# Multi-Authority ABE from Lattices without Random Oracles

Brent Waters
UT Austin and NTT Research
bwaters@cs.utexas.edu

Hoeteck Wee
NTT Research and ENS, Paris
wee@di.ens.fr

David J. Wu
UT Austin
dwu4@cs.utexas.edu

**Abstract**

Attribute-based encryption (ABE) extends public-key encryption to enable fine-grained control to encrypted data. However, this comes at the cost of needing a central *trusted* authority to issue decryption keys. A multi-authority ABE (MA-ABE) scheme decentralizes ABE and allows anyone to serve as an authority. Existing constructions of MA-ABE only achieve security in the random oracle model.

In this work, we develop new techniques for constructing MA-ABE for the class of subset policies (which captures policies such as conjunctions and DNF formulas) whose security can be based in the plain model *without* random oracles. We achieve this by relying on the recently-proposed "evasive" learning with errors (LWE) assumption by Wee (EUROCRYPT 2022) and Tsabury (CRYPTO 2022).

Along the way, we also provide a modular view of the MA-ABE scheme for DNF formulas by Datta et al. (EUROCRYPT 2021) in the random oracle model. We formalize this via a general version of a related-trapdoor LWE assumption by Brakerski and Vaikuntanathan (ITCS 2022), which can in turn be reduced to the plain LWE assumption. As a corollary, we also obtain an MA-ABE scheme for subset policies from *plain* LWE with a *polynomial* modulus-to-noise ratio in the random oracle model. This improves upon the Datta et al. construction which relied on LWE with a *sub-exponential* modulus-to-noise ratio. Moreover, we are optimistic that the generalized related-trapdoor LWE assumption will also be useful for analyzing the security of other lattice-based constructions.

## 1 Introduction

Attribute-based encryption (ABE) [SW05, GPSW06] extends classic public-key encryption to support fine-grained access control on encrypted data. For instance, in a ciphertext-policy ABE (CP-ABE) scheme, each ciphertext ct is associated with a policy $f$ together with a message $\mu$ while decryption keys sk are associated with an attribute $x$. Decryption successfully recovers the message $\mu$ when $x$ satisfies $f$. Security requires that an adversary who only possesses secret keys for a collection of attributes $x_1, \ldots, x_n$ that do not satisfy $f$ does not learn anything about the message. In this work, we are interested in systems that are secure against *unbounded* collusions: that is, security holds against an adversary that has any arbitrary (polynomial) number of non-satisfying attributes.

**Multi-authority ABE.** In a traditional ABE scheme, there exists a central trusted authority that generates and issues decryption keys. The central authority has the ability to decrypt *all* ciphertexts encrypted using the system. To mitigate the reliance on a single central trusted authority, a line of works [Cha07, LCLS08, MKE08, CC09] have introduced and studied the notion of a "multi-authority" ABE (MA-ABE) scheme where *anyone* can become an authority. In an MA-ABE scheme, each authority controls different attributes and can *independently* issue secret keys corresponding to the set of attributes under their control. Policies in an MA-ABE system are formulated with respect to the attributes of one or more authorities. To decrypt, a user combines the secret keys for attributes from a set of authorities that satisfy the policy. Security is still required to hold against users who possess an arbitrary number of unauthorized secret keys, with an additional challenge that some subset of the authorities (associated with the ciphertext policy) could now be corrupted and colluding with the adversary.

Earlier constructions of MA-ABE had various limitations in terms of functionality or security (or both). The first construction that achieved the first fully decentralized MA-ABE scheme was by Lewko and Waters [LW11]. Unlike previous schemes, the Lewko-Waters scheme allows any user to become an authority, and moreover, the

only coordination needed among users and authorities is a one-time sampling of a set of global parameters. The Lewko-Waters construction supports any access policy computable by an $\mathsf{NC}^1$ circuit (i.e., a Boolean formula) and security relies on assumptions on groups with bilinear maps and in the random oracle model. Subsequently, a number of works have realized new constructions for $\mathsf{NC}^1$ policies based on bilinear maps [RW15, DKW21b], and recently, Datta et al. [DKW21a] showed how to construct an MA-ABE scheme for access policies computable by DNF formulas (of *a priori* bounded size) from the learning with errors (LWE) assumption [DKW21a]. All of these constructions rely on the random oracle model. This motivates the following question:

*Can we construct a multi-authority ABE scheme without random oracles?*

## 1.1 Our Contributions

In this work, we show how to leverage the recently-introduced evasive LWE assumption [Wee22, Tsa22] to obtain an MA-ABE scheme for subset policies *without* random oracles. Subset policies capture DNF policies as in [DKW21a].[1] Moreover, our MA-ABE construction supports subset policies and DNFs of arbitrary polynomial size which improves upon the previous lattice-based construction in the random oracle model [DKW21a]. We summarize this result in the following informal theorem and provide the full details in Section 6:

**Theorem 1.1** (Informal)**.** *Assuming polynomial hardness of LWE and of evasive LWE (both with a sub-exponential modulus-to-noise ratio), there exists a statically-secure multi-authority ABE for subset policies (of arbitrary polynomial size).*

**Understanding the evasive LWE assumption.** While the evasive LWE assumption is much less well-understood compared to the plain LWE assumption, our construction provides a new avenue towards realizing MA-ABE *without* random oracles. In particular, putting assumptions aside, our construction constitutes the first heuristic MA-ABE without random oracles. In all previous constructions of multi-authority ABE, the random oracle was used to hash a global user identifier (denoted gid) to obtain common randomness that is used to bind different keys to a single user. For the particular case of [DKW21a], the random oracle was used to hash an identifier to obtain a discrete Gaussian sample. Our candidate replaces the random oracle with a subset product of public low-norm matrices. To prove security of the resulting scheme, we rely on the fact that under LWE, multiplying a secret key by a subset product of (public) low-norm matrices yields a pseudorandom function [BLMR13] in addition to the evasive LWE assumption.

**A modular approach in the random oracle model.** The starting point of our construction is the MA-ABE construction for (bounded-size) DNF policies by Datta et al. [DKW21a]. Along the way to our construction without random oracles (Theorem 1.1), we provide a more modular description of the Datta et al. scheme. Specifically, we extract a new trapdoor sampling lemma that is implicitly used in their construction. This lemma can be viewed as a generalization of the related trapdoor LWE lemma from the recent work of Brakerski and Vaikuntanathan [BV22], and may prove useful for constructing other primitives from the *standard* LWE assumption. We provide an overview of our generalized related-trapdoor lemma in Section 2 and provide the full details in Section 4.

Using our generalized related-trapdoor LWE lemma, we in turn provide a more modular description of the MA-ABE scheme of Datta et al. [DKW21a], and moreover, base hardness on the plain LWE assumption with a *polynomial* modulus-to-noise ratio in the random oracle model. Previously, Datta et al. relied on noise smudging for trapdoor sampling in their security analysis[2], and consequently, could only reduce security to LWE with a *sub-exponential* modulus-to-noise ratio. We summarize these results in the following (informal) theorem and provide the full details in Section 5:

**Theorem 1.2** (Informal)**.** *Let $\lambda$ be a security parameter. Assuming polynomial hardness of LWE with a polynomial modulus-to-noise ratio, there exists a statically-secure multi-authority ABE scheme for subset policies of* a priori *bounded length $L = L(\lambda)$ in the random oracle model. The size of the ciphertext is quasi-linear in the bound $L$.*

[1] As noted in [DKW21a, Remark 6.1], the MA-ABE scheme therein requires a *monotone* secret-sharing scheme where reconstruction has small coefficients and the joint distribution of the unauthorized shares are uniformly random; such a scheme is only known for subset policies and DNFs.

[2] See the descriptions of Hybrid 5 and the analysis of Lemmas 5.5 and 6.5 in [DKW21a], where noise smuging is used for simulating secret keys.

Like previous lattice-based MA-ABE constructions in the random oracle model [DKW21a], the global public parameters in Theorem 1.2 imposes an *a priori* bound $L$ on the size of the policies that can be associated with ciphertexts, and moreover, the ciphertext size increases as a function of $L$. We note that our construction based on the stronger evasive LWE assumption (Theorem 1.1) supports policies of arbitrary polynomial size in the plain model.

### 1.2 Additional Related Work

Kim [Kim19] and Wang et al. [WFL19] also studied constructions of multi-authority ABE (for bounded-depth circuits and Boolean formulas, respectively) from lattice-based assumptions. However, both schemes operate in a a model where there is a single central authority that generates the public keys and secret keys for each of the authorities in the system. Relying on a central trusted party runs against the original goal of *decentralizing* trust. Moreover, these constructions only ensure security against bounded collusions. In this work, we focus exclusively on the fully decentralized setting introduced by Lewko and Waters [LW11] that neither requires a centralized setup nor assumes an a priori bound on the number of authorities or corruptions.

Recently, Tsabury [Tsa22] and Vaikuntanathan et al. [VWW22] showed how to build witness encryption from a *stronger* variant of the evasive LWE assumption with private-coin auxiliary input and sub-exponential hardness. In contrast, our multi-authority ABE construction in the standard model relies on evasive LWE with *public-coin* auxiliary input and polynomial hardness with a sub-exponential modulus-to-noise ratio; this was also the case for the optimal broadcast encryption scheme by Wee [Wee22]. While vanilla witness encryption implies single-authority ABE [GGSW13], we currently do not know how to construct multi-authority ABE from *vanilla* witness encryption.

## 2 Technical Overview

In this section, we provide a technical overview of our lattice-based MA-ABE constructions. Throughout this work, we focus exclusively on *subset policies* (which suffices for supporting DNF formulas). In an ABE scheme for subset policies, ciphertexts are associated with a set $A$ and secret keys are associated with a set $B$. Decryption succeeds if $A \subseteq B$.

**Lattice preliminaries.** The learning with errors (LWE) assumption [Reg05] says that the distribution $(\mathbf{A}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal)$ is computationally indistinguishable from $(\mathbf{A}, \mathbf{u}^\intercal)$ where $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow D_{\mathbb{Z},\chi}^m$, and $\mathbf{u} \xleftarrow{R} \mathbb{Z}_q^m$, where $n, m, q, \chi$ are lattice parameters and $D_{\mathbb{Z},\chi}$ is the discrete Gaussian distribution with parameter $\chi$. To simplify the presentation in the technical overview, we will use *curly underlines* in place of (small) noise terms. Namely, instead of writing $\mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal$, we simply write $\mathbf{s}^\intercal\underset{\sim}{\mathbf{A}}$.

For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a target vector $\mathbf{y} \in \mathbb{Z}_q^n$, we write $\mathbf{A}_\chi^{-1}(\mathbf{y})$ to denote a random variable $\mathbf{x} \in \mathbb{Z}_q^m$ whose distribution is a discrete Gaussian distribution $D_{\mathbb{Z},\chi}^m$ conditioned on $\mathbf{Ax} = \mathbf{y}$. For ease of notation, we will drop the subscript $\chi$ in this technical overview. A sequence of works [Ajt96, GPV08, ABB10b, ABB10a, CHKP10, MP12] (see also Section 3.2) have shown how to sample a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ together with a trapdoor $\mathsf{td}_\mathbf{A}$ to enable efficient sampling from the distribution $\mathbf{A}^{-1}(\mathbf{y})$ for any target $\mathbf{y} \in \mathbb{Z}_q^n$.

In the following description, we write $\mathbf{I}_n \in \mathbb{Z}_q^{n\times n}$ to denote the $n$-by-$n$ identity matrix and $\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g}^\intercal \in \mathbb{Z}_q^{n\times m}$, where $\mathbf{g}^\intercal = [1 \mid 2 \mid \cdots \mid 2^{\lfloor \log q \rfloor}]$, to denote the standard gadget matrix [MP12].

### 2.1 Starting Point: Single-Authority CP-ABE for Subset Policies

We start by describing a simple CP-ABE for subset policies that lies at the core of our MA-ABE scheme. In the following, let $[L]$ be the universe of attributes. Each ciphertext is associated with a subset $A \subseteq [L]$ and each secret key is associated with a subset $B \subseteq [L]$; decryption succeeds as long as $A \subseteq B$.

- The master public key consists of $(\mathbf{A}_1, \mathbf{B}_1, \mathbf{p}_1), \ldots, (\mathbf{A}_L, \mathbf{B}_L, \mathbf{p}_L) \xleftarrow{R} \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^{n\times m(2L-1)} \times \mathbb{Z}_q^n$.
- The master secret key consist of the trapdoors $\mathsf{td}_{\mathbf{A}_1}, \ldots, \mathsf{td}_{\mathbf{A}_L}$ for $\mathbf{A}_1, \ldots, \mathbf{A}_L$, respectively.

- An encryption of a message bit $\mu \in \{0,1\}$ with respect to a set $X \subseteq [L]$ is a tuple

$$\mathsf{ct} = \left( \left\{ \mathbf{s}^\intercal \mathbf{A}_i \right\}_{i \in X},\ \mathbf{s}^\intercal \sum_{i \in X} \mathbf{B}_i,\ \mathbf{s}^\intercal \sum_{i \in X} \mathbf{p}_i + \mu \cdot \lfloor q/2 \rceil \right),$$

  where $\mathbf{s} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$.

- A secret key for a set $Y \subseteq [L]$ consists of a tuple

$$\mathsf{sk} = \left( \left\{ \mathbf{A}_i^{-1}(\mathbf{p}_i + \mathbf{B}_i \mathbf{r}) \right\}_{i \in Y},\ \mathbf{r} \right),$$

  where $\mathbf{r} \leftarrow D_{\mathbb{Z},\chi}^{m(2L-1)}$ is sampled from a discrete Gaussian distribution.

Decryption uses the fact that

$$-\left( \mathbf{s}^\intercal \sum_{i \in X} \mathbf{B}_i \right) \cdot \mathbf{r} + \sum_{i \in X} \mathbf{s}^\intercal \mathbf{A}_i \cdot \mathbf{A}_i^{-1}(\mathbf{p}_i + \mathbf{B}_i \mathbf{r}) \approx -\mathbf{s}^\intercal \sum_{i \in X} \mathbf{B}_i \mathbf{r} + \mathbf{s}^\intercal \sum_{i \in X} (\mathbf{p}_i + \mathbf{B}_i \mathbf{r}) = \mathbf{s}^\intercal \sum_{i \in X} \mathbf{p}_i,$$

since $\mathbf{r}$ and $\mathbf{A}^{-1}(\cdot)$ are small. Looking ahead to our multi-authority construction, observe that key generation can be carried out in a decentralized manner: given a "public" Gaussian vector $\mathbf{r}$, computing the secret-key components $\mathbf{A}_i^{-1}(\mathbf{p}_i + \mathbf{B}_i \mathbf{r})$ associated with index $i$ only requires knowledge of $\mathbf{B}_i, \mathbf{p}_i$ and the trapdoor for $\mathbf{A}_i$, which are all *specific* to attribute $i$ (and could be independently generated by the $i^{\text{th}}$ authority).

**Selective security.** To argue that this CP-ABE scheme is selectively secure[3], we proceed as follows:

1. First, we show how to sample a secret key for a set $Y \subseteq [L]$ given a trapdoor for $\mathbf{B}_Y$, where $\mathbf{B}_Y \in \mathbb{Z}_q^{n|Y| \times m(2L-1)}$ is the matrix formed by vertically concatenating $\mathbf{B}_i$ for all $i \in Y$.
2. Next, we show that under the LWE assumption, $\mathbf{s}^\intercal \sum_{i \in X} \mathbf{B}_i$ is pseudorandom even given an oracle for $\mathbf{B}_Y^{-1}(\cdot)$ for arbitrary $Y \subseteq [L]$ of the adversary's choosing, provided that for each $Y$, it is the case that $X \not\subseteq Y$. Here, $X \subseteq [L]$ is the set associated with the challenge ciphertext. Technically, we additionally require that $\mathbf{s}^\intercal \mathbf{A}_i$ and $\mathbf{s}^\intercal \sum_{i \in X} \mathbf{p}_i$ are also pseudorandom, but these components are easily handled by the standard LWE assumption. For ease of exposition, we do not focus on these additional components in this overview and refer instead to Sections 4 and 5 for the full description.

For the second step, we prove a more general statement which generalizes the related-trapdoor LWE lemma previously introduced by Brakerski and Vaikuntanathan [BV22] in the context of constructing compact CP-ABE for circuits.

**Generalized related-trapdoor LWE.** Our generalized related-trapdoor LWE assumption asserts that for any non-zero vector $\mathbf{u} \in \{0,1\}^L$, the vector $\mathbf{s}^\intercal (\mathbf{u}^\intercal \otimes \mathbf{I}_n)\mathbf{B}$ is pseudorandom given an oracle for the function $(\mathbf{M}, \mathbf{t}) \mapsto ((\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B})^{-1}(\mathbf{t})$, as long as the matrix $\bar{\mathbf{M}} = \left[ \begin{smallmatrix} \mathbf{M} \\ \mathbf{u}^\intercal \end{smallmatrix} \right] \in \mathbb{Z}_q^{(k+1) \times L}$ is full rank (and $k < L$).[4] To show that the standard LWE assumption implies the generalized related-trapdoor LWE assumption, we take an LWE matrix $\hat{\mathbf{A}}$ and the vector $\mathbf{u} \in \{0,1\}^L$, and we set the matrix $\mathbf{B}$ to be

$$\mathbf{B} = \left[ \hat{\mathbf{A}} \mid \hat{\mathbf{A}}\mathbf{R} + \mathbf{U}^\perp \otimes \mathbf{G} \right]$$

[3] In the selective security game, the adversary starts by committing to the set $X$ associated with the challenge ciphertext. The reduction algorithm is then allowed to program $X$ into the public parameters of the scheme.

[4] Some restriction on $\mathbf{M}$ is also necessary. For instance, it is easy to distinguish $\mathbf{s}^\intercal (\mathbf{u} \otimes \mathbf{I}_n)\mathbf{B}$ if $\mathbf{M} = \mathbf{u}^\intercal$, or more generally, if $\mathbf{u}_0^\intercal \mathbf{M} = \mathbf{u}$ for some $\mathbf{u}_0 \in \{0,1\}^k$.

where $\mathbf{R}$ is a (random) low-norm matrix and $\mathbf{U}^\perp \in \{0,1\}^{L\times(L-1)}$ is a full-rank basis for the kernel of $\mathbf{u}^\intercal$. By design, $(\mathbf{u}\otimes\mathbf{I}_n)\mathbf{B} = [(\mathbf{u}\otimes\mathbf{I}_n)\hat{\mathbf{A}} \mid (\mathbf{u}\otimes\mathbf{I}_n)\hat{\mathbf{A}}\mathbf{R}]$ which means we do *not* know a trapdoor for $(\mathbf{u}\otimes\mathbf{I}_n)\mathbf{B}$. On the other hand,

$$(\mathbf{M}\otimes\mathbf{I}_n)\mathbf{B}\underbrace{\begin{bmatrix}-\mathbf{R}\\ \mathbf{I}_{\hat{m}(L-1)}\end{bmatrix}}_{\tilde{\mathbf{R}}} = (\mathbf{M}\otimes\mathbf{I}_n)(\mathbf{U}^\perp\otimes\mathbf{G}) = \mathbf{M}\mathbf{U}^\perp\otimes\mathbf{G}.$$

When $\bar{\mathbf{M}} = \left[\begin{smallmatrix}\mathbf{M}\\ \mathbf{u}^\intercal\end{smallmatrix}\right]$ is full rank, then $\mathbf{M}\mathbf{U}^\perp$ is also full rank. Since $\tilde{\mathbf{R}}$ is low-norm, it is a trapdoor for $(\mathbf{M}\otimes\mathbf{I}_n)\mathbf{B}$ (see [MP12] and Corollary 3.12).

Returning to the proof of selective security for the above CP-ABE scheme, observe that showing $\mathbf{s}^\intercal\sum_{i\in X}\mathbf{B}_i$ given an oracle for $\mathbf{B}_Y^{-1}(\cdot)$ directly maps to an instance of the related-trapdoor LWE assumption:

- Let $\mathbf{B}\in\mathbb{Z}_q^{nL\times m(2L-1)}$ be the matrix obtained by vertically stacking $\mathbf{B}_1,\ldots,\mathbf{B}_L\in\mathbb{Z}_q^{n\times m(2L-1)}$.
- The vector $\mathbf{u}\in\{0,1\}^L$ is the indicator vector for the challenge set $X$. Namely, $u_i = 1$ if $i\in X$ and 0 otherwise. Then, $(\mathbf{u}\otimes\mathbf{I}_n)\mathbf{B} = \sum_{i\in X}\mathbf{B}_i$.
- The oracle $\mathbf{B}_Y^{-1}(\cdot)$ can be simulated by querying the related-trapdoor oracle on matrix $\mathbf{M}_Y\in\mathbb{Z}_q^{|Y|\times L}$ formed by taking the rows of $\mathbf{I}_L$ corresponding to the indices in $Y$. In this case $(\mathbf{M}\otimes\mathbf{I}_n)\mathbf{B} = \mathbf{B}_Y$ defined previously. Moreover, by construction of $\mathbf{M}_Y$, whenever $X\not\subseteq Y$, we have that $\mathbf{u}^\intercal$ is *not* in the row-span of $\mathbf{M}_Y$.

Finally, we remark here that the original version of the related-trapdoor LWE assumption formulated by Brakerski and Vaikuntanathan [BV22] considered the special case where the matrix $\mathbf{M}$ is a row vector with a specific structure.[5] Our formulation considers a general matrix $\mathbf{M}$ which is useful for constructing an ABE scheme with a *distributed* setup. We also note that this type of trapdoor sampling was also implicit in the CP-ABE construction of Datta et al. [DKW21a]; however, they critically relied on noise flooding to simulate the analog of the $((\mathbf{M}\otimes\mathbf{I}_n)\mathbf{B})^{-1}(\cdot)$ oracle. As a result, the security of their scheme relied on LWE with a super-polynomial modulus-to-noise ratio in the random oracle model. In this work, we both provide a modular description of the core trapdoor sampling lemma (Section 4) and then show how to leverage it to obtain a multi-authority ABE for subset policies using LWE with a polynomial modulus-to-noise ratio in the random oracle model (Section 5). We are optimistic that our generalized version of the related trapdoor LWE assumption will also be useful for analyzing the security of other lattice-based constructions.

## 2.2 MA-ABE for Subset Policies in the Random Oracle Model

First, we observe that our core CP-ABE scheme naturally extends to yield a MA-ABE scheme for subset policies in the random oracle model. We make the following modifications to the base scheme:

- The authority associated with attribute $i$ samples $\mathbf{A}_i,\mathbf{B}_i,\mathbf{p}_i$ along with a trapdoor $\mathsf{td}_{\mathbf{A}_i}$ for $\mathbf{A}_i$.
- To generate a key for a user with identifier gid, we derive $\mathbf{r}$ *deterministically* from $\mathsf{H}(\mathsf{gid})$ and output $\mathbf{A}^{-1}(\mathbf{p}_i+\mathbf{B}_i\mathbf{r})$.

Security of the core CP-ABE implies that the ensuing MA-ABE scheme remains secure as long as no authority is corrupted. On the other hand, it is easy to see that the scheme is insecure if we allow authority corruptions, since we can use an authority's trapdoor to recover the LWE secret $\mathbf{s}$ from $\mathbf{s}^\intercal\mathbf{A}_i$.

**Security with authority corruptions.** To defend against corrupted authorities, we modify the ciphertext structure. Instead of having a single LWE secret $\mathbf{s}$ that is shared across *authorities*, we instead sample a fresh $\mathbf{s}_i$ for each attribute $i\in X$. That is, the ciphertext is now given by:

$$\mathsf{ct} = \left(\left\{\mathbf{s}_i^\intercal\mathbf{A}_i\right\}_{i\in X},\ \sum_{i\in X}\mathbf{s}_i^\intercal\mathbf{B}_i,\ \sum_{i\in X}\mathbf{s}_i^\intercal\mathbf{p}_i+\mu\cdot\lfloor q/2\rceil\right)$$

[5] Concretely, $\mathbf{u}^\intercal = [1\mid\mathbf{x}^\intercal]$ and $\mathbf{M} = [1\mid\mathbf{y}^\intercal]$ for some $\mathbf{x},\mathbf{y}\in\{0,1\}^{L-1}$. The adversary is restricted to queries $\mathbf{y}\neq\mathbf{x}$, which is implied by our requirement that $\bar{\mathbf{M}}$ has full rank.

Key generation proceeds as before. Decryption still follows from a similar relation as before:

$$-\underbrace{\left(\sum_{i\in X} \mathbf{s}_i^\top \mathbf{B}_i\right)}\cdot \mathbf{r} + \sum_{i\in X} \underbrace{\mathbf{s}_i^\top \mathbf{A}_i}\cdot \mathbf{A}_i^{-1}(\mathbf{p}_i + \mathbf{B}_i\mathbf{r}) = \sum_{i\in X} \mathbf{s}_i^\top \mathbf{p}_i.$$

**Static security with authority corruptions.** We now argue that the resulting MA-ABE scheme is statically secure.[6] Let $C$ denote the set of authorities that are corrupted. The adversary gets to choose the public keys and secret keys for authorities in $C$. In the multi-authority setting, a secret-key query consists of a pair $(Y, \mathsf{gid})$ where $Y$ is a set of *honest* authorities (i.e., $Y \cap C = \emptyset$) and gid is the user identifier. Let $X$ be the set of authorities associated with the challenge ciphertext. The admissibility criterion is that $X \nsubseteq Y \cup C$.

The proof of security proceeds similarly to that of our core CP-ABE, except we replace the challenge set $X$ with the set $X \setminus C$. Since $Y \cap C = \emptyset$, the MA-ABE admissibility criterion $X \nsubseteq Y \cup C$ is equivalent to $X \setminus C \nsubseteq Y$, which coincides with the criterion from our CP-ABE analysis. In particular, the security reduction can basically *ignore* the ciphertext components associated with corrupted authorities (since the ciphertext component of each authority is associated with *independent* LWE secrets $\mathbf{s}_i$) and just focus on the attributes controlled by the honest authorities. The general argument again relies on our (generalized) related-trapdoor LWE assumption:

1. First, we show how to sample a secret key for $Y$ given a trapdoor for $\mathbf{B}_Y$ (where $\mathbf{B}_Y \in \mathbb{Z}_q^{n|Y|\times m(2L-1)}$ is again the matrix formed by vertically stacking the matrices $\mathbf{B}_i$ associated with the authorities $i \in Y$.

2. As in the analysis of the CP-ABE scheme, we use the oracle in the related-trapdoor LWE assumption to compute $\mathbf{B}_Y^{-1}(\cdot)$ in the proof. Arguing the correctness of this step additionally requires the ability to "program" the random oracle. This is because in the real scheme, the secret keys are sampled by computing $\mathbf{r} \leftarrow \mathsf{H}(\mathsf{gid})$ and then sampling $\mathbf{u}_i \leftarrow \mathbf{A}_i(\mathbf{p}_i + \mathbf{B}_i\mathbf{r})$ for each $i \in X$. The reduction algorithm will instead sample $\mathbf{u}_i \leftarrow D_{\mathbb{Z},\chi}^m$ itself and then obtain $\mathbf{r} \in \mathbb{Z}_q^{m(2L-1)}$ using its oracle $\mathbf{B}_Y^{-1}(\cdot)$. In the random oracle model, the reduction then programs $\mathsf{H}(\mathsf{gid})$ to $\mathbf{r}$. We refer to Section 5 for more details.

3. Finally, to simulate the challenge ciphertext, the reduction algorithm samples a random $\mathbf{s}_i \xleftarrow{R} \mathbb{Z}_q^n$ for each corrupted authority $i \in C$. For the honest authorities $i \in X \setminus C$, the reduction sets the secret key to be $\hat{\mathbf{s}}_i$ and programs $\mathbf{s}_i := \mathbf{s} + \hat{\mathbf{s}}_i$, where $\mathbf{s}$ is the secret in the related-trapdoor

We provide the formal analysis in Section 5. This construction yields a MA-ABE scheme for subset policies from the related-trapdoor LWE assumption in the random oracle model. The related-trapdoor LWE assumption we rely on here reduces to the standard LWE assumption with a polynomial modulus-to-noise ratio. This yields Theorem 1.2.

## 2.3 Removing Random Oracles via Evasive LWE

To obtain an MA-ABE construction *without* random oracles, we describe a way to *concretely* implement the hash function H in our basic construction above. Our specific instantiation relies on computing a subset product of low-norm matrices. Specifically, let $\mathbf{D}_0, \mathbf{D}_1 \in \mathbb{Z}_q^{m\times m}$ be low-norm matrices. These are fixed public matrices that will be included as part of the global parameters. For an input $x \in \{0,1\}^\ell$, we define $\mathsf{H}(x) := \left(\prod_{i\in[\ell]} \mathbf{D}_{x_i}\right)\boldsymbol{\eta} \in \mathbb{Z}_q^m$, where $\boldsymbol{\eta} \in \mathbb{Z}_q^m$ is the first canonical basis vector. Previously, Boneh et al. [BLMR13] showed that for any sequence of $x_1, \ldots, x_k \in \{0,1\}^\ell$ the values $\left\{\underbrace{\mathbf{s}^\top \mathsf{H}(x_i)}\right\}_{i\in[k]}$ are pseudorandom. While we do not know how to prove security of the MA-ABE construction instantiated with this subset-product hash function using the plain learning with errors assumption, we show how to do so using the recently-introduced evasive LWE assumption by Wee [Wee22] and Tsabury [Tsa22].

[6] In the static security model [RW15], we require the adversary to commit to the set of corrupted authorities, the secret-key queries, and the challenge ciphertext query at the beginning of the security game. Previous lattice-based MA-ABE constructions were also analyzed in the static security model [DKW21a].

**Evasive LWE.** We start by describing a variant of the evasive LWE assumption introduced by Wee [Wee22] and refer to Section 3.2 for the formal description. Let $\mathbf{P}_1, \ldots, \mathbf{P}_\ell$ be drawn from some efficiently-sampleable distribution of matrices. Roughly speaking, the evasive LWE assumption says that if the distribution $\{\mathbf{A}_i\,,\ \mathbf{s}^\intercal\mathbf{P}_i\}_{i\in[\ell]}$ is pseudorandom, then the distributions

$$\{\mathbf{A}_i\,,\ \mathbf{s}^\intercal\mathbf{A}_i\,,\ \mathbf{A}_i^{-1}(\mathbf{P}_i)\}_{i\in[\ell]} \quad \text{and} \quad \{\mathbf{A}_i\,,\ \mathbf{u}_i^\intercal\,,\ \mathbf{A}_i^{-1}(\mathbf{P}_i)\}_{i\in[\ell]}$$

are computationally indistinguishable. Intuitively, the evasive LWE assumption says that the presence of $\mathbf{A}_i^{-1}(\mathbf{P}_i)$ does *not* help break LWE so long as $\mathbf{s}^\intercal\mathbf{P}_i$ is pseudorandom. Indeed, if the distinguisher multiplied $\mathbf{s}^\intercal\mathbf{A}$ with $\mathbf{A}^{-1}(\mathbf{P})$, then it roughly obtains $\mathbf{s}^\intercal\mathbf{P}$, which is pseudorandom by assumption.

In the context of our MA-ABE scheme, the matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ will be associated with the public keys for the honest authorities, and the columns of $\mathbf{P}_i$ will consists of $\mathbf{p}_i + \mathbf{B}_i\mathbf{r}_{\mathsf{gid}}$ for the user identifiers gid that appear in the adversary's secret-key queries. By setting $\mathbf{P}_i$ properly (see Section 6), the reduction algorithm can in turn answer the secret-key queries *without* switching to using a trapdoor for $\mathbf{B}_Y$ to answer key queries. We highlight the key differences in reduction strategies here:

- Previously (Section 2.2), the reduction sampled $\mathbf{u}_i$ itself and used the trapdoor for $\mathbf{B}_Y$ to sample $\mathbf{r} = \mathsf{H}(\mathsf{gid})$. This was necessary because the reduction did not (and cannot) possess a trapdoor for each $\mathbf{A}_i$ to sample $\mathbf{u}_i$ as in the real scheme. If the reduction did possess such a trapdoor for every $i$ that appears in the challenge ciphertext, then it could trivially break security itself. Then, to ensure consistency of the sampled key with respect to the outputs of H, this requires the reduction to *program* the outputs of H. Hence, we model H as a random oracle in this case.
- In contrast, when we use evasive LWE, the reduction computes $\mathbf{r} = \mathsf{H}(\mathsf{gid})$ normally and then directly constructs $\mathbf{u}_i$ using the terms provided in the evasive LWE challenge. These terms can be simulated *without* knowledge of a trapdoor for $\mathbf{A}_i$. Observe that this strategy only relies on the ability to compute $\mathsf{H}(\cdot)$, *not* the ability to program its outputs. In general, the evasive LWE assumptions allows us to reduce the task of proving security to that of reasoning about the pseudorandomness of LWE samples with respect to correlated public matrices. In the latter distribution, there are no Gaussian samples, and no need to implement any kind of trapdoor sampling.

When we use evasive LWE, the computation of $\mathbf{s}^\intercal\mathbf{P}$ essentially translates to computing $\mathbf{s}^\intercal\mathsf{H}(\mathsf{gid})$, which is pseudorandom by the Boneh et al. [BLMR13] analysis. We refer to Section 6 for the formal description.

While the evasive LWE assumption is much less well understood compared to the classic LWE assumption, proving security under evasive LWE at the minimum indicates that replacing the random oracle with a subset-product hash function is a sound *heuristic* for constructing an MA-ABE scheme in the plain model. It is an interesting challenge to try and prove the security of our construction from the plain LWE assumption; such a proof would provide the first construction of MA-ABE from standard assumptions in the plain model. Alternatively, it is also interesting to further cryptanalyze the evasive LWE assumption.

# 3 Preliminaries

We write $\lambda$ to denote the security parameter. For a positive integer $n \in \mathbb{N}$, we write $[n]$ to denote the set $\{1, \ldots, n\}$. For a positive integer $q \in \mathbb{N}$, we write $\mathbb{Z}_q$ to denote the integers modulo $q$. We use bold uppercase letters to denote matrices (e.g., $\mathbf{A}, \mathbf{B}$) and bold lowercase letters to denote vectors (e.g., $\mathbf{u}$, $\mathbf{v}$). We use non-boldface letters to refer to their components: $\mathbf{v} = (v_1, \ldots, v_n)$. For matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n\times m}$, we write $\mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \in \mathbb{Z}_q^{n\ell\times m\ell}$ to denote the block diagonal matrix with blocks $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ along the main diagonal (and 0s elsewhere).

We write $\mathsf{poly}(\lambda)$ to denote a function that is $O(\lambda^c)$ for some $c \in \mathbb{N}$ and $\mathsf{negl}(\lambda)$ to denote a function that is $o(\lambda^{-c})$ for all $c \in \mathbb{N}$. An algorithm is efficient if it runs in probabilistic polynomial time in its input length. We say that two families of distributions $\mathcal{D}_1 = \{\mathcal{D}_{1,\lambda}\}_{\lambda\in\mathbb{N}}$ and $\mathcal{D}_2 = \{\mathcal{D}_{2,\lambda}\}_{\lambda\in\mathbb{N}}$ are computationally indistinguishable if no efficient algorithm can distinguish them with non-negligible probability. We denote this by writing $\mathcal{D}_1 \overset{c}{\approx} \mathcal{D}_2$. We say they are statistically indistinguishable if the statistical distance $\Delta(\mathcal{D}_1, \mathcal{D}_2)$ is bounded by a negligible function in $\lambda$ and denote this by writing $\mathcal{D}_1 \overset{s}{\approx} \mathcal{D}_2$. We say a distribution $\mathcal{D}$ is $B$-bounded if $\Pr[|x| \le B : x \leftarrow \mathcal{D}] = 1$.

## 3.1 Multi-Authority Attribute-Based Encryption

In this section, we introduce the syntax of a multi-authority ABE scheme [LW11]. We start with the definition of a monotone access structure [Bei96].

