Adaptively-Secure Flexible and Identity-Based Broadcast Encryption from Lattices

Rishab Goyal
UW-Madison*

Saikumar Yadugiri
UW-Madison†

Abstract

Broadcast encryption (BE) allows a sender to succinctly encrypt a message to any dynamically chosen subset of recipients. The gold-standard for BE is *optimal succinctness* (parameters independent of the number of users) and *adaptive security*. Attaining both from falsifiable post-quantum assumptions has been a central open problem. Recently, Goyal and Yadugiri (GY) [GY26b] gave the first adaptively-secure and optimally-succinct slotted distributed BE under a falsifiable lattice assumption, but their techniques inherently require an a-priori bound on the number of users and a slotted user structure. Two highly-sought-after generalizations thus remained open: *flexible BE* (FBE), where users asynchronously sample and register their own keys and *identity-based BE* (IBBE), where a trusted authority issues keys for identities drawn from a super-polynomially large space.

In this work, we present the first adaptively-secure FBE and IBBE schemes with all parameter sizes independent of the number of users, both under the same falsifiable lattice assumption (decomposed LWE) and in the same model (Random Oracle Model) as the prior state-of-the-art for slotted distributed BE. Our FBE additionally enjoys a *transparent setup*, in line with the trustless ethos motivating distributed and flexible BE. At the technical heart of our results, we extend the equivocal encryption framework of GY to capture *unbounded* and *dynamic* broadcast systems, and introduce *Equivocal Matrix Commitments*—a strengthening of matrix commitments that supports adaptive equivocation of the committed matrix. We expect this new abstraction to find broader applications in designing adaptively-secure trustless lattice-based encryption.

*Email: rishab@cs.wisc.edu.
†Email: saikumar@cs.wisc.edu.

Contents

1 Introduction **1**
- 1.1 Our Contributions . . . 2

2 Technical Overview **5**
- 2.1 Flexible Broadcast Encryption . . . 5
- 2.2 Identity-Based Broadcast Encryption . . . 10

3 Preliminaries **12**
- 3.1 Lattice Preliminaries . . . 13
- 3.2 ℓ-Decomposed LWE . . . 14

4 Equivocal Matrix Commitment Scheme **15**
- 4.1 Preliminaries . . . 16
- 4.2 Construction . . . 18
- 4.3 Security Analysis . . . 19

5 (Equivocal) Flexible Broadcast Encryption **20**
- 5.1 Flexible Broadcast Encryption . . . 20
- 5.2 Equivocal Flexible Broadcast Encryption . . . 21
- 5.3 Semi-Static FBE from EqFBE . . . 22

6 Equivocal Flexible Broadcast Encryption from Decomposed LWE **23**
- 6.1 Fake Setup, Key Generation Simulation, and Equivocation . . . 25
- 6.2 Mode Indistinguishability Analysis . . . 25
- 6.3 Selective Security for Fake Setup Analysis . . . 26

7 (Equivocal) Identity-Based Broadcast Encryption **26**
- 7.1 Identity-Based Broadcast Encryption . . . 26
- 7.2 Equivocal Identity-Based Broadcast Encryption . . . 27
- 7.3 IBBE from EqBE . . . 28

8 Equivocal Identity-Based Broadcast Encryption from Decomposed LWE **29**
- 8.1 Tools Required: Trapdoored Equivocal Matrix Commitment Schemes and Back-Samplable EqFBE . . . 29
- 8.2 Construction . . . 30
- 8.3 Fake Setup, Key Generation Simulation, and Equivocation . . . 32
- 8.4 Mode Indistinguishability Analysis . . . 32
- 8.5 Semantic Security Analysis . . . 33

A Additional Proofs **39**

B Proof of Theorem 4.5 **46**

C Proof of Theorem 5.6 **54**

D Proofs from Section 6 **56**
D.1 Proof of Theorem 6.2 56
D.2 Proof of Theorem 6.3 60

E Proof of Theorem 7.4 **62**

F Proofs from Section 8 **64**
F.1 Proof of Theorem 8.6 64
F.2 Proof of Theorem 8.7 70

1 Introduction

Broadcast Encryption (BE) [FN94] is a fundamental cryptographic primitive that allows a sender to encrypt a message to a dynamically chosen subset of recipients, where both the ciphertext size and the sender's running time are far smaller than what would be required to encrypt and send the message to each recipient individually. Since its inception in the early 90s, broadcast encryption has emerged as a powerful tool for guaranteeing message privacy with succinctness in multi-receiver systems. In the early days, broadcast encryption served as a key enabler for media distribution (most notably, in the AACS standard) and pay-TV systems. In recent years, its applications have been further explored in the context of cloud storage and emerging IoT technologies. Notably, Cloudflare has explored the applicability of broadcast encryption for fine-grained cloud access control [Clo20].

Ideally, one wants the ciphertext size, as well as the sender's running time, to be independent of the number of recipients. And, for security, one desires that even an attacker that can arbitrarily and adaptively corrupt recipients cannot decrypt any ciphertext whose intended recipients are uncorrupted. Jointly, optimal succinctness and adaptive security correspond to the gold-standard guarantees for broadcast encryption.

In recent years, there has been significant interest in designing broadcast encryption in a completely trustless setting, where every receiver samples its own public keys and there is no centrally trusted party that holds a master secret key. Such trustless broadcast encryption systems were initially formalized in the "ad-hoc" or *distributed setup* model [WQZD10, BZ14], and most recently, they were generalized to the fully *flexible setup* model [FWW23], where there is no a-priori bound on the number of users, nor is there any synchronization needed amongst different receivers.

The post-quantum frontier. Despite great potential, there are only a handful of approaches for building broadcast encryption and its trustless generalizations. Early constructions [BGW05, BW06, GW09, Del07] (non-exhaustive list) relied on bilinear pairings but incurred large public keys (growing with the number of users). Later, using heavy hammers such as code obfuscation, multilinear maps, functional encryption, and witness encryption, new constructions were proposed that achieved optimal parameter sizes [BZ14, GVW19, FWW23]. Unfortunately, we do not yet have any provable candidates for such objects from falsifiable post-quantum assumptions. Given the rapidly rising threat of quantum computing, as highlighted by NIST and industry estimates, the urgency of transitioning to and developing post-quantum constructions for broadcast encryption is extremely high. To address this gap, new approaches for designing broadcast encryption and its trustless generalizations from lattices have been proposed in a sequence of recent exciting works [Wee24, CW24, Wee25, CHW25, AMR25, WW25]. While earlier works relied on non-falsifiable lattice assumptions which have not resisted cryptanalytic attacks, more recent works prove security under new falsifiable lattice assumptions called succinct [Wee24] or decomposed LWE [AMR25], that mirror the "q-type" assumptions popularly considered in the pairing-based literature.

The adaptive security challenge. Unfortunately, designing broadcast encryption with (or without) a distributed setup, while simultaneously providing optimal succinctness and adaptive security, has been an incredibly challenging problem. Until very recently, there was no broadcast encryption in any trustless model that satisfied optimal succinctness and was adaptively secure under the hardness of a falsifiable lattice assumption. In a recent work, Goyal and Yadugiri [GY26b] broke through this barrier and designed the first post-quantum distributed broadcast encryption (DBE) with optimal parameters under the decomposed LWE assumption in the random oracle model. A

key technical innovation of their work was the introduction of a new framework for proving adaptive security in lattice-based schemes, which they called an '*Equivocal Encryption System*'. Their work filled an important gap in our understanding of adaptive security for advanced encryption systems.

The bounded-users barrier. Despite this significant progress, a crucial limitation persists in the state of the art. Concretely, the systems designed in [GY26b] cannot support an unbounded number of users, since the CRS as well as all keys have to be sampled with respect to an a-priori bound N on the number of users that can ever join. This also affects running times, since growing the user base on-the-fly incurs a prohibitive overhead. Moreover, their design inherently relies on a slotted structure where every user is pre-assigned an index, and encryption is not well-defined if two users share the same slot. We call such systems slotted DBE. Relaxing only the slot structure gives *unslotted* DBE, where key generation is no longer tied to any index, but where the parameter(s) still grow with an a-priori bound. For instance, the CRS in [WW26] grows with the bound on the size of the broadcast set. As a result, neither the [GY26b] system, nor any preceding lattice-based work, captures the following two highly-sought-after settings:

Flexible Broadcast Encryption (FBE): FBE [FWW23] removes both restrictions at once, avoiding the slotted structure while keeping every parameter independent of any bound. Here, each user simply samples a public-secret key pair, and registers its public key on a public bulletin board. During encryption, a sender can then arbitrarily pick an ordered list of public keys from the bulletin board as the intended recipients. Notably, unlike unslotted DBE, no system parameter is fixed in advance, and unlike slotted DBE, users never pre-commit to specific slots or coordinate during key generation.

Identity-Based Broadcast Encryption (IBBE): IBBE [Del07, SF07, GW09] is the centralized analog of FBE. Here, a trusted authority samples a master public-secret key pair, and issues secret keys for identities using the master secret key. An encryptor can dynamically pick any subset of recipient identities (from an exponentially large identity space) and succinctly encrypt to them given just the master public key. IBBE is an extremely well-studied primitive, but existing constructions rely on either pairings or heavy objects such as obfuscation, which is not known under falsifiable post-quantum assumptions.

In this work, we address both the above challenges, and build on the Equivocal Encryption framework to resolve the following open problem:

Can we design adaptively-secure and fully-succinct, flexible as well as identity-based, broadcast encryption from falsifiable lattice assumptions?

1.1 Our Contributions

We answer the above question affirmatively. We extend the Equivocal Encryption framework introduced by [GY26b], and generalize it to capture *unbounded* and *dynamic* broadcast encryption systems. In particular, we design the following adaptively-secure broadcast encryption systems with optimal parameters under the same hardness assumption as the prior state-of-the-art. We summarize our results and compare them with the state of the art in Table 1.

Result 1 – Flexible Broadcast Encryption. Our first broadcast encryption scheme is designed in the flexible model [FWW23], where receivers can arbitrarily join and leave the system, and can simply register their public keys on a public bulletin board. All system parameters in our FBE

are independent of the total number of receivers and senders. That is, it is a truly unbounded broadcast encryption system without requiring a trusted or synchronized setup. In words, our scheme has a *transparent setup* which is in line with the main motivation of trustlessness behind distributed/flexible broadcast encryption.

Theorem 1.1 (Informal)**.** Assuming hardness of decomposed LWE, there exists an adaptively-secure flexible broadcast encryption scheme, with all parameter sizes independent of the total number of users, in the Random Oracle Model.

Result 2 – Identity-Based Broadcast Encryption. Our second broadcast encryption scheme is designed in the traditional model where a trusted authority samples the global system keys. In this model, we design an identity-based broadcast encryption system, where there is no a-priori bound on the maximum number of identities in the system, and an encryptor can dynamically select any subset of recipient identities that it wants to encrypt to. All parameter sizes in our system are a fixed polynomial in the security parameter (and the identity space index).

Theorem 1.2 (Informal)**.** Assuming hardness of decomposed LWE, there exists an adaptively-secure identity-based broadcast encryption scheme, with all parameter sizes independent of the total number of identities or recipient set size, in the Random Oracle Model.

Both our broadcast encryption schemes are proven adaptively secure under the decomposed LWE assumption [AMR25] in the Random Oracle Model [BR93]. Our results are plausibly post-quantum secure, and rely on the same lattice assumption as the prior state-of-the-art, requiring no new hardness assumptions.

We remark that an optimal flexible broadcast encryption scheme does not generically imply an identity-based broadcast encryption scheme. Although a generic compilation is possible (by a-priori allocating one slot per identity), it would lead to massive public key sizes and inefficient setup (growing with the size of the identity space). Thus, an optimal IBBE requires new technical insights beyond optimal FBE.

As already noted by Gentry-Waters [GW09], their compiler for semi-static to adaptive security does not extend to IBBE because the reduction must commit to a subset of the user universe, which is super-polynomially large in IBBE. We must therefore design and prove our IBBE adaptively secure directly (more details in Section 2.2). This motivates our next contribution, a stronger equivocal encryption framework.

Strong equivocal encryption framework. An important contribution of our work is to introduce and instantiate a new stronger variant of equivocal encryption framework that we use to obtain our results. The main strengthening to the framework we introduce is an explicit SimKGen algorithm (see Section 7.2) that also enables adaptive equivocation of public keys. We instantiate this stronger notion of key simulation interface, and this is exactly what enables direct adaptive security for IBBE. The reduction can answer the adversary's adaptive secret-key queries on the fly, without ever committing in advance to a set of honest identities.

