putting the Two Cases Together

We combine the two constructions to handle both possible cases: $L_M \leq L_{\mathbf{x}}$ and $L_M \geq L_{\mathbf{x}}$. The resulting scheme $\Pi_{\mathrm{TM}} = (\mathsf{Setup}_{\mathrm{TM}}, \mathsf{KeyGen}_{\mathrm{TM}}, \mathsf{Enc}_{\mathrm{TM}}, \mathsf{Dec}_{\mathrm{TM}})$ runs both $\Pi^{\leq}$ and $\Pi^{\geq}$ during setup, key generation, and encryption. During decryption, the input lengths determine which component is used. The algorithms are defined as follows.

$\mathsf{Setup}_{\mathrm{TM}}(1^\lambda)$. The setup algorithm does the following.

1. Compute $\mathsf{msk}^{\leq} \leftarrow \mathsf{Setup}^{\leq}(1^\lambda)$ and $\mathsf{msk}^{\geq} \leftarrow \mathsf{Setup}^{\geq}(1^\lambda)$.
2. Output $\mathsf{msk}_{\mathrm{TM}} := (\mathsf{msk}^{\leq}, \mathsf{msk}^{\geq})$.

$\mathsf{KeyGen}_{\mathrm{TM}}(\mathsf{msk}_{\mathrm{TM}}, M)$. The key generation algorithm does the following.

1. Compute $\mathsf{sk}_M^{\leq} \leftarrow \mathsf{KeyGen}^{\leq}(\mathsf{msk}^{\leq}, M)$ and $\mathsf{sk}_M^{\geq} \leftarrow \mathsf{KeyGen}^{\geq}(\mathsf{msk}^{\geq}, M)$.
2. Output $\mathsf{sk}_M = (\mathsf{sk}_M^{\leq}, \mathsf{sk}_M^{\geq})$.

$\mathsf{Enc}_{\mathrm{TM}}(\mathsf{msk}_{\mathrm{TM}}, (\mathbf{x}, 1^t), \mu)$. The encryption algorithm does the following.

1. Compute $\mathsf{ct}_{\mathbf{x},t}^{\leq} \leftarrow \mathsf{Enc}^{\leq}(\mathsf{msk}^{\leq}, (\mathbf{x}, 1^t), \mu)$ and $\mathsf{ct}_{\mathbf{x},t}^{\geq} \leftarrow \mathsf{Enc}^{\geq}(\mathsf{msk}^{\geq}, (\mathbf{x}, 1^t), \mu)$.
2. Output $\mathsf{ct}_{\mathbf{x},t} = (\mathsf{ct}_{\mathbf{x},t}^{\leq}, \mathsf{ct}_{\mathbf{x},t}^{\geq})$.

$\mathsf{Dec}_{\mathrm{TM}}(\mathsf{ct}_{\mathbf{x},t}, (\mathbf{x}, 1^t), \mathsf{sk}_M, M)$. The decryption algorithm does the following.

1. Parse $\mathsf{sk}_M = (\mathsf{sk}_M^{\leq}, \mathsf{sk}_M^{\geq})$ and $\mathsf{ct}_{\mathbf{x},t} = (\mathsf{ct}_{\mathbf{x},t}^{\leq}, \mathsf{ct}_{\mathbf{x},t}^{\geq})$. Set $L_M = |M|$ and $L_{\mathbf{x}} = |(\mathbf{x}, 1^t)|$.
2. If $L_M \leq L_{\mathbf{x}}$, output $\mathsf{Dec}^{\leq}(\mathsf{ct}_{\mathbf{x},t}^{\leq}, (\mathbf{x}, 1^t), \mathsf{sk}_M^{\leq}, M)$. Otherwise, output $\mathsf{Dec}^{\geq}(\mathsf{ct}_{\mathbf{x},t}^{\geq}, (\mathbf{x}, 1^t), \mathsf{sk}_M^{\geq}, M)$.

**Lemma 5.8 (Correctness).** If $\Pi^{\leq}$ and $\Pi^{\geq}$ are correct, then the composed scheme $\Pi_{\mathrm{TM}}$ is correct.

*Proof.* Fix $\mu$, $(\mathbf{x}, 1^t)$, and $M$ with $R_{\mathrm{TM}}((\mathbf{x}, 1^t), M) = 0$. If $L_M \leq L_{\mathbf{x}}$, decryption uses the $\Pi^{\leq}$ component, which outputs $\mu$ with overwhelming probability by the correctness of $\Pi^{\leq}$. Otherwise, $L_M > L_{\mathbf{x}}$ and decryption uses the $\Pi^{\geq}$ component. By the correctness of $\Pi^{\geq}$, this component also outputs $\mu$ with overwhelming probability. □

**Theorem 5.9 (Very Selective Security).** Assume that PRF is secure, CP is very-selectively secure (as per Definition 2.9), and KP is very selectively secure (as per Definition 2.9). Then $\Pi_{\mathrm{TM}}$ is very-selectively secure (satisfying Definition 2.9).