**Definition 3.1** (Access Structure [Bei96])**.** Let $S$ be a set and let $2^S$ denote the power set of $S$ (i.e., the set of all subsets of $S$). An access structure on $S$ is a set $\mathbb{A} \subseteq 2^S \setminus \emptyset$ of non-empty subsets of $S$. We refer to the elements of $\mathbb{A}$ as the *authorized* sets and those not in $\mathbb{A}$ as the *unauthorized* sets. We say an access structure is *monotone* if for all sets $B, C \in 2^S$, if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$.

**Definition 3.2** (Multi-Authority ABE [LW11, RW15, adapted])**.** Let $\lambda$ be a security parameter, $\mathcal{M}$ be a message space, $\mathcal{AU} = \{\mathcal{AU}_\lambda\}_{\lambda\in\mathbb{N}}$ be the universe of authority identifiers, and $\mathcal{GID} = \{\mathcal{GID}_\lambda\}_{\lambda\in\mathbb{N}}$ be the universe of global identifiers for users. To simplify the exposition, we follow the convention in [RW15, DKW21a] and assume that each authority controls a single attribute; this definition generalizes naturally to the setting where each authority controls an arbitrary polynomial number of attributes (see [RW15]). A multi-authority attribute-based encryption scheme for a class of policies $\mathcal{P} = \{\mathcal{P}_\lambda\}_{\lambda\in\mathbb{N}}$ (each described by a monotone access structure on a subset of $\mathcal{AU}$) consists of a tuple of efficient algorithms $\Pi_{\mathsf{MA\text{-}ABE}} = (\mathsf{GlobalSetup}, \mathsf{AuthSetup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ with the following properties:

- $\mathsf{GlobalSetup}(1^\lambda) \to \mathsf{gp}$: On input the security parameter $\lambda$, the global setup algorithm outputs the global parameters gp.
- $\mathsf{AuthSetup}(\mathsf{gp}, \mathsf{aid}) \to (\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}})$: On input the global parameters gp and an authority identifier $\mathsf{aid} \in \mathcal{AU}$, the authority setup algorithm outputs a public key $\mathsf{pk}_{\mathsf{aid}}$ and a master secret key $\mathsf{msk}_{\mathsf{aid}}$.
- $\mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}, \mathsf{gid}) \to \mathsf{sk}$: On input the global parameters gp, the authority's master secret key msk, and the user identifier $\mathsf{gid} \in \mathcal{GID}$, the key-generation algorithm outputs a decryption key sk.
- $\mathsf{Encrypt}(\mathsf{gp}, \mathbb{A}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mu) \to \mathsf{ct}$: On input the global parameters gp, an access structure $\mathbb{A} \in \mathcal{P}$ on a set of authorities $A \subseteq \mathcal{AU}$, the set of public keys $\mathsf{pk}_{\mathsf{aid}}$ associated with each authority $\mathsf{aid} \in A$, and a message $\mu \in \mathcal{M}$, the encryption algorithm outputs a ciphertext ct.
- $\mathsf{Decrypt}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mathsf{ct}) \to \mu$: On input the global parameters gp, a collection of secret keys $\mathsf{sk}_{\mathsf{aid}}$ issued by a set of authorities $\mathsf{aid} \in A$, and a ciphertext ct, the decryption algorithm outputs a message $\mu \in \mathcal{M} \cup \{\perp\}$.

Moreover, $\Pi_{\mathsf{MA\text{-}ABE}}$ should satisfy the following properties:

- **Correctness:** The exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\lambda \in \mathbb{N}$, every message $\mu \in \mathcal{M}$, every identifier $\mathsf{gid} \in \mathcal{GID}_\lambda$, every set of authorities $A \subseteq \mathcal{AU}_\lambda$, every access structure $\mathbb{A} \in \mathcal{P}_\lambda$ on $A$, and every subset of authorized authorities $B \in \mathbb{A}$,

$$\Pr\left[\mu' = \mu : \begin{array}{c} \mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda); \\ \forall \mathsf{aid} \in A : (\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}}) \leftarrow \mathsf{AuthSetup}(\mathsf{gp}, \mathsf{aid}); \\ \forall \mathsf{aid} \in B : \mathsf{sk}_{\mathsf{gid},\mathsf{aid}} \leftarrow \mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, \mathsf{gid}); \\ \mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{gp}, \mathbb{A}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mu); \\ \mu' \leftarrow \mathsf{Decrypt}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{gid},\mathsf{aid}}\}_{\mathsf{aid}\in B}, \mathsf{ct}) \end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

- **Static security:** For a security parameter $\lambda \in \mathbb{N}$, an adversary $\mathcal{A}$, and a bit $b \in \{0, 1\}$, we define the static security game for an multi-authority ABE scheme as follows:
  - **Setup:** The challenger starts by sampling $\mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$ and gives gp to $\mathcal{A}$.
  - **Attacker queries.** The adversary $\mathcal{A}$ now specifies the following:
    - A set $C \subseteq \mathcal{AU}_\lambda$ of corrupt authorities together with a public key $\mathsf{pk}_{\mathsf{aid}}$ for each corrupt authority $\mathsf{aid} \in C$.
    - A set $\mathcal{N} \subseteq \mathcal{AU}_\lambda$ of non-corrupt authorities, where $\mathcal{N} \cap C = \emptyset$.

* A set $Q = \{(\mathsf{gid}, A)\}$ of secret key queries where each query consists of a global identifier $\mathsf{gid} \in \mathcal{GID}_\lambda$ and a subset of non-corrupt authorities $A \subset \mathcal{N}$.
* A pair of challenge messages $\mu_0, \mu_1 \in \mathcal{M}$, a set of authorities $A^* \subseteq C \cup \mathcal{N}$, and an access structure $\mathbb{A} \in \mathcal{P}_\lambda$ on $A^*$.

– **Challenge.** The challenger then samples $(\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}}) \leftarrow \mathsf{AuthSetup}(\mathsf{gp}, \mathsf{aid})$ for each authority $\mathsf{aid} \in \mathcal{N}$. It responds to the adversary with the following:
* The public keys $\mathsf{pk}_{\mathsf{aid}}$ for the non-corrupted authority $\mathsf{aid} \in \mathcal{N}$.
* For each secret-key query $(\mathsf{gid}, A)$, the secret keys $\mathsf{sk}_{\mathsf{gid},\mathsf{aid}} \leftarrow \mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, \mathsf{gid})$ for each $\mathsf{aid} \in A$.
* The challenge ciphertext $\mathsf{ct}_b \leftarrow \mathsf{Encrypt}(\mathsf{gp}, \mathbb{A}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid} \in A^*}, \mu_b)$.

– **Output phase:** Finally, algorithm $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$, which is the output of the experiment.

We say an adversary $\mathcal{A}$ is admissible for the above security game if $A^* \cap C \notin \mathbb{A}$ and moreover, for every secret key query $(\mathsf{gid}, A)$, it holds that $(A \cup C) \cap A^* \notin \mathbb{A}$. Finally, we say $\Pi_{\mathsf{MA\text{-}ABE}}$ satisfies static security if for all efficient and admissible adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $|\Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]| = \mathsf{negl}(\lambda)$ in the above security game.

**Remark 3.3** (Static Security in the Random Oracle Model)**.** Following [RW15, DKW21a], we also extend Definition 3.2 to the random oracle model [BR93]. In this setting, we assume that a global hash function $\mathsf{H}$ (modeled as a random oracle) is published as part of the global public parameters and accessible to all of the parties in the system. When extending static security to the random oracle model, we require that the adversary submits its random oracle queries as part of its initial query in the static security game. The challenger then includes the responses to the random oracle queries as part of the challenge. We also allow the adversary to further query the random oracle during the challenge phase of the game.

**Remark 3.4** (Security Notions)**.** The static security requirement in Definition 3.2 requires that the adversary commits to *all* of its queries upfront. A stronger notion of security is adaptive security under static corruptions [LW11] which requires the adversary pre-commit to the set of corrupted authorities, but thereafter, the adversary can adaptively make secret-key queries both before and after making its challenge ciphertext query. We can also consider intermediate notions where the adversary needs to commit to the policy associated with the challenge ciphertext, but can then issue secret key queries adaptively (i.e., the analog of "selective security" in single-authority ABE). Achieving stronger notions of security (beyond static security) for multi-authority ABE from lattice-based assumptions is an interesting open problem.

**Multi-authority ABE for subset policies.** Our focus in this work is on constructing multi-authority ABE for the class of subset policies. Here, the ciphertext is associated with a set of authorities $A$ and decryption succeeds whenever a user possesses keys from a set of authorities $B$ where $A \subseteq B$. We define this more formally below.

**Definition 3.5** (Multi-Authority ABE for Subset Policies)**.** Let $\lambda$ be a security parameter and $\mathcal{AU} = \{\mathcal{AU}_\lambda\}_{\lambda \in \mathbb{N}}$ be the universe of authority identifiers. We define the class of subset policies $\mathcal{P} = \{\mathcal{P}_\lambda\}_{\lambda \in \mathbb{N}}$ to be the set

$$\mathcal{P}_\lambda = \{\mathbb{A} : \mathbb{A} = \{B : A \subseteq B\} \text{ where } A \subseteq \mathcal{AU}_\lambda\}\,.$$

Notably, an access structure $\mathbb{A}$ for a subset policy is fully determined by the set $A \subseteq \mathcal{AU}_\lambda$. Thus, when describing an MA-ABE scheme $\Pi_{\mathsf{MA\text{-}ABE}} = (\mathsf{GlobalSetup}, \mathsf{AuthSetup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ for the class of subset policies, we omit the specification of $\mathbb{A}$ in the encryption algorithm and have the encryption algorithm only take as input the public keys associated with the authorities in $A$. More precisely, we modify the syntax of the encryption algorithm as follows:

- $\mathsf{Encrypt}(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid} \in A}, \mu) \to \mathsf{ct}$: On input the global parameters $\mathsf{gp}$, the set of public keys $\mathsf{pk}_{\mathsf{aid}}$ associated with each authority $\mathsf{aid} \in A$, and a message $\mu \in \mathcal{M}$, the encryption algorithm outputs a ciphertext $\mathsf{ct}$.

**Remark 3.6** (Multi-Authority ABE for DNFs)**.** A multi-authority ABE scheme for subset policies directly implies a multi-authority ABE scheme for access structures that can be decided by a polynomial-size conjunction or a DNF formula. First, we define the notion of an access structure decidable by a Boolean formula. Let $\mathbb{A}$ be an access structure on a set $A = \{a_1, \ldots, a_n\}$. For a subset $B \subseteq A$, we define indicator bits $b_1, \ldots, b_n$ where $b_i = 1$ if $a_i \in B$ and 0 otherwise. We say that $\mathbb{A}$ can be computed by a Boolean formula $\varphi$ if there exists a Boolean formula $\varphi\colon \{0,1\}^n \to \{0,1\}$ such that $B \in \mathbb{A}$ if and only if $\varphi(b_1, \ldots, b_n) = 1$. It is straightforward to use an MA-ABE scheme for subset policies to construct MA-ABE schemes for policies computable by either a conjunction or a DNF:

- **Conjunction**: Let $\mathbb{A}$ be an access structure on $A$ that is computable by a conjunction on variables $b_{i_1}, \ldots, b_{i_d}$. This is equivalent to a subset policy for the set $\{a_{i_1}, \ldots, a_{i_d}\}$.
- **DNF formulas:** Let $\mathbb{A}$ be an access structure on $A$ that is computable by a DNF $\varphi\colon \{0,1\}^n \to \{0,1\}$. By construction, we can write $\varphi(x_1, \ldots, x_n) = \bigvee_{i\in[t]} \varphi_i(x_1, \ldots, x_n)$, where each $\varphi_i$ is a conjunction. In this case, decryption succeeds as long as at least one of the $\varphi_i$ is satisfied. In this case, we simply concatenate $t$ ciphertexts together, where the $i^{\text{th}}$ ciphertext is an encryption to the $i^{\text{th}}$ conjunction $\varphi_i$. Correctness follows by construction while security follows by a standard hybrid argument.

**Remark 3.7** (Multi-Authority ABE for $k$-CNFs)**.** In the single-authority setting, ABE for subset policies implies an ABE scheme for $k$-CNF formulas for constant $k \in \mathbb{N}$ [Tsa19, GLW21]. However, this generic approach does not easily translate to the multi-authority setting. Here, a $k$-CNF formula $\varphi\colon \{0,1\}^n \to \{0,1\}$ can be written as $\varphi(x_1, \ldots, x_n) = \bigwedge_{i\in[t]} \varphi_i(x_1, \ldots, x_n)$, where each clause $\varphi_i(x_1, \ldots, x_n)$ is a disjunction on up to $k$ variables. To support $k$-CNF formulas $\varphi\colon \{0,1\}^n \to \{0,1\}$ on a set $A = \{a_1, \ldots, a_n\}$, the approach is to first define a universe $U$ of size $|U| = O(kn^k)$, where each element $u \in U$ is associated with a distinct subset of $S_u \subseteq A$ of size $|S_u| \leq k$. A secret key for $a_i$ consists of secret keys for all $u \in U$ where $a_i \in S_u$. A $k$-CNF policy $\varphi(x_1, \ldots, x_n) = \bigwedge_{i\in[t]} \varphi_i(x_1, \ldots, x_n)$ where each clause $\varphi_i$ depends on a set $T_i \subseteq A$ of at most $k$ variables corresponds to a subset policy for the set $\{u_{T_1}, \ldots, u_{T_t}\}$.

In the multi-authority setting, different authorities own the different attributes $a_1, \ldots, a_n$. To implement $k$-CNF policies as subset policies via the above transformation, we require a multi-authority ABE scheme that supports subset policies where the basic attributes are *combinations* of attributes from *different* authorities. This conflicts with the requirement that authorities be independent in the multi-authority setting. It is an interesting question to construct a multi-authority ABE scheme capable of supporting $k$-CNF formulas from one that supports subset policies.

## 3.2 Lattice Preliminaries

Throughout this work, we always use the $\ell_\infty$ norm for vectors and matrices. Specifically, for a vector $\mathbf{u}$, we write $\|\mathbf{u}\| := \max_i |x_i|$, and for a matrix $\mathbf{A}$, we write $\|\mathbf{A}\| = \max_{i,j} |A_{i,j}|$. For a dimension $k \in \mathbb{N}$, we write $\mathbf{I}_k \in \mathbb{Z}_q^{k\times k}$ to denote the $k$-by-$k$ identity matrix.

**Discrete Gaussians.** We write $D_{\mathbb{Z},\chi}$ to denote the (centered) discrete Gaussian distribution over $\mathbb{Z}$ with parameter $\chi \in \mathbb{R}^+$. For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times t}$, and a vector $\mathbf{v} \in \mathbb{Z}_q^n$, we write $\mathbf{A}_\chi^{-1}(\mathbf{v})$ to denote a random variable $\mathbf{x} \leftarrow D_{\mathbb{Z},\chi}^m$ conditioned on $\mathbf{Ax} = \mathbf{v} \bmod q$. We extend $\mathbf{A}_s^{-1}$ to matrices by applying $\mathbf{A}_s^{-1}$ to each column of the input. Throughout this work, we will use the following standard tail bound on Gaussian distributions:

**Fact 3.8** (Gaussian Tail Bound)**.** Let $\lambda$ be a security parameter and $s = s(\lambda)$ be a Gaussian width parameter. Then, for all polynomials $n = n(\lambda)$, there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$,

$$\Pr\left[\|\mathbf{v}\| > \sqrt{\lambda}s : \mathbf{v} \leftarrow D_{\mathbb{Z},s}^n\right] = \mathsf{negl}(\lambda).$$

**Assumption 3.9** (Learning with Errors [Reg05])**.** Let $\lambda$ be a security parameter and let $n = n(\lambda)$, $m = m(\lambda)$, $q = q(\lambda)$, $\chi = \chi(\lambda)$ be integers. Then, the decisional learning with errors assumption $\mathsf{LWE}_{n,m,q,\chi}$ states that for $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow D_{\mathbb{Z},\chi}^m$, and $\mathbf{u} \xleftarrow{R} \mathbb{Z}_q^m$,

$$(\mathbf{A}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal) \overset{c}{\approx} (\mathbf{A}, \mathbf{u}).$$

**The gadget matrix.** We recall the definition of the gadget matrix [MP12]. For positive integers $n, q \in \mathbb{N}$, let $\mathbf{G}_n = \mathbf{I}_n \otimes \mathbf{g}^\intercal \in \mathbb{Z}_q^{n \times m}$ be the gadget matrix where $\mathbf{g}^\intercal = [1, 2, \ldots, 2^{\log q - 1}]$ and $m = n \lceil \log q \rceil$. The inverse function $\mathbf{G}_n^{-1} : \mathbb{Z}_q^{n \times t} \to \mathbb{Z}_q^{m \times t}$ expands each entry $x \in \mathbb{Z}_q$ into a column of size $\lceil \log q \rceil$ consisting of the bits in the binary representation of $x$. By construction, for every matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times t}$, it follows that $\mathbf{G}_n \cdot \mathbf{G}_n^{-1}(\mathbf{A}) = \mathbf{A} \bmod q$. When the lattice dimension $n$ is clear, we will omit the subscript and simply write $\mathbf{G}$ and $\mathbf{G}^{-1}(\cdot)$ to denote $\mathbf{G}_n$ and $\mathbf{G}_n^{-1}(\cdot)$.

**Lattice trapdoors.** In this work, we use the gadget trapdoors introduced by Micciancio and Peikert [MP12]. Our description below follows many of the notational conventions from [BTVW17].

**Theorem 3.10** (Lattice Trapdoors [Ajt96, GPV08, ABB10b, ABB10a, CHKP10, MP12])**.** *Let $n, m, q$ be lattice parameters. Then there exist efficient algorithms* (TrapGen, SamplePre) *with the following syntax:*

- $\mathsf{TrapGen}(1^n, q, m) \to (\mathbf{A}, \mathsf{td}_\mathbf{A})$*: On input the lattice dimension $n$, the modulus $q$, the number of samples $m$, the trapdoor-generation algorithm outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ together with a trapdoor $\mathsf{td}_\mathbf{A}$.*
- $\mathsf{SamplePre}(\mathbf{A}, \mathsf{td}_\mathbf{A}, \mathbf{v}, s) \to \mathbf{u}$*: On input a matrix $\mathbf{A}$, a trapdoor $\mathsf{td}_\mathbf{A}$, a target vector $\mathbf{v}$, and a Gaussian width parameter $s$, the preimage-sampling algorithm outputs a vector $\mathbf{u}$.*

*Moreover, there exists a polynomial $m_0 = m_0(n, q) = O(n \log q)$ such that for all $m \geq m_0$, the above algorithms satisfy the following properties:*

- **Trapdoor distribution:** *The matrix $\mathbf{A}$ output by $\mathsf{TrapGen}(1^n, q, m)$ is statistically close to uniform. Specifically, if $(\mathbf{A}, \mathsf{td}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ and $\mathbf{A}' \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m}$, then $\Delta(\mathbf{A}, \mathbf{A}') \leq 2^{-n}$.*
- **Trapdoor quality:** *The trapdoor $\mathsf{td}_\mathbf{A}$ output by $\mathsf{TrapGen}(1^n, q, m)$ is a $\tau$-trapdoor where $\tau = O\left(\sqrt{n \log q \log n}\right)$. We refer to the parameter $\tau$ as the* quality *of the trapdoor.*
- **Preimage sampling:** *Suppose $\mathsf{td}_\mathbf{A}$ is a $\tau$-trapdoor for $\mathbf{A}$. Then, for all $s \geq \tau \cdot \omega(\sqrt{\log n})$ and all target vectors $\mathbf{v} \in \mathbb{Z}_q^n$, the statistical distance between the following distributions is at most $2^{-n}$:*

$$\{\mathbf{u} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathsf{td}_\mathbf{A}, \mathbf{v}, s)\} \quad \text{and} \quad \{\mathbf{u} \leftarrow \mathbf{A}_s^{-1}(\mathbf{v})\}.$$

**Gadget trapdoors.** In this work, we will work with the gadget trapdoors introduced by Micciancio and Peikert [MP12]. We recall the key properties of gadget trapdoors from [MP12] and then state a direct corollary that we will use in this work (Corollary 3.12).

**Theorem 3.11** (Gadget Trapdoors [MP12])**.** *The gadget matrix $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ has a public $\tau$-trapdoor $\mathsf{td}_\mathbf{G}$ where $\tau = O(1)$. In addition, if $\mathbf{AR} = \mathbf{HG}$ where $\mathbf{A} \in \mathbb{Z}_q^{n \times m'}$, $\mathbf{R} \in \mathbb{Z}_q^{m' \times m}$, $m = n \lceil \log q \rceil$, and $\mathbf{H} \in \mathbb{Z}_q^{n \times n}$ is invertible, then $\mathsf{td}_\mathbf{A} = (\mathbf{R}, \mathbf{H})$ can be used as a $\tau$-trapdoor (by extending* SamplePre *from Theorem 3.10 accordingly) for $\mathbf{A}$ where $\tau = s_1(\mathbf{R})$ and $s_1(\mathbf{R}) \leq \sqrt{mm'}\|\mathbf{R}\|$ denotes the largest singular value of $\mathbf{R}$.*

**Corollary 3.12** (Gadget Trapdoors)**.** *Let $\mathbf{H} \in \mathbb{Z}_q^{k \times t}$ be a full rank matrix where $k \leq t$ (i.e., $\mathbf{H}$ has full row rank). Suppose $\mathbf{AR} = \mathbf{H} \otimes \mathbf{G}$. Let $\mathbf{A} \in \mathbb{Z}_q^{kn \times m'}$ and $\mathbf{R} \in \mathbb{Z}_q^{m' \times mt}$ with $m = n \lceil \log q \rceil$. Then, $\mathsf{td}_\mathbf{A} = (\mathbf{R}, \mathbf{H})$ can be used as a $\tau$-trapdoor for $\mathbf{A}$ where $\tau \leq \sqrt{kmm'} \cdot mt\|\mathbf{R}\|$.*

*Proof.* We can write $\mathbf{H} \otimes \mathbf{G} = (\mathbf{H} \otimes \mathbf{I}_n)(\mathbf{I}_t \otimes \mathbf{G}) = (\mathbf{H} \otimes \mathbf{I}_n)\mathbf{G}_{nt}$. Since $\mathbf{H}$ is full rank (with $k \leq t$), there exists a matrix $\mathbf{H}^* \in \mathbb{Z}_q^{t \times k}$ such that $\mathbf{H}\mathbf{H}^* = \mathbf{I}_k$. Correspondingly, $(\mathbf{H} \otimes \mathbf{I}_n)(\mathbf{H}^* \otimes \mathbf{I}_n) = \mathbf{I}_{kn}$. Let $\bar{\mathbf{R}} = \mathbf{R}\mathbf{G}_{nt}^{-1}((\mathbf{H}^* \otimes \mathbf{I}_n)\mathbf{G}_{kn}) \in \mathbb{Z}_q^{m' \times km}$. Now, we can write

$$\mathbf{A}\bar{\mathbf{R}} = \mathbf{A}\mathbf{R}\mathbf{G}_{nt}^{-1}((\mathbf{H}^* \otimes \mathbf{I}_n)\mathbf{G}_{kn}) = (\mathbf{H} \otimes \mathbf{I}_n)\mathbf{G}_{nt}\mathbf{G}_{nt}^{-1}((\mathbf{H}^* \otimes \mathbf{I}_n)\mathbf{G}_{kn}) = \mathbf{G}_{kn},$$

and so $\bar{\mathbf{R}}$ is a trapdoor for $\mathbf{A}$ (Theorem 3.11). Moreover, $\|\bar{\mathbf{R}}\| \leq mt\|\mathbf{R}\|$, and the claim follows. □

**Preimage sampling.** We will also use the following property of discrete Gaussian distributions which follows from [GPV08]:

**Lemma 3.13** (Preimage Sampling [GPV08, adapted]). *Let $n, m, q$ be lattice parameters. There exists polynomials $m_0(n,q) = O(n \log q)$ and $\chi_0(n,q) = \sqrt{n \log q} \cdot \omega(\sqrt{\log n})$ such that for all $m \geq m_0(n,q)$ and $\chi \geq \chi_0(n,q)$, the statistical distance between the following distributions is* $\mathsf{negl}(n)$*:*

$$\left\{(\mathbf{A}, \mathbf{x}, \mathbf{Ax}) : \mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}, \mathbf{x} \leftarrow D_{\mathbb{Z},\chi}^m\right\} \text{ and } \left\{(\mathbf{A}, \mathbf{x}, \mathbf{y}) : \mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}, \mathbf{y} \xleftarrow{R} \mathbb{Z}_q^n, \mathbf{x} \leftarrow \mathbf{A}_\chi^{-1}(\mathbf{y})\right\}.$$

**Lemma 3.14** (Leftover Hash Lemma [ABB10a]). *Let $n, m, q$ be lattice parameters where $q > 2$ is prime. There exists a polynomial $m_0(n,q) = O(n \log q)$ such that for all $m \geq m_0(n,q)$, all vectors $\mathbf{e} \in \mathbb{Z}_q^m$, and all polynomials $k = k(n)$, the statistical distance between the following distributions is* $\mathsf{negl}(n)$*:*

$$\{(\mathbf{A}, \mathbf{AR}, \mathbf{e}^\intercal \mathbf{R}) : \mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}, \mathbf{R} \xleftarrow{R} \{-1,1\}^{m \times k}\} \quad \text{and} \quad \{(\mathbf{A}, \mathbf{B}, \mathbf{e}^\intercal \mathbf{R}) : \mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}, \mathbf{B} \xleftarrow{R} \mathbb{Z}_q^{n \times k}, \mathbf{R} \xleftarrow{R} \{-1,1\}^{m \times k}\}. \quad (3.1)$$

**Smudging lemma.** We will also use the following standard smudging lemma (see [BDE+18] for a proof):

**Lemma 3.15** (Smudging Lemma). *Let $\lambda$ be a security parameter. Take any $e \in \mathbb{Z}$ where $|e| \leq B$. Suppose $\chi \geq B \cdot \lambda^{\omega(1)}$. Then, the statistical distance between the distributions $\left\{z : z \leftarrow D_{\mathbb{Z},\chi}\right\}$ and $\left\{z + e : z \leftarrow D_{\mathbb{Z},\chi}\right\}$ is* $\mathsf{negl}(\lambda)$*.*

**The evasive LWE assumption.** We now state a variant of the evasive LWE assumption introduced by Wee [Wee22] and Tsabury [Tsa22]. We compare our formulation with the original version by Wee in Remark 3.18.

**Assumption 3.16** (Evasive LWE). Let $\lambda$ be a security parameter, and let $n = n(\lambda), m = m(\lambda), q = q(\lambda), \chi = \chi(\lambda)$, $s = s(\lambda)$ with $s \geq O(\sqrt{m \log q})$. Let Samp be an algorithm that takes the security parameter $1^\lambda$ as input and outputs a matrix $\mathbf{B} \in \mathbb{Z}_q^{n\ell \times m'}$, a set of $\ell$ target matrices $\mathbf{P}_1 \in \mathbb{Z}_q^{n \times N_1}, \ldots, \mathbf{P}_\ell \in \mathbb{Z}_q^{n \times N_\ell}$, and auxiliary information $\mathsf{aux} \in \{0,1\}^*$. Then, for adversaries $\mathcal{A}_0$ and $\mathcal{A}_1$, we define advantage functions

$$\begin{aligned}
\mathsf{Adv}_{\mathcal{A}_0}^{(\mathrm{PRE})}(\lambda) &:= \Big|\Pr\left[\mathcal{A}_0(\{(\mathbf{A}_i, \mathbf{s}_i^\intercal \mathbf{A}_i + \mathbf{e}_{1,i}^\intercal)\}_{i \in [\ell]}, \mathbf{B}, \mathbf{s}^\intercal \mathbf{B} + \mathbf{e}_2^\intercal, \{\mathbf{s}_i^\intercal \mathbf{P}_i + \mathbf{e}_{3,i}^\intercal\}_{i \in [\ell]}, \mathsf{aux}) = 1\right] \\
&\quad - \Pr\left[\mathcal{A}_0(\{(\mathbf{A}_i, \mathbf{u}_{1,i}^\intercal)\}_{i \in [\ell]}, \mathbf{B}, \mathbf{u}_2^\intercal, \{\mathbf{u}_{3,i}^\intercal\}_{i \in [\ell]}, \mathsf{aux}) = 1\right]\Big| \\
\mathsf{Adv}_{\mathcal{A}_1}^{(\mathrm{POST})}(\lambda) &:= \Big|\Pr\left[\mathcal{A}_1(\{(\mathbf{A}_i, \mathbf{s}_i^\intercal \mathbf{A}_i + \mathbf{e}_{1,i}^\intercal)\}_{i \in [\ell]}, \mathbf{B}, \mathbf{s}^\intercal \mathbf{B} + \mathbf{e}_2^\intercal, \{\mathbf{K}_i\}_{i \in [\ell]}, \mathsf{aux}) = 1\right] \\
&\quad - \Pr\left[\mathcal{A}_1(\{(\mathbf{A}_i, \mathbf{u}_{1,i}^\intercal)\}_{i \in [\ell]}, \mathbf{B}, \mathbf{u}_2^\intercal, \{\mathbf{K}_i\}_{i \in [\ell]}, \mathsf{aux}) = 1\right]\Big|,
\end{aligned}$$

where

$$\begin{aligned}
&(\mathbf{B}, \mathbf{P}_1, \ldots, \mathbf{P}_\ell, \mathsf{aux}) \leftarrow \mathsf{Samp}(1^\lambda), \\
&\mathbf{A}_1, \ldots, \mathbf{A}_\ell \xleftarrow{R} \mathbb{Z}_q^{n \times m}, \\
&\mathbf{s}_1, \ldots, \mathbf{s}_\ell \xleftarrow{R} \mathbb{Z}_q^n, \mathbf{s}^\intercal \leftarrow [\mathbf{s}_1^\intercal \mid \cdots \mid \mathbf{s}_\ell^\intercal] \in \mathbb{Z}_q^{n\ell}, \\
&\mathbf{u}_{1,i} \xleftarrow{R} \mathbb{Z}_q^m, \mathbf{e}_{1,i} \leftarrow D_{\mathbb{Z},\chi}^m \ \forall i \in [\ell], \\
&\mathbf{u}_2 \xleftarrow{R} \mathbb{Z}_q^{m'}, \mathbf{e}_2 \leftarrow D_{\mathbb{Z},\chi}^{m'}, \\
&\mathbf{u}_{3,i} \xleftarrow{R} \mathbb{Z}_q^{N_i}, \mathbf{e}_{3,i} \leftarrow D_{\mathbb{Z},\chi}^{N_i} \ \forall i \in [\ell], \\
&\mathbf{K}_i \leftarrow (\mathbf{A}_i)_s^{-1}(\mathbf{P}_i) \ \forall i \in [\ell].
\end{aligned}$$

We say that the evasive LWE assumption holds if for every efficient sampler Samp and every efficient adversary $\mathcal{A}_1$, there exists an efficient algorithm $\mathcal{A}_0$, polynomial $\mathsf{poly}(\cdot)$, and negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\mathsf{Adv}_{\mathcal{A}_0}^{(\mathrm{PRE})}(\lambda) \geq \mathsf{Adv}_{\mathcal{A}_1}^{(\mathrm{POST})}(\lambda)/\mathsf{poly}(\lambda) - \mathsf{negl}(\lambda).$$

**Remark 3.17** (Auxiliary Input Distribution). As in [Wee22], we only require that the assumption holds for samplers where aux additionally contains all of the coin tosses used by Samp (i.e., *public-coin* samplers). This avoids obfuscation-based counter-examples where aux contains an obfuscation of a program related to a trapdoor for matrix $\mathbf{B}$ or $\mathbf{P}_i$. This is a *weaker* assumption compared to the evasive LWE assumptions needed to realize witness encryption (which rely on security of evasive LWE to hold for private-coin samplers) [Tsa22, VWW22].

**Remark 3.18** (Comparison with [Wee22])**.** The original formulation of the evasive LWE assumption by Wee [Wee22] corresponds to the special case where $\ell = 1$ (i.e., there is just a single matrix $\mathbf{A}_1$ and single target $\mathbf{P}_1$). When constructing multi-authority ABE, we rely on multiple *independent* matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ (one associated with each authority). It is an interesting question to reduce Assumption 3.16 to the simpler setting of $\ell = 1$. We note that the justification given in [Wee22] for evasive LWE are equally applicable to this setting.

# 4 Generalized Related-Trapdoor LWE Assumption

In this section, we introduce a generalized variant of the related-trapdoor robust LWE assumption of Brakerski and Vaikuntanathan [BV22] and then show that its hardness can be based on the standard LWE assumption (Theorem 4.2). As described in Section 2, the generalized related-trapdoor LWE assumption essentially asserts that given a vector $\mathbf{u} \in \{0,1\}^L$, an LWE sample with respect to $(\mathbf{u} \otimes \mathbf{I}_n)\mathbf{B}$ is pseudorandom (where $\mathbf{B} \in \mathbb{Z}_q^{n \times mL}$) given an oracle that takes as input $(\mathbf{M}, \mathbf{t})$ and outputs $(\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B})^{-1}(\mathbf{t})$ whenever $\bar{\mathbf{M}} = \left[\begin{smallmatrix} \mathbf{M} \\ \mathbf{u}^\intercal \end{smallmatrix}\right] \in \mathbb{Z}_q^{(k+1) \times L}$ is full rank. The original formulation of the related trapdoor assumption in [BV22] (for the setting of *single-authority* ciphertext-policy ABE) considered the special case where the matrix $\mathbf{M} \in \mathbb{Z}_q^{1 \times L}$ is a row vector. Here, we consider the case where $\mathbf{M}$ can be an arbitrary matrix. This generalization will be useful for distributing the setup in an ABE scheme to obtain a multi-authority ABE (see Section 5).

A similar approach is also implicit in the ciphertext-policy ABE scheme by Datta et al. [DKW21a]. Their approach relied on noise smudging to simulate the preimage-sampling oracle, and as such, security relied on a super-polynomial modulus. In this work, we abstract out the core technique through the related-trapdoor LWE assumption and then show a direct reduction to LWE without relying on noise smudging. This allows us to base security on LWE with a *polynomial* modulus.