Aggregated public keys. An additional feature of our FBE scheme is that the encryption and decryption times can be driven down, to be independent of the number of users in the recipient set, by pre-processing the public keys into a short *aggregated master public key*. That is, any party can locally compress the list of public keys of all intended recipients into a short master public key (with associated per-user decryption hints), using which encryption and decryption can be performed more efficiently. This is similar in spirit to the pre-processing step considered in

registered ABE [GV20, HLWW23] and distributed monotone-policy encryption [AGY26, WW26]. The resulting FBE has all parameters, including encryption and decryption time, independent of the number of recipients.

On removing random oracles. Our results are proven secure in the Random Oracle Model [BR93]. Removing random oracles while retaining optimal succinctness, adaptive security, and transparent setup, is an exciting direction for future work. For slotted distributed broadcast encryption, this was recently addressed in [GY26b] by introducing the notion of a Programmable Pseudorandom Set Sampler (PPS). Extending their approach to the flexible and identity-based settings, while retaining fully-transparent setup for FBE, is a fascinating open problem.

Reference	Assumption	Type	$\lvert\mathsf{crs}\rvert$	$\lvert\mathsf{pk}\rvert$	$\lvert\mathsf{sk}\rvert$	$\lvert\mathsf{ct}\rvert$	TP	PQ	AD
[BZ14]	$i\mathcal{O}$ + OWF	slotted DBE	–	1	1	1	✓	–	✓
[FWW23]	WE + LWE	FBE	1	1	1	1	✓	–	✓
[KMW23]	BDHE	slotted DBE	N	N	1	1	✗	✗	✓
[GLWW23]	BDHE	unslotted DBE	N	N	1	1	✗	✗	✓
[GKPW24]	Generic Bilinear Group	unslotted DBE	N	N	1	1	✗	✗	✗
[WW26]	N-Extended BDHE	unslotted DBE	N	N	1	1	✗	✗	✗
[CW24]	succinct LWE	slotted DBE	N^2	N	1	1	✗	✓	✗
[CHW25]	succinct LWE	slotted DBE	N^2	N	1	1	✗	✓	✓
[AMR25]	decomposed LWE	unslotted DBE	N	1	1	1	✓	✓	✗
[WW25]	decomposed LWE	slotted DBE	1	1	1	1	✓	✓	✗
[GY26b]	decomposed LWE	slotted DBE	1	1	1	1	✓	✓	✓
Corollary 6.4	decomposed LWE	FBE	1	1	1	1	✓	✓	✓
Corollary 8.8	decomposed LWE	IBBE	1	1	1	1	–	✓	✓

Table 1: Comparison of our results with prior works on (distributed) broadcast encryption. $\lvert\mathsf{crs}\rvert, \lvert\mathsf{pk}\rvert, \lvert\mathsf{sk}\rvert$, and $\lvert\mathsf{ct}\rvert$ denote the sizes of the common random string / master public key, user public keys, user secret keys, and ciphertexts respectively. For slotted DBE, N denotes the number of users in the system and for unslotted DBE, N is a bound on the maximal size of recipient set. The values reported in these columns hide $\mathsf{poly}(\lambda, \log N)$ terms. For instance, if $\lvert\mathsf{crs}\rvert$ is reported as X, it signifies $\lvert\mathsf{crs}\rvert = X \cdot \mathsf{poly}(\lambda, \log N)$. **TP**, **PQ**, and **AD** denote transparent setup, post-quantum, and adaptive security respectively. All of the lattice-based adaptively secure schemes, including ours, are in the Random Oracle Model. We mark the Type column as DBE for slotted distributed BE, FBE for flexible BE, and IBBE for identity-based BE. BDHE is the Bilinear Diffie-Hellman Exponent assumption [Wat11]

Related Work. The notion of FBE was introduced by Freitag, Waters, and Wu [FWW23], who gave an adaptively-secure construction in the random oracle model [BR93] from witness encryption [GGSW13], a non-falsifiable assumption. While there are multiple approaches to design slotted and unslotted DBE with adaptive security from a variety of assumptions (see Table 1), the scheme of [FWW23] remained the only FBE prior to our work.

IBBE generalizes broadcast encryption to a superpolynomial number of users and was proposed by Delerablée [Del07] and by Sakai and Furukawa [SF07]. Gentry and Waters [GW09] gave the first adaptively secure construction under the DBDHE assumption [BBG05], with compact ciphertexts in the random oracle model and sub-linear size ciphertexts in the plain model. Agrawal and Yamada [AY20] constructed an adaptively secure IBBE from standard LWE in the random oracle and bilinear generic group models.

Comparison to Concurrent Work. In an independent and concurrent work, Abram, Malavolta, and Roy [AMR26a] construct compact shift-hiding PRFs [PS18], pseudorandom obfuscation [BDJ+25], and broadcast encryption from decomposed LWE. We provide a detailed comparison

below.

- **Adaptive Security and Parameter Succinctness.** Our work achieves adaptively secure FBE in the Random Oracle Model with fully succinct parameters that are independent of the number of users. The prior version of [AMR26a, ePrint 2026/741] claimed the same result, as well as an adaptively secure DBE in the standard model with a CRS whose size grows linearly with the number of users. However, that version contains an error in its parameter analysis. Its shift-hiding PRF is instantiated on inputs of length $N^2 \cdot \mathsf{poly}(\lambda, \log N)$, where N bounds the broadcast-set size, and the modulus placed in the CRS grows linearly with this input length. Consequently, its FBE CRS grows as $\mathsf{poly}(N)$, making the scheme an unslotted DBE rather than an FBE, and the claimed linear CRS bound for its standard-model DBE also does not hold. Both works rely on decomposed LWE.
- **Fast Encryption and Decryption.** We explicitly show that our schemes support public-coin preprocessing, which compresses the public keys into a fixed-size digest and makes encryption and decryption independent of the number of recipients. Although [AMR26a] does not discuss this property, its construction can support similar preprocessing using the structure of its shift-hiding functions.
- **Identity-Based Broadcast Encryption.** We additionally construct IBBE with optimal parameters in the Random Oracle Model. As discussed in Section 2.2, a direct adaptation of FBE to the identity-based setting generally results in large public keys. The framework of [AMR26a] does not provide IBBE.
- **Technical Foundations and Additional Results.** Our construction uses Equivocal Matrix Commitments to achieve adaptive security, while [AMR26a] develops compact shift-hiding functions. Their work also gives pseudorandom obfuscation for puncturable PRFs assuming indistinguishability obfuscation and proposes a heuristic candidate for indistinguishability obfuscation.

Subsequent Work. A later revision of Abram, Malavolta, and Roy [AMR26b] repairs the parameter issue using trapdoor hashing [DGI+19] and obtains an adaptively secure FBE scheme with optimal parameters from decomposed LWE in the Random Oracle Model. Their underlying semi-statically secure FBE scheme is in the standard model, while adaptive security follows by applying the transformation of [FWW23]. In another subsequent work, Goyal and Yadugiri [GY26a] use our EqMCOM primitive for distributed monotone-policy encryption and silent threshold encryption, further demonstrating the versatility of our tool.

2 Technical Overview

In this section, we provide a high-level technical overview of our results. We start by describing our flexible broadcast encryption (FBE) construction. Later, in Section 2.2, we use the algebraic structure of our FBE as a springboard to build our identity-based broadcast encryption (IBBE) scheme. Throughout this overview, we focus primarily on conveying the main technical ideas rather than precise definitions or parameter choices.

2.1 Flexible Broadcast Encryption

A Flexible Broadcast Encryption (FBE) [FWW23] scheme consists of four algorithms ($\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}$), and is syntactically similar to distributed broadcast encryption (DBE) [WQZD10, BZ14],

with one crucial difference. In FBE, the key generation algorithm KGen is not tied to any specific slot or user index. Each user simply samples a fresh $(\mathsf{pk}, \mathsf{sk})$ pair from a shared CRS. To encrypt, Enc takes an arbitrary ordered list of public keys $(\mathsf{pk}_1, \ldots, \mathsf{pk}_N)$ and a message μ, and outputs a ciphertext ct. To decrypt, Dec takes $(\mathsf{pk}_1, \ldots, \mathsf{pk}_N, \mathsf{sk}_i, \mathsf{ct})$ for some $i \in [N]$ and outputs μ'. Thus, FBE sidesteps the "slotted" structure of DBE entirely, and allows users to asynchronously join a system of arbitrary size without having to pre-commit to a unique slot.

As discussed earlier, the desired properties are compactness (all parameter sizes are a fixed polynomial in the security parameter λ, and in particular independent of N), and adaptive security, where an attacker can create and corrupt an a-priori unbounded number of $(\mathsf{pk}, \mathsf{sk})$ pairs and, at the end, challenge the system on a set of public keys none of which were corrupted. Full definitions are provided in Section 5.1. We also consider the weaker notion of *semi-static* security, which is the same as adaptive security, except that the attacker is not allowed to make any key corruption queries. Prior works [GW09, FWW23] have shown that a semi-statically secure FBE can be generically compiled into an adaptively secure one via the Gentry and Waters [GW09] two-key compiler. Due to this generic compiler, we focus on designing a *semi-statically secure* FBE, and generically boost it to adaptive security using the two-key compiler.

Our approach: the Equivocal framework, lifted to unbounded users. Our starting point is the recent equivocal encryption framework of Goyal and Yadugiri [GY26b] (GY from henceforth). At a high level, GY designs a (bounded-user, slotted) DBE by following a two-step template:

(1) Define a strengthening of DBE called Equivocal DBE (EqDBE), and prove generically that any EqDBE gives an adaptively-secure DBE.
(2) Design EqDBE from the decomposed LWE assumption, in the Random Oracle Model.

We too follow this broad template. Namely, we first define an *Equivocal Flexible Broadcast Encryption* (EqFBE) scheme, prove that any EqFBE yields a semi-statically secure FBE (which then can be compiled into an adaptively secure FBE), and finally construct EqFBE from decomposed LWE. Crucially, however, extending this template to the unbounded setting of FBE is not immediate. Before explaining the challenges and our ideas, let us first recall the equivocal framework of GY at an abstract level.

Reviewing Equivocal Encryption Systems. Consider a public-key encryption system where we would like to prove *adaptive* security with respect to some challenge input τ^* that an attacker can choose at the very end of the experiment. Abstractly, an 'Equivocal' version of this system introduces a 'fake mode' sampler in which:

- A "fake" CRS, together with a "fake" ciphertext $\widetilde{\mathsf{ct}}$ (generated for a *dummy* challenge input) and an auxiliary trapdoor aux, can be jointly sampled such that the fake CRS is indistinguishable from the real CRS.
- Later, given any adaptively chosen τ^*, one can use aux to *equivocate* $\widetilde{\mathsf{ct}}$ into a real-looking ciphertext ct^* w.r.t. τ^*.

An Equivocal Encryption System is useful for proving adaptive security as long as it satisfies two properties: (a) *mode indistinguishability*, i.e., the distribution of a real CRS/ciphertext pair is indistinguishable from a fake CRS and equivocated ciphertext (for *any* adaptively chosen τ^*); and (b) *selective security for fake setup*, i.e., the fake ciphertext on its own still hides the message even when the attacker has access to the auxiliary trapdoor aux. Taken together, these two properties

simplify the task of proving adaptive security, into proving (a) an indistinguishability statement about equivocability, and (b) a selective-looking security statement about fake ciphertexts. For further details, we refer the reader to [GY26b].

Although in GY, the framework was only instantiated for the (bounded-user, slotted) DBE setting, it is very general. Moving to the unbounded and flexible setting, we extend the framework as follows:

- **Unbounded key generation.** Since FBE does not take an upfront bound N on the number of users, the EqFBE fake setup $\mathsf{FakeSetup}$ only takes the security parameter and a message, and no longer takes N or any set as input. Public keys are instead generated on-the-fly through an explicit key-generation simulator $\mathsf{SimKGen}$, which takes aux and a state, and outputs a simulated public key (while maintaining consistency with aux).
- **Dynamic equivocation.** The $\mathsf{Equivocate}$ algorithm now takes, as input, an adaptively chosen subset S of public keys previously generated by $\mathsf{SimKGen}$, and equivocates $\widetilde{\mathsf{ct}}$ to a real-looking ciphertext under this dynamically chosen S. Crucially, the size of $\widetilde{\mathsf{ct}}$ must be independent of the number of simulated public keys.
- **Dynamic mode indistinguishability game.** Mode indistinguishability is now stated in terms of an interactive game, where an attacker makes polynomially many key-generation queries, then chooses the challenge set $S \subseteq [h]$ for some handle counter h (where h is not bounded a-priori), and only at that point receives the (equivocated) challenge ciphertext.

After making these extensions, one can generalize the GY template to prove that an EqFBE implies a semi-statically secure FBE. Combining with the two-key compiler of [GW09, HWW25], we obtain an adaptively secure FBE. A full definition of our EqFBE abstraction is in Definition 5.4, and the transformation in Theorem 5.6.