*Proof.* For very selective security, let $\mathcal{A}$ be a PPT adversary with challenge input $(x^\star, 1^{t^\star})$ and machines $M_1, \ldots, M_Q$. We consider the following three games. In each game, the challenge ciphertext is a pair $\mathsf{ct}^\star = (\mathsf{ct}^{\leq,\star}, \mathsf{ct}^{\geq,\star})$, where $\mathsf{ct}^{\leq,\star}$ and $\mathsf{ct}^{\geq,\star}$ are the ciphertexts generated by $\Pi^{\leq}$ and $\Pi^{\geq}$, respectively.

$\mathsf{Game}_0$**.** The real very selective-security experiment with challenge message $\mu_0$. The challenge ciphertext is generated as

$$\mathsf{ct}^\star = \left(\mathsf{Enc}^{\leq}(\mathsf{msk}^{\leq}, (\mathbf{x}^\star, 1^{t^\star}), \mu_0), \mathsf{Enc}^{\geq}(\mathsf{msk}^{\geq}, (\mathbf{x}^\star, 1^{t^\star}), \mu_0)\right).$$

$\mathsf{Game}_1$**.** This game is identical to $\mathsf{Game}_0$, except that $\mathsf{ct}^{\leq,\star}$ encrypts $\mu_1$. Thus,

$$\mathsf{ct}^\star = \left(\mathsf{Enc}^{\leq}(\mathsf{msk}^{\leq}, (\mathbf{x}^\star, 1^{t^\star}), \mu_1), \mathsf{Enc}^{\geq}(\mathsf{msk}^{\geq}, (\mathbf{x}^\star, 1^{t^\star}), \mu_0)\right).$$

A reduction to the very selective security of $\Pi^{\leq}$ (Theorem 5.5) generates the $\Pi^{\geq}$ component locally and answers all the queries componentwise. Hence, $\mathsf{Game}_0 \approx_c \mathsf{Game}_1$.

$\mathsf{Game}_2$**.** This game is identical to $\mathsf{Game}_1$, except that $\mathsf{ct}^{\geq,\star}$ encrypts $\mu_1$. Thus,

$$\mathsf{ct}_2^\star = \left(\mathsf{Enc}^{\leq}(\mathsf{msk}^{\leq}, (\mathbf{x}^\star, 1^{t^\star}), \mu_1), \mathsf{Enc}^{\geq}(\mathsf{msk}^{\geq}, (\mathbf{x}^\star, 1^{t^\star}), \mu_1)\right).$$

A reduction to the very selective security of $\Pi^{\geq}$ (Theorem 5.7) generates the $\Pi^{\leq}$ component locally and answers all the queries componentwise. Hence, $\mathsf{Game}_1 \approx_c \mathsf{Game}_2$.

By construction of $\Pi_{\mathrm{TM}}$, $\mathsf{Game}_0$ is exactly the real very selective security experiment with challenge message $\mu_0$, while $\mathsf{Game}_2$ is exactly the real experiment with the challenge message $\mu_1$. Therefore, $|\Pr[\mathsf{Game}_0 = 1] - \Pr[\mathsf{Game}_2 = 1]| \leq \mathsf{negl}(\lambda)$. Therefore, $\Pi_{\mathrm{TM}}$ is very selectively secure. □

Using the secret-key CP-ABE scheme in Section 4 and the bounded-depth KP-ABE scheme of [BGG+14, CW23] gives the following corollary.

**Corollary 5.10.** If LWE and csLWE hold (Assumption 2.3), then there exists a very selectively secure secret-key KP-ABE scheme for Turing machines.

# 6 Doubly Circular Assumption and its cryptanalysis

In the security proof of our ABE construction in Section 3.2, we formulate the doubly circular assumption (Assumption 3.3) and prove it in Shoup's bilinear GGM. If the assumption is instead postulated in the standard model, the same proof establishes the standard-model security of our construction. Motivated by this observation, in this section, we study the doubly circular assumption as a stand-alone computational assumption.

We restate the assumption and discuss its basic properties in Section 6.1. In Section 6.2, we show that it follows from the SXDH assumption and circular small-secret LWE when the functions $\{f_i\}_i$ are restricted to be constant. While this provides partial justification, our ABE construction requires the assumption for general functions. Finally, in Section 6.3, we examine several natural attack strategies against the assumption and explain why they do not apply.

## 6.1 Assumption

Here, we restate the Doubly Circular assumption.