**Assumption 4.1** (Generalized Related-Trapdoor LWE)**.** Let $\lambda \in \mathbb{N}$ be a security parameter, and $n = n(\lambda)$, $m = m(\lambda)$, $\hat{m} = \hat{m}(\lambda)$, and $\chi = \chi(\lambda)$ be lattice parameters. Let $q = q(\lambda)$ be a prime modulus. Let $L = L(\lambda)$ be a length parameter. For a bit $b \in \{0,1\}$, we define the related-trapdoor LWE game between a challenger and an adversary $\mathcal{A}$:

1. The adversary $\mathcal{A}$ starts by choosing a non-zero vector $\mathbf{u} \in \{0,1\}^L$.
2. The challenger samples matrices $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}$ and $\mathbf{B} \xleftarrow{R} \mathbb{Z}_q^{nL \times \hat{m}(2L-1)}$ and constructs the challenge as follows:
   - If $b = 0$, the challenger samples $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^n$, $\mathbf{R} \xleftarrow{R} \{-1,1\}^{\hat{m}L \times \hat{m}(L-1)}$, $\mathbf{e} \leftarrow D_{\mathbb{Z},\chi}^m$, $\hat{\mathbf{e}}_0 \leftarrow D_{\mathbb{Z},\chi}^{\hat{m}L}$, $\hat{\mathbf{e}}^\intercal \leftarrow \hat{\mathbf{e}}_0^\intercal[\mathbf{I}_{\hat{m}L} \mid \mathbf{R}] \in \mathbb{Z}_q^{\hat{m}(2L-1)}$, and gives $\left(\mathbf{A},\ \mathbf{B},\ \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal,\ \mathbf{s}^\intercal(\mathbf{u}^\intercal \otimes \mathbf{I}_n)\mathbf{B} + \hat{\mathbf{e}}^\intercal\right)$ to $\mathcal{A}$.
   - If $b = 1$, the challenger samples $\mathbf{v} \xleftarrow{R} \mathbb{Z}_q^m$, $\hat{\mathbf{v}} \xleftarrow{R} \mathbb{Z}_q^{\hat{m}(2L-1)}$ and gives $(\mathbf{A}, \mathbf{B}, \mathbf{v}^\intercal, \hat{\mathbf{v}}^\intercal)$ to $\mathcal{A}$.
3. Adversary $\mathcal{A}$ can now make queries of the form $(\mathbf{M}, \mathbf{t})$ where $\mathbf{M} \in \mathbb{Z}_q^{k \times L}$ where $k < L$ and $\mathbf{t} \in \mathbb{Z}_q^{kn}$.
   - Define the matrix $\bar{\mathbf{M}} = \left[\begin{smallmatrix} \mathbf{M} \\ \mathbf{u}^\intercal \end{smallmatrix}\right]$. If $\bar{\mathbf{M}}$ is not full rank (over $\mathbb{Z}_q$), the challenger replies with $\perp$.
   - If $\mathbf{t}$ is not in the column span of $(\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B}$, then the challenger also replies with $\perp$.
   - Otherwise, it samples and replies with $\mathbf{y} \leftarrow ((\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B})_\chi^{-1}(\mathbf{t})$. Namely, $\mathbf{y} \in \mathbb{Z}_q^{m(2L-1)}$ is sampled from the distribution $D_{\mathbb{Z},\chi}^{m(2L-1)}$ conditioned on $(\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B}\mathbf{y} = \mathbf{t}$.
4. At the end of the game, algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is also the output of the experiment.

We say that the $\mathsf{RTLWE}_{n,m,\hat{m},q,\chi,L}$ assumption holds if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathrm{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $|\Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1]| = \mathrm{negl}(\lambda)$ in the above security game.

**Theorem 4.2** (Generalized Related-Trapdoor LWE)**.** *Let $\lambda$ be a security parameter, and let $n = n(\lambda)$, $q = q(\lambda)$, $m = m(\lambda)$, $\hat{m} = \hat{m}(\lambda)$, and $\chi = \chi(\lambda)$ be lattice parameters. Suppose that $q > 2$ is a prime and $\chi > 2\hat{m}^2L^2 \cdot \omega(\sqrt{\log n})$. Then, there exists a fixed polynomial $m_0(n,q) = O(n \log q)$ such that for all $\hat{m} > m_0(n,q)$ and under the $\mathsf{LWE}_{n,m+\hat{m}L,q,\chi}$ assumption, the $\mathsf{RTLWE}_{n,m,\hat{m},q,\chi,L}$ assumption holds.*

*Proof.* Throughout the analysis, we use the fact that since $q$ is prime, $\mathbb{Z}_q$ is a field and $\mathbb{Z}_q^L$ is a vector space (where notions like "rank" are well-defined). We start by defining a sequence of hybrid experiments:

- $\mathsf{Hyb}_0$: This is the real experiment with bit $b = 0$.
- $\mathsf{Hyb}_1$: Same as $\mathsf{Hyb}_0$ except the challenger changes how it constructs $\mathbf{B}$ in the challenge:
  - First, since $\mathbf{u} \neq \mathbf{0}$, its kernel has dimension $L-1$. Let $\mathbf{U}^\perp \in \mathbb{Z}_q^{L\times(L-1)}$ be a full-rank matrix where $\mathbf{u}^\intercal\mathbf{U}^\perp = \mathbf{0}$ (i.e., the columns of $\mathbf{U}^\perp$ form a basis for the kernel of $\mathbf{u}^\intercal$). In the description here and in the proof, we assume that $\mathbf{U}^\perp$ is computed from $\mathbf{u}^\intercal$ using an efficient and deterministic algorithm (e.g., Gaussian elimination).
  - The challenger samples $\hat{\mathbf{A}} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{nL\times\hat{m}L}$ and $\mathbf{R} \xleftarrow{\mathrm{R}} \{-1,1\}^{\hat{m}L\times\hat{m}(L-1)}$, and sets $\mathbf{B} \leftarrow [\hat{\mathbf{A}} \mid \hat{\mathbf{A}}\mathbf{R} + \mathbf{U}^\perp \otimes \mathbf{G}] \in \mathbb{Z}_q^{nL\times\hat{m}(2L-1)}$.

  Note that the challenger uses the same *inefficient* procedure for answering oracle queries $(\mathbf{M}, \mathbf{t})$ as in $\mathsf{Hyb}_0$. Namely, when $[\mathbf{M}^\intercal \mid \mathbf{u}]$ is full rank and $\mathbf{t}$ is in the image of $((\mathbf{M}\otimes\mathbf{I}_n)\mathbf{B})$, it samples $\mathbf{y} \leftarrow ((\mathbf{M}\otimes\mathbf{I}_n)\mathbf{B})_\chi^{-1}(\mathbf{t})$, which only depends on $\mathbf{M}$, $\mathbf{B}$, and $\mathbf{t}$.
- $\mathsf{Hyb}_2$: Same as $\mathsf{Hyb}_1$, except the challenger changes how it samples $\hat{\mathbf{A}} \in \mathbb{Z}_q^{nL\times\hat{m}L}$ and how it responds to oracle queries:
  - Since $\mathbf{u} \in \{0,1\}^L$ and $\mathbf{u} \neq \mathbf{0}$, let $i \in [L]$ be the smallest index where $u_i = 1$. For all $j \neq i$, the challenger samples $\hat{\mathbf{A}}_j \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n\times\hat{m}L}$. Next, it samples $\mathbf{D} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n\times\hat{m}L}$ and sets $\hat{\mathbf{A}}_i = \mathbf{D} - \sum_{j\neq i} u_j\hat{\mathbf{A}}_j$. It sets
  $$\hat{\mathbf{A}} = \begin{bmatrix} \hat{\mathbf{A}}_1 \\ \vdots \\ \hat{\mathbf{A}}_L \end{bmatrix} \in \mathbb{Z}_q^{nL\times\hat{m}L}.$$
  In this experiment, $(\mathbf{u}^\intercal\otimes\mathbf{I}_n)\hat{\mathbf{A}} = \mathbf{D}$ and since $\mathbf{u}^\intercal\mathbf{U}^\perp = \mathbf{0}$,
  $$(\mathbf{u}^\intercal\otimes\mathbf{I}_n)\mathbf{B} = (\mathbf{u}^\intercal\otimes\mathbf{I}_n)[\hat{\mathbf{A}} \mid \hat{\mathbf{A}}\mathbf{R} + \mathbf{U}^\perp\otimes\mathbf{G}] = [\mathbf{D} \mid \mathbf{D}\mathbf{R} + \mathbf{u}^\intercal\mathbf{U}^\perp\otimes\mathbf{G}] = [\mathbf{D} \mid \mathbf{D}\mathbf{R}]. \tag{4.1}$$
  The challenge in this experiments can thus be written as $\left(\mathbf{A}, \mathbf{B}, \mathbf{s}^\intercal\mathbf{A}+\mathbf{e}^\intercal, (\mathbf{s}^\intercal\mathbf{D}+\hat{\mathbf{e}}_0^\intercal)[\mathbf{I}_{\hat{m}L} \mid \mathbf{R}]\right)$, where $\mathbf{s} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and $\hat{\mathbf{e}}_0 \leftarrow D_{\mathbb{Z},\chi}^{\hat{m}L}$.
  - When answering oracle queries $(\mathbf{M}, \mathbf{t})$ where $\bar{\mathbf{M}} = \left[\begin{smallmatrix}\mathbf{M}\\ \mathbf{u}^\intercal\end{smallmatrix}\right]$ is full rank and $\mathbf{t}$ is in the image of $(\mathbf{M}\otimes\mathbf{I}_n)\mathbf{B}$, the challenger computes
  $$\mathsf{td} = \begin{bmatrix} -\mathbf{R} \\ \mathbf{I}_{\hat{m}(L-1)} \end{bmatrix} \in \mathbb{Z}_q^{\hat{m}(2L-1)\times\hat{m}(L-1)} \tag{4.2}$$
  and samples $\mathbf{y} \leftarrow \mathsf{SamplePre}((\mathbf{M}\otimes\mathbf{I}_n)\mathbf{B}, \mathsf{td}, \mathbf{t}, \chi)$ (cf. Theorem 3.10 and Corollary 3.12).
- $\mathsf{Hyb}_3$: Same as $\mathsf{Hyb}_2$ except the challenger sets the challenge as $(\mathbf{A}, \mathbf{B}, \mathbf{v}^\intercal, \hat{\mathbf{v}}^\intercal)$ where $\mathbf{v} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^m$ and $\hat{\mathbf{v}} \leftarrow \hat{\mathbf{z}}^\intercal[\mathbf{I}_{\hat{m}L} \mid \mathbf{R}]$ where $\hat{\mathbf{z}} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{\hat{m}L}$.
- $\mathsf{Hyb}_4$: Same as $\mathsf{Hyb}_3$ except the challenge reverts to sampling $\hat{\mathbf{A}} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{nL\times\hat{m}L}$. When answering oracle queries $(\mathbf{M}, \mathbf{t})$ where $\bar{\mathbf{M}} = \left[\begin{smallmatrix}\mathbf{M}\\ \mathbf{u}^\intercal\end{smallmatrix}\right]$ is full rank and $\mathbf{t}$ is in the image of $(\mathbf{M}\otimes\mathbf{I}_n)\mathbf{B}$, the challenger reverts to sampling $\mathbf{y} \leftarrow ((\mathbf{M}\otimes\mathbf{I}_n)\mathbf{B})_\chi^{-1}(\mathbf{t})$ *inefficiently*. In particular, $\mathbf{y}$ here can be sampled given only $\mathbf{M}$, $\mathbf{B}$, and $\mathbf{t}$ (*without* knowledge of $\mathbf{R}$).
- $\mathsf{Hyb}_5$: Same as $\mathsf{Hyb}_4$ except the challenger samples $\mathbf{B} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{nL\times\hat{m}(2L-1)}$ and $\hat{\mathbf{v}} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{\hat{m}(2L-1)}$ when constructing the challenger ciphertext. This is the real experiment with bit $b = 1$.

For an adversary $\mathcal{A}$, we write $\mathsf{Hyb}_i(\mathcal{A})$ to denote the output of $\mathsf{Hyb}_i$ with adversary $\mathcal{A}$. We now show that each adjacent pair of hybrid experiments is computationally indistinguishable.

**Lemma 4.3.** *There exists a fixed polynomial $m_0(n,q) = O(n \log q)$ such that for all $\hat{m} > m_0(n,q)$ and all adversaries $\mathcal{A}$, $\mathsf{Hyb}_0(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_1(\mathcal{A})$.*

*Proof.* Suppose there exists an adversary $\mathcal{A}$ that distinguishes between $\mathsf{Hyb}_0$ and $\mathsf{Hyb}_1$ with non-negligible probability $\varepsilon$. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ that can distinguish between the distributions in Eq. (3.1):

- Algorithm $\mathcal{B}$ samples $\hat{\mathbf{e}}_0^\intercal \xleftarrow{R} \mathbb{Z}_q^{\hat{m}L}$ and gives the dimension $k = \hat{m}(L-1)$ and error vector $\hat{\mathbf{e}}_0$ to its challenger.
- Algorithm $\mathcal{B}$ receives a challenge $(\mathbf{D}_0, \mathbf{D}_1, \mathbf{u})$ where $\mathbf{D}_0 \in \mathbb{Z}_q^{nL \times \hat{m}L}$, $\mathbf{D}_1 \in \mathbb{Z}_q^{nL \times \hat{m}(2L-1)}$, and $\mathbf{u} \in \mathbb{Z}_q^{\hat{m}L}$, algorithm $\mathcal{B}$ computes $\mathbf{B} \leftarrow [\mathbf{D}_0 \mid \mathbf{D}_1 + \mathbf{U}^\perp \otimes \mathbf{G}]$ and $\hat{\mathbf{e}}^\intercal \leftarrow [\hat{\mathbf{e}}_0^\intercal \mid \mathbf{u}]$.
- Algorithm $\mathcal{B}$ constructs the remaining elements in the challenge exactly as in $\mathsf{Hyb}_0$ and $\mathsf{Hyb}_1$. Likewise, it responds to the oracle queries using the same procedure as in $\mathsf{Hyb}_0$ and $\mathsf{Hyb}_1$. In particular, all of the other components are independent of $\mathbf{R}$.
- At the end of the experiment, algorithm $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

We take $m_0$ to be the polynomial from Lemma 3.14. Then, if $\mathbf{D}_0, \mathbf{D}_1$ are uniform and $\mathbf{u} = \hat{\mathbf{e}}_0^\intercal \mathbf{R}$ for $\mathbf{R} \xleftarrow{R} \{-1,1\}^{\hat{m}L \times \hat{m}(L-1)}$, then $\mathcal{B}$ perfectly simulates the first distribution in Eq. (3.1) for $\mathcal{A}$. Alternatively, if $\mathbf{D}_1 = \mathbf{D}_0\mathbf{R}$ and $\mathbf{u} = \hat{\mathbf{e}}_0^\intercal \mathbf{R}$, then $\mathcal{B}$ perfectly simulates the second distribution in Eq. (3.1). The claim holds. □

**Lemma 4.4.** *Suppose $\chi \geq 2\hat{m}^2 L^2 \cdot \omega(\sqrt{\log n})$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_1(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_2(\mathcal{A})$.*

*Proof.* First, the distribution of $\hat{\mathbf{A}}$ is uniform over $\mathbb{Z}_q^{nL \times \hat{m}L}$ in both experiments (in $\mathsf{Hyb}_2$, all of the blocks $\hat{\mathbf{A}}_j$ for $j \neq i$ and $\mathbf{D}$ are independent and uniform over $\mathbb{Z}_q^{n \times \hat{m}L}$). Consider now the response to an oracle query $(\mathbf{M}, \mathbf{t})$. If $[\mathbf{M}^\intercal \mid \mathbf{u}]$ is not full rank or if $\mathbf{t}$ is not in the image of $(\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B}$, then the challenger's response in both experiments is $\perp$. Otherwise, the following holds:

- Suppose $\mathbf{M} \in \mathbb{Z}_q^{k \times L}$. Since $k + 1 \leq L$, there exists $[\mathbf{V} \mid \mathbf{v}] \in \mathbb{Z}_q^{L \times (k+1)}$ where $\mathbf{V} \in \mathbb{Z}_q^{L \times k}$ and $\mathbf{v} \in \mathbb{Z}_q^L$ such that $\left[\begin{smallmatrix} \mathbf{M} \\ \mathbf{u}^\intercal \end{smallmatrix}\right][\mathbf{V} \mid \mathbf{v}] = \mathbf{I}_{k+1}$, and in particular, that $\mathbf{MV} = \mathbf{I}_k$ and $\mathbf{u}^\intercal \mathbf{V} = \mathbf{0}$. Thus, $\mathbf{V} \in \mathrm{span}(\mathbf{U}^\perp)$ and so, there exists $\mathbf{V}' \in \mathbb{Z}_q^{(L-1) \times k}$ such that $\mathbf{MU}^\perp \mathbf{V}' = \mathbf{I}_k$. Equivalently, the matrix $\mathbf{MU}^\perp \in \mathbb{Z}_q^{k \times (L-1)}$ is full rank. Then

$$(\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B} \underbrace{\begin{bmatrix} -\mathbf{R} \\ \mathbf{I}_{\hat{m}(L-1)} \end{bmatrix}}_{\bar{\mathbf{R}}} = (\mathbf{M} \otimes \mathbf{I}_n)[\hat{\mathbf{A}} \mid \hat{\mathbf{A}}\mathbf{R} + \mathbf{U}^\perp \otimes \mathbf{G}] \begin{bmatrix} -\mathbf{R} \\ \mathbf{I}_{\hat{m}(L-1)} \end{bmatrix} = \mathbf{MU}^\perp \otimes \mathbf{G}. \tag{4.3}$$

  By construction, $\bar{\mathbf{R}} \in \{-1,1\}^{\hat{m}(2L-1) \times \hat{m}(L-1)}$.

- Since $\mathbf{MU}^\perp$ is full rank and $(\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B}\bar{\mathbf{R}} = \mathbf{MU}^\perp \otimes \mathbf{G}$, we appeal to Corollary 3.12 to conclude that $\mathsf{td} = (\bar{\mathbf{R}}, \mathbf{MU}^\perp)$ is a $\tau$-trapdoor for $(\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B}$ with $\tau \leq \sqrt{k\hat{m}^2(2L-1)}\hat{m}(L-1) < 2\hat{m}^2L^2$ since $k < L$.
- Since $\chi > 2\hat{m}^2L^2 \cdot \omega(\sqrt{\log n}) > \tau \cdot \omega(\sqrt{\log n})$, the distributions

$$\{\mathbf{y} \leftarrow \mathsf{SamplePre}((\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B}, \mathsf{td}, \mathbf{t}, \chi)\} \quad \textit{and} \quad \{\mathbf{y} \leftarrow ((\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B})_\chi^{-1}(\mathbf{t})\}.$$

  are statistically close by Theorem 3.10.

The claim now follows by a standard hybrid argument over the number of queries the adversary makes. □

**Lemma 4.5.** *Under the $\mathsf{LWE}_{n,mL,q,\chi}$ assumption, for all efficient adversaries $\mathcal{A}$, it follows that $\mathsf{Hyb}_2(\mathcal{A}) \overset{c}{\approx} \mathsf{Hyb}_3(\mathcal{A})$.*

*Proof.* Suppose there exists an efficient adversary $\mathcal{A}$ that is able to distinguish $\mathsf{Hyb}_2$ from $\mathsf{Hyb}_3$ with non-negligible advantage $\varepsilon$. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ for the $\mathsf{LWE}_{n,mL,q,\chi}$ assumption:

1. Algorithm $\mathcal{B}$ receives an LWE challenge $\left([\mathbf{A} \mid \mathbf{D}], [\mathbf{z}^\intercal \mid \hat{\mathbf{z}}^\intercal]\right)$ where $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{D} \in \mathbb{Z}_q^{n \times \hat{m}L}$, $\mathbf{z} \in \mathbb{Z}_q^m$, and $\hat{\mathbf{z}} \in \mathbb{Z}_q^{\hat{m}L}$.

2. Algorithm $\mathcal{B}$ starts running $\mathcal{A}$. Algorithm $\mathcal{A}$ starts by choosing a non-zero vector $\mathbf{u} \in \{0,1\}^L$. Let $i \in [L]$ be the smallest index where $u_i = 1$. For $j \neq i$, algorithm $\mathcal{A}$ samples $\hat{\mathbf{A}}_j \xleftarrow{R} \mathbb{Z}_q^{n \times \hat{m}L}$ and it computes $\hat{\mathbf{A}}_i \leftarrow \mathbf{D} - \sum_{j \neq i} u_j \hat{\mathbf{A}}_j$. Finally, set $\hat{\mathbf{A}} \leftarrow [\hat{\mathbf{A}}_1^\intercal \mid \cdots \mid \hat{\mathbf{A}}_L^\intercal]^\intercal$.

3. Algorithm $\mathcal{B}$ samples $\mathbf{R} \xleftarrow{R} \{-1,1\}^{\hat{m}L \times \hat{m}(L-1)}$ and sets $\mathbf{B} = [\hat{\mathbf{A}} \mid \hat{\mathbf{A}}\mathbf{R} + \mathbf{U}^\perp \otimes \mathbf{G}]$. It gives $(\mathbf{A}, \mathbf{B}, \mathbf{z}^\intercal, \hat{\mathbf{z}}^\intercal[\mathbf{I}_{\hat{m}L} \mid \mathbf{R}])$ to $\mathcal{A}$.

4. Whenever $\mathcal{A}$ makes an oracle query on input $(\mathbf{M}, \mathbf{t})$, algorithm $\mathcal{B}$ checks if $\left[\begin{smallmatrix} \mathbf{M} \\ \mathbf{u}^\intercal \end{smallmatrix}\right]$ is full rank and that $\mathbf{t}$ is in the column span of $(\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B}$ . If not, then algorithm $\mathcal{B}$ replies with $\perp$. Otherwise, algorithm $\mathcal{B}$ constructs the trapdoor $\mathsf{td} = \left[\begin{smallmatrix} -\mathbf{R} \\ \mathbf{I}_{\hat{m}(L-1)} \end{smallmatrix}\right]$ as in Eq. (4.2) and replies with $\mathbf{y} \leftarrow \mathsf{SamplePre}((\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B}, \mathsf{td}, \mathbf{t}, \chi)$.

5. At the end of the game, algorithm $\mathcal{A}$ outputs a bit $b \in \{0,1\}$, which algorithm $\mathcal{B}$ outputs as the output of the experiment.

Algorithm $\mathcal{B}$ constructs $\hat{\mathbf{A}}$ exactly as prescribed in $\mathsf{Hyb}_2$ and $\mathsf{Hyb}_3$ and answers the oracle queries using the same procedure in $\mathsf{Hyb}_2$ and $\mathsf{Hyb}_3$. From Eq. (4.1), in $\mathsf{Hyb}_2$ and $\mathsf{Hyb}_3$, we have that $(\mathbf{u}^\intercal \otimes \mathbf{I}_n)\mathbf{B} = [\mathbf{D} \mid \mathbf{D}\mathbf{R}]$. It suffices to consider the distribution of the challenge ciphertext:

- Suppose $\mathbf{z}^\intercal = \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal$ and $\hat{\mathbf{z}}^\intercal = \mathbf{s}^\intercal\mathbf{D} + \hat{\mathbf{e}}_0^\intercal$ for some $\mathbf{e} \leftarrow D_{\mathbb{Z},\chi}^m$ and $\hat{\mathbf{e}}_0 \leftarrow D_{\mathbb{Z},\chi}^{\hat{m}L}$. Then,

$$\hat{\mathbf{z}}^\intercal[\mathbf{I}_{\hat{m}L} \mid \mathbf{R}] = (\mathbf{s}^\intercal\mathbf{D} + \hat{\mathbf{e}}_0^\intercal)[\mathbf{I}_{\hat{m}L} \mid \mathbf{R}] = \mathbf{s}^\intercal[\mathbf{D} \mid \mathbf{D}\mathbf{R}] + \hat{\mathbf{e}}_0^\intercal[\mathbf{I}_{\hat{m}L} \mid \mathbf{R}] = \mathbf{s}^\intercal(\mathbf{u}^\intercal \otimes \mathbf{I}_n)\mathbf{B} + \hat{\mathbf{e}}_0^\intercal[\mathbf{I}_{\hat{m}L} \mid \mathbf{R}],$$

  which coincides with the challenge distribution in $\mathsf{Hyb}_2$.

- Suppose $\mathbf{z} \xleftarrow{R} \mathbb{Z}_q^m$ and $\hat{\mathbf{z}} \xleftarrow{R} \mathbb{Z}_q^{\hat{m}L}$. This is precisely the challenge distribution in $\mathsf{Hyb}_3$.

Thus, depending on whether the LWE challenge is pseudorandom or uniform, algorithm $\mathcal{B}$ perfectly simulates either $\mathsf{Hyb}_2$ or $\mathsf{Hyb}_3$ for $\mathcal{A}$, and the claim follows. □

**Lemma 4.6.** *Suppose $\chi \geq 2\hat{m}^2L^2 \cdot \omega(\sqrt{\log n})$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_3(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_4(\mathcal{A})$.*

*Proof.* Follows by an identical argument as in the proof of Lemma 4.4. □

**Lemma 4.7.** *There exists a fixed polynomial $m_0(n,q) = O(n \log q)$ such that for all $\hat{m} > m_0$ and all adversaries $\mathcal{A}$, $\mathsf{Hyb}_4(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_5(\mathcal{A})$.*

*Proof.* The only difference between $\mathsf{Hyb}_4$ and $\mathsf{Hyb}_5$ is the distribution of $\mathbf{B}$ and $\hat{\mathbf{v}}$. Notably, in both experiments, the responses to the oracle queries depend only on $\mathbf{M}$ and $\mathbf{B}$ and not $\mathbf{R}$ (i.e., the challenger in both experiments samples $\mathbf{y} \leftarrow ((\mathbf{M} \otimes \mathbf{I}_n)\mathbf{B})_\chi^{-1}(\mathbf{t})$ *inefficiently*). Consider the distribution of the challenge in the two experiments:

- In $\mathsf{Hyb}_4$, $\mathbf{B} = [\hat{\mathbf{A}} \mid \hat{\mathbf{A}}\mathbf{R}] + [\mathbf{0}^{nL \times \hat{m}L} \mid \mathbf{U}^\perp \otimes \mathbf{G}]$ and $\hat{\mathbf{v}} = [\hat{\mathbf{z}}^\intercal \mid \hat{\mathbf{z}}^\intercal\mathbf{R}]$, with $\mathbf{R} \xleftarrow{R} \{-1,1\}^{\hat{m}L \times \hat{m}(2L-1)}$.

- In $\mathsf{Hyb}_5$, $\mathbf{B}$ and $\hat{\mathbf{v}}$ are both uniform.

In both cases, $\hat{\mathbf{A}}$ and $\hat{\mathbf{z}}$ are uniform. The claim now follows by applying Lemma 3.14 and considering the setting $\mathbf{A} = [\hat{\mathbf{A}}^\intercal \mid \hat{\mathbf{z}}]^\intercal \in \mathbb{Z}_q^{(nL+1) \times \hat{m}L}$. Note that here, we can set $\mathbf{e}$ arbitrarily (i.e., the claim follows by the vanilla leftover hash lemma without leakage). □

Combining Lemmas 4.3 to 4.7, the claim holds. □

# 5 Multi-Authority ABE from LWE in the Random Oracle Model

In this section, we describe our construction of multi-authority ABE for the family of subset policies in the random oracle model. Our construction follows a similar structure as the multi-authority ABE scheme of Datta et al. [DKW21a] except we provide a direct reduction to the (generalized) related trapdoor LWE problem (Section 4). Notably, this allows us to base security on polynomial hardness of the plain LWE assumption with a *polynomial* modulus. The previous construction of Datta et al. relied on LWE with a *super-polynomial* modulus-to-noise ratio.

**Construction 5.1** (Multi-Authority ABE in the Random Oracle Model)**.** Let $\lambda$ be a security parameter, and $n = n(\lambda)$, $m = m(\lambda)$, $q = q(\lambda)$, and $\chi = \chi(\lambda)$ be lattice parameters. Let $L = L(\lambda)$ be a bound on the number of attributes associated with a ciphertext. Let $\mathcal{GID} = \{0,1\}^\lambda$ be the set of user identifiers and $\mathcal{AU} = \{0,1\}^\lambda$ be the set of authority identifiers. The construction will rely on a hash function $\mathsf{H} \colon \mathcal{GID} \to \mathbb{Z}_q^{m(2L-1)}$, which will be modeled as a random oracle as follows:

- For ease of exposition in the following description, we will start by assuming that the outputs of the random oracle $\mathsf{H}$ are distributed according to a *discrete Gaussian distribution*. Specifically, on every input $\mathsf{gid} \in \mathcal{GID}$, the output $\mathsf{H}(\mathsf{gid})$ is a sample from the distribution $D_{\mathbb{Z},\chi}^{m(2L-1)}$. In Section 5.1 and Remark 5.9, we show that using inversion sampling, we can implement $\mathsf{H}$ using a *standard* random oracle $\mathsf{H}' \colon \mathcal{GID} \to \{0,1\}^{\lambda m(2L-1)}$, where the outputs of $\mathsf{H}'(\mathsf{gid})$ are distributed uniformly over $\{0,1\}^{\lambda m(2L-1)}$ as usual.

We construct a multi-authority ABE scheme for subset policies with message space $\mathcal{M} = \{0,1\}$ as follows:

- $\mathsf{GlobalSetup}(1^\lambda)$: Output the global parameters $\mathsf{gp} = (\lambda, n, m, q, \chi, L, \mathsf{H})$.
- $\mathsf{AuthSetup}(\mathsf{gp}, \mathsf{aid})$: On input the global parameters $\mathsf{gp}$ and an authority identifier $\mathsf{aid} \in \mathcal{AU}$, sample $(\mathbf{A}_{\mathsf{aid}}, \mathsf{td}_{\mathsf{aid}}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$, $\mathbf{p}_{\mathsf{aid}} \xleftarrow{R} \mathbb{Z}_q^n$, and $\mathbf{B}_{\mathsf{aid}} \xleftarrow{R} \mathbb{Z}_q^{n \times m(2L-1)}$. Output the authority public key $\mathsf{pk}_{\mathsf{aid}} \leftarrow (\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$ and the authority secret key $\mathsf{msk}_{\mathsf{aid}} = \mathsf{td}_{\mathsf{aid}}$.
- $\mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}, \mathsf{pk}, \mathsf{gid})$: On input the global parameters $\mathsf{gp} = (\lambda, n, m, q, \chi, L, \mathsf{H})$, the master secret key $\mathsf{msk} = \mathsf{td}$, the public key $\mathsf{pk} = (\mathbf{A}, \mathbf{B}, \mathbf{p})$, and the user identifier $\mathsf{gid}$, the key-generation algorithm computes $\mathbf{r} \leftarrow \mathsf{H}(\mathsf{gid}) \in \mathbb{Z}_q^{m(2L-1)}$ and uses $\mathsf{td}$ to sample $\mathbf{u} \leftarrow \mathbf{A}_\chi^{-1}(\mathbf{p} + \mathbf{B}\mathbf{r})$. It outputs $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} = \mathbf{u}$.
- $\mathsf{Encrypt}(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid} \in A}, \mu)$: On input the global parameters $\mathsf{gp} = (\lambda, n, m, q, \chi, L, \mathsf{H})$, a set of of public keys $\mathsf{pk}_{\mathsf{aid}} = (\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$ associated with a set of authorities $A$, and the message $\mu \in \{0,1\}$, the encryption algorithm samples $\mathbf{s}_{\mathsf{aid}} \xleftarrow{R} \mathbb{Z}_q^n$, $\mathbf{e}_{1,\mathsf{aid}} \leftarrow D_{\mathbb{Z},\chi}^m$, $\mathbf{R} \xleftarrow{R} \{0,1\}^{mL \times m(L-1)}$, $\hat{\mathbf{e}}_2 \leftarrow D_{\mathbb{Z},\chi}^{mL}$, and $\mathbf{e}_2^\top \leftarrow \hat{\mathbf{e}}_2^\top [\mathbf{I}_{mL} \mid \mathbf{R}]$, and $e_3 \leftarrow D_{\mathbb{Z},\chi}$ for each $\mathsf{aid} \in A$. It outputs the ciphertext

$$\mathsf{ct} = \left( \left\{ \mathbf{s}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\top \right\}_{\mathsf{aid} \in A}, \sum_{\mathsf{aid} \in A} \mathbf{s}_{\mathsf{aid}}^\top \mathbf{B}_{\mathsf{aid}} + \mathbf{e}_2^\top, \sum_{\mathsf{aid} \in A} \mathbf{s}_{\mathsf{aid}}^\top \mathbf{p}_{\mathsf{aid}} + e_3 + \mu \cdot \lfloor q/2 \rceil \right).$$

- $\mathsf{Decrypt}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid} \in A}, \mathsf{ct}, \mathsf{gid})$: On input the global parameters $\mathsf{gp} = (\lambda, n, m, q, \chi, L, \mathsf{H})$, a set of secret keys $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} = \mathbf{u}_{\mathsf{aid},\mathsf{gid}}$ associated with authorities $\mathsf{aid} \in A$ and user identifier $\mathsf{gid}$, and a ciphertext $\mathsf{ct} = (\{\mathbf{c}_{1,\mathsf{aid}}^\top\}_{\mathsf{aid} \in A}, \mathbf{c}_2^\top, c_3)$, the decryption algorithm computes $\mathbf{r} \leftarrow \mathsf{H}(\mathsf{gid})$ and outputs

$$\left\lfloor \frac{2}{q} \cdot \left( c_3 + \mathbf{c}_2^\top \mathbf{r} - \sum_{\mathsf{aid} \in A} \mathbf{c}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid},\mathsf{gid}} \bmod q \right) \right\rceil.$$

**Theorem 5.2** (Correctness)**.** *Suppose the conditions of Theorem 3.10 and Lemma 3.13 hold (i.e., $m \geq m_0(n,q) = O(n \log q)$ and $\chi > \chi_0(n,q) = \sqrt{n \log q} \cdot \omega(\sqrt{\log n})$). Then, there exists a polynomial $q_0 = O(\lambda \chi^2 m^2 L^2)$ such that for all $q > q_0$, Construction 5.1 is correct.*

*Proof.* Take any message $\mu \in \{0,1\}$, an identifier $\mathsf{gid} \in \mathcal{GID}$, and set of authorities $A \subseteq \mathcal{AU}$. Sample the global parameters $\mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$, the authority keys $(\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}}) \leftarrow \mathsf{AuthSetup}(\mathsf{gp}, \mathsf{aid})$, the secret keys $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} \leftarrow \mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, \mathsf{gid})$, and the ciphertext $\mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid} \in A}, \mu)$. We now expand the various components appearing in the computation of $\mathsf{Decrypt}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid} \in A}, \mathsf{ct}, \mathsf{gid})$:

- The global parameters $\mathsf{gp} = (\lambda, n, m, q, \chi, L, \mathsf{H})$ consists of the lattice parameter and the description of a hash function $\mathsf{H}\colon \mathcal{GID} \to \mathbb{Z}_q^{m(2L-1)}$.
- The ciphertext $\mathsf{ct} = \left(\left\{\mathbf{c}_{1,\mathsf{aid}}^\top\right\}_{\mathsf{aid}\in A}, \mathbf{c}_2^\top, c_3\right)$ where

$$\mathbf{c}_{1,\mathsf{aid}}^\top = \mathbf{s}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\top \quad , \quad \mathbf{c}_2^\top = \sum_{\mathsf{aid}\in A} \mathbf{s}_{\mathsf{aid}}^\top \mathbf{B}_{\mathsf{aid}} + \mathbf{e}_2^\top \quad , \quad c_3 = \sum_{\mathsf{aid}\in A} \mathbf{s}_{\mathsf{aid}}^\top \mathbf{p}_{\mathsf{aid}} + e_3 + \mu \cdot \lfloor q/2 \rceil ,$$

  and $(\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$ is the public key associated with authority aid.
- The secret key $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} = \mathbf{u}_{\mathsf{aid},\mathsf{gid}} \leftarrow (\mathbf{A}_{\mathsf{aid}})_\chi^{-1}(\mathbf{p}_{\mathsf{aid}} + \mathbf{B}_{\mathsf{aid}}\mathbf{r})$ and $\mathbf{r} \leftarrow \mathsf{H}(\mathsf{gid})$. Since $\mathbf{p}_{\mathsf{aid}}$ is uniform over $\mathbb{Z}_q^n$ and independent of $\mathbf{B}_{\mathsf{aid}}\mathbf{r}$, the marginal distribution of $\mathbf{u}_{\mathsf{aid},\mathsf{gid}}$ is statistically close to $D_{\mathbb{Z},\chi}^m$ by Lemma 3.13. By construction, $\mathbf{r}$ is sampled from $D_{\mathbb{Z},\chi}^{m(2L-1)}$. Then, by Fact 3.8, with overwhelming probability, $\|\mathbf{u}_{\mathsf{aid},\mathsf{gid}}\| \le B$ and $\|\mathbf{r}\| \le B$ where $B = \sqrt{\lambda}\chi$.
- Then

$$\mathbf{c}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid},\mathsf{gid}} = \mathbf{s}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} \mathbf{u}_{\mathsf{aid},\mathsf{gid}} + \mathbf{e}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid},\mathsf{gid}} = \mathbf{s}_{\mathsf{aid}}^\top \mathbf{p}_{\mathsf{aid}} + \mathbf{s}_{\mathsf{aid}}^\top \mathbf{B}_{\mathsf{aid}} \mathbf{r} + \mathbf{e}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid},\mathsf{gid}}.$$

  Then, the main decryption relation becomes

$$c_3 + \mathbf{c}_2^\top \mathbf{r} - \sum_{\mathsf{aid}\in A} \mathbf{c}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid},\mathsf{gid}} = \mu \cdot \lfloor q/2 \rceil + e_3 + \mathbf{e}_2^\top \mathbf{r} - \sum_{\mathsf{aid}\in A} \mathbf{e}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid},\mathsf{gid}}.$$

  Decryption succeeds if the total error $\tilde{e} = e_3 + \mathbf{e}_2^\top \mathbf{r} - \sum_{\mathsf{aid}\in A} \mathbf{e}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid},\mathsf{gid}}$ satisfies $|\tilde{e}| < (q-1)/4$.
- To bound the error $\tilde{e}$, we bound each of its components. By definition, $\mathbf{e}_2^\top = \hat{\mathbf{e}}_2^\top [\mathbf{I}_{mL} \mid \mathbf{R}]$ where $\hat{\mathbf{e}}_2 \leftarrow D_{\mathbb{Z},\chi}^{mL}$ and $\|\mathbf{R}\| = 1$. By Fact 3.8, $\|\hat{\mathbf{e}}_2\| \le B = \sqrt{\lambda}\chi$ with overwhelming probability; in this case, $\|\mathbf{e}_2^\top\| \le BmL$. Thus, with overwhelming probability,

$$\begin{aligned} |e_3| &\le B \\ \|\mathbf{e}_2^\top \mathbf{r}\| &\le B^2 m^2 L(2L-1) \\ \|\mathbf{e}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid},\mathsf{gid}}\| &\le B^2 m \end{aligned}$$

  Finally, we have that $|A| \le L$, so we can now bound

$$|\tilde{e}| < B + B^2 m^2 L(2L-1) + B^2 mL = O(B^2 m^2 L^2) = O(\lambda \chi^2 m^2 L^2). \qquad \square$$

**Theorem 5.3** (Static Security). *Suppose the conditions of Theorem 3.10 and Lemma 3.13 hold (i.e., $m \ge m_0(n,q) = O(n \log q)$ and $\chi > \chi_0(n,q) = \sqrt{n \log q} \cdot \omega(\sqrt{\log n})$). Then, under the $\mathsf{RTLWE}_{n,Lm+1,m,q,\chi,L}$ assumption and modeling $\mathsf{H}\colon \mathcal{GID} \to \mathbb{Z}_q^{m(2L-1)}$ as a random oracle (with outputs distributed according to $D_{\mathbb{Z},\chi}^{m(2L-1)}$), Construction 5.1 is statically secure.*

*Proof.* We begin by defining a sequence of hybrid experiments:

- $\mathsf{Hyb}_0^{(b)}$: This is the static security experiment where the challenger encrypts message $\mu_b$ (where $b \in \{0,1\}$). Specifically, at the beginning of the game, the adversary outputs the following components:
  - A set of corrupted authorities $C \subset \mathcal{AU}$ and their public keys $\mathsf{pk}_{\mathsf{aid}} = (\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$ for each $\mathsf{aid} \in C$.
  - A list of non-corrupted authorities $\mathcal{N} \subseteq \mathcal{AU}$.
  - A list of secret-key queries $Q = \{(\mathsf{gid}, A)\}$ where each $A \subset \mathcal{N}$.
  - A pair of challenge messages $\mu_0, \mu_1 \in \{0,1\}$ and a set of authorities $A^* \subseteq C \cup \mathcal{N}$ where $|A^*| \le L$.