Designing EqFBE: why the GY system fails for unbounded users. In the GY construction, the CRS contains an LWE matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ alongside matrix commitment public parameters $\mathsf{com.pp} = (\mathbf{W}, \mathbf{D}, \mathbf{B})$ [Wee25]. A user i's public-secret key pair is $(\mathbf{t}_i, \mathbf{r}_i)$, where $\mathbf{t}_i = \mathbf{B}\mathbf{r}_i + \mathbf{p} - \mathbf{A}\mathbf{v}_i$, $\mathbf{r}_i \leftarrow \{\pm 1\}^{m'}$[1], and $\mathbf{v}_i$ is the i-th column of the verification matrix $\mathbf{V}_N$ associated with the commitment scheme for a universe of size N. Here, $\mathbf{v}_i$ effectively acts as a "slot encoding" for user i within a pre-declared universe of N users. During Enc (for a set $S \subseteq [N]$), the sender builds a matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$ whose j-th column is $\mathbf{t}_j$ if $j \in S$ and random otherwise, computes $\mathbf{C} = \mathsf{Com}(\mathbf{M})$, and then outputs roughly $(\mathbf{c}_1^\top := \mathbf{s}^\top \mathbf{B} + \mathsf{err}, \mathbf{c}_2^\top := \mathbf{s}^\top(\mathbf{A} + \mathbf{C}) + \mathsf{err}, c_3 := \mathbf{s}^\top \mathbf{p} + \mathsf{err} + \lfloor q/2 \rfloor \mu)$ along with auxiliary random masks needed for equivocation. GY then programs $\mathbf{A}$ in the security proof so that the committed matrix $\mathbf{M}$ can be *equivocated* to any subset S^* of a pre-declared set S'.

There are two fundamental reasons why this design is inherently limited to a bounded (in particular, polynomially-bounded) user base:

1. **Slot encoding tied to N.** Each user's public key is specifically designed using a slot encoding $\mathbf{v}_i$ that depends on N. Concretely, GY requires that N is polynomially-bounded so that the global commitment matrix and the associated slot encodings fit into the LWE matrix programming step. Going beyond a polynomial N, even with the succinct commitment techniques of [WW25], appears to break the matrix programming argument used for proving selective security of the fake-mode system.

[1] $m' = \mathsf{poly}(m)$. We use this notation through out this overview. For the exact value, check Assumption 3.7.

2. **Rigid equivocation window.** GY's equivocation is performed by per-user "one-time-pad"-style masking of specific columns within the committed matrix $\mathbf{M}$. Under this paradigm, the set of equivocatable columns is effectively *pre-declared* in the fake setup (i.e., the set S'), after which the system can equivocate to any $S^* \subseteq S'$, but not beyond. This rigidity is incompatible with the FBE setting where the attacker may request an a-priori unbounded number of public keys before committing to any challenge set.

Our key ideas. To overcome these obstacles, we restructure the design in three crucial ways:

1. **Commit only to the intended recipients.** Rather than committing to a globally-sized matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$ with slot-aligned columns, we have Enc directly commit to the matrix $\mathbf{M} = [\mathbf{t}_1 \mid \cdots \mid \mathbf{t}_{|S|}]$ of only the intended recipients' public keys, with no padding. This decouples encryption from any a-priori universe bound.
2. **Fully equivocating the commitment matrix.** Instead of the per-column one-time-pad masking approach of GY, we need to make the commitment matrix *fully* equivocal. That is, the committed matrix $\mathbf{C}$ in the fake mode should be openable to any matrix of public keys that later emerges from adaptively queried keys, not just a pre-declared superset.
3. **ROM-based compression of equivocation randomness.** For decryption correctness, the equivocator needs to reveal enough randomness so that the decryptor can reconstruct the committed matrix from the ciphertext. A-priori, this requires revealing $\Theta(N)$-size random masks, which would blow up the ciphertext size. We get around this by deriving all this randomness via a random oracle applied to a short seed included in the ciphertext. This way, the ciphertext stays compact, *and* the reduction can program the random oracle in the fake mode to adaptively explain the randomness needed to equivocate.

Our FBE construction: an Equivocal Matrix Commitment. The technical heart of our FBE is a new primitive we call an *Equivocal Matrix Commitment* (EqMCOM), which strengthens the matrix commitment scheme (MCOM) of [Wee25] with a novel equivocation interface. In MCOM, there exist three *deterministic* algorithms $\mathsf{Com}, \mathsf{Verify}, \mathsf{Open}$ which take $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ implicitly. Com takes as input $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$ and produces a short commitment $\mathbf{C}_\mathbf{M} \in \mathbb{Z}_q^{n \times m}$ such that $\mathbf{C}_\mathbf{M}\mathbf{V}_N = \mathbf{M} - \mathbf{B}\mathbf{Z}_\mathbf{M}$ where $\mathbf{Z}_\mathbf{M} = \mathsf{Open}(\mathbf{M})$ and $\mathbf{V}_N = \mathsf{Verify}(1^N)$ are low norm matrices.

EqMCOM augments this scheme by randomizing the commitment algorithm Com and introducing a new EqCom algorithm for equivocation. The scheme guarantees that for a matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$, given a sufficiently wide enough pre-image $\mathbf{Y}$ for $\mathbf{M}$ w.r.t. $\mathbf{B}$ (i.e., $\mathbf{M} = \mathbf{B}\mathbf{Y}$), any random $\mathbf{C}' \leftarrow \mathbb{Z}_q^{n \times m}$ can be 'explained' as a commitment to $\mathbf{M}$ using $\mathsf{EqCom}(\mathbf{Y})$. In other words, the following two distributions are computationally indistinguishable.

1. **Real World.** Given $(\mathbf{t}_1 = \mathbf{B}\mathbf{y}_1, \ldots, \mathbf{t}_L = \mathbf{B}\mathbf{y}_N)$ commit to $\mathbf{M} = [\mathbf{t}_1 \mid \ldots \mid \mathbf{t}_N]$ honestly using $\mathsf{Com}(\mathbf{M})$.
2. **Equivocated World.** Sample $\mathbf{C}' = \mathbf{B}\mathbf{R}_\mathbf{C}, \mathbf{R}_\mathbf{C} \leftarrow \{\pm 1\}^{m' \times m}$ early (independent of S) and then run $\mathsf{EqCom}(\mathbf{Y} = [\mathbf{y}_1 \mid \ldots \mathbf{y}_N], \mathbf{R}_\mathbf{C})$ to explain $\mathbf{C}'$ as a commitment to $\mathbf{M} = \mathbf{B}\mathbf{Y} = \mathbf{B}[\mathbf{y}_1 \mid \ldots \mid \mathbf{y}_N] = [\mathbf{t}_1 \mid \ldots \mid \mathbf{t}_N]$.

We call this commitment indistinguishability. Crucially, this must hold even when the adversary adaptively makes unbounded key generation queries (receiving fresh $\mathbf{t}_h = \mathbf{B}\mathbf{y}_h$) and corruption queries (revealing $\mathbf{y}_h$).

Constructing EqMCOM. Our EqMCOM construction combines three ingredients - (1) the MCOM of [Wee25], (2) an Explainable Re-Randomizer (ER) [CHW25, WW25], and (3) a random oracle $\mathcal{H}$. The ER scheme allows us to sample a 'random' commitment to $\mathbf{0} \in \mathbb{Z}_q^{n \times N}$ and also allows us to explain any $\mathbf{C}_0, \mathbf{Z}_0$ with low norm $\mathbf{Z}_0$ such that $\mathbf{C}_0\mathbf{V}_N = -\mathbf{B}\mathbf{Z}_0$ by deriving random coins for this sampling.

The commitment for matrix $\mathbf{M}$ in EqMCOM is $\mathbf{C} = \mathbf{C}_0 + \mathbf{C}_\mathbf{M}$ where $(\mathbf{C}_0, \mathbf{Z}_0) \leftarrow \mathsf{ER.Sample}(1^\lambda, 1^N)$. Here, the random coins of $\mathsf{ER.Sample}$ is derived from $\mathcal{H}$ using a short $\mathsf{aux} \leftarrow \{0,1\}^\lambda$. This way, $(\mathbf{C}, \mathsf{aux})$ together act as a short commitment to matrix $\mathbf{M}$. Correctness of EqMCOM is immediate and guarantees that $\mathbf{C}\mathbf{V}_N = (\mathbf{C}_0 + \mathbf{C}_\mathbf{M})\mathbf{V}_N = \mathbf{M} - \mathbf{B}(\mathbf{Z}_0 + \mathbf{Z}_M) = \mathbf{M} - \mathbf{B}\mathbf{Z}$ where $\mathbf{Z} = \mathsf{EqMCOM.Open}(\mathbf{M}, \mathsf{aux})$. This is the same functionality as MCOM.

The point of layering ER on top of MCOM is programmability. To equivocate a random $\mathbf{C}' = \mathbf{B}\mathbf{R}_\mathbf{C}$ and $\mathbf{R}_\mathbf{C} \leftarrow \{\pm 1\}^{m' \times m}$ as the commitment to $\mathbf{M} = \mathbf{B}[\mathbf{y}_1 \mid \ldots \mid \mathbf{y}_N] = \mathbf{B}\mathbf{Y}$, EqCom algorithm compute $\mathbf{C}_\mathbf{M}$ of MCOM honestly and sets $\mathbf{C}_0 = \mathbf{C}' - \mathbf{C}_\mathbf{M}$. Then, it computes $\mathbf{Z}_0 \leftarrow \mathbf{B}_{\widetilde{\sigma}}^{-1}(-\mathbf{C}_0\mathbf{V}_N)$ for sufficiently wide $\widetilde{\sigma}$. As $\mathbf{M} = \mathbf{B}\mathbf{Y}$ and $\mathbf{C}' = \mathbf{B}\mathbf{R}_\mathbf{C}$, we can use $\mathbf{Y}, \mathbf{R}_\mathbf{C}$ to derive $\mathbf{Z}_0$ efficiently without using a trapdoor for $\mathbf{B}$ as $\mathbf{Z}_0 = \mathbf{Y} - \mathbf{Z}_\mathbf{M} - \mathbf{R}_\mathbf{C}\mathbf{V}_N$. Then, we can use $\mathsf{ER.Explain}$ to derive new random coins for this $(\mathbf{C}_0, \mathbf{Z}_0)$ and program them in $\mathcal{H}$ at a random $\mathsf{aux}' \leftarrow \{0,1\}^\lambda$. Overall, the net effect is that a commitment $\mathbf{C} = \mathbf{B}\mathbf{R}_\mathbf{C}$ that was sampled before S was known can be opened to $\mathbf{M} = \mathbf{B}\mathbf{Y}$ for any adaptively chosen $\mathbf{Y}$ derived from adaptively queried keys.

EqFBE from EqMCOM. Plugging in EqMCOM into GY-style template of DBE (applied here to the unbounded/flexible setting) gives a remarkably clean FBE. The crs of EqFBE contains $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$ which are used by EqMCOM and a random vector $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. Notice that there is no matrix $\mathbf{A}$ unlike [GY26b] which creates a bounded user barrier. A user samples $\mathsf{sk} = \mathbf{r} \leftarrow \{\pm 1\}^{m'}$ and sets $\mathsf{pk} = \mathbf{t} = \mathbf{B}\mathbf{r} + \mathbf{p}$. To encrypt to public keys $(\mathbf{t}_1, \ldots, \mathbf{t}_N)$, we compute the matrix $\mathbf{M} = [\mathbf{t}_1 \mid \ldots \mid \mathbf{t}_N]$ and compute $(\mathbf{C}, \mathsf{aux}) \leftarrow \mathsf{EqMCOM.Com}(\mathbf{M})$. Then sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ and output

$$\mathsf{ct} = (\mathbf{c}_1^\top := \mathbf{s}^\top\mathbf{B} + \mathsf{err}, \mathbf{c}_2^\top := \mathbf{s}^\top\mathbf{C} + \mathsf{err}, c_3 := \mathbf{s}^\top\mathbf{p} + \mathsf{err} + \lfloor q/2 \rfloor \mu, \mathsf{aux})$$

Note that the size of the ciphertext is completely independent of recipient set size L. To decrypt, i-th user computes $\mathbf{V}_N, \mathbf{Z} = \mathsf{EqMCOM.Open}(\mathbf{M}, \mathsf{aux})$, uses their i-th columns $\mathbf{v}_i, \mathbf{z}_i$ (respectively), and computes

$$c_i = \mathbf{c}_2^\top\mathbf{v}_i + \mathbf{c}_1^\top(\mathbf{z}_i - \mathbf{r}_i) \approx \mathbf{s}^\top(\mathbf{C}\mathbf{v}_i + \mathbf{B}\mathbf{z}_i - \mathbf{B}\mathbf{r}_i) = \mathbf{s}^\top(\mathbf{t}_i - \mathbf{B}\mathbf{r}_i) = \mathbf{s}^\top\mathbf{p}$$

From there, user can compute μ with high probability.