*Assumption* 6.1 (Doubly Circular assumption (Restatement of Assumption 3.3)). Let $n, m, \delta, q, L', M, \sigma, \sigma_{\mathbf{s}}, \sigma_{\mathsf{fhe}} \in \mathbb{N}$ satisfy $q = Mp$, $\sigma \leq 2^{-\lambda}M$, and $\delta(\lambda) = \mathrm{poly}(\lambda)$. The Doubly Circular assumption states that for every PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ there exists a negligible function $\mathsf{negl}(\cdot)$ such that

$$\left|\Pr\left[\mathcal{A}_1\left(\{[\gamma_i]_2, [t_{i,\beta}]_2\}_{i\in[Q]}, \mathsf{st}\right) = \beta \;\middle|\; (\mathsf{st}, \{f_i\}_{i\in[Q]}) \leftarrow \mathcal{A}_0(\mathsf{pp}, \mathbf{A}, \mathbf{u}, \mathbf{b}, \mathbf{S})\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda),$$

where the variables are generated as follows. Let $\mathcal{G}_{\mathsf{BL}} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, [1]_2) \leftarrow \mathsf{BLGroupGen}(1^\lambda)$. The public parameters, challenge values, and auxiliary information are then sampled according to the distribution specified below.

$$\begin{array}{lll}
\mathsf{pp} = (\mathcal{G}_{\mathsf{BL}}, n, m, \delta, q, L', M, \sigma, \sigma_{\mathbf{s}}, \sigma_{\mathsf{fhe}}) & \mathbf{A} \leftarrow \mathbb{Z}_q^{(n+1)\times L'}, & \bar{\mathbf{A}}_{\mathsf{fhe}} \leftarrow \mathbb{Z}_q^{n\times m}, \\
\mathbf{R} \leftarrow \{0,1\}^{m\times m(n+1)\lceil \log q\rceil}, & \mathbf{u} \leftarrow \mathbb{Z}_q^{n+1}, & \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{L'}, \\
\mathbf{e}_{\mathsf{fhe}} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma_{\mathsf{fhe}}}^{m}, & \mathbf{r} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma_{\mathbf{s}}}^{n}, & \mathbf{s} = (\mathbf{r}^\intercal, -1)^\intercal, \\
\mathbf{A}_{\mathsf{fhe}} = \begin{pmatrix} \bar{\mathbf{A}}_{\mathsf{fhe}} \\ \mathbf{r}^\intercal \bar{\mathbf{A}}_{\mathsf{fhe}} + \mathbf{e}_{\mathsf{fhe}}^\intercal \end{pmatrix}, & \{\gamma_i \leftarrow \mathbb{Z}_p\}_{i\in[Q]}, & v = \lfloor \mathbf{s}^\intercal \mathbf{u}/M \rfloor, \\
\beta \leftarrow \{0,1\} & \{t_{i,0} = \gamma_i(v + f_i(\mathbf{s}))\}_{i\in[Q]}, & \{t_{i,1} \leftarrow \mathbb{Z}_p\}_{i\in[Q]} \\
\mathbf{b}^\intercal = \mathbf{s}^\intercal \mathbf{A} + \mathbf{e}^\intercal, & \mathbf{S} = \underbrace{\mathbf{A}_{\mathsf{fhe}}\mathbf{R} - \mathsf{bits}(\mathbf{s}) \otimes \mathbf{G}}_{\texttt{circular encoding}}, &
\end{array}$$

and for every $i \in [Q]$, $f_i$ is a function such that $f_i : \mathbb{Z}_q^{n+1} \to [-\delta, \delta]$.

As shown in Lemma 3.5, the Doubly Circular assumption holds for every polynomially bounded $\delta = \delta(\lambda)$ in the bilinear GGM, specifically in the RR model, under the circular small-secret LWE assumption $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma_{\mathbf{s}},\sigma}$. Thus, the assumption is secure against all the generic attacks captured by this model.

**Falsifiability and the Bounded-Output Formulation.** The bounded-output requirement on $f_i$ is syntactic rather than semantic. The challenger is not required to verify that an arbitrary circuit has output in $[-\delta, \delta]$ on every possible input. Instead, the circuit representation of every admissible function $f_i$ is required to have the fixed final output layer $\mathsf{Truncate}_\delta$. The part of the circuit preceding this layer may compute an arbitrary polynomial-size function $\widehat{f}_i \colon \mathbb{Z}_q^{n+1} \to \mathbb{Z}_p$, while the final layer ensures by construction that

$$f_i(\mathbf{z}) = \mathsf{Truncate}_\delta\big(\widehat{f}_i(\mathbf{z})\big) \in [-\delta, \delta]$$

for every input $\mathbf{z}$. Thus, whether a circuit represents an admissible function can be efficiently determined from its description alone, without deciding any semantic property of the function over the domain of possible secrets. In particular, the security experiment is efficiently implementable, and the resulting assumption is falsifiable. This formulation is fully compatible with the reduction in Section 3.2: the reduction constructs a circuit computing $-\mathsf{err}_i(\mathbf{s}')$ and appends the prescribed $\mathsf{Truncate}_\delta$ layer.