  The challenger initializes an empty table $\mathsf{T}\colon \mathcal{GID} \to \{0,1\}^{\lambda m(2L-1)}$ that it will use for answering random oracle queries. The challenger then constructs the public keys, secret keys and the challenge ciphertext as follows:

– **Public keys for non-corrupted authorities:** For each $\mathsf{aid} \in \mathcal{N}$, the challenger samples $(\mathbf{A}_{\mathsf{aid}}, \mathsf{td}_{\mathsf{aid}}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$, $\mathbf{p}_{\mathsf{aid}} \xleftarrow{R} \mathbb{Z}_q^n$, and $\mathbf{B}_{\mathsf{aid}} \xleftarrow{R} \mathbb{Z}_q^{n \times m(2L-1)}$ and sets the public key to be $\mathsf{pk}_{\mathsf{aid}} = (\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$.

– **Secret key queries:** For each secret key query $(\mathsf{gid}, A)$ where $A \subset \mathcal{N}$, the challenger computes $\mathbf{r}_{\mathsf{gid}} \leftarrow \mathsf{H}(\mathsf{gid})$ and samples $\mathbf{u}_{\mathsf{aid},\mathsf{gid}} \leftarrow (\mathbf{A}_{\mathsf{gid}})_\chi^{-1}(\mathbf{p}_{\mathsf{aid}} + \mathbf{B}_{\mathsf{aid}}\mathbf{r}_{\mathsf{gid}})$. It sets the secret key to $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} = \mathbf{u}_{\mathsf{aid},\mathsf{gid}}$.

– **Challenge ciphertext:** The challenger samples $\mathbf{s}_{\mathsf{aid}} \xleftarrow{R} \mathbb{Z}_q^n$ and $\mathbf{e}_{1,\mathsf{aid}} \leftarrow D_{\mathbb{Z},\chi}^m$ for each $\mathsf{aid} \in A^*$. It also samples $\mathbf{R} \xleftarrow{R} \{0,1\}^{mL \times m(L-1)}$, $\hat{\mathbf{e}}_2 \leftarrow D_{\mathbb{Z},\chi}^{mL}$, and $\mathbf{e}_2^\top \leftarrow \hat{\mathbf{e}}_2^\top[\mathbf{I}_{mL} \mid \mathbf{R}]$, and $e_3 \leftarrow D_{\mathbb{Z},\chi}$. Finally, it outputs the challenge ciphertext

$$\mathsf{ct} = \left( \left\{ \mathbf{s}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\top \right\}_{\mathsf{aid} \in A^*}, \sum_{\mathsf{aid} \in A^*} \mathbf{s}_{\mathsf{aid}}^\top \mathbf{B}_{\mathsf{aid}} + \mathbf{e}_2^\top, \sum_{\mathsf{aid} \in A^*} \mathbf{s}_{\mathsf{aid}}^\top \mathbf{p}_{\mathsf{aid}} + e_3 + \mu_b \cdot \lfloor q/2 \rceil \right).$$

– **Random oracle queries:** On input $\mathsf{gid} \in \mathcal{GID}$ (either from the adversary or when processing a secret-key query), the challenger checks whether there exists a mapping $(\mathsf{gid} \mapsto \mathbf{r}_{\mathsf{gid}})$ in $\mathsf{T}$. If so, it replies with $\mathbf{r}_{\mathsf{gid}}$. Otherwise, it samples $\mathbf{r}_{\mathsf{gid}} \leftarrow D_{\mathbb{Z},\chi}^{m(2L-1)}$, adds the mapping $(\mathsf{gid} \mapsto \mathbf{r}_{\mathsf{gid}})$ to $\mathsf{T}$, and replies with $\mathbf{r}_{\mathsf{gid}}$.

At the end of the game, the adversary outputs a bit $b' \in \{0,1\}$ which is the output of the experiment.

- $\mathsf{Hyb}_1^{(b)}$: Same as $\mathsf{Hyb}_0^{(b)}$ except the challenger changes how it constructs the secret keys. First, let $A^* \subseteq C \cup \mathcal{N}$ be the set of authorities associated with the challenge ciphertext and let $A^* \cap \mathcal{N} = \{\mathsf{aid}_1^*, \ldots, \mathsf{aid}_\ell^*\}$. The challenger defines the matrix

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_{\mathsf{aid}_1^*} \\ \vdots \\ \mathbf{B}_{\mathsf{aid}_\ell^*} \\ \mathbf{B}_{\mathsf{rest}} \end{bmatrix} \in \mathbb{Z}_q^{nL \times m(2L-1)},$$

where $\mathbf{B}_{\mathsf{aid}_i^*} \xleftarrow{R} \mathbb{Z}_q^{n \times m(2L-1)}$ is the matrix associated with part of the public key for authority $\mathsf{aid}_i^*$ and $\mathbf{B}_{\mathsf{rest}} \xleftarrow{R} \mathbb{Z}_q^{n(L-\ell) \times m(2L-1)}$ consist of additional *unused* components. The challenger responds to the secret key queries as follows:

– **Public keys for non-corrupted authorities:** For each $\mathsf{aid}_i^* \in A_{\mathsf{chal}}^*$, the challenger samples $\mathbf{A}_{\mathsf{aid}_i^*} \xleftarrow{R} \mathbb{Z}_q^{n \times m}$ and $\mathbf{p}_{\mathsf{aid}_i^*} \xleftarrow{R} \mathbb{Z}_q^n$, and sets the public key to be $\mathsf{pk}_{\mathsf{aid}_i^*} = (\mathbf{A}_{\mathsf{aid}_i^*}, \mathbf{B}_{\mathsf{aid}_i^*}, \mathbf{p}_{\mathsf{aid}_i^*})$. In particular, the challenger does *not* sample a trapdoor for $\mathsf{aid}_i^*$ anymore. For $\mathsf{aid} \in \mathcal{N} \notin A^*$, the challenger samples the public keys as in $\mathsf{Hyb}_0^{(b)}$.

– **Secret key queries:** For each secret key query $(\mathsf{gid}, A)$ where $A \subset \mathcal{N}$, the challenger first partitions $A = A_{\mathsf{chal}} \cup \bar{A}_{\mathsf{chal}} \subset \mathcal{N}$ where $A_{\mathsf{chal}} \subset A^*$ consists of the authorities appearing in the challenge ciphertext and $\bar{A}_{\mathsf{chal}} = A \setminus A_{\mathsf{chal}}$ consists of the authorities which do not appear in the challenge ciphertext. If $A_{\mathsf{chal}} \neq \emptyset$, then the challenger proceeds as follows:

  * Let $A_{\mathsf{chal}} = \{\mathsf{aid}_{j_1}^*, \ldots, \mathsf{aid}_{j_k}^*\}$ where $j_1, \ldots, j_k \in [\ell]$. For each $\mathsf{aid}_i^* \in A_{\mathsf{chal}}$, it samples $\mathbf{u}_{\mathsf{aid}_i^*,\mathsf{gid}} \leftarrow D_{\mathbb{Z},\chi}^m$.
  * Let $\mathbf{I}_L \in \mathbb{Z}_q^{L \times L}$ be the identity matrix. For each $i \in [\ell]$, associate authority $\mathsf{aid}_i^*$ with the $i^{\text{th}}$ row of $\mathbf{I}_L$. Define the matrix $\mathbf{M}_A \in \{0,1\}^{|A_{\mathsf{chal}}| \times L}$ to be the matrix formed by taking the rows of $\mathbf{I}_L$ associated with the identities in $A_{\mathsf{chal}}$.
  * The challenger samples

$$\mathbf{r}_{\mathsf{gid}} \leftarrow \begin{bmatrix} \mathbf{B}_{\mathsf{aid}_{j_1}^*} \\ \vdots \\ \mathbf{B}_{\mathsf{aid}_{j_k}^*} \end{bmatrix}_\chi^{-1} \left( \begin{bmatrix} \mathbf{A}_{\mathsf{aid}_{j_1}^*} \mathbf{u}_{\mathsf{aid}_{j_1}^*,\mathsf{gid}} - \mathbf{p}_{\mathsf{aid}_{j_1}^*} \\ \vdots \\ \mathbf{A}_{\mathsf{aid}_{j_k}^*} \mathbf{u}_{\mathsf{aid}_{j_k}^*,\mathsf{gid}} - \mathbf{p}_{\mathsf{aid}_{j_k}^*} \end{bmatrix} \right) \in \mathbb{Z}_q^{m(2L-1)}, \tag{5.1}$$

or more compactly, $\mathbf{r}_{\mathsf{gid}} \leftarrow ((\mathbf{M}_A \otimes \mathbf{I}_n)\mathbf{B})_\chi^{-1}(\mathbf{t}_{A,\mathsf{gid}})$, where

$$\mathbf{t}_{A,\mathsf{gid}} = \begin{bmatrix} \mathbf{A}_{\mathsf{aid}^*_{j_1}} \mathbf{u}_{\mathsf{aid}^*_{j_1},\mathsf{gid}} - \mathbf{p}_{\mathsf{aid}^*_{j_1}} \\ \vdots \\ \mathbf{A}_{\mathsf{aid}^*_{j_k}} \mathbf{u}_{\mathsf{aid}^*_{j_k},\mathsf{gid}} - \mathbf{p}_{\mathsf{aid}^*_{j_k}} \end{bmatrix}.$$

If $\mathbf{t}_{A,\mathsf{gid}}$ is not in the image of $(\mathbf{M}_A \otimes \mathbf{I}_n)\mathbf{B}$, then the challenger aborts the experiment with output 0.

* The challenger adds $(\mathsf{gid} \mapsto \mathbf{r}_{\mathsf{gid}})$ to T.

If $A_{\mathsf{chal}} = \emptyset$, then $\mathcal{B}$ samples $\mathbf{r}_{\mathsf{gid}} \leftarrow D_{\mathbb{Z},\chi}^{m(2L-1)}$ and adds the mapping $(\mathsf{gid} \mapsto \mathbf{r}_{\mathsf{gid}})$ to the table T.

Finally, the challenger constructs the secret keys for each authority $\mathsf{aid} \in A$ as follows:

* If $\mathsf{aid} \in A_{\mathsf{chal}}$, then $\mathsf{aid} = \mathsf{aid}_i^*$ for some $i \in [\ell]$. Algorithm $\mathcal{B}$ sets the secret key as $\mathsf{sk}_{\mathsf{aid}_i^*,\mathsf{gid}} = \mathbf{u}_{\mathsf{aid}_i^*,\mathsf{gid}}$.
* If $\mathsf{aid} \in \bar{A}_{\mathsf{chal}}$, the challenger computes $\mathbf{r}_{\mathsf{gid}} \leftarrow \mathsf{T}[\mathsf{gid}]$. It then sets $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} = \mathbf{u}_{\mathsf{aid},\mathsf{gid}} \leftarrow (\mathbf{A}_{\mathsf{aid}})_\chi^{-1}(\mathbf{p}_{\mathsf{aid}} + \mathbf{B}_{\mathsf{aid}}\mathbf{r}_{\mathsf{gid}})$ exactly as in $\mathsf{Hyb}_0^{(b)}$.

- $\mathsf{Hyb}_2^{(b)}$: Same as $\mathsf{Hyb}_1^{(b)}$, except the challenger constructs the challenge ciphertext as follows:
  - **Challenge ciphertext:** The challenger samples $\mathbf{s}_{\mathsf{aid}} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and $\mathbf{e}_{1,\mathsf{aid}} \leftarrow D_{\mathbb{Z},\chi}^m$ for each corrupted authority $\mathsf{aid} \in A^* \cap C$ (same as in $\mathsf{Hyb}_1^{(b)}$) and sets $\mathbf{c}_{1,\mathsf{aid}}^\top \leftarrow \mathbf{s}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\top$. For the honest authorities $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$, it samples $\mathbf{c}_{1,\mathsf{aid}_i^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^m$. Next, it samples $\mathbf{c}_2 \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{m(2L-1)}$ and $c_3 \xleftarrow{\mathrm{R}} \mathbb{Z}_q$. Finally, it outputs the challenge ciphertext
  $$\mathsf{ct} = \left(\left\{\mathbf{c}_{1,\mathsf{aid}}^\top\right\}_{\mathsf{aid} \in A}, \mathbf{c}_2, c_3\right).$$
  In particular, the challenge ciphertext is independent of $\mu_b$.

For an adversary $\mathcal{A}$, we write $\mathsf{Hyb}_i^{(b)}(\mathcal{A})$ to denote the output distribution of $\mathsf{Hyb}_i^{(b)}$ with adversary $\mathcal{A}$. We now show that each pair of adjacent distributions are computationally indistinguishable:

**Lemma 5.4.** *Suppose $q$ is prime and let $m_0(n,q) = O(n \log q)$ and $\chi_0(n,q) = \sqrt{n \log q} \cdot \omega(\sqrt{\log n})$ be the polynomials from Lemma 3.13. If $m \geq m_0$ and $\chi \geq \chi_0$, then for all adversaries $\mathcal{A}$ and $b \in \{0,1\}$, $\mathsf{Hyb}_0^{(b)}(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_1^{(b)}(\mathcal{A})$.*

*Proof.* Consider a secret key query $(\mathsf{gid}, A)$. Certainly if $A \subseteq \mathcal{N} \setminus A^*$ (i.e., $A_{\mathsf{chal}} = \emptyset$), the distribution of the secret keys is identical in $\mathsf{Hyb}_0^{(b)}$ and $\mathsf{Hyb}_1^{(b)}$. Consider the case where $A_{\mathsf{chal}} \neq \emptyset$. We consider the distribution of $\left(\mathbf{r}_{\mathsf{gid}}, \{\mathbf{u}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid} \in A_{\mathsf{chal}}}\right)$ in $\mathsf{Hyb}_0^{(b)}$ and $\mathsf{Hyb}_1^{(b)}$. In both experiments, these components are sampled so as to satisfy the linear system:

$$\left[\begin{array}{ccc|c} \mathbf{A}_{\mathsf{aid}^*_{j_1}} & & & -\mathbf{B}_{\mathsf{aid}^*_{j_1}} \\ & \ddots & & \vdots \\ & & \mathbf{A}_{\mathsf{aid}^*_{j_k}} & -\mathbf{B}_{\mathsf{aid}^*_{j_k}} \end{array}\right] \cdot \begin{bmatrix} \mathbf{u}_{\mathsf{aid}^*_{j_1},\mathsf{gid}} \\ \vdots \\ \mathbf{u}_{\mathsf{aid}^*_{j_k},\mathsf{gid}} \\ \mathbf{r}_{\mathsf{gid}} \end{bmatrix} = \begin{bmatrix} \mathbf{p}_{\mathsf{aid}^*_{j_1}} \\ \vdots \\ \mathbf{p}_{\mathsf{aid}^*_{j_k}} \end{bmatrix}. \tag{5.2}$$

Consider the joint distribution of $\mathbf{r}_{\mathsf{gid}}$ and the $\{\mathbf{u}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid} \in A_{\mathsf{chal}}}$ in $\mathsf{Hyb}_0^{(b)}$. We start by characterizing the marginal distribution of each $\mathbf{u}_{\mathsf{aid},\mathsf{gid}}$ in $\mathsf{Hyb}_0^{(b)}$:

- In $\mathsf{Hyb}_0^{(b)}$, the challenger samples $\mathbf{r}_{\mathsf{gid}} \leftarrow D_{\mathbb{Z},\chi}^{m(2L-1)}$. Define the matrix $\mathbf{B}^* \in \mathbb{Z}_q^{nk \times m(2L-1)}$ where

$$\mathbf{B}^* = \begin{bmatrix} \mathbf{B}_{\mathsf{aid}^*_{j_1}} \\ \vdots \\ \mathbf{B}_{\mathsf{aid}^*_{j_k}} \end{bmatrix} \in \mathbb{Z}_q^{nk \times m(2L-1)}.$$

In $\mathsf{Hyb}_0^{(b)}$, the distribution of $\mathbf{B}^*$ is uniform over $\mathbb{Z}_q^{nk \times m(2L-1)}$. Since $k < L$, and $m \geq m_0(n,q) = O(n \log q)$, we have that $m(2L-1) \geq m_0(nk, q)$. Moreover, since $\chi \geq \sqrt{n \log q} \cdot \omega(\sqrt{\log n})$, we conclude by Lemma 3.13 that the distribution of $\mathbf{B}^* \mathbf{r}_{\mathsf{gid}}$ is statistically close to uniform over $\mathbb{Z}_q^{nk}$. This means that for all $\mathsf{aid} \in A_{\mathsf{chal}}$, the distribution of $\mathbf{p}_{\mathsf{aid}} - \mathbf{B}_{\mathsf{aid}} \mathbf{r}_{\mathsf{gid}}$ is independent and uniform over $\mathbb{Z}_q^n$.

- Then, for each $\mathsf{aid} \in A_{\mathsf{chal}}$, the challenger samples $\mathbf{u}_{\mathsf{aid},\mathsf{gid}} \leftarrow (\mathbf{A}_{\mathsf{aid}})_\chi^{-1}(\mathbf{p}_{\mathsf{aid}} - \mathbf{B}_{\mathsf{aid}} \mathbf{r}_{\mathsf{gid}})$. From above, the marginal distribution of each $\mathbf{p}_{\mathsf{aid}} - \mathbf{B}_{\mathsf{aid}} \mathbf{r}_{\mathsf{gid}}$ is uniform and independent over $\mathbb{Z}_q^n$. Finally, since $\mathbf{A}_{\mathsf{aid}_{j_i}^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m}$ in $\mathsf{Hyb}_0^{(b)}$, we again appeal to Lemma 3.13 to conclude that the marginal distribution of each $\mathbf{u}_{\mathsf{aid},\mathsf{gid}}$ is statistically close to $D_{\mathbb{Z},\chi}^m$. Moreover, each of the $\mathbf{u}_{\mathsf{aid},\mathsf{gid}}$'s are independent.

Thus, in $\mathsf{Hyb}_0^{(b)}$, the distribution of each $\mathbf{u}_{\mathsf{aid},\mathsf{gid}}$ is statistically close to the discrete Gaussian distribution $D_{\mathbb{Z},\chi}^m$. We now characterize the conditional distribution of $\mathbf{r}_{\mathsf{gid}}$ given $\{\mathbf{u}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid} \in A_{\mathsf{chal}}}$ in $\mathsf{Hyb}_0^{(b)}$. By Eq. (5.2), this is a discrete Gaussian $D_{\mathbb{Z},\chi}^{m(2L-1)}$ conditioned on

$$\mathbf{B}^* \mathbf{r}_{\mathsf{gid}} = \begin{bmatrix} \mathbf{A}_{\mathsf{aid}_{j_1}^*} \mathbf{u}_{\mathsf{aid}_{j_1}^*,\mathsf{gid}} - \mathbf{p}_{\mathsf{aid}_{j_1}^*} \\ \vdots \\ \mathbf{A}_{\mathsf{aid}_{j_k}^*} \mathbf{u}_{\mathsf{aid}_{j_k}^*,\mathsf{gid}} - \mathbf{p}_{\mathsf{aid}_{j_k}^*} \end{bmatrix}.$$

By definition, the conditional distribution of $\mathbf{r}_{\mathsf{gid}}$ given $\{\mathbf{u}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid} \in A_{\mathsf{chal}}}$ is precisely

$$(\mathbf{B}^*)_\chi^{-1}\left(\begin{bmatrix} \mathbf{A}_{\mathsf{aid}_{j_1}^*} \mathbf{u}_{\mathsf{aid}_{j_1}^*,\mathsf{gid}} - \mathbf{p}_{\mathsf{aid}_{j_1}^*} \\ \vdots \\ \mathbf{A}_{\mathsf{aid}_{j_k}^*} \mathbf{u}_{\mathsf{aid}_{j_k}^*,\mathsf{gid}} - \mathbf{p}_{\mathsf{aid}_{j_k}^*} \end{bmatrix}\right).$$

This coincides precisely with the distribution in $\mathsf{Hyb}_1^{(b)}$ and the claim follows. □

**Lemma 5.5.** *Under the* $\mathsf{RTLWE}_{n,Lm+1,m,q,\chi,L}$ *assumption and modeling* $\mathsf{H}$ *as a random oracle, for all efficient adversaries* $\mathcal{A}$, $\mathsf{Hyb}_1^{(b)}(\mathcal{A}) \stackrel{c}{\approx} \mathsf{Hyb}_2^{(b)}(\mathcal{A})$.

*Proof.* Suppose there exists an efficient adversary $\mathcal{A}$ that distinguishes $\mathsf{Hyb}_1^{(b)}$ from $\mathsf{Hyb}_2^{(b)}$ with advantage $\varepsilon > 0$. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ for the RTLWE assumption:

1. Algorithm $\mathcal{B}$ starts running algorithm $\mathcal{A}$. In the static security model, algorithm $\mathcal{A}$ outputs the following:
   - A set of corrupted authorities $C \subset \mathcal{AU}$ and their public keys $\mathsf{pk}_{\mathsf{aid}} = (\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$ for each $\mathsf{aid} \in C$.
   - A list of non-corrupted authorities $\mathcal{N} \subseteq \mathcal{AU}$.
   - A list of secret-key queries $Q = \{(\mathsf{gid}, A)\}$ where each $A \subset \mathcal{N}$.
   - A pair of challenge messages $\mu_0, \mu_1 \in \{0,1\}$ and a set of authorities $A^* \subseteq C \cup \mathcal{N}$.
2. Let $\ell = |A^* \cap \mathcal{N}|$, and let $A^* \cap \mathcal{N} = \{\mathsf{aid}_1^*, \ldots, \mathsf{aid}_\ell^*\}$. Algorithm $\mathcal{B}$ constructs the vector $\mathbf{u} \in \{0,1\}^L$ where $u_i = 1$ for $i \in [\ell]$ and $u_i = 0$ for $i > \ell$. In the reduction, the first $\ell$ entries of $\mathbf{u}$ are associated with the $\ell$ non-corrupt authorities that appear in the challenge ciphertext. The remaining $L - \ell$ entries are unused.
3. In response, algorithm $\mathcal{B}$ receives a challenge $(\mathbf{A}, \mathbf{B}, \mathbf{z}^\intercal, \hat{\mathbf{z}}^\intercal)$ where $\mathbf{A} \in \mathbb{Z}_q^{n \times (Lm+1)}$, $\mathbf{B} \in \mathbb{Z}_q^{nL \times m(2L-1)}$, $\mathbf{z} \in \mathbb{Z}_q^{Lm+1}$, and $\hat{\mathbf{z}} \in \mathbb{Z}_q^{m(2L-1)}$. Algorithm $\mathcal{B}$ parses

$$\mathbf{A} = \left[\mathbf{A}_{\mathsf{aid}_1^*} \mid \cdots \mid \mathbf{A}_{\mathsf{aid}_\ell^*} \mid \mathbf{A}_{\mathsf{rest}} \mid \mathbf{p}\right]$$
$$\mathbf{z}^\intercal = \left[\mathbf{z}_{\mathsf{aid}_1^*}^\intercal \mid \cdots \mid \mathbf{z}_{\mathsf{aid}_\ell^*}^\intercal \mid \mathbf{z}_{\mathsf{rest}}^\intercal \mid t\right],$$

where $\mathbf{A}_{\mathsf{aid}_i^*} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{p} \in \mathbb{Z}_q^n$, $\mathbf{z}_{\mathsf{aid}_i^*} \in \mathbb{Z}_q^m$, $t \in \mathbb{Z}_q$; the remaining components $\mathbf{A}_{\mathsf{rest}} \in \mathbb{Z}_q^{n\times(L-\ell)m}$ and $\mathbf{z}_{\mathsf{rest}} \in \mathbb{Z}_q^{(L-\ell)m}$ denote *unused* elements. It also parses $\mathbf{B}$ as

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_{\mathsf{aid}_1^*} \\ \vdots \\ \mathbf{B}_{\mathsf{aid}_\ell^*} \\ \mathbf{B}_{\mathsf{rest}} \end{bmatrix} \in \mathbb{Z}_q^{nL\times m(2L-1)},$$

where each $\mathbf{B}_{\mathsf{aid}_i^*} \in \mathbb{Z}_q^{n\times m(2L-1)}$; the remaining block $\mathbf{B}_{\mathsf{rest}} \in \mathbb{Z}_q^{n(L-\ell)\times m(2L-1)}$ is *unused* in the reduction.

4. Algorithm $\mathcal{B}$ initializes an empty table $\mathsf{T}\colon \mathcal{GID} \to \mathbb{Z}_q^{m(2L-1)}$ that it will use for responding to random oracle queries. Algorithm $\mathcal{B}$ computes responses to the adversary's queries as follows:

   - **Public keys for non-corrupted authorities:** Algorithm $\mathcal{B}$ constructs the public keys for authorities in $\mathcal{N} \cap A^*$ and $\mathcal{N} \setminus A^*$ as follows:
     - For each $i \in [\ell]$, the challenger samples vectors $\mathbf{p}_{\mathsf{aid}_i^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ such that $\sum_{i\in[\ell]} \mathbf{p}_{\mathsf{aid}_i^*} = \mathbf{p}$. Then, for each $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$, algorithm $\mathcal{B}$ constructs the public key as $\mathsf{pk}_{\mathsf{aid}_i^*} = (\mathbf{A}_{\mathsf{aid}_i^*}, \mathbf{B}_{\mathsf{aid}_i^*}, \mathbf{p}_{\mathsf{aid}_i^*})$.
     - For authorities $\mathsf{aid} \in \mathcal{N} \setminus A^*$, the challenger samples $(\mathbf{A}_{\mathsf{aid}}, \mathsf{td}_{\mathsf{aid}}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$, $\mathbf{p}_{\mathsf{aid}} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and $\mathbf{B}_{\mathsf{aid}} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n\times m(2L-1)}$ as in the real scheme. It sets the public key to $\mathsf{pk}_{\mathsf{aid}} = (\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$.
   - **Secret keys:** Let $(\mathsf{gid}, A)$ be a secret-key query. The challenger partitions $A = A_{\mathsf{chal}} \cup \bar{A}_{\mathsf{chal}} \subset \mathcal{N}$ exactly as prescribed in $\mathsf{Hyb}_1^{(b)}$ and $\mathsf{Hyb}_2^{(b)}$.
     - If $A_{\mathsf{chal}} \neq \emptyset$, then algorithm $\mathcal{B}$ samples $\mathbf{u}_{\mathsf{aid}_i^*,\mathsf{gid}} \leftarrow D_{\mathbb{Z},\chi}^m$ for each $\mathsf{aid}_i^* \in A_{\mathsf{chal}}$ and constructs the matrix $\mathbf{M}_A$ and $\mathbf{t}_{A,\mathsf{gid}}$ as defined in $\mathsf{Hyb}_1^{(b)}$ and $\mathsf{Hyb}_2^{(b)}$. It makes an oracle query on $(\mathbf{M}_A, \mathbf{t}_{A,\mathsf{gid}})$ to obtain a vector $\mathbf{r}_{\mathsf{gid}} \in \mathbb{Z}_q^m$. The challenger adds the mapping $(\mathsf{gid} \mapsto \mathbf{r}_{\mathsf{gid}})$ to the table $\mathsf{T}$.
     - If $A_{\mathsf{chal}} = \emptyset$, then $\mathcal{B}$ samples $\mathbf{r}_{\mathsf{gid}} \leftarrow D_{\mathbb{Z},\chi}^{m(2L-1)}$ and adds the mapping $(\mathsf{gid} \mapsto \mathbf{r}_{\mathsf{gid}})$ to the table $\mathsf{T}$.

     Algorithm $\mathcal{B}$ then constructs the secret keys for each authority $\mathsf{aid} \in A$ as follows:
     - If $\mathsf{aid} \in A_{\mathsf{chal}}$, then $\mathsf{aid} = \mathsf{aid}_i^*$ for some $i \in [\ell]$. Algorithm $\mathcal{B}$ sets the secret key as $\mathsf{sk}_{\mathsf{aid}_i^*,\mathsf{gid}} = \mathbf{u}_{\mathsf{aid}_i^*,\mathsf{gid}}$.
     - If $\mathsf{aid} \in \bar{A}_{\mathsf{chal}}$, then algorithm $\mathcal{B}$ knows the associated trapdoor $\mathsf{td}_{\mathsf{aid}}$. Then, it computes $\mathbf{r}_{\mathsf{gid}} \leftarrow \mathsf{T}[\mathsf{gid}]$. It samples $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} = \mathbf{u}_{\mathsf{aid},\mathsf{gid}} \leftarrow \mathbf{A}_{\mathsf{aid}}^{-1}(\mathbf{p}_{\mathsf{aid}} + \mathbf{B}_{\mathsf{aid}}\mathbf{r}_{\mathsf{gid}})$ using the trapdoor $\mathsf{td}_{\mathsf{aid}}$.
   - **Challenge ciphertext:** To construct the challenge ciphertext, algorithm $\mathcal{B}$ proceeds as follows:
     - For each $\mathsf{aid} \in A^*$, algorithm $\mathcal{B}$ first samples $\mathbf{s}_{\mathsf{aid}} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$.
       - For each $\mathsf{aid} \in A^* \cap C$, algorithm $\mathcal{B}$ samples $\mathbf{e}_{1,\mathsf{aid}} \leftarrow D_{\mathbb{Z},\chi}^m$ and computes $\mathbf{c}_{1,\mathsf{aid}}^\mathsf{T} = \mathbf{s}_{\mathsf{aid}}^\mathsf{T}\mathbf{A}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\mathsf{T}$.
       - For each $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$, algorithm $\mathcal{B}$ computes $\mathbf{c}_{1,\mathsf{aid}_i^*}^\mathsf{T} \leftarrow \mathbf{s}_{\mathsf{aid}_i^*}^\mathsf{T}\mathbf{A}_{\mathsf{aid}_i^*} + \mathbf{z}_{\mathsf{aid}_i^*}$.
     - For the remaining ciphertext components $\mathbf{c}_2$ and $c_3$, algorithm $\mathcal{B}$ computes

$$\mathbf{c}_2^\mathsf{T} = \sum_{\mathsf{aid}\in A^*\cap C} \mathbf{s}_{\mathsf{aid}}^\mathsf{T}\mathbf{B}_{\mathsf{aid}} + \sum_{\mathsf{aid}_i^*\in A^*\cap\mathcal{N}} \mathbf{s}_{\mathsf{aid}_i^*}^\mathsf{T}\mathbf{B}_{\mathsf{aid}_i^*} + \hat{\mathbf{z}}^\mathsf{T}$$
$$c_3 = \sum_{\mathsf{aid}\in A^*\cap C} \mathbf{s}_{\mathsf{aid}}^\mathsf{T}\mathbf{p}_{\mathsf{aid}} + \sum_{\mathsf{aid}_i^*\in A^*\cap\mathcal{N}} \mathbf{s}_{\mathsf{aid}_i^*}^\mathsf{T}\mathbf{p}_{\mathsf{aid}_i^*} + \mu_b \cdot \lfloor q/2 \rceil + t.$$

     The challenger constructs the challenge ciphertext as $\mathsf{ct} = \left(\left\{\mathbf{c}_{1,\mathsf{aid}}\right\}_{\mathsf{aid}\in A^*}, \mathbf{c}_2, c_3\right)$.

5. Whenever $\mathcal{A}$ makes a random oracle query on an input gid, algorithm $\mathcal{B}$ checks if there exists a mapping $(\mathsf{gid} \mapsto \mathbf{r}_{\mathsf{gid}})$ in $\mathsf{T}$. If so, it replies with $\mathbf{r}_{\mathsf{gid}}$. Otherwise, it samples $\mathbf{r}_{\mathsf{gid}} \leftarrow D_{\mathbb{Z},\chi}^{m(2L-1)}$, adds the mapping $(\mathsf{gid} \mapsto \mathbf{r}_{\mathsf{gid}})$ to $\mathsf{T}$, and replies with $\mathbf{r}_{\mathsf{gid}}$.