Fake Setup, Key Generation Simulation, and Equivocation. The fake setup algorithm $\mathsf{FakeSetup}$ given a message μ, samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ and sets $\widetilde{\mathsf{ct}} = (\mathbf{c}_1^\top := \mathbf{s}^\top\mathbf{B} + \mathsf{err}, c_3 := \mathbf{s}^\top\mathbf{p} + \mathsf{err} + \lfloor q/2 \rfloor \mu)$ where $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. It also samples $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$ which are required for EqMCOM. This way, crs is exactly identical to the construction.

To simulate public keys queried by attacker $\mathsf{SimKGen}$ samples $\mathbf{t} = \mathbf{B}\mathbf{y}$ where $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{m'}$. Finally, to equivocate to a set of public keys $(\mathbf{t}_1, \ldots, \mathbf{t}_N)$, notice that each $\mathbf{t}_i = \mathbf{B}\mathbf{y}_i$ and thus $\mathbf{M} = [\mathbf{t}_1 \mid \ldots \mid \mathbf{t}_N] = \mathbf{B}[\mathbf{y}_1 \mid \ldots \mid \mathbf{y}_N]$. Thus, we can equivocate $\mathbf{M}$ to a random commitment $\mathbf{C} = \mathbf{B}\mathbf{R}_\mathbf{C}$ and set $\mathbf{c}_2^\top \approx \mathbf{c}_1^\top\mathbf{R}_\mathbf{C}$ and provide the ciphertext as $(\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{aux}')$ where $\mathsf{aux}' \leftarrow \mathsf{EqMCOM.EqCom}([\mathbf{y}_1 \mid \ldots \mid \mathbf{y}_N], \mathbf{R}_\mathbf{C})$.

Security. We prove that above design satisfies mode indistinguishability and selective security for fake setup.

Mode indistinguishability. We argue this using the following hybrids.

(1) This is the real semi-static FBE experiment where public keys are sampled as $\mathbf{t} = \mathbf{Br} + \mathbf{p}$ where $\mathbf{r} \leftarrow \{\pm 1\}^{m'}$ and $\mathbf{C}$ in Enc is computed using $\mathsf{EqMCOM.Com}(\mathbf{M})$.
(2) We use leftover hash lemma to sample $\mathbf{t} \leftarrow \mathbb{Z}_q^n$. Then, by another leftover hash lemma argument, sample $\mathbf{t} = \mathbf{By}$ where $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{m'}$.
(3) Use $\mathsf{EqMCOM.EqCom}([\mathbf{y}_1 \mid \ldots \mid \mathbf{y}_N], \mathbf{R}_{\mathbf{C}})$ instead of $\mathsf{EqMCOM.Com}$ during Enc. This step is indistinguishable from previous one by commitment indistinguishability property of EqMCOM.

After these hybrids, the experiment matches the fake-mode distribution produced by $\mathsf{FakeSetup}$, $\mathsf{SimKGen}, \mathsf{Equivocate}$ and thus mode indistinguishability holds.

Selective security for fake setup. The fake mode's crs is essentially $(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p})$ and the fake ciphertext is essentially $(\mathbf{s}^\top \mathbf{B} + \mathsf{err}, \mathbf{s}^\top \mathbf{p} + \mathsf{err} + \lfloor q/2 \rfloor \mu)$. Using leftover hash lemma, we sample $\mathbf{p} = \mathbf{Br_p}$ where $\mathbf{r_p} \leftarrow \{\pm 1\}^{m'}$. Then, by the decomposed LWE assumption, $\mathbf{c}_1^\top := \mathbf{s}^\top \mathbf{B} + \mathsf{err}$ is pseudorandom, which means $c_3 := \mathbf{c}_1^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu$ statistically hides μ by leftover hash lemma. This yields selective security for the fake setup, which concludes the proof. Full proofs are provided in Section 6.

Aggregating public keys. Above we described an FBE scheme whose CRS, public key, secret key, and ciphertext sizes are all $\mathsf{poly}(\lambda, \log N)$. However, encryption and decryption running time can still grow with the number of recipients N, since the encryptor has to read all N public keys to compute $\mathsf{EqMCOM.Com}(\mathbf{M})$. This can be addressed using the standard registered-ABE style aggregation [GV20, HLWW23]. We can use a public-coin pre-processing algorithm that takes a list $(\mathsf{pk}_1, \ldots, \mathsf{pk}_N)$ and produces a short master public key mpk plus per-user decryption hints $\{\mathsf{ht}_i\}_{i \in [N]}$. Using mpk, the encryptor and decryptor can run in time $\mathsf{poly}(\lambda, \log N)$. This is quite useful for quorums for which encryption happens frequently.

Removing random oracles. Our FBE construction crucially relies on programming the random oracle during the fake mode to perform equivocation. As already noted by GY in the slotted DBE setting, removing the random oracle while retaining all other optimal features is a delicate challenge. For slotted DBE, this gap was partially addressed by GY via a new tool called Programmable Pseudorandom Set Samplers (PPS). In our flexible setting, matters are even more subtle. The CRS is *transparent*, and it is not clear how to embed a PPS-style structure into a transparent CRS without a blow-up in size. We leave constructing adaptively-secure FBE with optimal parameters in the *standard model* (especially with a transparent setup) as an exciting open problem.

Finally, we subsequently realized that using our EqMCOM primitive (full definition in Definition 4.1), we could directly prove adaptive security of FBE in ROM. However, for ease of exposition, we demonstrated a semi-static version which can be compiled into an adaptively secure FBE using [GW09, FWW23].

2.2 Identity-Based Broadcast Encryption

We now move to the centralized setting, where a trusted authority samples a master public-secret key pair, and identities are drawn from an a-priori unbounded space $\mathcal{I} = \{\{0,1\}^\kappa\}_{\kappa \in \mathbb{N}}$. Our next goal is to design an adaptively-secure IBBE with all parameters $\mathsf{poly}(\lambda, \kappa)$, under the same decomposed LWE assumption.

A trivial approach would be to instantiate our FBE scheme with one public-secret key pair per identity during Setup, and include all 2^κ public keys in the master public key, with the corresponding

secret keys as the master secret. However, this is completely unworkable. The sizes of $|\mathsf{mpk}|, |\mathsf{msk}|$ grow with 2^κ, and Setup takes exponential time. Thus, we need a different approach.

Why Gentry-Waters compiler fails. A more sophisticated route is to first build a semi-statically-secure IBBE and apply the Gentry-Waters compiler [GW09, HWW25], as we did for FBE. This approach also fails in the IBBE setting. In the semi-static experiment, the adversary must commit in advance to a set S' containing the eventual challenge set S^*, while the reduction must answer every secret-key query for an identity outside S'. For broadcast encryption with a polynomial universe $[N]$, [GW09] showed a way for the reduction to construct S' using an N-bit indicator vector. In IBBE, the identity universe is super-polynomially large, so describing or guessing the analogous set S' cannot be done by a polynomial-time reduction. Consequently, the generic compiler cannot yield our result and we must prove adaptive security directly. Our stronger equivocal encryption framework provides the key simulation interface needed to answer adaptive secret-key queries without committing to the challenge identities in advance. We instantiate this interface by exploiting the algebraic structure of our FBE construction and making a few additional changes.

A dual-mode setup for public keys. The key observation is that, in our FBE construction, the public-secret key pair has a very specific structure. The public key $\mathbf{t}$ is nothing but an Ajtai commitment w.r.t. the matrix $\mathbf{B}$, since $\mathbf{t} = \mathbf{Br} + \mathbf{p}$ with $\mathbf{r}$ short. In particular, public keys are pseudorandom strings in $\mathbb{Z}_q^n$ (indistinguishable from uniform by the leftover hash lemma). So, there are really two equivalent ways to sample an FBE $(\mathsf{pk}, \mathsf{sk})$ pair:

1. **Forward sampling (FBE style):** Sample short $\mathbf{r}$, set $\mathbf{t} := \mathbf{Br} + \mathbf{p}$, output $(\mathsf{pk}, \mathsf{sk}) = (\mathbf{t}, \mathbf{r})$.
2. **Backward sampling (dual mode):** Sample $\mathbf{t} \leftarrow \mathbb{Z}_q^n$, sample a short $\mathbf{r}$ such that $\mathbf{Br} = \mathbf{t} - \mathbf{p}$ using a trapdoor for $\mathbf{B}$, output $(\mathsf{pk}, \mathsf{sk}) = (\mathbf{t}, \mathbf{r})$.

The second mode is infeasible in FBE because no party has the trapdoor of $\mathbf{B}$ (recall that $\mathbf{B}$ is part of the public CRS). But in IBBE, we have a trusted key authority. This naturally suggests the following design. The authority samples $\mathbf{B}$ together with its trapdoor $\mathbf{T_B}$, and puts $\mathbf{B}$ (without $\mathbf{T_B}$) in the master public key. To derive a secret key for an identity id, the authority first hashes id using a random oracle $\mathcal{H}_{\mathsf{pk}} : \{0,1\}^\kappa \to \mathbb{Z}_q^n$ to get $\mathbf{t}_{\mathsf{id}} = \mathcal{H}_{\mathsf{pk}}(\mathsf{id})$, and then uses its trapdoor to sample a short Gaussian pre-image $\mathbf{y}_{\mathsf{id}}$ with $\mathbf{B}\mathbf{y}_{\mathsf{id}} = \mathbf{t}_{\mathsf{id}} - \mathbf{p}$. On the encryptor side, anyone can compute $\mathbf{t}_{\mathsf{id}} = \mathcal{H}_{\mathsf{pk}}(\mathsf{id})$ for any identity id without needing the trapdoor, and then run our FBE encryption using the public keys $(\mathbf{t}_{\mathsf{id}_1}, \ldots, \mathbf{t}_{\mathsf{id}_N})$ where $S = (\mathsf{id}_1, \ldots, \mathsf{id}_N)$ is the target identity set.

This gives an IBBE scheme with all parameter sizes $\mathsf{poly}(\lambda, \kappa)$, since the public keys are now implicitly defined by the random oracle and the authority only needs to be invoked to produce secret keys (much like GPV-style IBE [GPV08]). We call the strengthening of IBBE with equivocation properties Equivocal IBBE (EqBE). Definition is provided in Definition 7.2 and full construction is provided in Construction 8.3.

Security intuition. The proof that our IBBE construction is adaptively secure is very similar to the proof for our FBE, with one additional twist to handle secret key queries. Since in IBBE the attacker can query secret keys for arbitrary identities (as long as they are not in the challenge set), the reduction must be able to answer these queries without knowing the challenge set in advance. For any queried id, the reduction programs $\mathcal{H}_{\mathsf{pk}}(\mathsf{id}) := \mathbf{B}\mathbf{y}'_{\mathsf{id}} + \mathbf{p}$ for a freshly sampled short Gaussian $\mathbf{y}'_{\mathsf{id}}$, and returns $\mathbf{y}'_{\mathsf{id}}$ as the secret key. This way of sampling $\mathcal{H}_{\mathsf{pk}}(\mathsf{id})$ and later revealing $\mathbf{y}'_{\mathsf{id}}$ constitutes the $\mathsf{SimKGen}$ algorithm. This lets us simulate secret keys on the fly consistently,

while retaining the ability to later equivocate the challenge ciphertext onto any subset of queried identities as long as $\mathbf{y}'_{\mathsf{id}}$ comes from a wide-enough gaussian distribution. Unlike EqFBE, the IBBE fake mode must hide μ while remaining consistent with adaptive secret-key queries, and the short representation $\mathbf{r_p}$ of $\mathbf{p} = \mathbf{B}\mathbf{r_p}$ is initially used inside equivocation. We first use decomposed LWE to make $\mathbf{c}_1^\top$ to be uniform and then use gaussian pre-image sampling and smudging to remove $\mathbf{r_p}$ from equivocation. Then a leftover hash lemma helps us in making the challenge independent of μ. More details are provided in Section 7 and Appendix F.

A general observation. Taking a step back, the observation that our FBE admits a dual-mode setup (where pk's are pseudorandom and sk's can be "reverse-sampled" with a trapdoor) may be useful beyond the IBBE setting. Concretely, any FBE whose $(\mathsf{pk}, \mathsf{sk})$ pair is a short-LWE-like sample w.r.t. a public matrix $\mathbf{B}$ can be converted into an IBBE by programming $\mathsf{pk} = \mathcal{H}_{\mathsf{pk}}(\mathsf{id})$ and putting the trapdoor of $\mathbf{B}$ into the master secret key, much as GPV [GPV08] converts SIS-based signatures into an identity-based encryption scheme via random oracle programming. We believe that this observation about translating an FBE to IBBE via a dual-setup mode will be useful in other contexts too, such as designing identity-based variants of other standard encryption schemes via their registration-based analogues.