Consequently, the resulting function satisfies the bounded-output requirement syntactically. The bounded-output restriction is essential. Without it, the adversary could choose $f_i(\mathbf{s}) = -v$, in which case

$$t_{i,0} = \gamma_i\big(v + f_i(\mathbf{s})\big) = 0,$$

allowing it to distinguish $t_{i,0}$ from a uniformly random $t_{i,1}$.

## 6.2 Reduction for Constant $\{f_i\}_i$

In fact, if each function in the function family is restricted to be a zero function, i.e., $f_i(\mathbf{s}) = 0$ for all $\mathbf{s}$ and $i$, then the assumption follows directly from circular small-secret LWE and SXDH assumptions. This holds true even if each function in the function family is restricted to be *any* constant function, i.e., for each $i \in [Q]$, there exists a constant $c_i \in \mathbb{Z}_p$ such that $f_i(\mathbf{s}) = c_i$, for all $\mathbf{s}$. We prove the above next.

**Lemma 6.2.** If the circular small-secret LWE assumption $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma_{\mathbf{s}},\sigma}$ and the SXDH assumption with respect to $\mathsf{BLGroupGen}$ hold, then the Doubly Circular assumption holds when each function $f_i\colon \mathbb{Z}_q^{n+1} \to [-\delta,\delta]$ chosen by the adversary is constant; namely, for every $i \in [Q]$, there exists $c_i \in [-\delta,\delta]$ such that $f_i(\mathbf{w}) = c_i$ for every $\mathbf{w} \in \mathbb{Z}_q^{n+1}$.

*Proof.* Let $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ be the adversary for Doubly Circular assumption. We prove the theorem via a series of games. Let us denote the event that $\mathcal{A}$ outputs a 1 at the end of $\mathsf{Game}_{\mathsf{x}}$ as $\mathsf{E}_{\mathsf{x}}$.

$\mathsf{Game}_0$**:** This is the same as the real world. Here we sample/compute terms $\mathsf{pp}, \mathbf{A}, \mathbf{u}, \mathbf{s}, \mathbf{b}, \mathbf{R}, \mathbf{A}_{\mathsf{fhe}}, \mathbf{S}$ as in Assumption 6.1 and forward to $\mathcal{A}_0$- $(\mathsf{pp}, \mathbf{A}, \mathbf{u}, \mathbf{b}^\intercal = \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal, \mathbf{S} = \mathbf{A}_{\mathsf{fhe}}\mathbf{R} - \mathsf{bits}(\mathbf{s}) \otimes \mathbf{G})$. Upon receiving $\{f_i\}_{i\in[Q]}$ from $\mathcal{A}_0$, we sample $\gamma_i$ for all $i \in [Q]$, compute $v = \lfloor \mathbf{s}^\intercal\mathbf{u}/M \rfloor$ and forward $\mathcal{A}_1$ the tuple $(\{[\gamma_i]_2, [t_i]_2 = [\gamma_i(v + c_i)]_2\}_{i\in[Q]})$.

$\mathsf{Game}_1$**:** In this game $v = \lfloor (\mathbf{s}^\intercal\mathbf{u} + e)/M \rfloor$ instead of $v = \lfloor \mathbf{s}^\intercal\mathbf{u}/M \rfloor$, where $e \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}$. Since $\mathbf{s}^\intercal\mathbf{u}$ is uniform over $\mathbb{Z}_q$, the two values can differ only if $\mathbf{s}^\intercal\mathbf{u}$ lies within distance $\|e\|$ of a rounding boundary. This occurs with probability at most $2\|e\|/M$, which is negligible since $\sigma/M = \mathsf{negl}(\lambda)$. Therefore, we have $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_0]| \le \mathsf{negl}(\lambda)$.

$\mathsf{Game}_2$**:** In this game we replace $\mathbf{b}^\intercal \leftarrow \mathbb{Z}_q^{1\times L'}$, $\mathbf{S} \leftarrow \mathbb{Z}_q^{(n+1)\times(n+1)m\lceil\log q\rceil}$ and $v \leftarrow \mathbb{Z}_p$. Assuming $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma_{\mathbf{s}},\sigma}$, we have $|\Pr[\mathsf{E}_2] - \Pr[\mathsf{E}_1]| \le \mathsf{negl}(\lambda)$. To see this, we construct a reduction $\mathcal{B}$ that breaks $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma_{\mathbf{s}},\sigma}$ given a distinguisher between $\mathsf{Game}_1$ and $\mathsf{Game}_2$. Here, we use the equivalent formulation of the assumption in Remark 2.4, with $m' = L' + 1$. $\mathcal{B}$ proceeds as follows.