6. At the end of the game, algorithm $\mathcal{A}$ outputs a bit $b \in \{0,1\}$, which $\mathcal{B}$ also outputs.

To complete the proof, we argue that $\mathcal{B}$ simulates either $\mathsf{Hyb}_1^{(b)}$ or $\mathsf{Hyb}_2^{(b)}$ for $\mathcal{A}$. We first consider the distribution of the public keys and the secret keys:

- **Public keys for non-corrupted authorities:** The public keys for authorities $\mathsf{aid} \in \mathcal{N} \setminus A^*$ that are not in the challenge ciphertext are generated exactly as in the real scheme (same as $\mathsf{Hyb}_0^{(b)}$ and $\mathsf{Hyb}_1^{(b)}$). For an authority $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$, the matrices $\mathbf{A}_{\mathsf{aid}_i^*}$ and $\mathbf{B}_{\mathsf{aid}_i^*}$ are from the RTLWE assumption, so they are uniformly and independently random. Finally, since $\mathbf{p} \in \mathbb{Z}_q^n$ is uniform, so is $\mathbf{p}_{\mathsf{aid}_i^*}$ for all $i \in [\ell]$. Thus, the public keys $\mathsf{pk}_{\mathsf{aid}_i^*}$ are all correctly distributed.
- **Secret keys:** Consider a secret key query $(\mathsf{gid}, A)$. If $A \subseteq \mathcal{N} \setminus A^*$ (i.e., $A_{\mathsf{chal}} = \emptyset$), then algorithm $\mathcal{B}$ constructs $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}}$ using the *same* procedure as in $\mathsf{Hyb}_1^{(b)}$ and $\mathsf{Hyb}_2^{(b)}$ (since it knows the trapdoor for all authorities not present in the challenge ciphertext). Consider the case where $A_{\mathsf{chal}} \neq \emptyset$. Let $A_{\mathsf{chal}} = \{\mathsf{aid}_{j_1}^*, \ldots, \mathsf{aid}_{j_k}^*\}$. Consider the query $(\mathbf{M}_A, \mathbf{t}_{A,\mathsf{gid}})$ algorithm $\mathcal{B}$ makes to its oracle. Since $\mathbf{M}_A$ is constructed by taking a subset of the rows of $\mathbf{I}_L$ (identified by the indices $j_1, \ldots, j_k \in [\ell]$), the matrix $\mathbf{M}_A$ is full rank. Moreover, since $\mathcal{A}$ is admissible $A_{\mathsf{chal}} \subsetneq [\ell]$, which means that $\mathbf{u}^\top$ is *not* in the row-span of $\mathbf{M}_A$. This means the matrix $\begin{bmatrix} \mathbf{M}_A \\ \mathbf{u}^\top \end{bmatrix}$ is full rank and represents a valid query to the RTLWE oracle. By construction of the RTLWE oracle, algorithm $\mathcal{B}$ perfectly simulates the distribution of secret keys in $\mathsf{Hyb}_1^{(b)}$ and $\mathsf{Hyb}_2^{(b)}$.

Next, we consider the distribution of the challenge ciphertext. Here, we consider two possibilities depending on the challenge distribution:

- Suppose $\mathbf{z}^\top = \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top$ and $\hat{\mathbf{z}}^\top = \mathbf{s}^\top (\mathbf{u}^\top \otimes \mathbf{I}_n)\mathbf{B} + \hat{\mathbf{e}}^\top$, where $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow D_{\mathbb{Z},\chi}^{Lm+1}$, $\hat{\mathbf{e}}^\top \leftarrow \hat{\mathbf{e}}_0^\top [\mathbf{I}_{mL} \mid \mathbf{R}]$, $\hat{\mathbf{e}}_0^\top \leftarrow D_{\mathbb{Z},\chi}^{mL}$, and $\mathbf{R} \xleftarrow{R} \{-1, 1\}^{mL \times m(L-1)}$. Parse $\mathbf{e}^\top = \left[\tilde{\mathbf{e}}_{\mathsf{aid}_1^*}^\top \mid \cdots \mid \tilde{\mathbf{e}}_{\mathsf{aid}_L^*}^\top \mid \tilde{e}\right]$ where each $\tilde{\mathbf{e}}_{\mathsf{aid}_i^*} \in \mathbb{Z}_q^m$. In particular, this means that $\mathbf{z}_{\mathsf{aid}_i^*}^\top = \mathbf{s}^\top \mathbf{A}_{\mathsf{aid}_i^*} + \tilde{\mathbf{e}}_{\mathsf{aid}_i^*}^\top$. Consider each component of the ciphertext:
  - Consider the first ciphertext component. For $\mathsf{aid} \in A^* \cap C$, let $\tilde{\mathbf{s}}_{\mathsf{aid}} = \mathbf{s}_{\mathsf{aid}}$ and $\tilde{\mathbf{e}}_{\mathsf{aid}} = \mathbf{e}_{1,\mathsf{aid}}$. For $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$, let $\tilde{\mathbf{s}}_{\mathsf{aid}_i^*} = \mathbf{s} + \mathbf{s}_{\mathsf{aid}_i^*}$ By construction each $\tilde{\mathbf{s}}_{\mathsf{aid}}$ is uniform and independent over $\mathbb{Z}_q^n$. For corrupted authorities $\mathsf{aid} \in A^* \cap C$, algorithm $\mathcal{B}$ computes
  $$\mathbf{s}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\top = \tilde{\mathbf{s}}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} + \tilde{\mathbf{e}}_{\mathsf{aid}}^\top.$$
  For the honest authorities $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$, algorithm $\mathcal{B}$ computes
  $$\mathbf{s}_{\mathsf{aid}_i^*}^\top \mathbf{A}_{\mathsf{aid}_i^*} + \mathbf{z}_{\mathsf{aid}_i^*}^\top = (\mathbf{s}_{\mathsf{aid}_i^*} + \mathbf{s})^\top \mathbf{A}_{\mathsf{aid}_i^*} + \tilde{\mathbf{e}}_{\mathsf{aid}_i^*}^\top = \tilde{\mathbf{s}}_{\mathsf{aid}_i^*}^\top \mathbf{A}_{\mathsf{aid}_i^*} + \tilde{\mathbf{e}}_{\mathsf{aid}_i^*}^\top,$$
  Thus, we have that
  $$\left\{\mathbf{s}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} + \tilde{\mathbf{e}}_{1,\mathsf{aid}}^\top\right\}_{\mathsf{aid} \in A^* \cap C} \cup \left\{\mathbf{s}_{\mathsf{aid}_i^*}^\top \mathbf{A}_{\mathsf{aid}_i^*} + \mathbf{z}_{\mathsf{aid}_i^*}^\top\right\}_{\mathsf{aid}_i^* \in A^* \cap \mathcal{N}} = \left\{\tilde{\mathbf{s}}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} + \tilde{\mathbf{e}}_{\mathsf{aid}}^\top\right\}_{\mathsf{aid} \in A^*},$$
  which matches the distribution in $\mathsf{Hyb}_1^{(b)}$.
  - Consider the second ciphertext component $\mathbf{c}_2$. Recall that $\mathbf{u} = [\mathbf{1}^\ell \mid \mathbf{0}^{L-\ell}]$ and that $A^* \cap \mathcal{N} = \{\mathsf{aid}_1^*, \ldots, \mathsf{aid}_\ell^*\}$. Thus,
  $$\mathbf{s}^\top (\mathbf{u}^\top \otimes \mathbf{I}_n)\mathbf{B} = \sum_{i \in [\ell]} \mathbf{s}^\top \mathbf{B}_{\mathsf{aid}_i^*} = \sum_{\mathsf{aid}_i^* \in A^* \cap \mathcal{N}} \mathbf{s}^\top \mathbf{B}_{\mathsf{aid}_i^*}.$$
  Then, we have
  $$\begin{aligned} \mathbf{c}_2^\top &= \sum_{\mathsf{aid} \in A^* \cap C} \mathbf{s}_{\mathsf{aid}}^\top \mathbf{B}_{\mathsf{aid}} + \sum_{\mathsf{aid}_i^* \in A^* \cap \mathcal{N}} \mathbf{s}_{\mathsf{aid}_i^*}^\top \mathbf{B}_{\mathsf{aid}_i^*} + \mathbf{s}^\top (\mathbf{u}^\top \otimes \mathbf{I}_n)\mathbf{B} + \hat{\mathbf{e}}^\top \\ &= \sum_{\mathsf{aid} \in A^* \cap C} \tilde{\mathbf{s}}_{\mathsf{aid}}^\top \mathbf{B}_{\mathsf{aid}} + \sum_{\mathsf{aid}_i^* \in A^* \cap \mathcal{N}} \tilde{\mathbf{s}}_{\mathsf{aid}_i^*}^\top \mathbf{B}_{\mathsf{aid}_i^*} + \hat{\mathbf{e}}_0^\top [\mathbf{I}_{mL} \mid \mathbf{R}] \\ &= \sum_{\mathsf{aid} \in A^*} \tilde{\mathbf{s}}_{\mathsf{aid}}^\top \mathbf{B}_{\mathsf{aid}} + \hat{\mathbf{e}}_0^\top [\mathbf{I}_{mL} \mid \mathbf{R}], \end{aligned}$$
  which is precisely the distribution of $\mathbf{c}_2$ in $\mathsf{Hyb}_1^{(b)}$.

– Consider the final ciphertext component $c_3$. Recall that $\mathbf{p} = \sum_{i\in[\ell]} \mathbf{p}_i$. Thus,

$$\mathbf{s}^\mathsf{T}\mathbf{p} = \sum_{i\in[\ell]} \mathbf{s}^\mathsf{T}\mathbf{p}_{\mathsf{aid}_i^*} = \sum_{\mathsf{aid}_i^*\in A^*\cap\mathcal{N}} \mathbf{s}^\mathsf{T}\mathbf{p}_{\mathsf{aid}_i^*}.$$

Then, we have

$$\begin{aligned} c_3 &= \sum_{\mathsf{aid}\in A^*\cap C} \mathbf{s}_{\mathsf{aid}}^\mathsf{T}\mathbf{p}_{\mathsf{aid}} + \sum_{\mathsf{aid}_i^*\in A^*\cap\mathcal{N}} \mathbf{s}_{\mathsf{aid}_i^*}^\mathsf{T}\mathbf{p}_{\mathsf{aid}_i^*} + \mu_b\cdot\lfloor q/2\rceil + \mathbf{s}^\mathsf{T}\mathbf{p} + \tilde{e} \\ &= \sum_{\mathsf{aid}\in A^*\cap C} \tilde{\mathbf{s}}_{\mathsf{aid}}^\mathsf{T}\mathbf{p}_{\mathsf{aid}} + \sum_{\mathsf{aid}_i^*\in A^*\cap\mathcal{N}} \tilde{\mathbf{s}}_{\mathsf{aid}_i^*}\mathbf{p}_{\mathsf{aid}_i^*} + \tilde{e} + \mu_b\cdot\lfloor q/2\rceil \\ &= \sum_{\mathsf{aid}\in A^*} \tilde{\mathbf{s}}_{\mathsf{aid}}^\mathsf{T}\mathbf{p}_{\mathsf{aid}} + \tilde{e} + \mu_b\cdot\lfloor q/2\rceil, \end{aligned}$$

which again coincides with the distribution in $\mathsf{Hyb}_1^{(b)}$.

In this case, the challenge ciphertext is distributed exactly as in $\mathsf{Hyb}_1^{(b)}$.

- Suppose $\mathbf{z} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{Lm+1}$ and $\hat{\mathbf{z}} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{m(2L-1)}$. Similar to before, we consider each component of the ciphertext:
  – Consider the first ciphertext component. Since $\mathbf{z}$ is uniform and sampled independently of $\mathbf{s}_{\mathsf{aid}^*}$ and $\mathbf{A}$, the distribution of $\mathbf{s}_{\mathsf{aid}_i^*}^\mathsf{T}\mathbf{A}_{\mathsf{aid}_i^*} + \mathbf{z}_{\mathsf{aid}_i^*}$ is independently uniform for all $\mathsf{aid}_i^* \in A^*\cap\mathcal{N}$. Finally, algorithm $\mathcal{B}$ constructs the components $\mathbf{s}_{\mathsf{aid}}^\mathsf{T}\mathbf{A}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\mathsf{T}$ for $\mathsf{aid}\in A^*\cap C$ exactly as in the real scheme. Thus, the distribution of the first ciphertext component matches that in $\mathsf{Hyb}_2^{(b)}$.
  – The remaining ciphertext components $\mathbf{c}_2$ and $c_3$ are independent and uniform over $\mathbb{Z}_q^{m(2L-1)}$ and $\mathbb{Z}_q$, respectively (since $\hat{\mathbf{z}} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{m(2L-1)}$ and $t \xleftarrow{\mathrm{R}} \mathbb{Z}_q$ are uniform and independent of all other components). Once more, this coincides with the distribution in $\mathsf{Hyb}_2^{(b)}$.

  In this case, the challenge ciphertext is distributed as in $\mathsf{Hyb}_2^{(b)}$.

Thus, if the RTLWE challenge is pseudorandom, then algorithm $\mathcal{B}$ simulates $\mathsf{Hyb}_1^{(b)}$ for $\mathcal{A}$. Conversely, if the RTLWE challenge is random, then algorithm $\mathcal{B}$ simulates $\mathsf{Hyb}_2^{(b)}$ for $\mathcal{A}$. □

Since the distribution of $\mathsf{Hyb}_2^{(b)}$ is independent of the bit $b$, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_2^{(0)}(\mathcal{A}) \equiv \mathsf{Hyb}_2^{(1)}(\mathcal{A})$. Then, combining Lemmas 5.4 and 5.5, for all efficient adversaries $\mathcal{A}$, $\mathsf{Hyb}_0^{(0)}(\mathcal{A}) \overset{c}{\approx} \mathsf{Hyb}_0^{(1)}(\mathcal{A})$ and Construction 5.1 satisfies static security. □

**Parameter setting.** Let $\lambda$ be a security parameter. We can now instantiate Construction 5.1 as follows:

- We set the lattice dimension $n = \lambda$.
- To rely on Theorem 5.3, we rely on the $\mathsf{RTLWE}_{n,Lm+1,m,q,\chi,L}$ assumption. By Theorem 4.2, this reduces to $\mathsf{LWE}_{n,2Lm+1,q,\chi}$ if we set $m = O(n\log q)$, $q > 2$ to a prime, and $\chi = O(m^2L^2\log n)$.
- For correctness (Theorem 5.2), we additionally require $q = O(\lambda\chi^2m^2L^2)$.

In particular, this means we can choose $m, q, \chi$ to be polynomials in $\lambda$, and thus, base hardness on LWE with a *polynomial* modulus-to-noise ratio. We summarize the instantiation below:

**Corollary 5.6** (Multi-Authority ABE for Subset Policies in the Random Oracle Model)**.** *Let $\lambda$ be a security parameter. Assuming polynomial hardness of LWE with a polynomial modulus-to-noise ratio, there exists a statically-secure multi-authority ABE scheme for subset policies of* a priori *bounded length $L = L(\lambda)$ in the random oracle model. The size of the ciphertext scales quasi-linearly with the bound $L$.*

## 5.1 Instantiating using a Random Oracle with Uniform Outputs

As described, Construction 5.1 and Corollary 5.6 relies on a random oracle $\mathsf{H}\colon \mathcal{GID} \to \mathbb{Z}_q^{m(2L-1)}$ whose output distribution is the discrete Gaussian distribution $D_{\mathbb{Z},\chi}^{m(2L-1)}$. Since $\chi = \mathrm{poly}(\lambda)$ in our setting, we describe a simple way to instantiate $\mathsf{H}$ using a random oracle $\mathsf{H}'\colon \mathcal{GID} \to \{0,1\}^{\lambda m(2L-1)}$ whose output distribution is the uniform distribution via inversion sampling. The function $\mathsf{H}'$ coincides with the usual way we model the output distribution of a random oracle [BR93].

Previously, Brakerski et al. [BCTW16] sketched an alternative approach for instantiating a random oracle outputting samples from a discrete Gaussian distribution by adapting the rejection sampler of Lyubashevsky and Wichs [LW15]. Datta et al. [DKW21a] rely on noise smudging in their setting (which would in turn necessitate using a super-polynomial modulus-to-noise ratio). In our setting where we have a distribution with polynomial-size support, we describe a simple alternative based on inversion sampling. This is a simple approach used in concrete implementations of lattice-based cryptography [BCD+16].

**Lemma 5.7** (Inversion Sampling)**.** *Let $\lambda$ be a security parameter, $t = t(\lambda)$ be an input length, and $D$ be a discrete $B$-bounded distribution with an efficiently-computable cumulative distribution function. Then, there exists a pair of efficient algorithms* $(\mathsf{Project}, \mathsf{SampleR})$ *with the following properties:*

- $\mathsf{Project}(x) \to y$*: On input an input $x \in \{0,1\}^t$, the projection algorithm outputs a sample $y \in [-B, B]$. The projection algorithm is* deterministic*.*
- $\mathsf{SampleR}(y) \to x$*: On input a value $y \in [-B, B]$, the reverse sampling algorithm outputs an $x \in \{0,1\}^t$.*

*In addition, the following properties hold:*

- **Correctness:** *For all $y \in [-B, B]$,* $\Pr[\mathsf{Project}(\mathsf{SampleR}(y)) = y] = 1$.
- **Reverse-sampleability:** *For all $t > \log B + \omega(\log \lambda)$, the following two distributions are statistically indistinguishable:*

$$\{(x, \mathsf{Project}(x)) : x \xleftarrow{\mathrm{R}} (\{0,1\}^t)\} \quad \text{and} \quad \{(\mathsf{SampleR}(y), y) : y \leftarrow D\}.$$

*Proof.* We take $(\mathsf{Project}, \mathsf{SampleR})$ to be the standard inversion sampling algorithm. Let $f\colon [-B-1, B] \to [0,1]$ be the cumulative distribution function for $D$, and let $T = 2^t - 1$. We construct the two algorithms as follows:

- $\mathsf{Project}(x)$: On input $x \in \{0,1\}^t$, let $X \in [0, T]$ be the integer whose binary representation is $x$. Output $y \in [-B, B]$ where $T \cdot f(y-1) < X \le T \cdot f(y)$.
- $\mathsf{SampleR}(y)$: On input $y \in [-B, B]$, let $x_0 \leftarrow T \cdot f(y-1)$ and $x_1 \leftarrow T \cdot f(y)$. Output the binary representation of the element $x \xleftarrow{\mathrm{R}} (x_0, x_1] \cap \mathbb{Z}$.

Since the cumulative distribution function $f$ is efficiently-computable and the Project algorithm can be computed with $\mathrm{polylog}(B)$ calls to $f$ (e.g., using binary search), the Project algorithm is efficiently-computable. The SampleR algorithm only requires making two calls to $f$ and is likewise efficient. Next, correctness of the algorithm follows by construction. Finally, for the reverse-sampleability property, take any $Y \in [-B, B]$. Then,

$$\begin{aligned}\Pr[\mathsf{Project}(x) = Y : x \xleftarrow{\mathrm{R}} \{0,1\}^t] &= \frac{\lfloor T \cdot f(Y) \rfloor - \lceil T \cdot f(Y-1) \rceil}{T} = f(Y) - f(Y-1) + e \\ &= \Pr[y = Y : y \leftarrow D] + e\end{aligned}$$

where $|e| \le 2/T$. Thus, the statistical distance between $\{\mathsf{Project}(x) : x \xleftarrow{\mathrm{R}} \{0,1\}^t\}$ and $D$ is at most $2(2B+1)/T = \mathrm{negl}(\lambda)$. Finally, on input $y \in [-B, B]$, $\mathsf{SampleR}(y)$ outputs a uniform $x \xleftarrow{\mathrm{R}} \{0,1\}^t$ conditioned on $\mathsf{Project}(x) = y$. □

**Remark 5.8** (Extending to Product Distributions)**.** We can extend $(\mathsf{Project}, \mathsf{SampleR})$ to sample from a product distribution $D^n$ in the natural way. The projection algorithm takes as input a vector of bit-strings $\mathbf{x} \in (\{0,1\}^t)^n$ and applies the projection operator component-wise. The reverse sampling algorithm is defined analogously. Correctness and reverse-sampleability then follow via a standard hybrid argument.

**Remark 5.9** (Implementing the Random Oracle in Corollary 5.6)**.** We can now implement the random oracle $\mathsf{H}\colon \mathcal{GID} \to \mathbb{Z}_q^{m(2L-1)}$ in Corollary 5.6 (whose outputs are distributed according to $D_{\mathbb{Z},\chi}^{m(2L-1)}$) with a random oracle $\mathsf{H}'\colon \mathcal{GID} \to \{0,1\}^{\lambda m(2L-1)}$ whose outputs are uniform as follows:

- Let $\tilde{D}_{\mathbb{Z},\chi}$ be the discrete Gaussian distribution $D_{\mathbb{Z},\chi}$ truncated to the interval $[-\sqrt{\lambda}\chi, \sqrt{\lambda}\chi]$. Namely, to sample $\tilde{x} \leftarrow \tilde{D}_{\mathbb{Z},\chi}$, we first sample $x \leftarrow D_{\mathbb{Z},\chi}$ and output $x$ if $x \in [-\sqrt{\lambda}\chi, \sqrt{\lambda}\chi]$ and output 0 otherwise. By Fact 3.8, $\tilde{D}_{\mathbb{Z},\chi}$ is statistically indistinguishable from $D_{\mathbb{Z},\chi}$. In addition, $\tilde{D}_{\mathbb{Z},\chi}$ is $B$-bounded for $B = \sqrt{\lambda}\chi$.
- Let $(\mathsf{Project}, \mathsf{SampleR})$ be the inversion sampling algorithm from Lemma 5.7 and Remark 5.8 for the product distribution $\tilde{D}_{\mathbb{Z},\chi}^{m(2L-1)}$. We now define

  $$\mathsf{H}(\mathsf{gid}) \coloneqq \mathsf{Project}(\mathsf{H}'(\mathsf{gid})).$$

  Since $\chi = \chi(\lambda)$ is polynomially-bounded, the cumulative distribution function of $\tilde{D}_{\mathbb{Z},\chi}$ is efficiently-computable. Then, by Lemma 5.7 and Remark 5.8, for all polynomial-size collections of distinct inputs $\mathsf{gid}_1, \ldots, \mathsf{gid}_\ell \in \mathcal{GID}$, the joint distributions of

  $$\left\{\mathsf{H}(\mathsf{gid}_i)\right\}_{i\in[\ell]} \quad \text{and} \quad \left\{\mathbf{r}_i \leftarrow D_{\mathbb{Z},\chi}^{m(2L-1)}\right\}_{i\in[\ell]}$$

  are statistically indistinguishable.
- Finally, the proof of Theorem 5.3 critically relies on the ability to *program* the outputs of the random oracle in the reduction. Here, we rely on the $\mathsf{SampleR}$ algorithm. Namely, to program $\mathsf{H}(\mathsf{gid})$ to a vector $\mathbf{r}_{\mathsf{gid}} \leftarrow D_{\mathbb{Z},\chi}^{m(2L-1)}$, the reduction algorithm would sample $x_{\mathsf{gid}} \leftarrow \mathsf{SampleR}(\mathbf{r}_{\mathsf{gid}})$ and program $\mathsf{H}'(\mathsf{gid})$ to $x_{\mathsf{gid}}$. This induces the correct distribution by Lemma 5.7 and Remark 5.8.

# 6 Multi-Authority ABE without Random Oracles

We now give our construction of a multi-authority ABE scheme without random oracles. Specifically, we instantiate the hash function from Construction 5.1 with a subset-product construction (i.e., the lattice-based PRF from Theorem 6.1) and then prove security under the evasive LWE assumption (Assumption 3.16) and lattice-based PRFs [BPR12, BLMR13].

**Lattice-based PRFs.** Our analysis will rely on an *unrounded* lattice-based PRF. We state the theorem and provide a proof sketch below, and refer readers to [BPR12, Theorem 5.2] for a more formal exposition. Our presentation here is adapted from the work of Chen et al. [CVW18, Lemma 7.4] who use a similar theorem for analyzing the security of their private constrained PRF construction.

**Theorem 6.1** (Lattice-Based PRFs [BPR12, BLMR13])**.** *Let $\lambda$ be a security parameter and let $n = n(\lambda)$, $q = q(\lambda)$, $\chi = \chi(\lambda)$, $k = k(\lambda)$ be integers. Let $\chi_{\mathsf{smudge}} = \chi_{\mathsf{smudge}}(\lambda)$ be a noise parameter that will used for noise smudging. Let $\boldsymbol{\eta} \in \mathbb{Z}_q^k$ be the first elementary basis vector (i.e., $\eta_1 = 1$ and $\eta_i = 0$ for all $i \neq 1$). For a bit $b \in \{0,1\}$, an input length $\tau = \tau(\lambda)$, and an adversary $\mathcal{A}$, define the following pseudorandomness game between a challenger and $\mathcal{A}$:*

1. *The challenger begins by sampling $(\mathbf{D}_0, \mathbf{D}_1) \xleftarrow{R} D_{\mathbb{Z},\chi}^{k\times k}$ and a secret key $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^k$. It gives $\mathbf{D}_0$ and $\mathbf{D}_1$ to $\mathcal{A}$.*
2. *Algorithm $\mathcal{A}$ can now adaptively submit queries $x \in \{0,1\}^\tau$ to the challenger. If $b = 0$, the challenger samples $e_x \xleftarrow{R} D_{\mathbb{Z},\chi_{\mathsf{smudge}}}$ and outputs*

   $$f_{\mathbf{D}_0,\mathbf{D}_1,\mathbf{s}}(x) \coloneqq \mathbf{s}^\intercal \left(\prod_{i\in[\tau]} \mathbf{D}_{x_i}\right)\boldsymbol{\eta} + e_x \in \mathbb{Z}_q. \tag{6.1}$$

   *Otherwise, if $b = 1$, the challenger replies with $y \xleftarrow{R} \mathbb{Z}_q$.*
3. *After $\mathcal{A}$ is done making queries, it outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.*

*An adversary $\mathcal{A}$ is admissible if all of the queries it submits are distinct. Then, for all polynomials $\tau = \tau(\lambda)$, $q = q(\lambda)$, parameters $k \geq 6n \log q$, $\chi = \Omega(\sqrt{n \log q})$, $\chi_{\mathsf{smudge}} > \lambda^{\tau+\omega(1)} \cdot (k\chi)^{\tau}$, and assuming the $\mathsf{LWE}_{n,m,q,\chi}$ assumption for some $m = \mathsf{poly}(k, \tau, Q)$, for all efficient and admissible adversaries $\mathcal{A}$ making up to $Q$ queries, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $|\Pr[b' = 1 : b = 0] - \Pr[b' = 1 : b = 1]| = \mathsf{negl}(\lambda)$.*

*Proof (Sketch).* Our proof follows the same structure as [CVW18, Lemma 7.4] and [BPR12, Lemma 5.5]. Specifically, we start by defining the "expanded" evaluation function $\tilde{f}_{\mathbf{D}_0,\mathbf{D}_1,\mathbf{s}}(x)$:

$$\tilde{f}_{\mathbf{D}_0,\mathbf{D}_1,\mathbf{s}}(x) := \left(\left(\cdots\left((\mathbf{s}^{\mathsf{T}}\mathbf{D}_{x_1} + \mathbf{e}_1^{\mathsf{T}})\mathbf{D}_{x_2} + \mathbf{e}_2^{\mathsf{T}}\right)\mathbf{D}_{x_3} + \cdots + \mathbf{e}_{\tau-1}^{\mathsf{T}}\right)\mathbf{D}_{x_\tau} + \mathbf{e}_\tau^{\mathsf{T}}\right)\boldsymbol{\eta} + e_x, \tag{6.2}$$

where $\mathbf{e}_1, \ldots, \mathbf{e}_\tau \leftarrow D_{Z,\chi}^k$ and $e_x \leftarrow D_{\mathbb{Z},\chi_{\mathsf{smudge}}}$. Then,

$$\tilde{f}_{\mathbf{D}_0,\mathbf{D}_1,\mathbf{s}}(x) = f_{\mathbf{D}_0,\mathbf{D}_1,\mathbf{s}}(x) + \underbrace{\sum_{i\in[\tau]} \mathbf{e}_i^{\mathsf{T}}\left(\prod_{j=i+1}^{\tau} \mathbf{D}_{x_j}\right)\boldsymbol{\eta}}_{e_x^*}.$$

Since $\mathbf{D}_0, \mathbf{D}_1 \leftarrow D_{\mathbb{Z},\chi}^{k\times k}$, we appeal to Fact 3.8 to conclude that with overwhelming probability, $\|\mathbf{D}_0\|, \|\mathbf{D}_1\| \leq \sqrt{\lambda}\chi$. Similarly, with overwhelming probability, $\|\mathbf{e}_i\| \leq \sqrt{\lambda}\chi$. This means that $|e_x^*| \leq \tau \cdot (k\sqrt{\lambda}\chi)^{\tau}$. When $\chi_{\mathsf{smudge}} > \lambda^{\tau+\omega(1)}(k\chi)^{\tau}$, we can appeal to Lemma 3.15 to conclude that the distribution of $e_x$ and $e_x + e_x^*$ are statistically close. Correspondingly, the distributions of $f_{\mathbf{D}_0,\mathbf{D}_1,\mathbf{s}}(x)$ and $\tilde{f}_{\mathbf{D}_0,\mathbf{D}_1,\mathbf{s}}(x)$ are statistically close.

Finally, we use a standard hybrid argument (combining [BPR12, Theorem 5.2] and [BLMR13, Corollary 4.6]) to argue that the distribution of $\tilde{f}_{\mathbf{D}_0,\mathbf{D}_1,\mathbf{s}}(x)$ is computationally indistinguishable from the uniform distribution over $\mathbb{Z}_q$ under the $\mathsf{LWE}_{n,m,q,\chi}$ assumption for some $m = \mathsf{poly}(k, \tau, Q)$. This step relies on the distribution of $(\mathbf{D}, \mathbf{s}^{\mathsf{T}}\mathbf{D} + \mathbf{e}^{\mathsf{T}})$ being computationally indistinguishable from $(\mathbf{D}, \mathbf{u}^{\mathsf{T}})$ when $\mathbf{D} \leftarrow D_{\mathbb{Z},\chi}^{k\times k}$, $\mathbf{s} \xleftarrow{\mathsf{R}} \mathbb{Z}_q^k$, $\mathbf{e} \leftarrow D_{\mathbb{Z},\chi}^k$, and $\mathbf{u} \xleftarrow{\mathsf{R}} \mathbb{Z}_q^k$. This is implied by the $\mathsf{LWE}_{n,m,q,\chi}$ assumption when $k \geq 6n \log q$, $\chi = \Omega(\sqrt{n \log q})$, and $m = \mathsf{poly}(k)$ [BLMR13, Corollary 4.6]. □

**MA-ABE for subset policies without random oracles.** We now give the full construction of our MA-ABE scheme without random oracles. As described in Section 2, our construction essentially instantiates the random oracle in Construction 5.1 with a subset-product of low-norm matrices (which can be used as the basis for constructing a PRF according to Theorem 6.1). Arguing security in turn relies on the evasive LWE assumption (Assumption 3.16). Using the evasive LWE assumption to argue security has the extra benefit of allowing support for policies of arbitrary (polynomial) length (recall that Construction 5.1 as well as the previous lattice-based construction of Datta et al. [DKW21a] required imposing an *a priori* bound on the policy length, and the size of the ciphertext in turn grew with the maximum length).

**Construction 6.2** (Multi-Authority ABE without Random Oracles)**.** Let $\lambda$ be a security parameter, and $n = n(\lambda)$, $m = m(\lambda)$, $q = q(\lambda)$, and $\chi = \chi(\lambda)$ be lattice parameters. Let $\chi_{\mathsf{PRF}} = \chi_{\mathsf{PRF}}(\lambda)$ be a Gaussian width parameter used to define the hash function. Let $\tau = \tau(\lambda)$ be the bit-length of identities and let $\mathcal{GID} = \{0,1\}^\tau$ be the set of user identifiers. Let $\mathcal{AU} = \{0,1\}^\lambda$ be the set of authorities. We construct an MA-ABE scheme for subset policies (Definition 3.5) with message space $\mathcal{M} = \{0,1\}$ as follows:

- $\mathsf{GlobalSetup}(1^\lambda)$: Sample $\mathbf{D}_0, \mathbf{D}_1 \leftarrow D_{\mathbb{Z},\chi_{\mathsf{PRF}}}^{m\times m}$. Define the hash function $\mathsf{H}\colon \{0,1\}^\tau \to \mathbb{Z}_q^m$ by the function $\mathsf{H}(x) := \left(\prod_{i\in[\tau]} \mathbf{D}_{x_i}\right)\boldsymbol{\eta}$ where $\boldsymbol{\eta} \in \mathbb{Z}_q^m$ is the first canonical basis vector (i.e., $\eta_1 = 1$ and $\eta_i = 0$ for all $i \neq 1$). Output
$$\mathsf{gp} = (\lambda, n, m, q, \chi, \chi_{\mathsf{PRF}}, \tau, \mathbf{D}_0, \mathbf{D}_1).$$
For ease of exposition, whenever we write $\mathsf{H}(\cdot)$ in the following, we refer to the hash function defined by the matrices $\mathbf{D}_0, \mathbf{D}_1$ in the global parameters.

- $\mathsf{AuthSetup}(\mathsf{gp}, \mathsf{aid})$: On input the global parameters $\mathsf{gp} = (\lambda, n, m, q, \chi, \chi_{\mathsf{PRF}}, \tau, \mathbf{D}_0, \mathbf{D}_1)$ and an authority identifier $\mathsf{aid} \in \mathcal{AU}$, sample $(\mathbf{A}_{\mathsf{aid}}, \mathsf{td}_{\mathsf{aid}}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$, $\mathbf{p}_{\mathsf{aid}} \xleftarrow{\mathsf{R}} \mathbb{Z}_q^n$, and $\mathbf{B}_{\mathsf{aid}} \xleftarrow{\mathsf{R}} \mathbb{Z}_q^{n\times m}$. Output the authority public key $\mathsf{pk}_{\mathsf{aid}} \leftarrow (\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$ and the authority secret key $\mathsf{msk}_{\mathsf{aid}} = \mathsf{td}_{\mathsf{aid}}$.