Removing the random oracle from $\mathcal{H}_{\mathsf{pk}}$ for secret-key IBBE. In the secret-key IBBE setting, where both Enc and KGen are performed by parties holding msk, we can replace $\mathcal{H}_{\mathsf{pk}}$ with a PRF key K (included in msk). This avoids the need for a random oracle specifically for $\mathcal{H}_{\mathsf{pk}}$. More details are provided in Section 7. The random oracle present in EqMCOM, used for compressing equivocation randomness, remains necessary for our current approach. Finally, we also remark that as we are in the centralized setting, we could use succinct LWE [Wee24] to get more compact ciphertexts. However, for uniformity across our constructions, we chose to write our scheme using decomposed LWE.

3 Preliminaries

Notation. By PPT we denote probabilistic polynomial-time. We denote the security parameter by λ and the set of positive integers by $\mathbb{N}$. For any $a, b \in \{0\} \cup \mathbb{N}$, $a \leq b$, we denote by $[a, b]$, the set of all integers from a to b including a and b. In other words, $[a, b] = \{a, \ldots, b\}$. We denote by $[n] := [1, n]$. We denote by $x \leftarrow \mathcal{X}$, the process of sampling an element x from the set $\mathcal{X}$, with uniform probability. Similarly, for any PPT algorithm $\mathcal{A}$, $x \leftarrow \mathcal{A}(y)$ denotes the process of sampling x from the output distribution of $\mathcal{A}$ when run on y with randomly chosen coins. By $\mathsf{negl}(\cdot)$, we denote negligible functions. By $\mathsf{poly}(\cdot)$, we denote positive polynomials. All of our logarithms log are in base 2. Consider two efficiently samplable probability distributions $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_{\lambda\in\mathbb{N}}$ over support Ω. We say $\mathcal{X}$ and $\mathcal{Y}$ are computationally indistinguishable if for any non-uniform PPT distinguisher $\mathcal{D} = \{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$, and for large enough $\lambda \in \mathbb{N}$,

$$\left| \Pr_{\alpha\leftarrow\mathcal{X}_\lambda}\left[1 \leftarrow \mathcal{D}(1^\lambda, \alpha)\right] - \Pr_{\alpha\leftarrow\mathcal{Y}_\lambda}\left[1 \leftarrow \mathcal{D}(1^\lambda, \alpha)\right] \right| \leq \mathsf{negl}(\lambda)$$

We write $\mathcal{X} \approx_c \mathcal{Y}$ to denote that distributions $\mathcal{X}$ and $\mathcal{Y}$ computationally indistinguishable. Similarly, by $\approx_s$, we denote statistical indistinguishability. Concretely, $\mathcal{X} \approx_s \mathcal{Y}$ if for large enough $\lambda \in \mathbb{N}$, $\mathsf{SD}(\mathcal{X}_\lambda, \mathcal{Y}_\lambda) = \frac{1}{2}\sum_{\alpha\in\Omega} |\Pr[\alpha \leftarrow \mathcal{X}_\lambda] - \Pr[\alpha \leftarrow \mathcal{Y}_\lambda]| \leq \mathsf{negl}(\lambda)$.

3.1 Lattice Preliminaries

For prime power $q \in \mathbb{N}$, we denote the finite field with q elements as $\mathbb{Z}_q$. Following popular convention, we denote column vectors by lowercase bold letters, e.g., $\mathbf{x} \in \mathbb{Z}_q^n$ and denote matrices by uppercase bold letters, e.g., $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$. We use x_i to denote the i-th element in $\mathbf{x}$ for $i \in [n]$ and $a_{i,j}$ to denote the element in the i-th row and j-th column of matrix $\mathbf{A}$ for $i \in [n], j \in [m]$. We define $\mathbf{1}_n$ to be a vector of all ones, $\mathbf{0}_n$ to be a vector of all zeros, and $\mathbb{1}_{n,j}$ to denote the n-dimensional canonical basis vector that has 1 in the j-th position, i.e., $\mathbf{1}_n = [1\ 1\ \ldots\ 1]^\top$ and $\mathbb{1}_{n,j} = [\ \ldots\ 0\ 1\ 0\ \ldots]^\top$. We frequently drop n from subscript in these vectors when dimension is clear. We denote the $n \times n$ identity matrix by $\mathbf{I}_n$ and drop the subscript when dimension is clear. For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathsf{vec}(\mathbf{A}) \in \mathbb{Z}_q^{nm}$ denoted the vector obtained by concatenating the columns of $\mathbf{A}$. For any vector $\mathbf{a} \in \mathbb{Z}_q^n$ or matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, we use $\|\mathbf{a}\|$ and $\|\mathbf{A}\|$ to denote the maximum of all the entries in $\mathbf{a}$ and $\mathbf{A}$ respectively. We use $\mathbf{A} \otimes \mathbf{B}$ to denote the Kronecker product between matrices $\mathbf{A}, \mathbf{B}$.

Discrete Gaussian. We write $\mathcal{D}_{\mathbb{Z},\sigma}$ to denote the discrete gaussian distribution with width parameter $\sigma > 0$. We state the following important lemma about the maximum norm of a value sampled from discrete gaussian distribution.

Lemma 3.1 ([Lyu12, MP12])**.** For any $k > 0$, $\Pr[|x| > k\sigma : x \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}] \leq 2\exp\left(\frac{-k^2}{2}\right)$.

Gadget Matrix. For positive integers n, m, q, and $m = \Omega(n \log q)$, we write $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ to denote the gadget matrix [MP12]. It is defined as $\mathbf{G} = (\mathbf{I}_n \otimes \mathbf{g}^\top)$, $\mathbf{g} = [1\ 2\ 2^2\ \ldots\ 2^{\lfloor \log q \rfloor}\ 0\ \ldots\ 0]^\top \in \mathbb{Z}_q^m$. We write $\mathbf{G}^{-1}$ to denote the standard entry-wise bit decomposition. Note that $\mathbf{G}^{-1} : \mathbb{Z}_q^n \to \mathbb{Z}_q^m$ is an operation such that $\mathbf{G}\mathbf{G}^{-1}(\mathbf{a}) = \mathbf{a}$ and not inverse of $\mathbf{G}$. Output of $\mathbf{G}^{-1}$ is padded with 0s to match the dimensions. We often overload the notation and write $\mathbf{G}^{-1}(\mathbf{A})$ where $\mathbf{G}^{-1}$ is performed for each column.

Leftover Hash Lemma. We recall the (generalized) leftover hash lemma adapted from [ILL89, DRS04, GPV08, ABB10].

Lemma 3.2 (Generalized Leftover Hash Lemma)**.** Let $n, m, q \in \mathbb{N}$ and q is an odd prime.

1. $\{(\mathbf{B}, \mathbf{BR}, \mathbf{e}^\top\mathbf{R})\} \approx_s \{(\mathbf{B}, \mathbf{A}, \mathbf{e}^\top\mathbf{R})\}$ where $m = \Omega(n \log q)$, $\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{R} \leftarrow \{\pm 1\}^{m \times \ell}$(or $\mathbf{R} \leftarrow \{0,1\}^{m \times \ell}$), $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times \ell}$, $\ell = \mathsf{poly}(n)$ and $\mathbf{e}^\top \in \mathbb{Z}_q^m$ is any fixed vector.
2. $\{(\mathbf{B}, \mathbf{BR}, \mathbf{ER})\} \approx_s \{(\mathbf{B}, \mathbf{A}, \mathbf{ER})\}$ where $\mathbf{E} \in \mathbb{Z}_q^{N \times m}$ is any fixed vector, $m > (n + N + 1)\lceil \log q \rceil + \omega(\log \lambda)$, $\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{R} \leftarrow \{\pm 1\}^{m \times \ell}$(or $\mathbf{R} \leftarrow \{0,1\}^{m \times \ell}$), $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times \ell}$, $\ell = \mathsf{poly}(n)$.
3. $\{(\mathbf{B}, \mathbf{BR})\} \approx_s \{(\mathbf{B}, \mathbf{A})\}$ where $m = \Omega(n \log q)$, $\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{R} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times \ell}, \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times \ell}$, $\ell = \mathsf{poly}(n)$ and $\sigma = \omega(\sqrt{\log m})$.

Gadget Trapdoors. Let (n, m, q) be LWE parameters such that $m \geq 3n \log q$. Then, there exists two polynomial time algorithms defined as follows.

$\mathsf{TrapGen}(1^n, 1^m, q) \to (\mathbf{A}, \mathbf{T_A})$**.** The randomized trapdoor generation algorithm takes as input lattice dimension n, number of columns m, modulus q, outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and its trapdoor $\mathbf{T_A}$.

$\mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{y}, \sigma) \to \mathbf{x}$. The randomized pre-image sampling algorithm takes as input $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, its trapdoor $\mathbf{T_A}$, a vector $\mathbf{y} \in \mathbb{Z}_q^n$, gaussian width parameter σ, and outputs a vector $\mathbf{x} \in \mathbb{Z}_q^m$.

These algorithms satisfy the following properties.

Matrix Distribution. If $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, then $\|\mathbf{T_A}\| = 1$, $\mathbf{AT_A} = \mathbf{G}$, and

$$\{\mathbf{A} : (\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)\} \approx_s \{\mathbf{A} : \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}\}$$

Sampling Correctness. For all $\sigma > 0$, any $\mathbf{y} \in \mathbb{Z}_q^n$, $\mathbf{Ax} = \mathbf{y}$ where $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ and $\mathbf{x} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{y}, \sigma)$.

Pre-image Distribution. Let $(\mathbf{A}, \mathbf{T_A})$ be such that $\mathbf{AT_A} = \mathbf{G}$. Then, for any $\sigma \geq \|\mathbf{T_A}\| \, m \log n$, any $\mathbf{y} \in \mathbb{Z}_q^n$,

$$\{\mathbf{x} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{y}, \sigma)\} \approx_s \{\mathbf{x} : \mathbf{A}_\sigma^{-1}(\mathbf{y})\}$$

Here, $\mathbf{A}_\sigma^{-1}(\mathbf{y})$ denotes the (possibly inefficient) process of sampling a discrete gaussian vector with width parameter σ from the coset lattice $\Lambda_q^{\mathbf{y}}(\mathbf{A}) = \{\mathbf{x} : \mathbf{Ax} = \mathbf{y} \mod q, \mathbf{x} \in \mathbb{Z}^m\}$.

Remark 3.3. $\mathsf{SamplePre}$ can be naturally extended to sample pre-images for matrices by sampling a pre-image vector for each column independently. We will use this version in our work. Sampling correctness and pre-image distribution properties can be naturally extended to matrices as well.

Lemma 3.4 ([MP12])**.** Let n, m, q be lattice parameters such that $m \geq 2n \log q$. For all but $\mathsf{negl}(n)$ fraction of matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\sigma \geq \log m$, any vector $\mathbf{y}$ in the column space of $\mathbf{A}$, $\Pr[\|\mathbf{x}\| > \sqrt{m}\sigma : \mathbf{x} \leftarrow \mathbf{A}_\sigma^{-1}(\mathbf{y})] \leq \mathsf{negl}(m)$.

Lemma 3.5 (Smudging Lemma [AJL+12])**.** Let $B_1 = B_1(\lambda)$ and $B_2 = B_2(\lambda)$ be positive integers and let $e_1 \in [-B_1, B_1]$ be a fixed integer. Let $e_2 \leftarrow [-B_2, B_2]$ be chosen uniformly at random. Then the distribution of e_2 is statistically indistinguishable from that of $e_2 + e_1$ as long as $B_1/B_2 = \mathsf{negl}(\lambda)$.

Assumption 3.6 ([Reg05])**.** The LWE assumption with parameters (n, m, q, χ) states that

$$\left\{(\mathbf{A}, \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top)\right\} \approx_c \left\{(\mathbf{A}, \mathbf{c}^\top)\right\}$$

where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{c} \leftarrow \mathbb{Z}_q^m$.

3.2 ℓ-Decomposed LWE

We provide the ℓ-decomposed LWE assumption which was originally proposed in [AMR25]. Our formulation is adapted from [Wee25, WW25].

Assumption 3.7 ([AMR25], adapted from [Wee25, WW25])**.** Let (n, m, q, χ) be LWE parameters. Then, the (ℓ, σ)-decomposed LWE assumption states that

$$\left\{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top)\right\} \approx_c \left\{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{c}^\top)\right\}$$

where $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times \ell m}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times \ell m}, \mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{\ell^2 m}, \mathbf{c} \leftarrow \mathbb{Z}_q^{\ell^2 m}$ and $\mathbf{B} = \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_\ell)^\top \otimes \mathbf{G} \in \mathbb{Z}_q^{n \times \ell^2 m}$. When $\sigma = \mathsf{poly}(\lambda, \ell, m)$ we simply call this as ℓ-decomposed LWE assumption.