- It receives $(\mathbf{A}_{\mathsf{fhe}}, \mathbf{S}, \mathbf{A}_0, \mathbf{b}_0)$ the instance of $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma_{\mathbf{s}},\sigma}$.
- It parses $\mathbf{A}_0 = (\mathbf{A}' \in \mathbb{Z}_q^{(n+1)\times L'},\ \mathbf{u}' \in \mathbb{Z}_q^{n+1})$ and $\mathbf{b}_0^\intercal = ((\mathbf{b}')^\intercal \in \mathbb{Z}_q^{1\times L'},\ b \in \mathbb{Z}_q)$.
- It forwards $(\mathsf{pp}, \mathbf{A} = \mathbf{A}', \mathbf{u} = \mathbf{u}', \mathbf{b} = \mathbf{b}', \mathbf{S})$ to $\mathcal{A}_0$.
- Upon receiving $\{f_i\}_{i\in[Q]}$ from $\mathcal{A}_0$, it computes $\left(\left\{[\gamma_i]_2,\ [t_i]_2 = \left[\gamma_i \cdot \left(\left\lfloor \frac{b}{M} \right\rfloor + c_i\right)\right]_2\right\}\right)_{i\in[Q]}$ and gives it to $\mathcal{A}_1$.
- Finally, $\mathcal{B}$ outputs the same bit as $\mathcal{A}_1$.

It is easy to see that if $\beta = 0$, then $\mathcal{B}$ simulated $\mathsf{Game}_1$ for $\mathcal{A}$, else it simulated $\mathsf{Game}_2$.

$\mathsf{Game}_3$**:** In this game, $t_i = w_i + \gamma_i c_i$ for $w_i \leftarrow \mathbb{Z}_p$. We have $|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_2]| \le \mathsf{negl}(\lambda)$ from the SXDH assumption (Assumption 2.5) where $x, \{y_i, r_i\}_{i\in[Q]}$ in SXDH respectively corresponds to $v, \{\gamma_i, w_i\}_{i\in[Q]}$ here.

$\mathsf{Game}_4$: In this game $t_i \leftarrow \mathbb{Z}_p$. We have $\Pr[\mathsf{E}_4] = \Pr[\mathsf{E}_3]$ since $w_i$'s are not used anywhere else.

$\mathsf{Game}_5$: In this game we compute $\mathbf{b}^\intercal = \mathbf{s}^\intercal \mathbf{A} + \mathbf{e}^\intercal, \mathbf{S} = \mathbf{A}_{\mathsf{fhe}}\mathbf{R} - \mathsf{bits}(\mathbf{s}) \otimes \mathbf{G}$. The indistinguishability argument is the same as that for $\mathsf{Game}_1$ and $\mathsf{Game}_2$. Hence, assuming $\mathsf{csLWE}_{n,m,L'+1,q,\sigma_{\mathsf{fhe}},\sigma_{\mathbf{s}},\sigma}$, we have $|\Pr[\mathsf{E}_5] - \Pr[\mathsf{E}_4]| \leq \mathsf{negl}(\lambda)$.

From the above, we have $|\Pr[\mathsf{E}_0] - \Pr[\mathsf{E}_5]| \leq \mathsf{negl}(\lambda)$. Hence, the proof. □

### 6.3 Cryptanalysis

Here, we investigate potential attack strategies against our assumption. Note that a broad class of attacks, namely generic attacks, is ruled out by Lemma 3.5. Therefore, any successful attack must explicitly exploit the group encoding or the group description. The only apparent place where such information could be embedded is in the functions $\{f_i\}_i$. However, due to the randomization by $\{\gamma_i\}_i$, embedding useful information into $f_i$ appears difficult. In the following, we discuss a restricted class of attacks, connections to existing security notions, and an attack on a related variant.

**Noise Correlation Attack.** Here, we discuss a special class of attacks that falls within the category of generic attacks. Although this class is already ruled out by Lemma 3.5, the discussion is instructive, as it clarifies the difference between our assumption and the evasive (circular) LWE assumption, which is known to be insecure in general [AMYY25, HJL25]. Recall that the output range of each function $f_i$ is $[-\delta, \delta]$, where $\delta = \mathrm{poly}(\lambda)$. Consider the term

$$[t_i]_2 = [\gamma_i(v + f_i(\mathbf{s}))]_2, \qquad v := \lfloor \mathbf{s}^\intercal \mathbf{u}/M \rceil .$$

Here, $\gamma_i$ is sampled uniformly from $\mathbb{Z}_p$, while $v$ is pseudorandom under the LWE assumption. A potential concern is that $f_i(\mathbf{s})$ may be correlated with $v$. For example, interpreting $v$ using a fixed integer representation, consider the choice $f_i(\mathbf{s}) = -(v \bmod \delta)$. In this case, we obtain

$$[t_i]_2 = [\gamma_i \delta \lfloor v/\delta \rfloor]_2.$$

Thus, before multiplication by $\gamma_i$, the exponent is divisible by $\delta$. At a high level, this attack strategy resembles the recent noise-correlation attacks on evasive LWE [AMYY25, HJL25]. In the evasive LWE setting, the resulting divisibility can be tested directly because the relevant values are given in the clear. In our setting, by contrast, the value $\delta \lfloor v/\delta \rfloor$ appears only in the exponent and is available to the adversary only through its group encoding. Hence, its divisibility is not directly observable, and the corresponding attack does not carry over to our setting.