- KeyGen(gp, msk, pk, gid): On input the global parameters $\mathsf{gp} = (\lambda, n, m, q, \chi, \chi_{\mathsf{PRF}}, \tau, \mathbf{D}_0, \mathbf{D}_1)$, the master secret key $\mathsf{msk} = \mathsf{td}$, the public key $\mathsf{pk} = (\mathbf{A}, \mathbf{B}, \mathbf{p})$, the user identifier $\mathsf{gid} \in \{0,1\}^\tau$, the key-generation algorithm computes $\mathbf{r} \leftarrow \mathsf{H}(\mathsf{gid}) \in \mathbb{Z}_q^m$ and uses td to sample $\mathbf{u} \leftarrow \mathbf{A}_\chi^{-1}(\mathbf{p} + \mathbf{B}\mathbf{r})$. It outputs $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} = \mathbf{u}$.
- Encrypt(gp, $\{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mu$): On input the global parameters $\mathsf{gp} = (\lambda, n, m, q, \chi, \chi_{\mathsf{PRF}}, \tau, \mathbf{D}_0, \mathbf{D}_1)$, a set of of public keys $\mathsf{pk}_{\mathsf{aid}} = (\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$ associated with a set of authorities $A$, and the message $\mu \in \{0,1\}$, the encryption algorithm samples $\mathbf{s}_{\mathsf{aid}} \xleftarrow{R} \mathbb{Z}_q^n$, $\mathbf{e}_{1,\mathsf{aid}} \leftarrow D_{\mathbb{Z},\chi}^m$, $\mathbf{e}_2 \leftarrow D_{\mathbb{Z},\chi}^m$, and $e_3 \leftarrow D_{\mathbb{Z},\chi}$ for each $\mathsf{aid} \in A$. It outputs the ciphertext

$$\mathsf{ct} = \left( \left\{ \mathbf{s}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\top \right\}_{\mathsf{aid}\in A} , \sum_{\mathsf{aid}\in A} \mathbf{s}_{\mathsf{aid}}^\top \mathbf{B}_{\mathsf{aid}} + \mathbf{e}_2^\top , \sum_{\mathsf{aid}\in A} \mathbf{s}_{\mathsf{aid}}^\top \mathbf{p}_{\mathsf{aid}} + e_3 + \mu \cdot \lfloor q/2 \rceil \right).$$

- Decrypt(gp, $\{\mathsf{sk}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid}\in A}, \mathsf{ct}, \mathsf{gid}$): On input the global parameters $\mathsf{gp} = (\lambda, n, m, q, \chi, \chi_{\mathsf{PRF}}, \tau, \mathbf{D}_0, \mathbf{D}_1)$, a set of secret keys $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} = \mathbf{u}_{\mathsf{aid},\mathsf{gid}}$ associated with authorities $\mathsf{aid} \in A$ and user identifier gid, a ciphertext $\mathsf{ct} = \left(\{\mathbf{c}_{1,\mathsf{aid}}^\top\}_{\mathsf{aid}\in A}, \mathbf{c}_2, c_3\right)$, the decryption algorithm computes $\mathbf{r} \leftarrow \mathsf{H}(\mathsf{gid})$ and outputs

$$\left\lfloor \frac{2}{q} \cdot \left( c_3 + \mathbf{c}_2^\top \mathbf{r} - \sum_{\mathsf{aid}\in A} \mathbf{c}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid},\mathsf{gid}} \bmod q \right) \right\rceil .$$

**Theorem 6.3** (Correctness). *Let $L = L(\lambda)$ be a bound on the number of attributes associated with a ciphertext. Suppose the conditions of Theorem 3.10 and Lemma 3.13 hold (i.e., $m \geq m_0(n,q) = O(n \log q)$ and $\chi > \chi_0(n,q) = \sqrt{n \log q} \cdot \omega(\sqrt{\log n})$). Then, there exists $q_0 = O\left(Lm\lambda\chi^2 + (\sqrt{\lambda} m \chi_{\mathsf{PRF}})^{\tau+1}\chi\right)$ such that for all $m > m_0$, $q > q_0$, and $\chi > \chi_0$, Construction 6.2 is correct.*

*Proof.* This follows by a similar argument as the proof of Theorem 5.2. Specifically, take any message $\mu \in \{0,1\}$, an identifier $\mathsf{gid} \in \{0,1\}^\tau$, and set of authorities $A \subseteq \mathcal{AU}$. Sample the global parameters $\mathsf{gp} \leftarrow \mathsf{GlobalSetup}(1^\lambda)$, the authority keys $(\mathsf{pk}_{\mathsf{aid}}, \mathsf{msk}_{\mathsf{aid}}) \leftarrow \mathsf{AuthSetup}(\mathsf{gp}, \mathsf{aid})$, the secret keys $\mathsf{sk}_{\mathsf{gid},\mathsf{aid}} \leftarrow \mathsf{KeyGen}(\mathsf{gp}, \mathsf{msk}_{\mathsf{aid}}, \mathsf{gid})$, and the ciphertext $\mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{gp}, \{\mathsf{pk}_{\mathsf{aid}}\}_{\mathsf{aid}\in A}, \mu)$. We now expand the various components appearing in the computation of $\mathsf{Decrypt}(\mathsf{gp}, \{\mathsf{sk}_{\mathsf{aid},\mathsf{gid}}\}_{\mathsf{aid}\in A}, \mathsf{ct}, \mathsf{gid})$:

- First, $\mathsf{gp} = (\lambda, n, m, q, \chi, \chi_{\mathsf{PRF}}, \tau, \mathbf{D}_0, \mathbf{D}_1)$, where $\mathbf{D}_0, \mathbf{D}_1$ are sampled from $D_{\mathbb{Z},\chi_{\mathsf{PRF}}}^{m\times m}$. By Fact 3.8, with overwhelming probability, $\|\mathbf{D}_0\|, \|\mathbf{D}_1\| \leq \sqrt{\lambda}\chi_{\mathsf{PRF}}$.
- The ciphertext ct is given by $\mathsf{ct} = \left(\left\{\mathbf{c}_{1,\mathsf{aid}}^\top\right\}_{\mathsf{aid}\in A}, \mathbf{c}_2^\top, c_3\right)$ where

$$\mathbf{c}_{1,\mathsf{aid}}^\top = \mathbf{s}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\top \quad , \quad \mathbf{c}_2^\top = \sum_{\mathsf{aid}\in A} \mathbf{s}_{\mathsf{aid}}^\top \mathbf{B}_{\mathsf{aid}} + \mathbf{e}_2^\top \quad , \quad c_3 = \sum_{\mathsf{aid}\in A} \mathbf{s}_{\mathsf{aid}}^\top \mathbf{p}_{\mathsf{aid}} + e_3 + \mu \cdot \lfloor q/2 \rceil ,$$

  and $(\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$ is the public key associated with authority aid.
- Each secret key $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}} = \mathbf{u}_{\mathsf{aid},\mathsf{gid}} \leftarrow (\mathbf{A}_{\mathsf{aid}})_\chi^{-1}(\mathbf{p}_{\mathsf{aid}} + \mathbf{B}_{\mathsf{aid}}\mathbf{r})$. Since $\mathbf{p}_{\mathsf{aid}}$ is uniform over $\mathbb{Z}_q^n$ and independent of $\mathbf{B}_{\mathsf{aid}}\mathbf{r}$, the marginal distribution of $\mathbf{u}_{\mathsf{aid},\mathsf{gid}}$ is statistically close to $D_{\mathbb{Z},\chi}^m$ by Lemma 3.13. Then, by Fact 3.8, with overwhelming probability, $\|\mathbf{u}_{\mathsf{aid},\mathsf{gid}}\| \leq \sqrt{\lambda}\chi$.
- Since $\mathbf{r} = \left(\prod_{i\in[\tau]} \mathbf{D}_{\mathsf{gid}_i}\right)\boldsymbol{\eta}$, by Fact 3.8, $\|\mathbf{r}\| \leq (\sqrt{\lambda} m \chi_{\mathsf{PRF}})^\tau$ with overwhelming probability.
- By construction,

$$\mathbf{c}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid}} = \mathbf{s}_{\mathsf{aid}}^\top \mathbf{A}_{\mathsf{aid}} \mathbf{u}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid}} = \mathbf{s}_{\mathsf{aid}}^\top \mathbf{p}_{\mathsf{aid}} + \mathbf{s}_{\mathsf{aid}}^\top \mathbf{B}_{\mathsf{aid}} \mathbf{r} + \mathbf{e}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid}}.$$

  The main decryption relation then becomes

$$c_3 + \mathbf{c}_2^\top \mathbf{r} - \sum_{\mathsf{aid}\in A} \mathbf{c}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid}} = \mu \cdot \lfloor q/2 \rceil + e_3 + \mathbf{e}_2^\top \mathbf{r} - \sum_{\mathsf{aid}\in A} \mathbf{e}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid}}.$$

  Decryption succeeds if the total error $\tilde{e} = e_3 + \mathbf{e}_2^\top \mathbf{r} - \sum_{\mathsf{aid}\in A} \mathbf{e}_{1,\mathsf{aid}}^\top \mathbf{u}_{\mathsf{aid}}$ satisfies $|\tilde{e}| < (q-1)/4$.

- To bound the error $\tilde{e}$, we bound each of its components. Since the components of $\mathbf{e}_1$, $\mathbf{e}_2$, and $e_3$ are all independent samples from a discrete Gaussian distribution with width $\chi$, we can appeal to Fact 3.8 to conclude that with overwhelming probability, they are bounded by $\sqrt{\lambda}\chi$. Thus, with overwhelming probability,

$$\begin{aligned} |e_3| &\le \sqrt{\lambda}\chi \\ \|\mathbf{e}_2^\intercal \mathbf{r}\| &\le \left(\sqrt{\lambda} m\right)^{\tau+1} \chi_{\mathsf{PRF}}^{\tau} \chi \\ \|\mathbf{e}_{1,\mathsf{aid}}^\intercal \mathbf{u}_{\mathsf{aid}}\| &\le m\lambda\chi^2 \end{aligned}$$

  Combining these relations, we obtain the desired bound

$$|\tilde{e}| < \sqrt{\lambda}\chi + \left(\sqrt{\lambda} m\right)^{\tau+1} \chi_{\mathsf{PRF}}^{\tau} \chi + |A| \cdot m\lambda\chi^2 = O(Lm\lambda\chi^2 + (\sqrt{\lambda} m \chi_{\mathsf{PRF}})^{\tau+1}\chi).$$

  Note that there is a significant amount of slack in the above bound. □

**Theorem 6.4** (Static Security)**.** *There exists a polynomial $m_0(n, q) = O(n \log q)$ such that under the following conditions and assumptions, Construction 6.2 is statically secure:*

- *The number of samples $m$ satisfies $m \ge m_0$.*
- *Let $\chi_{\mathsf{smudge}} = \chi_{\mathsf{smudge}}(\lambda)$ be a smudging parameter where $\chi_{\mathsf{smudge}} > \lambda^{\tau+\omega(1)}(m\chi_{\mathsf{PRF}})^{\tau+1}$.*
- *The noise parameter $\chi$ satisfies $\chi > \lambda^{\omega(1)}\ell\chi_{\mathsf{smudge}}$.*
- *The $\mathsf{LWE}_{n,m',q,\chi_{\mathsf{PRF}}}$ assumption holds where $m' = \mathsf{poly}(m, \tau, Q)$ and $Q$ is a bound on the number of secret-key queries the adversary makes.*
- *The evasive LWE assumption with parameters $n, m, q, \chi, s = \chi$ holds (in particular, the preimages $\mathbf{K} \leftarrow \mathbf{A}^{-1}(\mathbf{P})$ are distributed according to a discrete Gaussian with parameter $s = \chi$).*

*Proof.* We start by defining a sequence of hybrid experiments:

- $\mathsf{Hyb}_0^{(\mathsf{main})}$: This is the static security experiment where the challenger encrypts message $\mu_0$. At the end of the game, the adversary outputs a bit $b \in \{0, 1\}$ which is the output of the experiment.
- $\mathsf{Hyb}_1^{(\mathsf{main})}$: Same as $\mathsf{Hyb}_0^{(\mathsf{main})}$, except the challenger uses the following modified procedure to construct the challenge ciphertext:
  - **Challenge ciphertext:** The challenger samples $\mathbf{s}_{\mathsf{aid}} \xleftarrow{R} \mathbb{Z}_q^n$ and $\mathbf{e}_{1,\mathsf{aid}} \leftarrow D_{\mathbb{Z},\chi}^m$ for each corrupted authority $\mathsf{aid} \in A^* \cap C$ (same as in $\mathsf{Hyb}_1$) and sets $\mathbf{c}_{1,\mathsf{aid}}^\intercal \leftarrow \mathbf{s}_{\mathsf{aid}}^\intercal \mathbf{A}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\intercal$. For the honest authorities $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$, it samples $\mathbf{c}_{1,\mathsf{aid}_i^*} \xleftarrow{R} \mathbb{Z}_q^m$. Next, it samples $\mathbf{c}_2 \xleftarrow{R} \mathbb{Z}_q^m$ and $c_3 \xleftarrow{R} \mathbb{Z}_q$. Finally, it outputs the challenge ciphertext

$$\mathsf{ct} = \left(\left\{\mathbf{c}_{1,\mathsf{aid}}^\intercal\right\}_{\mathsf{aid}\in A^*}, \mathbf{c}_2, c_3\right).$$

    In particular, the challenge ciphertext is independent of the message.
- $\mathsf{Hyb}_2^{(\mathsf{main})}$: This is the static security experiment where the challenger encrypts message $\mu_1$.

For an adversary $\mathcal{A}$, we write $\mathsf{Hyb}_i^{(\mathsf{main})}(\mathcal{A})$ to denote the output distribution of $\mathsf{Hyb}_i^{(\mathsf{main})}$ with adversary $\mathcal{A}$. Next, we note that for this setting of parameters, the conditions in Theorem 3.10 hold. Thus, in the following analysis, we implicitly assume that using a trapdoor output by $(\mathbf{A}, \mathsf{td}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$, it is possible to sample from a distribution that is statistically close to $\mathbf{A}_\chi^{-1}(\mathbf{t})$ for any target $\mathbf{t}$. We now show that each pair of adjacent distributions is computationally indistinguishable.

**Lemma 6.5.** *Under the same conditions as Theorem 6.4, for every efficient adversary $\mathcal{A}$, $\mathsf{Hyb}_0^{(\mathsf{main})}(\mathcal{A}) \overset{c}{\approx} \mathsf{Hyb}_1^{(\mathsf{main})}(\mathcal{A})$.*

*Proof.* Suppose there exists an efficient adversary $\mathcal{A}$ that distinguishes $\mathsf{Hyb}_0^{(\mathsf{main})}$ from $\mathsf{Hyb}_1^{(\mathsf{main})}$ with advantage $\varepsilon > 0$. First, we use $\mathcal{A}$ to construct a sampling algorithm $\mathsf{Samp}_{\mathcal{A}}$ (that depends on $\mathcal{A}$) for the evasive LWE assumption:

**Algorithm** $\mathsf{Samp}_{\mathcal{A}}(1^\lambda)$

On input the security parameter $\lambda$, the sampling algorithm proceeds as follows:

1. Let $\kappa = \kappa(\lambda)$ be a bound on the number of bits of randomness algorithm $\mathcal{A}$ uses. Sample $r \xleftarrow{R} \{0,1\}^\kappa$ and run algorithm $\mathcal{A}(1^\lambda; r)$.
2. Algorithm $\mathcal{A}$ outputs a set of corrupted authorities $C \subset \mathcal{AU}$ along with their public keys, a list of non-corrupted authorities $\mathcal{N} \subseteq \mathcal{AU}$, a set of secret key queries $Q$, a pair of challenge messages $\mu_0, \mu_1 \in \{0,1\}$, and a challenge identity set $A^* \subseteq C \cup \mathcal{N}$.
3. Let $\ell = |A^* \cap \mathcal{N}|$ and write $A^* \cap \mathcal{N} = \{\mathsf{aid}_1^*, \ldots, \mathsf{aid}_\ell^*\}$.
4. Sample $\mathbf{B} \xleftarrow{R} \mathbb{Z}_q^{n\ell \times (m+1)}$ and parse
$$\mathbf{B} = \left[\begin{array}{c|c} \mathbf{B}_{\mathsf{aid}_1^*} & \mathbf{p}_{\mathsf{aid}_1^*} \\ \vdots & \vdots \\ \mathbf{B}_{\mathsf{aid}_\ell^*} & \mathbf{p}_{\mathsf{aid}_\ell^*} \end{array}\right] \in \mathbb{Z}_q^{n\ell \times (m+1)},$$
where each $\mathbf{B}_{\mathsf{aid}_i^*} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{p}_{\mathsf{aid}_i^*} \in \mathbb{Z}_q^n$.
5. Sample matrices $\mathbf{D}_0, \mathbf{D}_1 \leftarrow D_{\mathbb{Z}, \chi_{\mathsf{PRF}}}^{m \times m}$.
6. Let $(\mathsf{gid}_1, A_1), \ldots, (\mathsf{gid}_Q, A_Q)$ be the secret-key queries algorithm $\mathcal{A}$ makes. For each $i \in [\ell]$, let $N_i \in [Q]$ be the number of indices $j \in [Q]$ where $\mathsf{aid}_i^* \in A_j$ (i.e., $N_i$ is the number of secret-key queries that involve authority $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$). The sampling algorithm constructs matrices $\mathbf{P}_i \in \mathbb{Z}_q^{n \times N_i}$ for $i \in [\ell]$ as follows:
   - Suppose authority $\mathsf{aid}_i^*$ is contained in the sets $A_{j_1}, \ldots, A_{j_{N_i}}$ for indices $j_1, \ldots, j_{N_i} \in [Q]$. These are the sets associated with the identifiers $\mathsf{gid}_{j_1}, \ldots, \mathsf{gid}_{j_{N_i}}$.
   - Define the matrix $\mathbf{P}_i$ as follows:
$$\mathbf{P}_i = \left[\mathbf{p}_{\mathsf{aid}_i^*} + \mathbf{B}_{\mathsf{aid}_i^*} \cdot \mathsf{H}(\mathsf{gid}_{j_1}) \mid \cdots \mid \mathbf{p}_{\mathsf{aid}_i^*} + \mathbf{B}_{\mathsf{aid}_i^*} \cdot \mathsf{H}(\mathsf{gid}_{j_{N_i}})\right].$$
7. Output $\mathbf{B}, \mathbf{P}_1, \ldots, \mathbf{P}_\ell$, and $\mathsf{aux} = (r, \mathbf{D}_0, \mathbf{D}_1)$. In this case, observe that aux can also just be the set of random coins used by the sampling algorithm (Remark 3.17).

To invoke the evasive LWE assumption, we now show that for all efficient distinguishers $\mathcal{D}$, $\mathsf{Adv}_{\mathcal{D}}^{(\mathrm{PRE})}(\lambda)$ is negligible.

**Claim 6.6.** *Suppose the lattice parameters satisfy the following conditions:*

- *The number of samples $m$ satisfy $m > 6n \log q$.*
- *Let $\chi_{\mathsf{smudge}} = \chi_{\mathsf{smudge}}(\lambda)$ be a smudging parameter where $\chi_{\mathsf{smudge}} > \lambda^{\tau+\omega(1)} \cdot (m\chi_{\mathsf{PRF}})^\tau$.*
- *The noise parameter $\chi$ satisfies $\chi > \lambda^{\omega(1)} \cdot \chi_{\mathsf{smudge}}$ and $\chi > \lambda^{\omega(1)} \cdot (\sqrt{\lambda} m \chi_{\mathsf{PRF}})^{\tau+1} \ell$.*

*Suppose the $\mathsf{LWE}_{n,m',q,\chi_{\mathsf{PRF}}}$ assumption holds where $m' = \mathsf{poly}(m, \tau, Q)$ and $Q$ is a bound on the number of secret-key queries adversary $\mathcal{A}$ makes. Then, for all efficient distinguishers $\mathcal{D}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, we have that $\mathsf{Adv}_{\mathcal{D}}^{(\mathrm{PRE})}(\lambda) = \mathsf{negl}(\lambda)$, where $\mathsf{Adv}_{\mathcal{D}}^{(\mathrm{PRE})}$ is the advantage of distinguisher $\mathcal{D}$ in the evasive LWE assumption (Assumption 3.16).*

*Proof.* We start by defining a sequence of hybrid experiments:

- $\mathsf{Hyb}_0$: This is the pseudorandom distribution in the definition of $\mathsf{Adv}_{\mathcal{D}}^{(\mathrm{PRE})}$. Without loss of generality, assume that $\mathcal{A}$ makes exactly $Q$ secret-key queries (an adversary that makes fewer than $Q$ queries can be padded to make exactly $Q$ queries). In this experiment, the challenger constructs the components as follows:

– Sample $(\mathbf{B}, \mathbf{P}_1, \ldots, \mathbf{P}_\ell, \mathsf{aux}) \leftarrow \mathsf{Samp}_{\mathcal{A}}(1^\lambda)$. By construction, $\mathbf{B} \xleftarrow{R} \mathbb{Z}_q^{n\ell \times (m+1)}$ where

$$\mathbf{B} = \left[\begin{array}{c|c} \mathbf{B}_1 & \mathbf{p}_1 \\ \vdots & \vdots \\ \mathbf{B}_\ell & \mathbf{p}_\ell \end{array}\right],$$

$\mathbf{B}_i \in \mathbb{Z}_q^{n \times m}$, $\mathbf{p}_i \in \mathbb{Z}_q^n$. In addition, for each $i \in [\ell]$, $\mathbf{P}_i \in \mathbb{Z}_q^{n \times N_i}$, where $N_i \in [Q]$. Finally, sample $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \xleftarrow{R} \mathbb{Z}_q^{n \times m}$.

– Let $(A_1, \mathsf{gid}_1), \ldots, (A_Q, \mathsf{gid}_Q)$ be the secret-key queries made by $\mathcal{A}$ in $\mathsf{Samp}_{\mathcal{A}}$. For each $i \in [\ell]$, let $N_i \in [Q]$ be the number of indices $j \in [Q]$ where $\mathsf{aid}_i^* \in A_j$. Suppose authority $\mathsf{aid}_i^*$ is contained in the sets $A_{j_1}, \ldots, A_{j_{N_i}}$ for (sorted) indices $j_1, \ldots, j_{N_i} \in [Q]$. Define the mapping $\rho_i \colon [N_i] \to [Q]$ that maps $\ell \in [N_i] \mapsto j_\ell \in [Q]$. In particular, for each $i \in [\ell]$, we can now write

$$\mathbf{P}_i = \left[\mathbf{p}_i + \mathbf{B}_i \cdot \mathsf{H}\big(\mathsf{gid}_{\rho_i(1)}\big) \mid \cdots \mid \mathbf{p}_i + \mathbf{B}_i \cdot \mathsf{H}\big(\mathsf{gid}_{\rho_i(N_i)}\big)\right].$$

– Sample $\mathbf{s}_1, \ldots, \mathbf{s}_\ell \xleftarrow{R} \mathbb{Z}_q^n$ and let $\mathbf{s}^\top = [\mathbf{s}_1^\top \mid \cdots \mid \mathbf{s}_\ell^\top] \in \mathbb{Z}_q^{n\ell}$. Sample $\mathbf{e}_{1,i} \leftarrow D_{\mathbb{Z},\chi}^m$, $\mathbf{e}_{3,i} \leftarrow D_{\mathbb{Z},\chi}^{N_i}$ for each $i \in [\ell]$. Then sample $\mathbf{e}_2 \leftarrow D_{\mathbb{Z},\chi}^{m+1}$.

– Compute $\mathbf{u}_{1,i}^\top \leftarrow \mathbf{s}_i^\top \mathbf{A}_i + \mathbf{e}_{1,i}^\top \in \mathbb{Z}_q^m$, $\mathbf{u}_2^\top \leftarrow \mathbf{s}^\top \mathbf{B} + \mathbf{e}_2^\top \in \mathbb{Z}_q^{m+1}$, and $\mathbf{u}_{3,i}^\top \leftarrow \mathbf{s}_i^\top \mathbf{P}_i + \mathbf{e}_{3,i}^\top \in \mathbb{Z}_q^{N_i}$ for each $i \in [\ell]$. Equivalently, if we define $\mathbf{t}_{i,j} = \mathbf{p}_i + \mathbf{B}_i \cdot \mathsf{H}\big(\mathsf{gid}_{\rho_i(j)}\big)$ and $v_{i,j} = \mathbf{s}_i^\top \mathbf{t}_{i,j}$, then we can rewrite the above quantities more compactly as

$$\begin{aligned}
\mathbf{u}_{1,i}^\top &= \mathbf{s}_i^\top \mathbf{A}_i + \mathbf{e}_{1,i}^\top \\
\mathbf{u}_2^\top &= \mathbf{s}^\top \mathbf{B} + \mathbf{e}_2^\top = \left[\sum_{i \in [\ell]} \mathbf{s}_i^\top \mathbf{B}_i \;\middle|\; \sum_{i \in [\ell]} \mathbf{s}_i^\top \mathbf{p}_i\right] + \mathbf{e}_2^\top \\
\mathbf{u}_{3,i}^\top &= \mathbf{s}^\top \mathbf{P}_i + \mathbf{e}_{3,i}^\top = \left[\mathbf{s}_i^\top \mathbf{t}_{i,1} \;\middle|\; \cdots \;\middle|\; \mathbf{s}_i^\top \mathbf{t}_{i,N_i}\right] + \mathbf{e}_{3,i}^\top = \left[v_{i,1} \mid \cdots \mid v_{i,N_i}\right] + \mathbf{e}_{3,i}^\top.
\end{aligned}$$

– The challenger gives the challenge $(\{(\mathbf{A}_i, \mathbf{u}_{1,i}^\top)\}_{i \in [\ell]}, \mathbf{B}, \mathbf{u}_2^\top, \{\mathbf{u}_{3,i}^\top\}_{i \in [\ell]}, \mathsf{aux})$ to the distinguisher who then outputs a bit $b \in \{0, 1\}$. This is the output of the experiment.

- $\mathsf{Hyb}_1$: Same as $\mathsf{Hyb}_0$, except the challenger changes the distribution from which $\mathbf{u}_{1,i}, \mathbf{u}_2, \mathbf{u}_{3,i}$ are sampled:
  – For each $i \in [\ell]$, sample $\hat{\mathbf{e}}_{1,i}^\top \leftarrow D_{\mathbb{Z},\chi_{\mathsf{PRF}}}^m$ and set $\mathbf{u}_{1,i}^\top \leftarrow (\mathbf{s}_i^\top \mathbf{A}_i + \hat{\mathbf{e}}_{1,i}^\top) + \mathbf{e}_{1,i}^\top$.
  – For each $i \in [\ell]$, sample $\hat{\mathbf{e}}_{2,i} \leftarrow D_{\mathbb{Z},\chi_{\mathsf{PRF}}}^m$ and $\hat{e}'_{2,i} \leftarrow D_{\mathbb{Z},\chi_{\mathsf{PRF}}}$ and let $\hat{\mathbf{u}}_{2,i} = \mathbf{s}_i^\top \mathbf{B}_i + \hat{\mathbf{e}}_{2,i}^\top$ and $\hat{u}'_{2,i} = \mathbf{s}_i^\top \mathbf{p}_i + \hat{\mathbf{e}}'_{2,i}$. Then, set

$$\mathbf{u}_2^\top \leftarrow \left[\sum_{i \in [\ell]} \hat{\mathbf{u}}_{2,i} \;\middle|\; \sum_{i \in [\ell]} \hat{u}'_{2,i}\right] = \left[\sum_{i \in [\ell]} (\mathbf{s}_i^\top \mathbf{B}_i + \hat{\mathbf{e}}_{2,i}^\top) \;\middle|\; \sum_{i \in [\ell]} (\mathbf{s}_i^\top \mathbf{p}_i + \hat{e}'_{2,i})\right] + \mathbf{e}_2^\top,$$

  – For each $i \in [\ell]$ and $j \in [N_i]$, compute

$$\begin{aligned}
v_{i,j} &= \big(\mathbf{s}_i^\top \mathbf{p}_i + \hat{e}'_{2,i}\big) + \big(\mathbf{s}_i^\top \mathbf{B}_i + \hat{\mathbf{e}}_{2,i}^\top\big) \cdot \mathsf{H}\big(\mathsf{gid}_{\rho_i(j)}\big) \\
&= \hat{u}'_{2,i} + \hat{\mathbf{u}}_{2,i}^\top \mathsf{H}\big(\mathsf{gid}_{\rho_i(j)}\big),
\end{aligned} \tag{6.3}$$

  and let

$$\mathbf{u}_{3,i}^\top = \left[v_{i,1} \mid \cdots \mid v_{i,N_i}\right] + \mathbf{e}_{3,i}^\top, \tag{6.4}$$

  where $\mathbf{e}_{3,i} \leftarrow D_{\mathbb{Z},\chi}^{N_i}$ as in $\mathsf{Hyb}_0$.

- $\mathsf{Hyb}_2$: Same as $\mathsf{Hyb}_1$, except the challenger changes the distribution from which $\mathbf{u}_{1,i}, \mathbf{u}_2, \mathbf{u}_{3,i}$ are sampled:

– For each $i \in [\ell]$, sample $\mathbf{u}_{1,i} \xleftarrow{R} \mathbb{Z}_q^m$.

– For each $i \in [\ell]$, sample $\hat{\mathbf{u}}_{2,i} \xleftarrow{R} \mathbb{Z}_q^m$ and $\hat{u}'_{2,i} \xleftarrow{R} \mathbb{Z}_q$. Set

$$\mathbf{u}_2^\intercal \leftarrow \left[ \sum_{i \in [\ell]} \hat{\mathbf{u}}_{2,i} \;\middle|\; \sum_{i \in [\ell]} \hat{u}'_{2,i} \right].$$

– Compute $v_{i,j} \leftarrow \hat{u}'_{2,i} + \hat{\mathbf{u}}_{2,i}^\intercal \mathsf{H}(\mathsf{gid}_{\rho_i(j)})$ as in Eq. (6.3). Then sample $\mathbf{e}_{3,i} \leftarrow D_{\mathbb{Z},\chi}^{N_i}$ and set $\mathbf{u}_{3,i}^\intercal = [v_{i,1} \mid \cdots \mid v_{i,N_i}] + \mathbf{e}_{3,i}^\intercal$ as in Eq. (6.4).

- $\mathsf{Hyb}_3$: Same as $\mathsf{Hyb}_2$ except the challenger samples $\mathbf{u}_{3,1} \xleftarrow{R} \mathbb{Z}_q^{n \times N_1}$. For all $i > 1$, the components $\mathbf{u}_{3,i}$ are constructed as in $\mathsf{Hyb}_2$.

  Notably, this hybrid "breaks the correlation" between the components of $\mathbf{u}_2$ and the $\mathbf{u}_{3,i}$'s. The transition from $\mathsf{Hyb}_2$ to $\mathsf{Hyb}_3$ (Lemma 6.9) critically relies on admissibility of the MA-ABE adversary (i.e., for every key query $(A, \mathsf{gid})$ the adversary makes, it must be the case that $A \notin A^*$, where $A^*$ is the set of authorities associated with the challenger ciphertext.

- $\mathsf{Hyb}_4$: Same as $\mathsf{Hyb}_3$ except the challenger samples $\mathbf{u}_2 \xleftarrow{R} \mathbb{Z}_q^{m+1}$ and $\mathbf{u}_{3,i} \xleftarrow{R} \mathbb{Z}_q^{n \times N_i}$ for all $i \in [\ell]$. This is the random distribution in the definition of $\mathsf{Adv}_{\mathcal{D}}^{(\mathsf{PRE})}$.

For a distinguisher $\mathcal{D}$, we write $\mathsf{Hyb}_i(\mathcal{D})$ to denote the output distribution of $\mathsf{Hyb}_i(\mathcal{D})$ with distinguisher $\mathcal{D}$. We now show that each pair of adjacent distributions are indistinguishable.

**Lemma 6.7.** *Suppose $\chi \geq \lambda^{\omega(1)} \cdot \left(\sqrt{\lambda} m \chi_{\mathsf{PRF}}\right)^{\tau+1} \ell$. Then, for all distinguishers $\mathcal{D}$, $\mathsf{Hyb}_0(\mathcal{D}) \overset{s}{\approx} \mathsf{Hyb}_1(\mathcal{D})$.*

*Proof.* The only difference in the two distribution is the distribution of the errors $\mathbf{e}_{1,i}, \mathbf{e}_2, \mathbf{e}_{3,i}$ associated with vectors $\mathbf{u}_{1,i}, \mathbf{u}_2, \mathbf{u}_{3,i}$. The claim follows by the smudging lemma (Lemma 3.15). Formally, we consider each term separately:

- In $\mathsf{Hyb}_0$, the error term associated with each $\mathbf{u}_{1,i}$ is $\mathbf{e}_{1,i} \leftarrow D_{\mathbb{Z},\chi}^m$ and in $\mathsf{Hyb}_1$, it is $\mathbf{e}_{1,i} + \hat{\mathbf{e}}_{1,i}$ where $\hat{\mathbf{e}}_{1,i} \leftarrow D_{\mathbb{Z},\chi_{\mathsf{PRF}}}^m$. By Fact 3.8, $\|\hat{\mathbf{e}}_1\| \leq \sqrt{\lambda}\chi_{\mathsf{PRF}}$ with overwhelming probability. Since $\chi > \lambda^{\omega(1)}\chi_{\mathsf{PRF}}$, the distributions of $\mathbf{e}_1$ and $\mathbf{e}_1 + \hat{\mathbf{e}}_1$ are statistically close by Lemma 3.15.

- In $\mathsf{Hyb}_0$, the error term associated with $\mathbf{u}_2$ is $\mathbf{e}_2 \leftarrow D_{\mathbb{Z},\chi}^{m+1}$ and in $\mathsf{Hyb}_1$, it is $\mathbf{e}_2 + \tilde{\mathbf{e}}$ where $\tilde{\mathbf{e}} = \left[\sum_{i \in [\ell]} \hat{\mathbf{e}}_{2,i}^\intercal \mid \sum_{i \in [\ell]} \hat{e}_{2,i}\right]^\intercal$, $\hat{\mathbf{e}}_{2,i} \leftarrow D_{\mathbb{Z},\chi_{\mathsf{PRF}}}^m$ and $\hat{e}_{2,i} \leftarrow D_{\mathbb{Z},\chi_{\mathsf{PRF}}}$. By Fact 3.8, $\|\tilde{\mathbf{e}}\| \leq \ell \cdot \sqrt{\lambda}\chi_{\mathsf{PRF}}$ with overwhelming probability. Since $\chi > \lambda^{\omega(1)}\ell\chi_{\mathsf{PRF}}$, the distributions of $\mathbf{e}_2$ and $\mathbf{e}_2 + \tilde{\mathbf{e}}_2$ are statistically close by Lemma 3.15.

- In $\mathsf{Hyb}_0$, the error associated with $\mathbf{u}_{3,i}$ is $\mathbf{e}_{3,i} \leftarrow D_{\mathbb{Z},\chi}^{N_i}$ and in $\mathsf{Hyb}_1$, it is $\mathbf{e}_{3,i} + \tilde{\mathbf{e}}_i$ where

$$\tilde{e}_{i,j} = \hat{e}'_{2,i} + \hat{\mathbf{e}}_{2,i}^\intercal \mathsf{H}(\mathsf{gid}_{\rho_i(j)}) = \hat{e}'_{2,i} + \hat{\mathbf{e}}_{2,i}^\intercal \prod_{k \in [\tau]} \mathbf{D}_{\mathsf{gid}_{\rho_i(j),k}} \boldsymbol{\eta}.$$

  Now, $\mathbf{D}_0, \mathbf{D}_1$ are both sampled from $D_{\mathbb{Z},\chi_{\mathsf{PRF}}}^{m \times m}$ and $\boldsymbol{\eta} \in \{0,1\}^m$. By Fact 3.8, with overwhelming probability $\|\mathbf{D}_0\|, \|\mathbf{D}_1\| \leq \sqrt{\lambda}\chi_{\mathsf{PRF}}$. Similarly, since $\hat{\mathbf{e}}_{2,i} \leftarrow D_{\mathbb{Z},\chi_{\mathsf{PRF}}}^m$, we also have that $\|\hat{\mathbf{e}}_{2,i}\| \leq \sqrt{\lambda}\chi_{\mathsf{PRF}}$. Thus, we conclude that with overwhelming probability, $\|\tilde{\mathbf{e}}_i\| \leq \sqrt{\lambda}\chi_{\mathsf{PRF}}(1 + (m\sqrt{\lambda}\chi_{\mathsf{PRF}})^\tau)$. Since $\chi > \lambda^{\omega(1)} \cdot \left(m\sqrt{\lambda}\chi_{\mathsf{PRF}}\right)^{\tau+1}$, the distributions of $\mathbf{e}_{3,i}$ and $\mathbf{e}_{3,i} + \tilde{\mathbf{e}}_i$ are statistically close by Lemma 3.15. □

**Lemma 6.8.** *Under the $\mathsf{LWE}_{n,2m+1,q,\chi_{\mathsf{PRF}}}$ assumption, for all efficient distinguishers $\mathcal{D}$, $\mathsf{Hyb}_1(\mathcal{D}) \overset{c}{\approx} \mathsf{Hyb}_2(\mathcal{D})$.*

*Proof.* For each $d \in [\ell]$, we define a sequence of intermediate hybrids:

- $\mathsf{Hyb}_{1,d}$: Same as $\mathsf{Hyb}_1$ except the challenger changes the distribution of $\mathbf{u}_{1,i}, \mathbf{u}_2, \mathbf{u}_{3,i}$:

– If $i \le d$, sample $\mathbf{u}_{1,i} \xleftarrow{R} \mathbb{Z}_q^m$. Otherwise, if $i > d$, sample $\hat{\mathbf{e}}_{1,i} \leftarrow D^m_{\mathbb{Z},\chi_{\mathsf{PRF}}}$ and $\mathbf{e}_{1,i} \leftarrow D^m_{\mathbb{Z},\chi}$ and set $\mathbf{u}_{1,i}^\top \leftarrow (\mathbf{s}_i^\top \mathbf{A}_i + \hat{\mathbf{e}}_{1,i}^\top) + \mathbf{e}_{1,i}^\top$.