Leftover Hash Lemma for Structured B. We remark that $\mathbf{B}$ defined in above assumption is a structured matrix uniquely defined by $\mathbf{W}, \mathbf{D}$. Thus, it might not be immediately clear that Lemma 3.2 works with $\mathbf{B}$. We prove that this holds with respect to structured $\mathbf{B}$ from the decomposed LWE assumption (and more) in Appendix A.

Trapdoor for B. We show an alternate way to derive $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ of the (ℓ, σ)-decomposed LWE assumption assumption such that there is a way to construct $\mathbf{T_B}$ such that $\mathbf{BT_B} = \mathbf{G}, \|\mathbf{T_B}\| = 1$. The procedure is described below.

1. Sample $(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, sample $[\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_\ell] \leftarrow \mathbb{Z}_q^{n \times (\ell-1)m}$. Set $\mathbf{W} := [\mathbf{W}_1 \mid \mathbf{W}_2 \mid \ldots \mid \mathbf{W}_\ell] \in \mathbb{Z}_q^{n \times \ell m}$.
2. Sample $\mathbf{D}_1 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times m}, [\mathbf{D}_3 \mid \ldots \mid \mathbf{D}_\ell] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (\ell-2)m}$. Sample $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ Sample $\mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma)$. Set $\mathbf{D} = [\mathbf{D}_1 \mid \mathbf{D}_2 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_\ell] \in \mathbb{Z}_q^{m \times \ell m}$.
3. Set $\mathbf{B} := \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_\ell)^\top \otimes \mathbf{G}$, $\mathbf{T_B} = \mathbb{1}_{\ell^2,2} \otimes \mathbf{T_A}$.
4. Output $(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{T_B})$.

It is straightforward to see that $\|\mathbf{T_B}\| = \|\mathbf{T_A}\| = 1$. Now, $\mathbf{BT_B} = [\mathbf{W}_1\mathbf{D}_1 + \mathbf{G} \mid \mathbf{W}_1\mathbf{D}_2 \mid \ldots \mid \mathbf{W}_\ell\mathbf{D}_\ell + \mathbf{G}][\mathbf{0} \mid \mathbf{T_A}^\top \mid \mathbf{0} \mid \ldots \mid \mathbf{0}]^\top = \mathbf{W}_1\mathbf{D}_2\mathbf{T_A} = \mathbf{AT_A} = \mathbf{G}$. We also remark that $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ given out by the procedure is statistically close to uniformly random values. Proof is provided in Appendix A.

4 Equivocal Matrix Commitment Scheme

In this section, we present a strengthening of the matrix commitment scheme (MCOM) from [Wee25] which we call Equivocal Matrix Commitment scheme (EqMCOM).

Syntax. An Equivocal Matrix Commitment schemes (EqMCOM) consists of the following polynomial time algorithms.

$\mathsf{Setup}(1^n, 1^m, q, \sigma, B) \to (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$**.** The randomized *public-coin* setup algorithm takes as input LWE parameters (n, m, q, σ) where $\sigma = \mathsf{poly}(\lambda, m)$, bound $B \in \mathbb{N}$, outputs public parameters pp, $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times m_1}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times m_1}$, and $\mathbf{B} = \mathbf{W}(\mathbf{I}_{m_2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{m_2})^\top \otimes \mathbf{G}$. The following algorithms take $(\widetilde{\sigma}, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$ implicitly.

$\mathsf{Com}(\mathbf{M}) \to (\mathbf{C}, \mathsf{aux})$**.** The randomized commitment algorithm takes as input matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$, outputs commitment matrix $\mathbf{C} \in \mathbb{Z}_q^{n \times m}$ and auxiliary information aux.

$\mathsf{Verify}(1^N) \to \mathbf{V}_N$**.** The deterministic verification algorithm takes as input column length N and outputs verification matrix $\mathbf{V}_N \in \mathbb{Z}_q^{m \times N}$.

$\mathsf{Open}(\mathbf{M}, \mathsf{aux}) \to \mathbf{Z}$**.** The deterministic opening algorithm takes as input matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$, auxiliary information aux, and outputs opening matrix $\mathbf{Z} \in \mathbb{Z}_q^{m_1 m_2 \times N}$.

$\mathsf{EqCom}(\mathbf{Y}, \mathbf{R_C}) \to \mathsf{aux}$**.** The randomized equivocal commitment algorithm takes as input low-norm matrix $\mathbf{Y} \in \mathbb{Z}_q^{m_1 m_2 \times N}$, $\mathbf{R_C} \in \mathbb{Z}_q^{m_1 m_2 \times m}$ such that $\|\mathbf{R_C}\| = 1$, and outputs auxiliary information aux.

Definition 4.1 (EqMCOM)**.** An EqMCOM scheme $(\mathsf{Setup}, \mathsf{Com}, \mathsf{Verify}, \mathsf{Open}, \mathsf{EqCom})$ should satisfy the following properties.

Correctness. For any $n, q \in \mathbb{N}, m = \Omega(n \log q), \sigma = \mathsf{poly}(\lambda, m), B \in \mathbb{N}, N = \mathsf{poly}(\lambda)$, any matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$,

$$\Pr\left[\begin{array}{lll} \mathbf{CV}_N = \mathbf{M} - \mathbf{BZ} & : & (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{Setup}(1^n, 1^m, q, \sigma, B), \\ & & (\mathbf{C}, \mathsf{aux}) \leftarrow \mathsf{Com}(\mathbf{M}), \mathbf{V}_N = \mathsf{Verify}(1^N), \\ & & \mathbf{Z} = \mathsf{Open}(\mathbf{M}, \mathsf{aux}) \end{array}\right] = 1$$

Compactness. There exist polynomials $p_1(\cdot), p_2(\cdot), \mathsf{poly}_{\mathsf{aux}}(\cdot,\cdot)$ such that for any $\lambda \in \mathbb{N}, N \leq 2^\lambda, \mathbf{M} \in \mathbb{Z}_q^{n \times N}, n, q \in \mathbb{N}, m = \Omega(n \log q), \sigma = \mathsf{poly}(\lambda, m)$, and $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{Setup}(1^n, 1^m, q, \sigma, B)$, $m_1 = p_1(m), m_2 = p_2(m), |\mathsf{aux}| \leq \mathsf{poly}_{\mathsf{aux}}(\lambda, \log N)$ where $(\mathbf{C}, \mathsf{aux}) \leftarrow \mathsf{Com}(\mathbf{M})$.

Moreover, we also need that $\|\mathbf{V}_N\| \leq O(\|\mathbf{D}\| m^4 \log q)$ and $\|\mathbf{Z}\| \leq O(\sqrt{m}\widetilde{\sigma})$ where $\mathbf{V}_N = \mathsf{Verify}(1^N)$, $\mathbf{Z} = \mathsf{Open}(\mathbf{M}, \mathsf{aux})$, and $\widetilde{\sigma}$ is implicit in pp.

Commitment Indistinguishability. For any stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$, the following probability is at most $1/2 + \mathsf{negl}(\lambda)$.

$$\Pr\left[\begin{array}{lll} b \leftarrow \mathcal{A}^{\mathcal{O}(\cdot,\cdot)}(\mathbf{C}^{(b)}, \mathsf{aux}^{(b)}) & : & (1^n, 1^m, q, \sigma, B) \leftarrow \mathcal{A}(1^\lambda), h := 1, \mathcal{Y} := \emptyset, \\ & & (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{Setup}(1^n, 1^m, q, \sigma, B), \\ & & (S, \mathbf{R}) \leftarrow \mathcal{A}^{\mathcal{O}(\cdot,\cdot)}(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}), \\ & & b \leftarrow \{0,1\}, \\ & & \text{If } b = 0, (\mathbf{C}^{(0)}, \mathsf{aux}^{(0)}) \leftarrow \mathsf{Com}(\mathbf{M}) \\ & & \text{If } b = 1, \mathbf{R}_{\mathbf{C}} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{C}^{(1)} = \\ & & \mathbf{BR}_{\mathbf{C}}, \mathsf{aux}^{(1)} \leftarrow \mathsf{EqCom}(\mathbf{Y} + \mathbf{R}, \mathbf{R}_{\mathbf{C}}) \end{array}\right]$$

where $\mathbf{Y} = [\mathcal{Y}[h_1] \mid \ldots \mid \mathcal{Y}[h_N]]$ and $\mathbf{M} = \mathbf{B}(\mathbf{Y} + \mathbf{R})$. Here, S is parsed as $(h_1, \ldots, h_N)$. $\mathcal{O}$ is a stateful oracle that responds to the following queries.

- Gen**.** $\mathcal{A}$ queries for generation. Sample $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{m_1 m_2}$. Compute $\mathbf{t} = \mathbf{By}$. Send $(\mathbf{t}, h)$ to $\mathcal{A}$. Store $\mathcal{Y}[h] := \mathbf{y}$ and update $h := h + 1$.
- (Corr, h')**.** $\mathcal{A}$ queries for corruption with $h' \in [h]$. If there is no entry for h' in $\mathcal{Y}$, send $\perp$ to $\mathcal{A}$. Otherwise, send $\mathcal{Y}[h']$ to $\mathcal{A}$. Remove entry corresponding to h' from $\mathcal{Y}$.

A stateful PPT machine $\mathcal{A}$ is said to be admissible if $S \subseteq [h]$ and for each $h^* \in S$, there is an entry in $\mathcal{Y}$ and $\|\mathbf{R}\| \leq B$. Moreover, $\mathcal{A}$ doesn't corrupt any $h^* \in S$ in post-challenge queries.

4.1 Preliminaries

We provide the necessary preliminaries required for our construction below.

Matrix Commitments. A Matrix Commitment scheme (MCOM) consists of the following polynomial time algorithms.

- $\mathsf{Setup}(1^n, 1^m, q, \sigma) \to (\mathbf{W}, \mathbf{D}, \mathbf{B})$**.** The randomized *public-coin* setup algorithm takes as input LWE parameters (n, m, q, σ) where $\sigma = \mathsf{poly}(\lambda, m)$ and outputs $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times 2m^3}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times 2m^3}$, and $\mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}$. The following algorithms take $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ implicitly.

$\mathsf{Com}(\mathbf{M}) \to \mathbf{C}$. The deterministic commitment algorithm takes as input matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$ and outputs commitment matrix $\mathbf{C} \in \mathbb{Z}_q^{n \times m}$.

$\mathsf{Verify}(1^N) \to \mathbf{V}_N$. The deterministic verification algorithm takes as input column length N and outputs verification matrix $\mathbf{V}_N \in \mathbb{Z}_q^{m \times N}$.

$\mathsf{Open}(\mathbf{M}) \to \mathbf{Z}$. The deterministic opening algorithm takes as input matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$ and outputs opening matrix $\mathbf{Z} \in \mathbb{Z}_q^{4m^5 \times N}$.

Definition 4.2 ([Wee25]). A MCOM scheme $(\mathsf{Setup}, \mathsf{Com}, \mathsf{Verify}, \mathsf{Open})$ satisfies the following properties.

Correctness. For any $n, q \in \mathbb{N}, m = \Omega(n \log q), \sigma = \mathsf{poly}(n, m), N = \mathsf{poly}(\lambda)$, any $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{Setup}(1^n, 1^m, q, \sigma)$, any matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$, $\mathbf{C}\mathbf{V}_N = \mathbf{M} - \mathbf{B}\mathbf{Z}$ where $\mathbf{C} = \mathsf{Com}(\mathbf{M}), \mathbf{V}_N = \mathsf{Verify}(1^N), \mathbf{Z} = \mathsf{Open}(\mathbf{M})$.

Compactness. For any $n, q \in \mathbb{N}, m = \Omega(n \log q), \sigma = \mathsf{poly}(n, m), N = \mathsf{poly}(\lambda)$, any $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{Setup}(1^n, 1^m, q, \sigma)$, any matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$, $\|\mathbf{V}_N\| \leq O(\|\mathbf{D}\| m^4 \log q)$ and $\|\mathbf{Z}\| \leq O(\|\mathbf{D}\| m^7 \log q \log N)$ where $\mathbf{V}_N = \mathsf{Verify}(1^N)$ and $\mathbf{Z} = \mathsf{Open}(\mathbf{M})$.

Explainable Re-Randomizer. Essentially, an Explainable Re-randomizer scheme (ER) lets us sample a *random* commitment to a $\mathbf{0} \in \mathbb{Z}_q^{n \times N}$ of MCOM above. An ER scheme consists of the following polynomial time algorithms.

$\mathsf{Setup}(1^n, 1^m, q, \sigma) \to (\mathbf{W}, \mathbf{D}, \mathbf{B})$. Same as Setup for MCOM Definition 4.2. The following algorithms take $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ implicitly.