**Relation to Key Dependent Message Security.** Our assumption combines aspects of ElGamal encryption and GSW encryption with key-dependent message (KDM) security in a inseparable manner. To build intuition, consider the following modification of the ElGamal encryption with a small message space. A message $m \in \mathbb{Z}_p$ is encrypted under the public key $[x]$ as $(c_1, c_2) = ([y], [x]^y \cdot [1]^{my} = [xy + my])$ where $x, y \leftarrow \mathbb{Z}_p$. Decryption using the secret key $x$ proceeds by computing $c_3 = c_2 \cdot c_1^{-x}$ and recovering the message as $\log_{c_1} c_3$. Since this final step requires solving a discrete logarithm, the message space must be small. In our setting, the adversary obtains a ciphertext $([\gamma_i]_2, [t_i = \gamma_i(v + f_i(\mathbf{s}))]_2)$ of the message $f_i(\mathbf{s})$ under the public key $[v]_2$. As the secret $v$ itself is derived from another secret $\mathbf{s}$, this ciphertext can be viewed as a KDM encryption, where a message derived from the secret key is encrypted under the corresponding public key. Moreover, the adversary is also given a GSW encryption $\mathbf{S}$ of the secret $\mathbf{s}$. This setting is reminiscent of strong KDM security with auxiliary information [DK18]. Roughly speaking, this notion requires that if encryption under a possibly maliciously generated public key remains secure in the presence of some auxiliary information $\mathsf{aux}$, then security should also hold in the KDM setting under the

same joint distribution of the public key and aux. Indeed, our assumption can be interpreted as asserting the KDM security of the above modified ElGamal scheme with respect to the auxiliary information $\mathsf{aux} = (\mathbf{A}, \mathbf{u}, \mathbf{b}, \mathbf{S})$. It is known that, in general, KDM security in the presence of unrestricted auxiliary information is impossible for any IND-CPA secure encryption scheme [FKP18]. The impossibility result relies on constructing a contrived auxiliary input that embeds an obfuscation of the decryption circuit with the secret key hardwired, where the lock value equals the KDM function output. It is easy to see that this attack strategy does not apply to our assumption as there is no unrestricted aux in our assumption, leaving no space to embed any problematic information such as the LO circuit above. Note that even though $\mathbf{S}$ in our assumption depends on the secret $\mathbf{s}$, it does not help the distinguisher since it is computationally hidden by circular small secret LWE.

**Attack on a Variant.** We show that if the assumption is modified so that $f_i$ is allowed to depend on $[\gamma_i]_2$ and has access to the group description and operation, then an attack becomes possible. In such a case, we define $f_i$ as follows. On input $\mathbf{s}$, the function first computes $v = \lfloor \mathbf{s}^\intercal \mathbf{u}/M \rfloor$. It then searches for the smallest integer $w \in [-\delta, \delta]$ such that the least significant bit (LSB) of $[\gamma_i(v+w)]_2$ is 0, where we use the binary representation of the group encoding $[\gamma_i(v+w)]_2$, and outputs $w$ if found. Assuming that the LSBs of the group encodings obtained by this operation are balanced over $\{0,1\}$[11], such a value $w$ can be found with overwhelming probability for polynomially large $\delta$. As a result, the adversary can distinguish $[t_{i,0}]_2 = [\gamma_i(v + f_i(\mathbf{s}))]_2$ from $[t_{i,1}]_2$ simply by inspecting the LSB. This attack does not apply to our assumption, since $f_i$ cannot depend on $\gamma_i$. We note that this attack cannot be captured by the TS model [Zha22], since the function $f_i$ explicitly uses labeling information. In contrast, the attack can be captured in the RR model if we allow the functions $f_i$ to have access to the group oracles. Our formulation of the assumption in Lemma 3.5 captures this type of attack by permitting oracle access to $f_i$ and by adopting the RR model rather than the TS model.

## Acknowledgement

Shweta Agrawal, Supriyo Banerjee, Anuja Modi, and Rajeev Raghunath are supported by the Arvind Raghunathan Center for the Theory of Computing (aRtCS), the Mphasis F1 Foundation, and the National Quantum Mission initiative of the Department of Science and Technology, India. Shota Yamada was supported in part by JST K Program Grant Number JPMJKP24U3 and JST CREST Grant Number JPMJCR22U5, Japan. We only used AI for editorial polishing.