– If $i \le d$, sample $\hat{\mathbf{u}}_{2,i} \xleftarrow{R} \mathbb{Z}_q^m$ and $\hat{u}'_{2,i} \xleftarrow{R} \mathbb{Z}_q$. If $i > d$, sample $\hat{\mathbf{e}}_{2,i} \leftarrow D^m_{\mathbb{Z},\chi_{\mathsf{PRF}}}$ and $\hat{e}'_{2,i} \leftarrow D_{\mathbb{Z},\chi_{\mathsf{PRF}}}$, and set $\hat{\mathbf{u}}_{2,i}^\top \leftarrow \mathbf{s}_i^\top \mathbf{B}_i + \hat{\mathbf{e}}_{2,i}^\top$ and $\hat{u}'_{2,i} \leftarrow \mathbf{s}_i^\top \mathbf{p}_i + \hat{e}'_{2,i}$. Finally, sample $\mathbf{e}_2 \leftarrow D^{m+1}_{\mathbb{Z},\chi}$ and compute

$$\mathbf{u}_2^\top \leftarrow \left[ \sum_{i\in[\ell]} \hat{\mathbf{u}}_{2,i}^\top \;\middle|\; \sum_{i\in[\ell]} \hat{u}'_{2,i} \right] + \mathbf{e}_2^\top.$$

– For each $i \in [\ell]$ and $j \in [N_i]$, compute $v_{i,j} \leftarrow \hat{u}'_{2,i} + \hat{\mathbf{u}}_{2,i}^\top \cdot \mathsf{H}\big(\mathsf{gid}_{\rho_i(j)}\big)$. Sample $\mathbf{e}_{3,i} \leftarrow D^{N_i}_{\mathbb{Z},\chi}$ and set $\mathbf{u}_{3,i}^\top \leftarrow \big[v_{i,1} \mid \cdots \mid v_{i,N_i}\big] + \mathbf{e}_{3,i}^\top$.

We define $\mathsf{Hyb}_{1,0} \equiv \mathsf{Hyb}_1$, and by construction, $\mathsf{Hyb}_{1,\ell} \equiv \mathsf{Hyb}_2$. We now show that for all $d \in [\ell]$, under the $\mathsf{LWE}_{n,2m+1,q,\chi_{\mathsf{PRF}}}$ assumption, hybrids $\mathsf{Hyb}_{1,d-1}(\mathcal{D})$ and $\mathsf{Hyb}_{1,d}(\mathcal{D})$ are computationally indistinguishable. Suppose there exists a distinguisher $\mathcal{D}$ such that $\big|\Pr[\mathsf{Hyb}_{1,d-1}(\mathcal{D}) = 1] - \Pr[\mathsf{Hyb}_{1,d}(\mathcal{D}) = 1]\big| = \varepsilon$. We use $\mathcal{D}$ to construct an adversary $\mathcal{B}$ for the LWE assumption:

1. At the beginning of the game, algorithm $\mathcal{B}$ receives an LWE challenge $(\mathbf{D}, \mathbf{z})$ where $\mathbf{D} \in \mathbb{Z}_q^{n\times(2m+1)}$ and $\mathbf{z} \in \mathbb{Z}_q^{2m+1}$. Algorithm $\mathcal{B}$ parses $\mathbf{D} = [\mathbf{A}_d \mid \mathbf{B}_d \mid \mathbf{p}_d]$ where $\mathbf{A}_d, \mathbf{B}_d \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{p}_d \in \mathbb{Z}_q^n$.

2. Algorithm $\mathcal{B}$ starts simulating an execution of $\mathsf{Samp}_{\mathcal{A}}(1^\lambda)$ as follows:

   - It starts running algorithm $\mathcal{A}$ with randomness $r \xleftarrow{R} \{0,1\}^\kappa$. Let $\ell$ be the number of non-corrupted authorities associated with the challenge ciphertext.
   - Algorithm $\mathcal{B}$ constructs the matrix $\mathbf{B}$ in Samp by first sampling $\mathbf{B}_i \xleftarrow{R} \mathbb{Z}_q^{n\times m}$ and $\mathbf{p}_i \xleftarrow{R} \mathbb{Z}_q^n$ for each $i \ne d$. The matrix $\mathbf{B}_d$ and vector $\mathbf{p}_d$ is taken from the LWE challenge. It then sets

   $$\mathbf{B} = \left[\begin{array}{c|c} \mathbf{B}_1 & \mathbf{p}_1 \\ \vdots & \vdots \\ \mathbf{B}_\ell & \mathbf{p}_\ell \end{array}\right].$$

   - Algorithm $\mathcal{B}$ constructs the remaining components $\mathbf{D}_0, \mathbf{D}_1, \mathbf{P}_1, \ldots, \mathbf{P}_\ell$, and aux exactly as described in the specification of $\mathsf{Samp}_{\mathcal{A}}$.

3. Algorithm $\mathcal{B}$ samples $\mathbf{A}_i \xleftarrow{R} \mathbb{Z}_q^{n\times m}$ for all $i \ne d$. Similar to above, the matrix $\mathbf{A}_d$ is taken from the LWE challenge.

4. For each $i > d$, algorithm $\mathcal{B}$ samples a secret key $\mathbf{s}_i \xleftarrow{R} \mathbb{Z}_q^n$. It also parses the challenge as $\mathbf{z}^\top = [\mathbf{z}_1^\top \mid \mathbf{z}_2^\top \mid z'_2]$ where $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{Z}_q^m$ and $z'_2 \in \mathbb{Z}_q$. It now constructs the components $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$ as follows:

   - **Component $\mathbf{u}_{1,i}$:** For each $i < d$, algorithm $\mathcal{B}$ samples $\mathbf{u}_{1,i} \xleftarrow{R} \mathbb{Z}_q^m$. For $i = d$, it samples $\mathbf{e}_{1,d} \leftarrow D^m_{\mathbb{Z},\chi}$ and sets $\mathbf{u}_{1,d} \leftarrow \mathbf{z}_1 + \mathbf{e}_{1,d}$. For $i > d$, it samples $\hat{\mathbf{e}}_{1,i} \leftarrow D^m_{\mathbb{Z},\chi_{\mathsf{PRF}}}$ and $\mathbf{e}_{1,i} \leftarrow D^m_{\mathbb{Z},\chi}$ and sets $\mathbf{u}_{1,i}^\top \leftarrow \mathbf{s}_i^\top \mathbf{A}_i + \hat{\mathbf{e}}_{1,i}$.
   - **Component $\mathbf{u}_2$:** For each $i < d$, algorithm $\mathcal{B}$ samples $\hat{\mathbf{u}}_{2,i} \xleftarrow{R} \mathbb{Z}_q^m$ and $\hat{u}'_{2,i} \xleftarrow{R} \mathbb{Z}_q$. For $i > d$, it samples $\hat{\mathbf{e}}_{2,i} \leftarrow D^m_{\mathbb{Z},\chi_{\mathsf{PRF}}}$ and $\hat{e}'_{2,i} \leftarrow D_{\mathbb{Z},\chi_{\mathsf{PRF}}}$. It then sets $\hat{\mathbf{u}}_{2,i}^\top \leftarrow \mathbf{s}_i^\top \mathbf{B}_i + \hat{\mathbf{e}}_{2,i}^\top$ and and $\hat{u}'_{2,i} \leftarrow \mathbf{s}_i^\top \mathbf{p}_i + \hat{e}'_{2,i}$. For $i = d$, it sets $\hat{\mathbf{u}}_{2,d} \leftarrow \mathbf{z}_2$ and $\hat{u}'_{2,d} \leftarrow z'_2$. Finally, it samples $\mathbf{e}_2 \leftarrow D^{m+1}_{\mathbb{Z},\chi}$ and computes

   $$\mathbf{u}_2^\top \leftarrow \left[ \sum_{i\in[\ell]} \hat{\mathbf{u}}_{2,i}^\top \;\middle|\; \sum_{i\in[\ell]} \hat{u}'_{2,i} \right] + \mathbf{e}_2^\top.$$

   - **Component $\mathbf{u}_{3,i}$:** It computes $\mathbf{u}_{3,i}$ for each $i \in [\ell]$ using the same procedure described in $\mathsf{Hyb}_{2,d}$ and $\mathsf{Hyb}_{2,d+1}$. Namely algorithm $\mathcal{B}$ samples $\mathbf{e}_{3,i} \leftarrow D^{N_i}_{\mathbb{Z},\chi}$ and computes computes $v_{i,j} \leftarrow \hat{u}'_{2,i} + \hat{\mathbf{u}}_{2,i}^\top \mathsf{H}\big(\mathsf{gid}_{\rho_i(j)}\big)$ as in Eq. (6.3) and $\mathbf{u}_{3,i} = [v_{i,1} \mid \cdots \mid v_{i,N_i}] + \mathbf{e}_{3,i}^\top$ as in Eq. (6.4).

5. Algorithm $\mathcal{B}$ gives $(\{(\mathbf{A}_i, \mathbf{u}_{1,i}^\intercal)\}_{i\in[\ell]}, \mathbf{B}, \mathbf{u}_2^\intercal, \{\mathbf{u}_{3,i}^\intercal\}_{i\in[\ell]}, \mathsf{aux})$ to $\mathcal{D}$ and outputs whatever $\mathcal{D}$ outputs.

By definition, $\mathbf{A}_d, \mathbf{B}_d \xleftarrow{R} \mathbb{Z}_q^{n\times m}$ and $\mathbf{p}_d \xleftarrow{R} \mathbb{Z}_q^n$ so algorithm $\mathcal{B}$ perfectly simulates the distribution of $(\{\mathbf{A}_i\}_{i\in[\ell]}, \mathbf{B}, \mathsf{aux})$ for $\mathcal{D}$. It suffices to consider the remaining components $\mathbf{u}_{1,i}, \mathbf{u}_2, \mathbf{u}_{3,i}$. First, observe that the distribution of $\mathbf{u}_{1,i}$, $\mathbf{u}_{2,i}$ and $\hat{u}_{2,i}$ for all $i \neq d$ are distributed exactly as required in $\mathsf{Hyb}_{2,d-1}$ and $\mathsf{Hyb}_{2,d}$, so consider the distribution of $\mathbf{u}_{1,d}$, $\mathbf{u}_{2,d}$, and $\hat{u}_{2,d}$:

- Suppose $\mathbf{z}_1^\intercal = \mathbf{s}^\intercal\mathbf{A}_d + \mathbf{e}_1^\intercal$, $\mathbf{z}_2^\intercal = \mathbf{s}^\intercal\mathbf{B}_d + \mathbf{e}_2^\intercal$, and $z_2' = \mathbf{s}^\intercal\mathbf{p}_d + e_2'$ for some $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^n$, $\mathbf{e}_1, \mathbf{e}_2 \leftarrow D^m_{\mathbb{Z},\chi_{\mathsf{PRF}}}$ and $e_2' \leftarrow D_{\mathbb{Z},\chi_{\mathsf{PRF}}}$. Then, $\mathbf{u}_{1,d}$, $\mathbf{u}_{2,d}$ and $\hat{u}_{2,d}$ are distributed exactly as in $\mathsf{Hyb}_{1,d-1}$.
- Suppose $\mathbf{z}_1, \mathbf{z}_2 \xleftarrow{R} \mathbb{Z}_q^m$ and $z_2' \xleftarrow{R} \mathbb{Z}_q$. Then $\mathbf{u}_{1,d}$, $\mathbf{u}_{2,d}$ and $\hat{u}_{2,d}$ are distributed exactly as in $\mathsf{Hyb}_{1,d}$. In particular, in this case, $\mathbf{z}_1, \mathbf{z}_2, z_2'$ are uniform and entirely independent of all other scheme parameters (e.g., $\mathbf{e}_1, \mathbf{e}_2$).

Thus, we conclude that the components $\mathbf{u}_{1,i}$, $\mathbf{u}_{2,i}$, $\hat{u}_{2,i}'$ are simulated exactly as in either $\mathsf{Hyb}_{1,d-1}$ or $\mathsf{Hyb}_{1,d}$. Since these components fully determine the distribution of $\mathbf{u}_2$ and $\mathbf{u}_{3,i}$ in $\mathsf{Hyb}_{1,d-1}$ and $\mathsf{Hyb}_{1,d}$ (and via identical relations), we conclude that if $\mathbf{z}$ is sampled from the LWE distribution, then algorithm $\mathcal{B}$ successfully simulated $\mathsf{Hyb}_{1,d-1}$ and if $\mathbf{z}$ is uniformly random, then $\mathcal{B}$ successfully simulated $\mathsf{Hyb}_{1,d}$. The claim now follows by a hybrid argument. □

**Lemma 6.9.** *Let $\chi_{\mathsf{smudge}} = \chi_{\mathsf{smudge}}(\lambda)$ be a smudging parameter where $\chi_{\mathsf{smudge}} > \lambda^{\tau+\omega(1)} \cdot (m\chi_{\mathsf{PRF}})^\tau$, and suppose moreover that $\chi > \lambda^{\omega(1)} \cdot \chi_{\mathsf{smudge}}$. Suppose $m > 6n\log q$. Then, under the $\mathsf{LWE}_{n,m',q,\chi_{\mathsf{PRF}}}$ assumption for some $m' = \mathsf{poly}(m,\tau,Q)$ where $Q$ is a bound on the number of secret-key queries adversary $\mathcal{A}$ makes, it holds that for all efficient distinguishers $\mathcal{D}$, $\mathsf{Hyb}_2(\mathcal{D}) \overset{c}{\approx} \mathsf{Hyb}_3(\mathcal{D})$.*

*Proof.* Let $N_1 \leq Q$ be the number of secret-key queries algorithm $\mathcal{A}$ makes that contains the first authority $\mathsf{aid}_1^*$. For each $d \in [N_1 + 1]$, we define a sequence of intermediate hybrids:

- $\mathsf{Hyb}_{2,d}$: Same as $\mathsf{Hyb}_2$ except the challenger changes the distribution of $v_{1,j}$:
  - If $j < d$, then the challenger samples $v_{1,j} \xleftarrow{R} \mathbb{Z}_q$.
  - Otherwise, the challenge samples $v_{1,j}$ as in $\mathsf{Hyb}_2$. Namely, $v_{1,j} = \hat{u}_{2,1}' + \hat{\mathbf{u}}_{2,1}^\intercal \mathsf{H}(\mathsf{gid}_{\rho_1(j)})$.
- $\mathsf{Hyb}_{2,d}^{(1)}$: Same as $\mathsf{Hyb}_{2,d}$ except the challenger changes the distribution of $v_{i,j}$. We start by defining a few useful quantities:
  - Let $\gamma = \rho_1(d)$. Namely, $\gamma$ is the index of the $d^{\text{th}}$ secret-key query that contains authority $\mathsf{aid}_1^*$.
  - Let $(\mathsf{gid}_\gamma, A_\gamma)$ be the $\gamma^{\text{th}}$ secret-key query that algorithm $\mathcal{A}$ makes. Since $\mathcal{A}$ is admissible, it must be the case that $A^* \not\subseteq A_\gamma$, where $A^*$ is the set of authorities associated with the challenge ciphertext, so there exists some other index $1 < i^* \leq \ell$ such that $\mathsf{aid}_{i^*}^* \notin A_\gamma$. Let $i^* \in [\ell]$ be the *smallest* such index where $\mathsf{aid}_{i^*}^* \notin A_\gamma$.
  - In particular, this means that $\rho_1(d) = \gamma$ and moreover, that $\rho_{i^*}(j) \neq \gamma$ for all $j \in [N_{i^*}]$. Recall that $\rho_{i^*}(\cdot)$ ranges over the secret-key query indices that contain $\mathsf{aid}_{i^*}^*$, and by construction $\mathsf{aid}_{i^*}^* \notin A_\gamma$.

  Then, sample $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^m$, $\hat{\mathbf{u}}_{2,1}', \hat{\mathbf{u}}_{2,i^*}' \xleftarrow{R} \mathbb{Z}_q^m$, and set $\hat{\mathbf{u}}_{2,1} \leftarrow \hat{\mathbf{u}}_{2,1}' + \mathbf{s}$ and $\hat{\mathbf{u}}_{2,i^*} \leftarrow \hat{\mathbf{u}}_{2,i^*}' - \mathbf{s}$. Specifically, the challenger constructs components $v_{i,j}$ as follows:
  - If $i = 1$ and $j < d$, then $v_{1,j} \xleftarrow{R} \mathbb{Z}_q$.
  - Otherwise, $v_{i,j} = \hat{u}_{2,i}' + \hat{\mathbf{u}}_{2,i}^\intercal \mathsf{H}(\mathsf{gid}_{\rho_i(j)})$. In particular, this means that if $i = 1$ and $j \geq d$, then

$$v_{1,j} = \hat{u}_{2,1}' + \hat{\mathbf{u}}_{2,1}^\intercal \mathsf{H}(\mathsf{gid}_{\rho_1(j)}) = \hat{u}_{2,1}' + (\hat{\mathbf{u}}_{2,1}')^\intercal \mathsf{H}(\mathsf{gid}_{\rho_1(j)}) + \mathbf{s}^\intercal \mathsf{H}(\mathsf{gid}_{\rho_1(j)}),$$

  and if $i = i^*$, then

$$v_{i^*,j} = \hat{u}_{2,i^*}' + \hat{\mathbf{u}}_{2,i^*}^\intercal \mathsf{H}(\mathsf{gid}_{\rho_{i^*}(j)}) = \hat{u}_{2,i^*}' + (\hat{\mathbf{u}}_{2,i^*}')^\intercal \mathsf{H}(\mathsf{gid}_{\rho_{i^*}(j)}) - \mathbf{s}^\intercal \mathsf{H}(\mathsf{gid}_{\rho_{i^*}(j)}).$$

- $\mathsf{Hyb}^{(2)}_{2,d}$: Same as $\mathsf{Hyb}^{(1)}_{2,d}$ except the challenger samples $\hat{e}_i \leftarrow D_{\mathbb{Z},\chi_{\mathsf{smudge}}}$ for each $i \in [Q]$ and modifies $v_{i,j}$ as follows:
  - If $i = 1$ and $j \geq d$, the challenger sets
  $$v_{1,j} = \hat{u}'_{2,1} + (\hat{\mathbf{u}}'_{2,1})^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_1(j)}) + (\mathbf{s}^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_1(j)}) + \hat{e}_{\rho_1(j)}).$$
  - If $i = i^*$, the challenger sets
  $$v_{i^*,j} = \hat{u}'_{2,i^*} + (\hat{\mathbf{u}}'_{2,i^*})^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_{i^*}(j)}) - (\mathbf{s}^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_{i^*}(j)}) + \hat{e}_{\rho_{i^*}(j)}).$$

  The vectors $\mathbf{u}_{3,i}$ is still computed according to Eq. (6.4): $\mathbf{u}^{\mathsf{T}}_{3,i} = \left[v_{i,1} \mid \cdots \mid v_{i,N_i}\right] + \mathbf{e}^{\mathsf{T}}_{3,i}$, where $\mathbf{e}_{3,i} \leftarrow D^{N_i}_{\mathbb{Z},\chi}$.
- $\mathsf{Hyb}^{(3)}_{2,d}$: Same as $\mathsf{Hyb}^{(2)}_{2,d}$ except the challenger replaces $\mathbf{s}^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_i) + \hat{e}_i$ with $r_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q$ for each $i \in [Q]$. Specifically, the challenger constructs $v_{i,j}$ as follows:
  - If $i = 1$ and $j \geq d$, the challenger sets
  $$v_{1,j} = \hat{u}'_{2,1} + (\hat{\mathbf{u}}'_{2,1})^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_1(j)}) + r_{\rho_1(j)}.$$
  - If $i = i^*$, the challenger sets
  $$v_{i^*,j} = \hat{u}'_{2,i^*} + (\hat{\mathbf{u}}'_{2,i^*})^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_{i^*}(j)}) - r_{\rho_{i^*}(j)}.$$
- $\mathsf{Hyb}^{(4)}_{2,d}$: Same as $\mathsf{Hyb}^{(3)}_{2,d}$ except the challenger samples $v_{1,d} \xleftarrow{\mathrm{R}} \mathbb{Z}_q$.
- $\mathsf{Hyb}^{(5)}_{2,d}$: Same as $\mathsf{Hyb}^{(4)}_{2,d}$ except the challenger replaces $r_i$ with $\mathbf{s}^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_i) + \hat{e}_i$ where $\hat{e}_i \leftarrow D_{\mathbb{Z},\chi_{\mathsf{smudge}}}$ for all $i \in [Q]$:
  - If $i = 1$ and $j = d$ the challenge samples $v_{1,d} \xleftarrow{\mathrm{R}} \mathbb{Z}_q$.
  - If $i = 1$ and $j > d$, the challenger sets
  $$v_{1,j} = \hat{u}'_{2,1} + (\hat{\mathbf{u}}'_{2,1})^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_1(j)}) + (\mathbf{s}^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_1(j)}) + \hat{e}_{\rho_1(j)}).$$
  - If $i = i^*$, the challenger sets
  $$v_{i^*,j} = \hat{u}'_{2,i^*} + (\hat{\mathbf{u}}'_{2,i^*})^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_{i^*}(j)}) - (\mathbf{s}^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_{i^*}(j)}) + \hat{e}_{\rho_{i^*}(j)}).$$
- $\mathsf{Hyb}^{(6)}_{2,d}$: Same as $\mathsf{Hyb}^{(5)}_{2,d}$ except the challenger modifies $v_{i,j}$ as follows:
  - If $i = 1$ and $j > d$, the challenger sets
  $$v_{1,j} = \hat{u}'_{2,1} + (\hat{\mathbf{u}}'_{2,1})^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_1(j)}) + \mathbf{s}^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_1(j)}).$$
  - If $i = i^*$, the challenger sets
  $$v_{i,j} = \hat{u}'_{2,i^*} + (\hat{\mathbf{u}}'_{2,i^*})^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_{i^*}(j)}) - \mathbf{s}^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_{\rho_{i^*}(j)}).$$

By construction, $\mathsf{Hyb}_{2,1} \equiv \mathsf{Hyb}_2$ and $\mathsf{Hyb}_{2,Q+1} \equiv \mathsf{Hyb}_3$. We now show that each adjacent pair of hybrids are indistinguishable.

**Claim 6.10.** *For all distinguishers $\mathcal{D}$, we have that* $\mathsf{Hyb}_{2,d}(\mathcal{D}) \equiv \mathsf{Hyb}^{(1)}_{2,d}(\mathcal{D})$.

*Proof.* This transition is syntactic. In both experiments, the distribution of $\hat{\mathbf{u}}_{2,1}$ and $\hat{\mathbf{u}}_{2,i^*}$ is uniform over $\mathbb{Z}^m_q$. □

**Claim 6.11.** *Suppose that $\chi \geq \lambda^{\omega(1)} \cdot \chi_{\mathsf{smudge}}$. Then, for all distinguishers $\mathcal{D}$, $\mathsf{Hyb}^{(1)}_{2,d}(\mathcal{D}) \stackrel{s}{\approx} \mathsf{Hyb}^{(2)}_{2,d}(\mathcal{D})$.*

*Proof.* The only difference between $\mathsf{Hyb}^{(1)}_{2,d}$ and $\mathsf{Hyb}^{(2)}_{2,d}$ is the extra $\hat{e}_j$ components in some of the $v_{i,j}$ terms. By construction, the challenger samples $\hat{e}_j \leftarrow D_{\mathbb{Z},\chi_{\mathsf{smudge}}}$, so $|\hat{e}_j| \leq \sqrt{\lambda}\chi_{\mathsf{smudge}}$ with overwhelming probability (Fact 3.8). For $i \in [\ell]$ and $j \in [N_i]$, let $\hat{v}_{i,j} \in \mathbb{Z}_q$ denote the value of $v_{i,j}$ computed according to the specification of $\mathsf{Hyb}^{(1)}_{2,d}$. Let $e_{3,i,k}$ to denote the $k^{\text{th}}$ component of $\mathbf{e}_{3,i}$. Consider the distribution of each component $u_{3,i,j}$ of $\mathbf{u}_{3,i}$:

- In $\mathsf{Hyb}^{(1)}_{2,d}$, we have $u_{3,i,j} = \hat{v}_{i,j} + e_{3,i,j}$.
- In $\mathsf{Hyb}^{(2)}_{2,d}$, we have $u_{3,i,j} = \hat{v}_{i,j} + e_{3,i,j} + c \cdot \hat{e}_j$, where $c \in \{-1, 0, 1\}$. Then, by Lemma 3.15, the distribution of $c \cdot \hat{e}_i + e_{3,i}$ and $e_{3,i}$ where $e_{3,i} \leftarrow D_{\mathbb{Z},\chi}$ is statistically close when $\chi \geq \chi_{\mathsf{smudge}} \cdot \lambda^{\omega(1)}$.

The claim now follows by a hybrid argument. □

**Claim 6.12.** *Suppose $m > 6n \log q$ and $\chi_{\mathsf{smudge}} > \lambda^{\tau+\omega(1)} \cdot (m\chi_{\mathsf{PRF}})^{\tau}$. Then, under the $\mathsf{LWE}_{n,m',q,\chi_{\mathsf{PRF}}}$ assumption for some $m' = \mathsf{poly}(m, \tau, Q)$ where $Q$ is a bound on the number of secret-key queries adversary $\mathcal{A}$ makes, it holds that for all efficient distinguishers $\mathcal{D}$, $\mathsf{Hyb}^{(2)}_{2,d}(\mathcal{D}) \stackrel{c}{\approx} \mathsf{Hyb}^{(3)}_{2,d}(\mathcal{D})$.*

*Proof.* The only difference between these two distributions is that we replace each output $\mathbf{s}^{\mathsf{T}}\mathsf{H}(\mathsf{gid}_i) + \hat{e}_i$ of the lattice-based PRF with truly random strings $r_{\rho_i(j)} \stackrel{\mathrm{R}}{\leftarrow} \mathbb{Z}_q$. This follows by pseudorandomness; specifically, under the given hypothesis, Theorem 6.1 holds. Formally, suppose there exists an efficient $\mathcal{D}$ such that $\left|\Pr[\mathsf{Hyb}^{(2)}_{2,d}(\mathcal{D}) = 1] - \Pr[\mathsf{Hyb}^{(3)}_{2,d}(\mathcal{D}) = 1]\right| = \varepsilon$. We use $\mathcal{D}$ to construct an efficient adversary $\mathcal{B}$ that breaks the lattice-based PRF from Theorem 6.1:

1. At the beginning of the game, algorithm $\mathcal{B}$ receives matrices $\mathbf{D}_0, \mathbf{D}_1 \in \mathbb{Z}_q^{m \times m}$ from the challenger.
2. Algorithm $\mathcal{B}$ runs $(\mathbf{B}, \mathbf{P}_1, \ldots, \mathbf{P}_\ell, \mathsf{aux}) \leftarrow \mathsf{Samp}_{\mathcal{A}}(1^\lambda)$, except it uses the matrices $\mathbf{D}_0, \mathbf{D}_1$ it received from the challenger instead of sampling them itself. It also samples $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \stackrel{\mathrm{R}}{\leftarrow} \mathbb{Z}_q^{n \times m}$. Let $Q = \{(\mathsf{gid}_1, A_1), \ldots, (\mathsf{gid}_Q, A_Q)\}$ be the set of secret key queries algorithm $\mathcal{A}$ makes (in the execution of $\mathsf{Samp}_{\mathcal{A}}$). For each $i \in [\ell]$, define the mapping $\rho_i \colon [N_i] \to [Q]$ exactly as in the specification of $\mathsf{Hyb}_0$.
3. Algorithm $\mathcal{B}$ makes queries on inputs $\mathsf{gid}_1, \ldots, \mathsf{gid}_Q$. Let $y_1, \ldots, y_Q \in \mathbb{Z}_q$ be the responses.
4. Let $(\mathsf{gid}_d, A_d)$ be the $d^{\text{th}}$ secret-key query chosen by $\mathcal{A}$. Algorithm $\mathcal{B}$ samples $\hat{\mathbf{u}}_i \stackrel{\mathrm{R}}{\leftarrow} \mathbb{Z}_q^m$, $\hat{u}'_i \stackrel{\mathrm{R}}{\leftarrow} \mathbb{Z}_q$, and $\mathbf{e}_{3,i} \leftarrow D^{N_i}_{\mathbb{Z},\chi}$ for each $i \in [\ell]$. It constructs the components $v_{i,j}$ as follows:
   - If $i = 1$ and $j < d$, algorithm $\mathcal{B}$ samples $v_{i,j} \stackrel{\mathrm{R}}{\leftarrow} \mathbb{Z}_q$.
   - If $i = 1$ and $j \geq d$, algorithm $\mathcal{B}$ sets $v_{i,j} \leftarrow \hat{u}'_1 + \hat{\mathbf{u}}_1^{\mathsf{T}}\mathsf{H}\left(\mathsf{gid}_{\rho_1(j)}\right) + y_{\rho_1(j)}$.
   - If $i = i^*$, algorithm $\mathcal{B}$ sets $v_{i,j} \leftarrow \hat{u}'_{i^*} + \hat{\mathbf{u}}_{i^*}^{\mathsf{T}}\mathsf{H}\left(\mathsf{gid}_{\rho_{i^*}(j)}\right) - y_{\rho_{i^*}(j)}$.
   - If $i \notin \{1, i^*\}$, algorithm $\mathcal{B}$ sets $v_{i,j} \leftarrow \hat{u}'_i + \hat{\mathbf{u}}_i^{\mathsf{T}}\mathsf{H}\left(\mathsf{gid}_{\rho_i(j)}\right)$.
5. Finally, algorithm $\mathcal{B}$ samples $\mathbf{u}_{1,i} \stackrel{\mathrm{R}}{\leftarrow} \mathbb{Z}_q^m$ and $\mathbf{e}_{3,i} \leftarrow D^{N_i}_{\mathbb{Z},\chi}$ for each $i \in [\ell]$, and constructs vectors
$$\mathbf{u}_2^{\mathsf{T}} \leftarrow \left[\sum_{i \in [\ell]} \hat{\mathbf{u}}_i \;\middle|\; \sum_{i \in [\ell]} \hat{u}'_i\right]$$
and $\mathbf{u}_{3,i}^{\mathsf{T}} = [v_{i,1} \mid \cdots \mid v_{i,N_i}] + \mathbf{e}_{3,i}^{\mathsf{T}}$ according to the specification of $\mathsf{Hyb}^{(2)}_{2,d}$ and $\mathsf{Hyb}^{(3)}_{2,d}$.
6. Algorithm $\mathcal{B}$ gives the challenge $(\{(\mathbf{A}_i, \mathbf{u}_{1,i}^{\mathsf{T}})\}_{i \in [\ell]}, \mathbf{B}, \mathbf{u}_2^{\mathsf{T}}, \{\mathbf{u}_{3,i}^{\mathsf{T}}\}_{i \in [\ell]}, \mathsf{aux})$ to $\mathcal{D}$ and outputs whatever $\mathcal{D}$ outputs.

By construction, the components $(\{\mathbf{A}_i, \mathbf{u}_{1,i}^\intercal)\}_{i\in[\ell]}, \mathbf{B}, \{\mathbf{P}_i\}_{i\in[\ell]}, \mathsf{aux})$ are distributed exactly as in $\mathsf{Hyb}_{2,d}^{(2)}$ and $\mathsf{Hyb}_{2,d}^{(3)}$. Consider now the distributions of $\mathbf{u}_2$ and $\mathbf{u}_{3,i}$ that algorithm $\mathcal{B}$ induces:

- Suppose $y_i = \mathbf{s}^\intercal \mathsf{H}(\mathsf{gid}_i) + \hat{e}_i$ where $\mathbf{s} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^m$ and $\hat{e}_i \leftarrow D_{\mathbb{Z},\chi_{\mathsf{smudge}}}$. In this case, algorithm $\mathcal{B}$ perfectly simulates an execution of $\mathsf{Hyb}_{2,d}^{(2)}$ with secret $\mathbf{s}$, and components $\hat{\mathbf{u}}_{2,i} = \hat{\mathbf{u}}_i$ and $\hat{u}'_{2,i} = \hat{u}'_i$ for all $i \in [\ell]$.
- Suppose $y_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q$. Then, algorithm $\mathcal{B}$ perfectly simulates an execution of $\mathsf{Hyb}_{2,d}^{(3)}$ with components $\hat{\mathbf{u}}_{2,i} = \hat{\mathbf{u}}_i$ and $\hat{u}'_{2,i} = \hat{u}'_i$ for all $i \in [\ell]$.