$\mathsf{Sample}(1^\lambda, 1^N, \sigma_{\mathsf{er}}) \to (\mathbf{C}, \mathbf{Z})$. The randomized sampling algorithm takes as input security parameter λ, column length N, width parameter $\sigma_{\mathsf{er}} > 0$, and outputs matrices $\mathbf{C} \in \mathbb{Z}_q^{n \times m}, \mathbf{Z} \in \mathbb{Z}_q^{m \times N}$.

$\mathsf{Explain}(1^\lambda, 1^T, \mathbf{C}, \mathbf{Z}, \sigma_{\mathsf{er}}) \to r'$. The randomized explanation algorithm takes as input $T = \mathsf{poly}(\lambda)$, matrices $\mathbf{C} \in \mathbb{Z}_q^{n \times m}, \mathbf{Z} \in \mathbb{Z}_q^{m \times N}$, width parameter $\sigma_{\mathsf{er}} > 0$, and outputs $r' \in \{0,1\}^\rho$.

Definition 4.3 ([CHW25, WW25]). An ER scheme $(\mathsf{Setup}, \mathsf{Sample}, \mathsf{Explain})$ is said to be an ER scheme if it satisfies the following properties.

Correctness. For any $\lambda \in \mathbb{N}, N \in \mathbb{N}, (\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{Setup}(1^n, 1^m, q, \sigma)$, there exists a polynomial $\rho = \mathsf{poly}(\lambda, N)$ such that $\forall\ n \geq \lambda, m \geq 2n\lceil \log q \rceil, \sigma_{\mathsf{er}} > 0$, $\mathbf{C}\mathbf{V}_N = -\mathbf{B}\mathbf{Z}$ where $(\mathbf{C}, \mathbf{Z}) \leftarrow \mathsf{Sample}(1^\lambda, 1^N, \sigma_{\mathsf{er}})$, $\mathbf{V}_N = \mathsf{MCOM.Verify}(1^N)$.

Randomness. For any $\lambda \in \mathbb{N}, N \in \mathbb{N}, (\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{Setup}(1^n, 1^m, q, \sigma)$, there exists a polynomial $\rho = \mathsf{poly}(\lambda, N)$ such that $\forall\ n \geq \lambda, m \geq 2n\lceil \log q \rceil, \|\mathbf{D}\|\, O(m^{12} N^3) < \sigma_{\mathsf{er}} < 2^{O(\lambda)}$,

$$\left\{ (\mathbf{C}, \mathbf{Z}) \ : \ \begin{array}{l} r \leftarrow \{0,1\}^\rho, \\ (\mathbf{C}, \mathbf{Z}) = \mathsf{Sample}(1^\lambda, 1^N, \sigma_{\mathsf{er}}; r) \end{array} \right\} \approx_s \left\{ (\mathbf{C}, \mathbf{Z}) \ : \ \begin{array}{l} \mathbf{C} \leftarrow \mathbb{Z}_q^{n \times m}, \\ \mathbf{V}_N = \mathsf{MCOM.Verify}(1^N), \\ \mathbf{Z} \leftarrow \mathbf{B}_{\sigma_{\mathsf{er}}}^{-1}(-\mathbf{C}\mathbf{V}_N) \end{array} \right\}$$

Explainability. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $N \in \mathbb{N}, T = \mathsf{poly}(\lambda)$, $\|\mathbf{D}\| \, O(m^{12}N^3) < \sigma_{\mathsf{er}} < 2^{O(\lambda)}$, the following holds: $\mathsf{SD}(\mathcal{D}_0, \mathcal{D}_1) \leq \frac{1}{T} + \mathsf{negl}(\lambda)$, where $\mathcal{D}_0, \mathcal{D}_1$ are defined as:

$$\mathcal{D}_0 = \left\{ (\mathbf{C}, \mathbf{Z}, r) \ : \ \begin{array}{l} r \leftarrow \{0,1\}^{\rho}, \\ (\mathbf{C}, \mathbf{Z}) = \mathsf{Sample}(1^{\lambda}, 1^{N}, \sigma_{\mathsf{er}}; r) \end{array} \right\},$$

$$\mathcal{D}_1 = \left\{ (\mathbf{C}, \mathbf{Z}, r') \ : \ \begin{array}{l} (\mathbf{C}, \mathbf{Z}) \leftarrow \mathsf{Sample}(1^{\lambda}, 1^{N}, \sigma_{\mathsf{er}}), \\ r' \leftarrow \mathsf{Explain}(1^{\lambda}, 1^{T}, \mathbf{C}, \mathbf{Z}, \sigma_{\mathsf{er}}) \end{array} \right\}$$

4.2 Construction

Construction 4.4. We provide the construction of an EqMCOM scheme (Definition 4.1) using

- An MCOM scheme (Definition 4.2)
- An ER scheme (Definition 4.3)
- A hash function $\mathcal{H} : \{0,1\}^{\lambda} \times \{0,1\}^{\lambda} \to \{0,1\}$ which will be modeled as random oracle in security proof.

$\mathsf{Setup}(1^n, 1^m, q, \sigma, B)$**.** Sample $\widetilde{\sigma} \in \mathbb{N}$ such that $\widetilde{\sigma} > \max\{2^{3\lambda} \cdot \sigma \cdot O(m^{25/2}), \lambda^{\omega(1)} \cdot (B + m^{15/2} \cdot \sigma \cdot \log q \cdot \log N)\}$. Sample hash function $\mathcal{H} : \{0,1\}^{\lambda} \times \{0,1\}^{\lambda} \to \{0,1\}$. Sample $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times 2m^3}$, $\mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times 2m^3}$. Compute $\mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^{\top} \otimes \mathbf{G}$.

Output $\mathsf{pp} := (n, m, q, \sigma, B, \widetilde{\sigma}, \mathcal{H}, \mathbf{W}, \mathbf{D}, \mathbf{B})$.

$\mathsf{Com}(\mathbf{M})$**.** Let N be the column length of $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$. Do the following.

1. Compute $\mathbf{C}_{\mathbf{M}} = \mathsf{MCOM.Com}(\mathbf{M})$.
2. Sample $\mathsf{sd} \leftarrow \{0,1\}^{\lambda}$.
3. Let $\rho = \mathsf{poly}(\lambda, N)$ be the length of the random coins used by ER for column length N. Sample $r_i = \mathcal{H}(\mathsf{sd}, i)$ for each $i \in [\rho]$. Set random coins $r = (r_i)_{i \in [\rho]}$.
4. Compute $(\mathbf{C}_0, \mathbf{Z}_0) = \mathsf{ER.Sample}(1^{\lambda}, 1^{N}, \widetilde{\sigma}; r)$.

Output $\mathbf{C} := \mathbf{C}_0 + \mathbf{C}_{\mathbf{M}}$ and $\mathsf{aux} := \mathsf{sd}$.

$\mathsf{Verify}(1^N)$**.** Compute and output $\mathbf{V}_N = \mathsf{MCOM.Verify}(1^N)$.

$\mathsf{Open}(\mathbf{M}, \mathsf{aux})$**.** Let N be the column length of $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$. Do the following.

1. Compute $\mathbf{Z}_{\mathbf{M}} = \mathsf{MCOM.Open}(\mathbf{M})$.
2. Perform steps 3-4 from Com to get $\mathbf{Z}_0$ using sd.

Output $\mathbf{Z} := \mathbf{Z}_0 + \mathbf{Z}_{\mathbf{M}}$.

$\mathsf{EqCom}^{\mathsf{H}(\cdot,\cdot)}(\mathbf{Y}, \mathbf{R}_{\mathbf{C}})$**.** This algorithm is defined with respect to a programmable random oracle which is treated as a dictionary H. Let N be the column length of $\mathbf{Y} \in \mathbb{Z}_q^{4m^5 \times N}$. Do the following.

1. Sample $\mathsf{sd} \leftarrow \{0,1\}^\lambda$. If there is an entry for (sd, j) for any $j \in \{0,1\}^\lambda$ in H, abort and output $\perp$. Otherwise, proceed as follows.
2. Compute $\mathbf{M} = \mathbf{BY}$. Then compute $\mathbf{C_M} = \mathsf{MCOM.Com}(\mathbf{M}), \mathbf{V}_N = \mathsf{MCOM.Verify}(1^N)$, and $\mathbf{Z_M} = \mathsf{MCOM.Open}(\mathbf{M})$.
3. Set $\mathbf{C}_0 = \mathbf{BR_C} - \mathbf{C_M}$ and $\mathbf{Z}_0 = \mathbf{Y} - \mathbf{Z_M} - \mathbf{R_C V}_N$.
4. Sample $r' \leftarrow \mathsf{ER.Explain}(1^\lambda, 1^T, \mathbf{C}_0, \mathbf{Z}_0, \widetilde{\sigma})$.
5. Set $\mathsf{H}[(\mathsf{sd}, i)] = r'_i$ for each $i \in [\rho]$ where $\rho = |r'|$ is the length of the random coins used by ER for column length N.

Output $\mathsf{aux} := \mathsf{sd}$.

Correctness. By correctness of MCOM, we have that $\mathbf{C_M V}_N = \mathbf{M} - \mathbf{BZ_M}$ for the values computed in Com. Moreover, if sd is same in Com and Open, we have that same $\mathbf{C}_0, \mathbf{Z}_0$ are computed in Com and Open. Then, by correctness of ER, we have that $\mathbf{C}_0\mathbf{V}_N = -\mathbf{BZ}_0$. Hence, $\mathbf{CV}_N = (\mathbf{C}_0 + \mathbf{C_M})\mathbf{V}_N = \mathbf{M} - \mathbf{B}(\mathbf{Z}_0 + \mathbf{Z_M}) = \mathbf{M} - \mathbf{BZ}$. Thus, correctness holds.

Compactness. As $\mathsf{aux} \in \{0,1\}^\lambda$, we have that compactness for aux holds and $p_1(m) = 2m^3$ and $p_2(m) = 2m^2$. Moreover, $\|\mathbf{V}_N\| \leq O(\|\mathbf{D}\| m^4 \log q)$ by compactness of MCOM. Similarly, $\|\mathbf{Z}\| \leq \|\mathbf{Z}_0\| + \|\mathbf{Z_M}\| \leq \sqrt{m}\widetilde{\sigma} + O(m^{15/2}\sigma \log q \log N) = O(\sqrt{m}\widetilde{\sigma})$ by our parameter choices.

4.3 Security Analysis

Theorem 4.5. If MCOM is a matrix commitment scheme (Definition 4.2), ER is an explainable re-randomizer scheme (Definition 4.3), then Construction 4.4 is an equivocal matrix commitment scheme (Definition 4.1) in the random oracle model.

Proof Sketch. We define the following hybrids.

Hybrid 0. This is the commitment indistinguishability game from Definition 4.1 with $b = 0$.

Hybrid 1. In this hybrid, we abort and output a random bit if the seed sd sampled during the challenge phase already appears in H, and we program $\mathsf{H}[(\mathsf{sd}, i)] = r_i$ for uniformly sampled $r = (r_1, \ldots, r_\rho)$ instead of obtaining r by querying $\mathcal{H}$. As $\mathcal{A}$ is a PPT machine, the abort occurs with probability at most $Q/2^\lambda$ for a polynomial Q and conditioned on not aborting, the reprogramming is identically distributed.

Hybrid 2. In this hybrid, we sample $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ together with a trapdoor $\mathbf{T_B}$ during setup. Hybrids 1 and 2 are statistically close by the matrix and pre-image distribution properties of gadget trapdoors (Lemma A.4).

Hybrid 3. In this hybrid, generation queries return a uniformly sampled $\mathbf{t} \leftarrow \mathbb{Z}_q^n$, and the corresponding $\mathbf{y} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, \mathbf{t}, \widetilde{\sigma})$ is sampled on-the-fly, only if the handle is later corrupted. Hybrids 2 and 3 are statistically close by the gaussian pre-image distribution (Lemma A.6 and Lemma A.1).

Hybrid 4. In this hybrid, we sample $\mathbf{C}_0 \leftarrow \mathbb{Z}_q^{n \times m}$ and $\mathbf{Z}_0 \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, -\mathbf{C}_0\mathbf{V}_N, \widetilde{\sigma})$ during the challenge phase, and program H with coins $r' \leftarrow \mathsf{ER.Explain}(1^\lambda, 1^T, \mathbf{C}_0, \mathbf{Z}_0, \widetilde{\sigma})$ that explain this pair. Hybrids 3 and 4 are within statistical distance $1/T + \mathsf{negl}(\lambda)$ by the explainability and randomness properties of ER. We discuss the choice of T later in the proof.

Hybrid 5. In this hybrid, we sample $\mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$ and set $\mathbf{C}_0 = \mathbf{BR_C} - \mathbf{C_M}$, so that the commitment sent to $\mathcal{A}$ is $\mathbf{C} = \mathbf{BR_C}$. Hybrids 4 and 5 are statistically close by leftover hash lemma (Lemma A.5).