[11] This assumption can be removed by replacing the LSB with an output of a $2\delta + 1$-wise independent hash function on input the group encoding.

## References


[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. (Cited on page 8.)

[AKY24a] Shweta Agrawal, Simran Kumari, and Shota Yamada. Compact pseudorandom functional encryption from evasive LWE. Cryptology ePrint Archive, Paper 2024/1719, 2024. (Cited on page 3.)

[AKY24b] Shweta Agrawal, Simran Kumari, and Shota Yamada. Attribute Based Encryption for Turing Machines from Lattices. In *Crypto*, 2024. (Cited on page 1, 3, 15, 16, 40.)

[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. (Cited on page 8, 23.)

[AMVY21] Shweta Agrawal, Monosij Maitra, Narasimha Sai Vempati, and Shota Yamada. Functional encryption for Turing machines with dynamic bounded collusion from LWE. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 239–269, Virtual Event, August 2021. Springer, Heidelberg. (Cited on page 15, 16.)

[AMY19] Shweta Agrawal, Monosij Maitra, and Shota Yamada. Attribute based encryption (and more) for nondeterministic finite automata from LWE. Cryptology ePrint Archive, Paper 2019/629, 2019. (Cited on page 13.)

[AMYY25] Shweta Agrawal, Anuja Modi, Anshu Yadav, and Shota Yamada. Zeroizing attacks against evasive & circular evasive lwe. In *Theory of Cryptography: 23rd International Conference, TCC 2025, Aarhus, Denmark, December 1–5, 2025, Proceedings, Part II*, page 259–290, Berlin, Heidelberg, 2025. Springer-Verlag. (Cited on page 1, 3, 6, 7, 16, 47.)

[ARYY23] Shweta Agrawal, Mélissa Rossi, Anshu Yadav, and Shota Yamada. Constant input attribute based (and predicate) encryption from evasive and tensor LWE. In *CRYPTO 2023, Part IV*, LNCS, pages 532–564. Springer, Heidelberg, August 2023. (Cited on page 7.)

[AWY20] Shweta Agrawal, Daniel Wichs, and Shota Yamada. Optimal broadcast encryption from lwe and pairings in the standard model. In *Theory of Cryptography: 18th International Conference, TCC 2020, Durham, NC, USA, November 16–19, 2020, Proceedings, Part I*, page 149–178, Berlin, Heidelberg, 2020. Springer-Verlag. (Cited on page 9, 16.)

[AY20a] Shweta Agrawal and Shota Yamada. CP-ABE for circuits (and more) in the symmetric key setting. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part I*, volume 12550 of *LNCS*, pages 117–148. Springer, Heidelberg, November 2020. (Cited on page 3, 12, 13, 14.)

[AY20b] Shweta Agrawal and Shota Yamada. Optimal broadcast encryption from pairings and lwe. In *Advances in Cryptology – EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part I*, page 13–43, Berlin, Heidelberg, 2020. Springer-Verlag. (Cited on page 3, 7, 8, 16.)

[BDJ+25] Pedro Branco, Nico Döttling, Abhishek Jain, Giulio Malavolta, Surya Mathialagan, Spencer Peters, and Vinod Vaikuntanathan. Pseudorandom obfuscation and applications. In *CRYPTO*, Cham, 2025. Springer Nature Switzerland. (Cited on page 1, 3.)

[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 *EUROCRYPT*. Springer, 2014. (Cited on page 3, 5, 6, 7, 12, 13, 14, 15, 25, 37, 39, 44.)

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, page 309–325, New York, NY, USA, 2012. Association for Computing Machinery. (Cited on page 4, 6.)

[BJK15] Allison Bishop, Abhishek Jain, and Lucas Kowalczyk. Function-hiding inner product encryption. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part I*, volume 9452 of *LNCS*, pages 470–491. Springer, Heidelberg, November / December 2015. (Cited on page 8.)

[BÜW25] Chris Brzuska, Akin Ünal, and Ivy K. Y. Woo. Evasive lwe assumptions: Definitions, classes, and counterexamples. In *ASIACRYPT 2024*, pages 418–449. Springer Nature Singapore, 2025. (Cited on page 1, 3.)

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS '11, page 97–106, USA, 2011. IEEE Computer Society. (Cited on page 4, 6.)

[BW19] Ward Beullens and Hoeteck Wee. Obfuscating simple functionalities from knowledge assumptions. In Dongdai Lin and Kazue Sako, editors, *Public-Key Cryptography – PKC 2019*, pages 254–283, Cham, 2019. Springer International Publishing. (Cited on page 22, 23.)

[CH85] Stephen A. Cook and H. James Hoover. A depth-universal circuit. *SIAM J. Comput.*, 14(4):833–839, November 1985. (Cited on page 34.)