Thus algorithm $\mathcal{B}$ breaks security of the lattice-based PRF in Theorem 6.1 with advantage $\varepsilon$ and the claim follows. □

**Claim 6.13.** *For all distinguishers $\mathcal{D}$, we have that $\mathsf{Hyb}_{2,d}^{(3)}(\mathcal{D}) \equiv \mathsf{Hyb}_{2,d}^{(4)}(\mathcal{D})$.*

*Proof.* This is purely syntactic. The only difference between these two distributions is the distribution of $v_{1,d}$. By construction of $i^*$ (see the description of $\mathsf{Hyb}_{2,d}^{(1)}$), we have that $\rho_{i^*}(j) \neq \rho_1(d)$ for all $j \in [N_{i^*}]$. Moreover, $\rho_1(\cdot)$ is an injective function so $\rho_1(j) \neq \rho_1(d)$ for all $j > d$. This means the *only* component in hybrids $\mathsf{Hyb}_{2,d}^{(3)}(\mathcal{D})$ and $\mathsf{Hyb}_{2,d}^{(4)}(\mathcal{D})$ that depends on $r_{\rho_1(d)}$ is $v_{1,d}$. Finally, $r_d$ is uniform over $\mathbb{Z}_q$ and independent of *all* other quantities, so the distribution of $v_{1,d}$ in $\mathsf{Hyb}_{2,d}^{(3)}$ is also uniform. This is identical to the distribution in $\mathsf{Hyb}_{2,d}^{(4)}$. □

**Claim 6.14.** *Under the conditions of Claim 6.12, for all efficient distinguishers $\mathcal{D}$, $\mathsf{Hyb}_{2,d}^{(4)}(\mathcal{D}) \overset{c}{\approx} \mathsf{Hyb}_{2,d}^{(5)}(\mathcal{D})$.*

*Proof.* Follows by a similar argument as in the proof of Claim 6.12. □

**Claim 6.15.** *Under the conditions of Claim 6.11, for all distinguishers $\mathcal{D}$, $\mathsf{Hyb}_{2,d}^{(5)}(\mathcal{D}) \overset{s}{\approx} \mathsf{Hyb}_{2,d}^{(6)}(\mathcal{D})$.*

*Proof.* Follows by the same argument as the proof of Claim 6.11. □

**Claim 6.16.** *For all distinguishers $\mathcal{D}$, we have that $\mathsf{Hyb}_{2,d}^{(6)}(\mathcal{D}) \equiv \mathsf{Hyb}_{2,d+1}(\mathcal{D})$.*

*Proof.* Follows by the same argument as the proof of Claim 6.10. □

Combining Claims 6.10 to 6.16, the output distributions of $\mathsf{Hyb}_2$ and $\mathsf{Hyb}_3$ are computationally indistinguishable. □

**Lemma 6.17.** *Suppose $m > 6n\log q$, $\chi_{\mathsf{smudge}} > \lambda^{\tau+\omega(1)} \cdot (m\chi_{\mathsf{PRF}})^\tau$, and $\chi > \lambda^{\omega(1)} \cdot \chi_{\mathsf{smudge}}$. Then, under the $\mathsf{LWE}_{n,m',q,\chi_{\mathsf{PRF}}}$ assumption for some $m' = \mathsf{poly}(m,\tau,Q)$ where $Q$ is a bound on the number of secret-key queries adversary $\mathcal{A}$ makes, it holds that for all efficient distinguishers $\mathcal{D}$, $\mathsf{Hyb}_3(\mathcal{D}) \overset{c}{\approx} \mathsf{Hyb}_4(\mathcal{D})$.*

*Proof.* First, we note that by construction, in $\mathsf{Hyb}_3$ and $\mathsf{Hyb}_4$, the vector $\mathbf{u}_3$ is *independent* of $\hat{\mathbf{u}}_{2,1}$ and $\hat{u}'_{2,1}$. The only component in $\mathsf{Hyb}_3$ and $\mathsf{Hyb}_4$ that depends on $\hat{\mathbf{u}}_{2,1}$ and $\hat{u}'_{2,1}$ is the vector $\mathbf{u}_2$. Since $\hat{\mathbf{u}}_{2,1} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^m$ and $\hat{u}'_{2,1} \xleftarrow{\mathrm{R}} \mathbb{Z}_q$, this means that the distribution of $\mathbf{u}_2$ is uniform over $\mathbb{Z}_q^{m+1}$ and independent of all other components in $\mathsf{Hyb}_3$ and $\mathsf{Hyb}_4$. Thus, it suffices to reason about the distribution of the components of $\mathbf{u}_{3,i}$ (for $i > 1$) in the two experiments (all other components are identically distributed). To do so, we define a sequence of hybrid experiments indexed by $d \in [2, \ell]$:

- $\mathsf{Hyb}_{3,d}$: Same as $\mathsf{Hyb}_3$ except the challenger changes the distribution of $v_{i,j}$:
  - If $i < d$, the challenger samples $v_{i,j} \xleftarrow{\mathrm{R}} \mathbb{Z}_q$.
  - If $i \geq d$, the challenger sets $v_{i,j} \leftarrow \hat{u}'_{2,i} + \hat{\mathbf{u}}_{2,i}^\intercal \mathsf{H}(\mathsf{gid}_{\rho_i(j)})$.
- $\mathsf{Hyb}_{3,d}^{(1)}$: Same as $\mathsf{Hyb}_{3,d}$ except the challenger changes the distribution of $v_{d,j}$:
  - If $i < d$, the challenger samples $v_{i,j} \xleftarrow{\mathrm{R}} \mathbb{Z}_q$.
  - If $i = d$, the challenger sets $v_{d,j} \leftarrow \hat{u}'_{2,d} + \left(\hat{\mathbf{u}}_{2,d}^\intercal \mathsf{H}(\mathsf{gid}_{\rho_d(j)}) + \hat{e}_{\rho_d(j)}\right)$ where $\hat{e}_{\rho_d(j)} \leftarrow D_{\mathbb{Z},\chi_{\mathsf{smudge}}}$.
  - If $i > d$, the challenger sets $v_{i,j} \leftarrow \hat{u}'_{2,i} + \hat{\mathbf{u}}_{2,i}^\intercal \mathsf{H}(\mathsf{gid}_{\rho_i(j)})$.

- $\mathsf{Hyb}^{(2)}_{3,d}$: Same as $\mathsf{Hyb}^{(1)}_{3,d}$ except the challenger sets $v_{d,j} \leftarrow \hat{u}'_{2,d} + r_{\rho_d(j)}$ where $r_{\rho_d(j)} \xleftarrow{R} \mathbb{Z}_q$.

In all experiments, each vector $\mathbf{u}_{3,i}$ is still constructed from $v_{i,j}$ and $\mathbf{e}_{3,i} \leftarrow D^{N_i}_{\mathbb{Z},\chi}$ according to Eq. (6.4). By construction $\mathsf{Hyb}_{3,2} \equiv \mathsf{Hyb}_3$ and $\mathsf{Hyb}_{3,\ell} \equiv \mathsf{Hyb}_4$. We now show that each adjacent pair of hybrid experiments are indistinguishable.

**Claim 6.18.** *Suppose $\chi \geq \lambda^{\omega(1)} \cdot \chi_{\mathsf{smudge}}$. Then, for all distinguishers $\mathcal{D}$, $\mathsf{Hyb}_{3,d}(\mathcal{D}) \overset{s}{\approx} \mathsf{Hyb}^{(1)}_{3,d}(\mathcal{D})$.*

*Proof.* This follows via a similar argument as the proof of Claim 6.11. Specifically, the only difference between $\mathsf{Hyb}_{3,d}$ and $\mathsf{Hyb}^{(1)}_{3,d}$ is the extra $\hat{e}_{\rho_d(j)}$ term in the components $v_{d,j}$. By construction, the challenger samples $\hat{e}_{\rho_d(i)} \leftarrow D_{\mathbb{Z},\chi_{\mathsf{smudge}}}$, so $|\hat{e}_{\rho_d(i)}| \leq \sqrt{\lambda}\chi_{\mathsf{smudge}}$ with overwhelming probability (Fact 3.8). For $j \in [N_d]$, let $\hat{v}_{d,j} \in \mathbb{Z}_q$ denote the value of $v_{d,j}$ computed according to the specification of $\mathsf{Hyb}_{3,d}$. Let $e_{3,d,j}$ denote the $j^{\text{th}}$ component of $\mathbf{e}_{3,d}$. Consider the distribution of each component $u_{3,d,j}$ of $\mathbf{u}_{3,d}$:

- In $\mathsf{Hyb}_{3,d}$, $u_{3,d,j} = \hat{v}_{d,j} + e_{3,d,j}$.
- In $\mathsf{Hyb}^{(1)}_{3,d}$, $u_{3,d,j} = \hat{v}_{d,j} + e_{3,d,j} + \hat{e}_{\rho_d(j)}$. By Lemma 3.15, the distribution of $\hat{e}_{\rho_d(j)} + e_{3,d,j}$ and $e_{3,d,j}$ where $e_{3,d,j} \leftarrow D_{\mathbb{Z},\chi}$ is statistically close when $\chi \geq \chi_{\mathsf{smudge}} \cdot \lambda^{\omega(1)}$.

The claim now follows by a hybrid argument. □

**Claim 6.19.** *Suppose $m > 6n \log q$ and $\chi_{\mathsf{smudge}} > \lambda^{\tau+\omega(1)} \cdot (m\chi_{\mathsf{PRF}})^{\tau}$. Then, under the $\mathsf{LWE}_{n,m',q,\chi_{\mathsf{PRF}}}$ assumption for some $m' = \mathsf{poly}(m, \tau, Q)$ where $Q$ is a bound on the number of secret-key queries adversary $\mathcal{A}$ makes, it holds that for all efficient distinguishers $\mathcal{D}$, $\mathsf{Hyb}^{(1)}_{3,d}(\mathcal{D}) \overset{c}{\approx} \mathsf{Hyb}^{(2)}_{3,d}(\mathcal{D})$.*

*Proof.* The only difference between these two distributions is that we replace each output $\hat{\mathbf{u}}^{\mathsf{T}}_{2,d}\mathsf{H}(\mathsf{gid}_i) + \hat{e}_i$ of the lattice-based PRF with a truly random string $r_i \xleftarrow{R} \mathbb{Z}_q$ for each $i \in [Q]$ (technically, for only the subset of $[Q]$ that is in the image of $\rho_d$). This follows by pseudorandomness; specifically, under the given hypothesis, Theorem 6.1 holds. The argument is a simpler version of the proof of Claim 6.12. Formally, suppose there exists an efficient distinguisher $\mathcal{D}$ such that $\left|\Pr[\mathsf{Hyb}^{(1)}_{3,d}(\mathcal{D}) = 1] - \Pr[\mathsf{Hyb}^{(2)}_{3,d}(\mathcal{D}) = 1]\right| = \varepsilon$. We use $\mathcal{D}$ to construct an efficient adversary $\mathcal{B}$ that breaks the lattice-based PRF from Theorem 6.1:

1. At the beginning of the game, algorithm $\mathcal{B}$ receives matrices $\mathbf{D}_0, \mathbf{D}_1 \in \mathbb{Z}_q^{m \times m}$ from the challenger.
2. Algorithm $\mathcal{B}$ runs $(\mathbf{B}, \mathbf{P}_1, \ldots, \mathbf{P}_\ell, \mathsf{aux}) \leftarrow \mathsf{Samp}_{\mathcal{A}}(1^\lambda)$, except it uses the matrices $\mathbf{D}_0, \mathbf{D}_1$ it received from the challenger instead of sampling them itself. It samples $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \xleftarrow{R} \mathbb{Z}_q^{n \times m}$. Next, let $\boldsymbol{Q} = \{(\mathsf{gid}_1, A_1), \ldots, (\mathsf{gid}_Q, A_Q)\}$ be the set of secret key queries algorithm $\mathcal{A}$ makes (in the execution of $\mathsf{Samp}_{\mathcal{A}}$). For each $i \in [Q]$, define the mapping $\rho_i \colon [N_i] \to [Q]$ exactly as in the specification of $\mathsf{Hyb}_0$.
3. Algorithm $\mathcal{B}$ makes queries on inputs $\mathsf{gid}_1, \ldots, \mathsf{gid}_Q$. Let $y_1, \ldots, y_Q \in \mathbb{Z}_q$ be the responses.
4. Algorithm $\mathcal{B}$ samples $\mathbf{u}_{1,i} \xleftarrow{R} \mathbb{Z}_q^m$ for each $i \in [\ell]$ and $\mathbf{u}_2 \xleftarrow{R} \mathbb{Z}_q^{m+1}$.
5. For each $i > d$, algorithm $\mathcal{B}$ samples $\hat{\mathbf{u}}_i \xleftarrow{R} \mathbb{Z}_q^m$. For $i \geq d$, it also samples $\hat{u}'_i \xleftarrow{R} \mathbb{Z}_q$. Then, for each $i \in [\ell]$ and $j \in [N_i]$, it constructs $v_{i,j}$ as follows:
   - If $i < d$, it samples $v_{i,j} \xleftarrow{R} \mathbb{Z}_q$.
   - If $i = d$, it sets $v_{d,j} \leftarrow \hat{u}'_d + y_{\rho_d(j)}$.
   - If $i > d$, it sets $v_{i,j} \leftarrow \hat{u}'_i + \hat{\mathbf{u}}^{\mathsf{T}}_i \mathsf{H}\big(\mathsf{gid}_{\rho_i(j)}\big)$.

   Finally, for each $i \in [\ell]$, it samples $\mathbf{e}_{3,i} \leftarrow D^{N_i}_{\mathbb{Z},\chi}$ and sets $\mathbf{u}^{\mathsf{T}}_{3,i} = [v_{i,1} \mid \cdots \mid v_{i,N_i}] + \mathbf{e}^{\mathsf{T}}_{3,i}$. It gives the challenge $(\{(\mathbf{A}_i, \mathbf{u}^{\mathsf{T}}_{1,i})\}_{i\in[\ell]}, \mathbf{B}, \mathbf{u}^{\mathsf{T}}_2, \{\mathbf{u}^{\mathsf{T}}_{3,i}\}_{i\in[\ell]}, \mathsf{aux})$ to $\mathcal{D}$. Algorithm $\mathcal{B}$ outputs whatever $\mathcal{D}$ outputs.

By construction, the components $(\{(\mathbf{A}_i, \mathbf{u}^{\mathsf{T}}_{1,i})\}_{i\in[\ell]}, \mathbf{B}, \mathbf{u}^{\mathsf{T}}_2, \{\mathbf{P}_i\}_{i\in[\ell]}, \mathsf{aux})$ are distributed exactly as in $\mathsf{Hyb}^{(1)}_{3,d}$ and $\mathsf{Hyb}^{(2)}_{3,d}$. Consider now the distribution of $\mathbf{u}_{3,i}$ that algorithm $\mathcal{B}$ induces:

- Suppose $y_i = \mathbf{s}^\intercal \mathsf{H}(\mathsf{gid}_i) + \hat{e}_i$ where $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^m$ and $\hat{e}_i \leftarrow D_{\mathbb{Z},\chi_{\mathsf{smudge}}}$. In this case, algorithm $\mathcal{B}$ perfectly simulates an execution of $\mathsf{Hyb}_{3,d}^{(1)}$ with $\hat{\mathbf{u}}_{2,d} = \mathbf{s}$ and $\hat{\mathbf{u}}_{2,i} = \hat{\mathbf{u}}_i$ for $i > d$.
- Suppose $y_i \xleftarrow{R} \mathbb{Z}_q$. Then, algorithm $\mathcal{B}$ perfectly simulates an execution of $\mathsf{Hyb}_{3,d}^{(2)}$ with $\hat{\mathbf{u}}_{2,i} = \hat{\mathbf{u}}_i$ for $i > d$.

Thus, algorithm $\mathcal{B}$ breaks security of the lattice-based PRF in Theorem 6.1 with advantage $\varepsilon$ and the claim follows. □

**Claim 6.20.** *For all distinguishers $\mathcal{D}$, $\mathsf{Hyb}_{3,d}^{(2)}(\mathcal{D}) \equiv \mathsf{Hyb}_{3,d+1}(\mathcal{D})$.*

*Proof.* This is a syntactic change and follows from the fact that $\rho_d$ is an injective function. Since each $r_{\rho_d(j)}$ is uniform and independent of all other components, the distribution of $v_{d,j}$ in $\mathsf{Hyb}_{3,d}^{(2)}$ for all $j \in [N_d]$ is uniform over $\mathbb{Z}_q$, which is precisely the distribution in $\mathsf{Hyb}_{3,d+1}$. □

Combining Claims 6.18 to 6.20, the output distributions of $\mathsf{Hyb}_3$ and $\mathsf{Hyb}_4$ are computationally indistinguishable. □

The claim follows by combining Lemmas 6.7 to 6.9 and 6.17. □

To complete the proof, we show that if there exists an adversary $\mathcal{A}$ that can distinguish between the main hybrids $\mathsf{Hyb}_0^{(\mathsf{main})}$ and $\mathsf{Hyb}_1^{(\mathsf{main})}$ in the proof of Theorem 6.4 with non-negligible advantage $\varepsilon$, then we can construct an efficient algorithm $\mathcal{B}$ such that $\mathsf{Adv}_{\mathcal{B}}^{(\mathsf{POST})}(\lambda) = \varepsilon$ in the evasive LWE assumption (Assumption 3.16) and with respect to the sampling algorithm $\mathsf{Samp}_{\mathcal{A}}$:

1. At the beginning of the game, algorithm $\mathcal{B}$ receives an evasive LWE challenge $(\{(\mathbf{A}_i, \mathbf{z}_{1,i}^\intercal)\}_{i\in[\ell]}, \mathbf{B}, \mathbf{z}_2^\intercal, \{\mathbf{K}_i\}_{i\in[\ell]}, \mathsf{aux})$ where $\mathbf{A}_i \in \mathbb{Z}_q^{n\times m}$, $\mathbf{z}_{1,i} \in \mathbb{Z}_q^m$, $\mathbf{B} \in \mathbb{Z}_q^{n\ell\times(m+1)}$, $\mathbf{z}_2 \in \mathbb{Z}_q^{m+1}$, $\mathbf{K}_i \in \mathbb{Z}_q^{m\ell\times N_i}$, and $\mathsf{aux} = (r, \mathbf{D}_0, \mathbf{D}_1)$.
2. Algorithm $\mathcal{B}$ starts running algorithm $\mathcal{A}$ with randomness $r$. Algorithm $\mathcal{A}$ outputs
   - A set of corrupted authorities $C \subset \mathcal{AU}$ and their public keys $\mathsf{pk}_{\mathsf{aid}} = (\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$ for each $\mathsf{aid} \in C$.
   - A list of non-corrupted authorities $\mathcal{N} \subseteq \mathcal{AU}$.
   - A list of secret-key queries $Q = \{(\mathsf{gid}_i, A_i)\}_{i\in[Q]}$ where each $A_i \subset \mathcal{N}$.
   - A pair of challenge messages $\mu_0, \mu_1 \in \{0,1\}$ and a set of authorities $A^* \subseteq C \cup \mathcal{N}$.
3. Let $A^* \cap \mathcal{N} = \{\mathsf{aid}_1^*, \ldots, \mathsf{aid}_\ell^*\}$. Algorithm $\mathcal{B}$ parses $\mathbf{B}$ as
$$\mathbf{B} = \left[\begin{array}{c|c} \mathbf{B}_{\mathsf{aid}_1^*} & \mathbf{p}_{\mathsf{aid}_1^*} \\ \vdots & \vdots \\ \mathbf{B}_{\mathsf{aid}_\ell^*} & \mathbf{p}_{\mathsf{aid}_\ell^*} \end{array}\right] \in \mathbb{Z}_q^{n\ell\times(m+1)},$$
   where $\mathbf{B}_{\mathsf{aid}_i^*} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{p}_{\mathsf{aid}_i^*} \in \mathbb{Z}_q^n$. In addition, for each $i \in [\ell]$, it sets $\mathbf{A}_{\mathsf{aid}_i^*} \leftarrow \mathbf{A}_i$.
4. Let $Q = \{(A_1, \mathsf{gid}_1), \ldots, (A_Q, \mathsf{gid}_Q)\}$. For each $i \in [Q]$, the challenger partitions $A_i = A_{i,\mathsf{chal}} \cup \bar{A}_{i,\mathsf{chal}} \subset \mathcal{N}$ where $A_{i,\mathsf{chal}} \subset A^*$ consists of the authorities appearing in the challenge ciphertext and $\bar{A}_{i,\mathsf{chal}} = A \setminus A_{i,\mathsf{chal}}$ consists of the authorities that do not appear in the challenge ciphertext.
5. Note that because $\mathcal{B}$ runs $\mathcal{A}$ with the *same* randomness as $\mathsf{Samp}_{\mathcal{A}}$, the queries $\mathcal{A}$ outputted in the invocation of $\mathsf{Samp}_{\mathcal{A}}$ exactly coincide with those in $\mathcal{B}$'s execution. Algorithm $\mathcal{B}$ now responds as follows:
   - **Public keys for non-corrupted authorities:** Algorithm $\mathcal{B}$ constructs the public keys for authorities in $\mathcal{N} \cap A^*$ and $\mathcal{N} \setminus A^*$ as follows:
     - For each $\mathsf{aid}_i^* \in \mathcal{N} \cap A^*$, algorithm $\mathcal{B}$ sets $\mathsf{pk}_{\mathsf{aid}_i^*} = (\mathbf{A}_{\mathsf{aid}_i^*}, \mathbf{B}_{\mathsf{aid}_i^*}, \mathbf{p}_{\mathsf{aid}_i^*})$.
     - For authorities $\mathsf{aid} \in \mathcal{N} \setminus A^*$, the challenger samples $(\mathbf{A}_{\mathsf{aid}}, \mathsf{td}_{\mathsf{aid}}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$, $\mathbf{p}_{\mathsf{aid}} \xleftarrow{R} \mathbb{Z}_q^n$ and $\mathbf{B}_{\mathsf{aid}} \leftarrow \mathbb{Z}_q^{n\times m}$. It sets the public key to $\mathsf{pk}_{\mathsf{aid}} = (\mathbf{A}_{\mathsf{aid}}, \mathbf{B}_{\mathsf{aid}}, \mathbf{p}_{\mathsf{aid}})$.

- **Secret keys:** We start by showing how to construct the secret keys corresponding to the honest authorities appearing in the challenge ciphertext from the components in $\mathbf{K}_1, \ldots, \mathbf{K}_\ell$:
  - By construction, $N_i \in [Q]$ is the number of indices $j \in [Q]$ where $\mathsf{aid}_i^* \in A_j$. Suppose authority $\mathsf{aid}_i^*$ is contained in the sets $A_{j_1}, \ldots, A_{j_{N_i}}$ for (sorted) indices $j_1, \ldots, j_{N_i} \in [Q]$. Define the mapping $\rho_i \colon [N_i] \to [Q]$ that maps $\ell \in [N_i] \mapsto j_\ell \in [Q]$. This is the definition from the proof of Claim 6.6.
  - For each $i \in [\ell]$ and $j \in [N_i]$, let $\mathsf{sk}_{\mathsf{aid}_i^*, \mathsf{gid}_{\rho_i(j)}} \leftarrow \mathbf{k}_{i,j}$ where $\mathbf{k}_{i,j} \in \mathbb{Z}_q^m$ denotes the $j^{\text{th}}$ column of $\mathbf{K}_i$.

  Then, for each $i \in [\ell]$ and authority $\mathsf{aid} \in \bar{A}_{i,\mathsf{chal}}$, the challenger computes $\mathbf{r}_{\mathsf{gid}_i} \leftarrow \mathsf{H}(\mathsf{gid}_i)$ and samples $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}_i} = \mathbf{u}_{\mathsf{aid},\mathsf{gid}_i} \leftarrow (\mathbf{A}_{\mathsf{aid}})_\chi^{-1}(\mathbf{p}_{\mathsf{aid}} + \mathbf{B}_{\mathsf{aid}}\mathbf{r}_{\mathsf{gid}_i})$ using the trapdoor $\mathsf{td}_{\mathsf{aid}}$ (which $\mathcal{B}$ sampled when constructing the public key). The challenger responds to the secret-key query $(\mathsf{gid}_i, A_i)$ with the set $\{\mathsf{sk}_{\mathsf{aid},\mathsf{gid}_i}\}_{\mathsf{aid} \in A_i}$.
- **Challenge ciphertext:** Algorithm $\mathcal{B}$ starts by parsing $\mathbf{z}_2^\mathsf{T} = [\hat{\mathbf{z}}_2^\mathsf{T} \mid z_3]$ where $\hat{\mathbf{z}}_2 \in \mathbb{Z}_q^m$ and $z_3 \in \mathbb{Z}_q$. For each $\mathsf{aid} \in A^* \cap C$, the challenger samples $\mathbf{s}_{\mathsf{aid}} \xleftarrow{\mathsf{R}} \mathbb{Z}_q^n$ and $\mathbf{e}_{1,\mathsf{aid}} \leftarrow D_{\mathbb{Z},\chi}^m$ and computes $\mathbf{c}_{1,\mathsf{aid}}^\mathsf{T} = \mathbf{s}_{\mathsf{aid}}^\mathsf{T}\mathbf{A}_{\mathsf{aid}} + \mathbf{e}_{1,\mathsf{aid}}^\mathsf{T}$. For $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$, it defines $\mathbf{c}_{1,\mathsf{aid}_i^*} \leftarrow \mathbf{z}_{1,i}$. It now constructs the challenge ciphertexts as

$$\mathsf{ct} = \left( \left\{ \mathbf{c}_{1,\mathsf{aid}}^\mathsf{T} \right\}_{\mathsf{aid} \in A^*}, \sum_{\mathsf{aid} \in A^* \cap C} \mathbf{s}_{\mathsf{aid}}^\mathsf{T}\mathbf{B}_{\mathsf{aid}} + \hat{\mathbf{z}}_2^\mathsf{T}, \sum_{\mathsf{aid} \in A^* \cap C} \mathbf{s}_{\mathsf{aid}}^\mathsf{T}\mathbf{p}_{\mathsf{aid}} + z_3 + \mu_0 \cdot \lfloor q/2 \rceil \right).$$

6. At the end of the game, algorithm $\mathcal{A}$ outputs a bit $b \in \{0,1\}$. Algorithm $\mathcal{B}$ outputs the same bit.

By construction of $\mathcal{B}$ and $\mathsf{Samp}_{\mathcal{A}}$, the public keys for the non-corrupted authorities are distributed exactly as in the real scheme (which corresponds to the distribution in $\mathsf{Hyb}_0^{(\mathsf{main})}$ and $\mathsf{Hyb}_1^{(\mathsf{main})}$). The secret-key queries are also distributed as in the real scheme. To verify this, consider again the $i^{\text{th}}$ secret-key query $(\mathsf{gid}_i, A_i)$ and once more partition $A_i = A_{i,\mathsf{chal}} \cup \bar{A}_{i,\mathsf{chal}}$ where $A_{i,\mathsf{chal}} = A_i \cap A^*$. Consider the secret-key components $\{\mathsf{sk}_{\mathsf{aid},\mathsf{gid}_i}\}_{\mathsf{aid} \in A_i}$ chosen by the challenger:

- When $\mathsf{aid} \in \bar{A}_{i,\mathsf{chal}}$, the secret key $\mathsf{sk}_{\mathsf{aid},\mathsf{gid}_i}$ is sampled exactly as in the real scheme (which coincides with the distribution in $\mathsf{Hyb}_0^{(\mathsf{main})}$ and $\mathsf{Hyb}_1^{(\mathsf{main})}$).
- For each $i \in [\ell]$ and $j \in [N_i]$, we have that $\mathsf{sk}_{\mathsf{aid}_i^*, \mathsf{gid}_{\rho_i(j)}} \leftarrow \mathbf{k}_{i,j}$. By construction of $\mathbf{k}_{i,j}$, this is equal to

$$\mathbf{k}_{i,j} \leftarrow (\mathbf{A}_i)_\chi^{-1}(\mathbf{p}_{\mathsf{aid}_i^*} + \mathbf{B}_{\mathsf{aid}_i^*} \cdot \mathsf{H}(\mathsf{gid}_{\rho_i(j)})),$$

  which is precisely the secret key distribution for the real scheme (which coincides with the distribution in $\mathsf{Hyb}_0^{(\mathsf{main})}$ and $\mathsf{Hyb}_1^{(\mathsf{main})}$).

It suffices to consider the distribution of the challenge ciphertext:

- Suppose $\mathbf{z}_{1,i}^\mathsf{T} = \mathbf{s}_i^\mathsf{T}\mathbf{A}_i + \mathbf{e}_{1,i}^\mathsf{T}$ and $\mathbf{z}_2^\mathsf{T} = \mathbf{s}^\mathsf{T}\mathbf{B} + \mathbf{e}_2^\mathsf{T}$ where $\mathbf{s}_i \xleftarrow{\mathsf{R}} \mathbb{Z}_q^n$, $\mathbf{s}^\mathsf{T} \leftarrow [\mathbf{s}_1^\mathsf{T} \mid \cdots \mid \mathbf{s}_\ell^\mathsf{T}]$, $\mathbf{e}_{1,i} \leftarrow D_{\mathbb{Z},\chi}^m$ and $\mathbf{e}_2 \leftarrow D_{\mathbb{Z},\chi}^{m+1}$. We can write $\mathbf{e}_2^\mathsf{T} = [\hat{\mathbf{e}}_2^\mathsf{T} \mid e_3]$ where $\hat{\mathbf{e}}_2 \in \mathbb{Z}_q^m$ and $e_3 \in \mathbb{Z}_q$. Then, by construction of algorithm $\mathcal{B}$, the following hold:
  - $\mathbf{c}_{1,\mathsf{aid}_i^*}^\mathsf{T} = \mathbf{z}_{1,i}^\mathsf{T} = \mathbf{s}_i^\mathsf{T}\mathbf{A}_{\mathsf{aid}_i^*} + \mathbf{e}_{1,i}^\mathsf{T}$ for each $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$.
  - $\hat{\mathbf{z}}_2^\mathsf{T} = \sum_{i \in [\ell]} \mathbf{s}_i^\mathsf{T}\mathbf{B}_{\mathsf{aid}_i^*} + \hat{\mathbf{e}}_2^\mathsf{T}$.
  - $z_3 = \sum_{i \in [\ell]} \mathbf{s}_i^\mathsf{T}\mathbf{p}_{\mathsf{aid}_i^*} + e_3$.

  This corresponds to a valid ciphertext where the secret keys associated with authority $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$ is $\mathbf{s}_i$. Moreover, the randomness $\mathbf{e}_{1,i}$, $\hat{\mathbf{e}}_2$, and $e_3$ are all distributed exactly as in the real scheme. The ciphertext components associated with the corrupted authorities $\mathsf{aid} \in A^* \cap C$ are simulated exactly as in the real scheme. This precisely coincides with the distribution in $\mathsf{Hyb}_0^{(\mathsf{main})}$.
- Suppose $\mathbf{z}_{1,i} \xleftarrow{\mathsf{R}} \mathbb{Z}_q^m$ for all $i \in [\ell]$ and $\mathbf{z}_2 \xleftarrow{\mathsf{R}} \mathbb{Z}_q^{m+1}$. In this case $\mathbf{c}_{1,\mathsf{aid}_i^*}$ is uniform and independent over $\mathbb{Z}_q^m$ for all $\mathsf{aid}_i^* \in A^* \cap \mathcal{N}$. Moreover, since $\mathbf{z}_2$ is uniform and independent of all other quantities, the second and third ciphertexts components are also uniform over $\mathbb{Z}_q^m$ and $\mathbb{Z}_q$, respectively. This precisely coincides with the distribution in $\mathsf{Hyb}_1^{(\mathsf{main})}$.

We conclude that $\mathsf{Adv}^{(\mathrm{POST})}_{\mathcal{B}}(\lambda) = \varepsilon$ in the evasive LWE assumption (Assumption 3.16). □

By an identical argument, we can show that under the evasive LWE assumption, for all adversaries $\mathcal{A}$, the output distributions of $\mathsf{Hyb}^{(\mathsf{main})}_1(\mathcal{A})$ and $\mathsf{Hyb}^{(\mathsf{main})}_2(\mathcal{A})$ are also computationally indistinguishable. Static security of Construction 6.2 holds. □

**Parameter setting.** Let $\lambda$ be a security parameter. We now instantiate Construction 6.2 as follows:

- Let the lattice dimension be $n = \lambda^{1/\varepsilon}$ for some constant $\varepsilon > 0$.
- We can set the length of the identities gid to be $\tau = \lambda$.
- For security (Theorem 6.4), we require that $\chi_{\mathsf{smudge}} > \lambda^{\lambda+\omega(1)}(m\chi_{\mathsf{PRF}})^{\lambda+1}$ and $\chi > \lambda^{\omega(1)}\ell\chi_{\mathsf{smudge}}$. Each of $\ell = \ell(\lambda), m = m(\lambda), \chi_{\mathsf{PRF}} = \chi_{\mathsf{PRF}}(\lambda)$ are polynomially-bounded. Thus, we can set $\chi = 2^{\tilde{O}(n^{\varepsilon})}$ to satisfy these requirements, where $\tilde{O}(\cdot)$ suppresses constant and logarithmic factors.
- To support arbitrary polynomial-size ciphertext policies, we set the bound $L = 2^{\lambda}$ in Theorem 6.3. To ensure correctness, we can set $m = O(n \log q)$ and $q = O(2^{\lambda} m\lambda\chi^2 + (\lambda m\chi_{\mathsf{PRF}})^{\lambda+1}\chi)$. Setting $q = 2^{\tilde{O}(n^{\varepsilon})}$ suffices to satisfy these requirements.

This yields the following corollary:

**Corollary 6.21** (Multi-Authority ABE for Subset Policies from Evasive LWE)**.** *Assuming polynomial hardness of LWE and of the evasive LWE assumption (both with a sub-exponential modulus-to-noise ratio), there exists a statically-secure multi-authority ABE for subset policies (of arbitrary polynomial size).*

## Acknowledgments

We thanks the TCC reviewers for helpful suggestions. B. Waters is supported by NSF CNS-1908611, a Simons Investigator award, and the Packard Foundation Fellowship. D. J. Wu is supported by NSF CNS-2151131, CNS-2140975, a Microsoft Research Faculty Fellowship, and a Google Research Scholar award.

## References

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

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

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC*, pages 99–108, 1996.

[BCD+16] Joppe W. Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the ring! practical, quantum-secure key exchange from LWE. In *ACM CCS*, pages 1006–1018, 2016.

[BCTW16] Zvika Brakerski, David Cash, Rotem Tsabary, and Hoeteck Wee. Targeted homomorphic attribute-based encryption. In *TCC*, pages 330–360, 2016.

[BDE+18] Jonathan Bootle, Claire Delaplace, Thomas Espitau, Pierre-Alain Fouque, and Mehdi Tibouchi. LWE without modular reduction and improved side-channel attacks against BLISS. In *ASIACRYPT*, pages 494–524, 2018.

[Bei96] Amos Beimel. *Secure Schemes for Secret Sharing and Key Distribution.* PhD thesis, Technion, 1996.

[BLMR13] Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic prfs and their applications. In *CRYPTO*, pages 410–428, 2013.

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *EUROCRYPT*, pages 719–737, 2012.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS*, pages 62–73, 1993.

[BTVW17] Zvika Brakerski, Rotem Tsabary, Vinod Vaikuntanathan, and Hoeteck Wee. Private constrained prfs (and more) from LWE. In *TCC*, pages 264–302, 2017.

[BV22] Zvika Brakerski and Vinod Vaikuntanathan. Lattice-inspired broadcast encryption and succinct ciphertext-policy ABE. In *ITCS*, pages 28:1–28:20, 2022.

[CC09] Melissa Chase and Sherman S. M. Chow. Improving privacy and security in multi-authority attribute-based encryption. In *ACM CCS*, pages 121–130, 2009.

[Cha07] Melissa Chase. Multi-authority attribute based encryption. In *TCC*, pages 515–534, 2007.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, pages 523–552, 2010.

[CVW18] Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. GGH15 beyond permutation branching programs: Proofs, attacks, and candidates. In *CRYPTO*, pages 577–607, 2018.

[DKW21a] Pratish Datta, Ilan Komargodski, and Brent Waters. Decentralized multi-authority ABE for DNFs from LWE. In *EUROCRYPT*, pages 177–209, 2021.

[DKW21b] Pratish Datta, Ilan Komargodski, and Brent Waters. Decentralized multi-authority ABE for nc^1 from computational-bdh. *IACR Cryptol. ePrint Arch.*, page 1325, 2021.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *STOC*, pages 467–476, 2013.

[GLW21] Rishab Goyal, Jiahui Liu, and Brent Waters. Adaptive security via deletion in attribute-based encryption: Solutions from search assumptions in bilinear groups. In *ASIACRYPT*, pages 311–341, 2021.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM CCS*, pages 89–98, 2006.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.

[Kim19] Sam Kim. Multi-authority attribute-based encryption from LWE in the OT model. *IACR Cryptol. ePrint Arch.*, page 280, 2019.

[LCLS08] Huang Lin, Zhenfu Cao, Xiaohui Liang, and Jun Shao. Secure threshold multi authority attribute based encryption without a central authority. In *INDOCRYPT*, pages 426–436, 2008.

[LW11] Allison B. Lewko and Brent Waters. Decentralizing attribute-based encryption. In *EUROCRYPT*, pages 568–588, 2011.

[LW15] Vadim Lyubashevsky and Daniel Wichs. Simple lattice trapdoor sampling from a broad class of distributions. In *PKC*, pages 716–730, 2015.

[MKE08] Sascha Müller, Stefan Katzenbeisser, and Claudia Eckert. Distributed attribute-based encryption. In *ICISC*, pages 20–36, 2008.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pages 700–718, 2012.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93, 2005.

[RW15] Yannis Rouselakis and Brent Waters. Efficient statically-secure large-universe multi-authority attribute-based encryption. In *Financial Cryptography and Data Security*, pages 315–332, 2015.

[SW05] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473, 2005.

[Tsa19] Rotem Tsabary. Fully secure attribute-based encryption for t-CNF from LWE. In *CRYPTO*, pages 62–85, 2019.

[Tsa22] Rotem Tsabary. Candidate witness encryption from lattice techniques. In *CRYPTO*, 2022.

[VWW22] Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. Witness encryption and null-IO from evasive LWE. In *ASIACRYPT*, 2022.

[Wee22] Hoeteck Wee. Optimal broadcast encryption and CP-ABE from evasive lattice assumptions. In *EUROCRYPT*, 2022.

[WFL19] Zhedong Wang, Xiong Fan, and Feng-Hao Liu. FE for inner products and its application to decentralized ABE. In *PKC*, pages 97–127, 2019.