[CW23] Valerio Cini and Hoeteck Wee. Abe for circuits with poly (lambda) -sized keys from lwe. In *FOCS*, 2023. (Cited on page 14, 15, 21, 34, 39, 44.)

[CW25] Valerio Cini and Hoeteck Wee. Faster abe for turing machines from circular evasive lwe. In *EUROCRYPT*, page 94–125, Berlin, Heidelberg, 2025. Springer-Verlag. (Cited on page 1, 3.)

[DJM+25] Nico Döttling, Abhishek Jain, Giulio Malavolta, Surya Mathialagan, and Vinod Vaikuntanathan. Simple and general counterexamples for private-coin evasive lwe. In *CRYPTO*, Cham, 2025. Springer Nature Switzerland. (Cited on page 1, 3.)

[DK18] Apoorvaa Deshpande and Yael Kalai. Proofs of ignorance and applications to 2-message witness hiding. Cryptology ePrint Archive, Report 2018/896, 2018. https://eprint.iacr.org/2018/896. (Cited on page 47.)

[FKP18] Bernardo Freitas, Julia Komlo, and Rafael Pass. An algorithmic framework for ad hoc public keys. In Dario Catalano and Roberto De Prisco, editors, *Security and Cryptography Networks - 11th International Conference, SCN 2018, Amalfi, Italy, September 12-14, 2018,*

*Proceedings*, volume 11035 of *Lecture Notes in Computer Science*, pages 543–561. Springer, 2018. (Cited on page 48.)

[GKP+13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 555–564. ACM Press, June 2013. (Cited on page 13, 14, 22.)

[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. (Cited on page 3.)

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. (Cited on page 5, 7.)

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Heidelberg, August 2013. (Cited on page 4, 25.)

[GVW12] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 162–179. Springer, Heidelberg, August 2012. (Cited on page 22.)

[GVW13a] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In *STOC*, 2013. (Cited on page 3.)

[GVW13b] 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. (Cited on page 12.)

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. *Journal of the ACM*, 62(6):45:1–45:33, 2015. (Cited on page 3.)

[HHY26] Tzu-Hsiang Huang, Wei-Hsiang Hung, and Shota Yamada. A note on obfuscation-based attacks on private-coin evasive lwe. *IACR Communications in Cryptology*, 2(4):20, 2026. (Cited on page 1, 3.)

[HJL25] Yao-Ching Hsieh, Aayush Jain, and Huijia Lin. Lattice-based post-quantum io from circular security with random opening assumption (part ii: zeroizing attacks against private-coin evasive lwe assumptions). *Cryptology ePrint Archive*, 2025. (Cited on page 1, 3, 47.)

[HLL23] Yao-Ching Hsieh, Huijia Lin, and Ji Luo. Attribute-based encryption for circuits of unbounded depth from lattices. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 415–434. IEEE, 2023. (Cited on page 1, 3, 4, 6, 13, 14, 15, 16, 17, 22, 24, 34.)

[Lin17] Huijia Lin. Indistinguishability obfuscation from sxdh on 5-linear maps and locality-5 prgs. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017*, pages 599–629, Cham, 2017. Springer International Publishing. (Cited on page 22, 23.)

[LL20] Huijia Lin and Ji Luo. Compact adaptively secure ABE from $k$-Lin: Beyond $\mathsf{NC}^1$ and towards $\mathsf{NL}$. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 247–277. Springer, Heidelberg, May 2020. (Cited on page 38.)

[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. (Cited on page 3, 7, 9.)

[LV16] Huijia Lin and Vinod Vaikuntanathan. Indistinguishability Obfuscation from DDH-Like Assumptions on Constant-Degree Graded Encodings . In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 11–20, Los Alamitos, CA, USA, October 2016. IEEE Computer Society. (Cited on page 22, 23.)

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755. Springer, Heidelberg, April 2012. (Cited on page 17.)

[PF79] Nicholas Pippenger and Michael J Fischer. Relations among complexity measures. *Journal of the ACM (JACM)*, 26(2), 1979. (Cited on page 38.)

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In *Proceedings of the 16th Annual International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT'97, page 256–266, Berlin, Heidelberg, 1997. Springer-Verlag. (Cited on page 18.)

[SS10] Amit Sahai and Hakan Seyalioglu. Worry-free encryption: functional encryption with public keys. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 2010*, pages 463–472. ACM Press, October 2010. (Cited on page 16.)

[SW05] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, 2005. (Cited on page 3.)

[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. (Cited on page 23.)

[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. (Cited on page 3, 7.)

[Zha22] Mark Zhandry. To label, or not to label (in generic groups). In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part III*, volume 13509 of *LNCS*, pages 66–96. Springer, Heidelberg, August 2022. (Cited on page 10, 16, 18, 48.)