Hybrid 6. In this hybrid, we set $\mathbf{Z}_0 = \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, [\mathbf{t}_{h_1} \mid \ldots \mid \mathbf{t}_{h_N}], \widetilde{\sigma}) + \mathbf{R} - \mathbf{Z_M} - \mathbf{R_C}\mathbf{V}_N$. Hybrids 5 and 6 are statistically close by the gaussian pre-image smudging lemma (Lemma A.7 and Lemma A.2).

Hybrid 7. In this hybrid, generation queries again sample $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and return $\mathbf{By}$, so that $\mathbf{Z}_0 = (\mathbf{Y} + \mathbf{R}) - \mathbf{Z_M} - \mathbf{R_C}\mathbf{V}_N$. As $\mathbf{T_B}$ is then unused, setup reverts to sampling $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ without a trapdoor. Hybrids 6 and 7 are statistically close by the gaussian pre-image distribution and Lemma A.4. This is the commitment indistinguishability game from Definition 4.1 with $b = 1$.

We provide full proof in Appendix B. □

5 (Equivocal) Flexible Broadcast Encryption

5.1 Flexible Broadcast Encryption

Syntax. A Flexible Broadcast Encryption scheme (FBE) consists of the following polynomial-time algorithms.

$\mathsf{GSetup}(1^\lambda) \to \mathsf{crs}$**.** The possibly randomized *public-coin* global setup algorithm takes as input security parameter λ and outputs common random string crs. The following algorithms take crs implicitly.

$\mathsf{KGen} \to (\mathsf{pk}, \mathsf{sk})$**.** The randomized key generation algorithm outputs public-secret key pair $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Enc}((\mathsf{pk}_1, \ldots, \mathsf{pk}_N), \mu) \to \mathsf{ct}$**.** The randomized encryption algorithm takes as input N public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_N$, message $\mu \in \{0,1\}$, and outputs a ciphertext ct.

$\mathsf{Dec}((\mathsf{pk}_1, \ldots, \mathsf{pk}_N), (i, \mathsf{sk}_i), \mathsf{ct}) \to \mu'$ The deterministic decryption algorithm takes as input N public keys $(\mathsf{pk}_1, \ldots, \mathsf{pk}_N)$, secret key for $i \in [N]$, sk_i, ciphertext ct, and outputs $\mu' \in \{0,1\} \cup \{\perp\}$.

Compressing public keys. The above is the standard syntax for FBE. We can also define a *public-coin* pre-processing algorithm that takes as input $(\mathsf{pk}_1, \ldots, \mathsf{pk}_N)$ (in above definition this is given as input to Enc) and produces short 'master public key' mpk and decryption hints dht_i for each $i \in [N]$. This way, we can drive the running time of Enc and Dec to be independent of N. Similar syntax was considered in the context of registered ABE [GV20, HLWW23] and distributed monotone-policy encryption [GKPW24, WW26, AGY26].

Definition 5.1 (FBE)**.** A FBE scheme $(\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is said to be a FBE scheme if it satisfies the following properties.

Correctness. For any $\lambda \in \mathbb{N}, N = \mathsf{poly}(\lambda)$, any $i \in [N], \mu \in \{0,1\}$,

$$\Pr\left[\begin{array}{c|l} \mu = \mathsf{Dec}((\mathsf{pk}_1, \ldots, \mathsf{pk}_N), & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda), \\ (i, \mathsf{sk}_i), \mathsf{ct}) & \forall\ j \in [N], (\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{KGen}, \\ & \mathsf{ct} \leftarrow \mathsf{Enc}((\mathsf{pk}_1, \ldots, \mathsf{pk}_N), \mu) \end{array}\right] = 1$$

Compactness. There exists polynomials $\mathsf{poly}_{\mathsf{crs}}(\cdot), \mathsf{poly}_{\mathsf{pk}}(\cdot), \mathsf{poly}_{\mathsf{sk}}(\cdot), \mathsf{poly}_{\mathsf{ct}}(\cdot,\cdot)$ such that for any $\lambda \in \mathbb{N}, N \leq 2^\lambda$, $|\mathsf{crs}| = \mathsf{poly}_{\mathsf{crs}}(\lambda), |\mathsf{pk}| = \mathsf{poly}_{\mathsf{pk}}(\lambda), |\mathsf{sk}| = \mathsf{poly}_{\mathsf{sk}}(\lambda), |\mathsf{ct}| = \mathsf{poly}_{\mathsf{ct}}(\lambda, \log N)$.

Adaptive Security. For any stateful admissible PPT adversary $\mathcal{A}$, there exists a negligible function such that $\forall\ \lambda \in \mathbb{N}$,

$$\left|\Pr\left[\begin{array}{lcl} & & \mathcal{K} := \emptyset, h := 1 \\ b \leftarrow \mathcal{A}(\mathsf{ct}) & : & \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda), \\ & & S \leftarrow \mathcal{A}^{\mathcal{O}(\cdot,\cdot)}(\mathsf{crs}), b \leftarrow \{0,1\}, \\ & & \mathsf{ct} \leftarrow \mathsf{Enc}((\mathsf{pk}_1, \ldots, \mathsf{pk}_N), b) \end{array}\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)$$

Here, $(\mathsf{pk}_1, \ldots, \mathsf{pk}_N)$ are derived as follows: let $(h_1, \ldots, h_N)$ be lexicographical ordering of S with each $h_i \in [h]$ for $i \in [N]$. Then, $\mathsf{pk}_i = \mathsf{pk}_{h_i}$. $\mathcal{O}$ is a stateful oracle that responds to the following queries.

- $(\mathsf{KGen}, \mathsf{id})$**.** $\mathcal{A}$ queries for key generation. $\mathcal{O}$ samples $(\mathsf{pk}_h, \mathsf{sk}_h) \leftarrow \mathsf{KGen}$, updates $\mathcal{K} := \mathcal{K} \cup \{(h, \mathsf{pk}_h, \mathsf{sk}_h)\}$ and sends (h, pk_h) to $\mathcal{A}$. Then, $\mathcal{O}$ updates $h := h + 1$.
- (Corr, h')**.** $\mathcal{A}$ queries for key corruption for handle $h' \in [h]$. If there is no entry for $(h', \mathsf{pk}_{h'}, \mathsf{sk}_{h'})$ in $\mathcal{K}$ send $\perp$ to $\mathcal{A}$. Otherwise, $\mathcal{O}$ responds with $\mathsf{sk}_{h'}$ to $\mathcal{A}$. $\mathcal{O}$ also updates $\mathcal{K} := \mathcal{K} \setminus \{(h', \mathsf{pk}_{h'}, \mathsf{sk}_{h'})\}$.

A stateful PPT machine $\mathcal{A}$ is said to be admissible if $S \subseteq [h]$ for each $h^* \in S$, there exists an entry $(h^*, \mathsf{pk}_{h^*}, \mathsf{sk}_{h^*}) \in \mathcal{K}$.

Definition 5.2 (Semi-Static Security)**.** An FBE scheme $(\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is said to be a semi-statically secure FBE scheme if satisfies correctness and compactness of Definition 5.1 and semi-static security which is the same as adaptive security except that $\mathcal{A}$ is no longer allowed any (Corr, h') queries.

Theorem 5.3 ([FWW23])**.** Assuming the existence of a semi-statically secure FBE scheme (Definition 5.2), there exists an adaptively-secure FBE scheme in the random oracle model.

5.2 Equivocal Flexible Broadcast Encryption

In this section, we instantiate our equivocal encryption framework in the context of FBE schemes and define Equivocal Flexible Broadcast Encryption (EqFBE).

Syntax. An Equivocal Flexible Broadcast Encryption scheme (EqFBE) consists of the following polynomial time algorithms.

$\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}$**.** These are defined similarly to an FBE scheme (Definition 5.1).

$\mathsf{FakeSetup}(1^\lambda, \mu) \to (\widetilde{\mathsf{crs}}, \widetilde{\mathsf{ct}}, \mathsf{aux})$**.** The randomized fake setup algorithm takes as input security parameter λ, message $\mu \in \{0,1\}$, outputs fake CRS, $\widetilde{\mathsf{crs}}$, fake ciphertext $\widetilde{\mathsf{ct}}$, and auxiliary information aux.

$\mathsf{SimKGen}(\mathsf{aux}, \mathsf{st}) \to (\mathsf{pk}, \mathsf{st}')$**.** The randomized *stateful* key generation simulation algorithm takes as input auxiliary information aux, state st, and outputs a public key, pk, and updated state st'.

$\mathsf{Equivocate}(\mathsf{aux}, \mathsf{st}, S) \to \mathsf{ct}$. The possibly randomized equivocation algorithm takes as input auxiliary information aux, state st, set S, and outputs ciphertext ct.

Definition 5.4. An EqFBE scheme ($\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FakeSetup}, \mathsf{SimKGen}, \mathsf{Equivocate}$) should satisfy the following properties.

Correctness and Compactness. These are defined for ($\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}$) exactly as correctness and compactness for an FBE scheme (Definition 5.1).

Mode Indistinguishability. We define the following game between a stateful PPT adversary $\mathcal{A}$ and a challenger Chal.

1. $\mathcal{A}$ sends $\mu \in \{0,1\}$ to Chal. Chal flips a coin $b \leftarrow \{0,1\}$. Chal initiates dictionary $\mathcal{K} := \emptyset$ and counter $h := 1$.
2. If $b = 0$, Chal samples $\mathsf{crs}^{(0)} \leftarrow \mathsf{GSetup}(1^\lambda)$, and sends $\mathsf{crs}^{(0)}$ to $\mathcal{A}$.
   Otherwise, if $b = 1$, Chal samples $(\mathsf{crs}^{(1)}, \widetilde{\mathsf{ct}}, \mathsf{aux}) \leftarrow \mathsf{FakeSetup}(1^\lambda, \mu)$ and sends $\mathsf{crs}^{(1)}$ to $\mathcal{A}$. Chal initiates $\mathsf{st} := (h, \mathcal{K})$.
3. $\mathcal{A}$ (polynomially-many times) queries for a key. If $b = 0$, Chal computes $(\mathsf{pk}_h^{(0)}, \mathsf{sk}_h) \leftarrow \mathsf{KGen}$ and sends $(h, \mathsf{pk}_h^{(0)})$ to $\mathcal{A}$. Chal also updates $\mathcal{K} := \mathcal{K} \cup \{(h, \mathsf{pk}_h^{(0)}, \mathsf{sk}_h)\}$ and $h := h+1$.
   Otherwise, if $b = 1$, Chal computes $(\mathsf{pk}_h^{(1)}, \mathsf{st}') \leftarrow \mathsf{SimKGen}(\mathsf{aux}, \mathsf{st})$ and sends $(h, \mathsf{pk}_h^{(1)})$ to $\mathcal{A}$. Chal also updates $\mathsf{st} := \mathsf{st}'$.
4. $\mathcal{A}$ sends $S \subseteq [h]$. Let $|S| = N$. Chal computes $(\mathsf{pk}_{h^*})_{h^* \in S}$ and orders them as $(\mathsf{pk}_1, \ldots, \mathsf{pk}_N)$ according to h^*. If $b = 0$, Chal samples $\mathsf{ct}^{(0)} \leftarrow \mathsf{Enc}((\mathsf{pk}_i)_{i \in [N]}, \mu)$ and sends $\mathsf{ct}^{(0)}$ to $\mathcal{A}$.
   Otherwise, if $b = 1$, Chal samples $\mathsf{ct}^{(1)} \leftarrow \mathsf{Equivocate}(\mathsf{aux}, \mathsf{st}, S)$ and sends $\mathsf{ct}^{(1)}$ to $\mathcal{A}$.
5. $\mathcal{A}$ guess b as b'.

We say that EqFBE satisfies mode indistinguishability property if for any stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$,

$$\left| \Pr[b' = b] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda)$$

Selective Security for Fake Setup. For any stateful admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$,

$$\left| \Pr\left[b \leftarrow \mathcal{A}(\widetilde{\mathsf{crs}}, \widetilde{\mathsf{ct}}, \mathsf{aux}) \quad : \quad \begin{array}{l} b \leftarrow \{0,1\}, \\ (\widetilde{\mathsf{crs}}, \widetilde{\mathsf{ct}}, \mathsf{aux}) \leftarrow \mathsf{FakeSetup}(1^\lambda, b) \end{array} \right] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda)$$

5.3 Semi-Static FBE from EqFBE

Construction 5.5. We construct a semi-statically secure FBE scheme (Definition 5.2) using an EqFBE scheme (Definition 5.4) as follows.

$\mathsf{GSetup}(1^\lambda)$**.** Sample $\mathsf{fbe.crs} \leftarrow \mathsf{EqFBE.GSetup}(1^\lambda)$. Output $\mathsf{crs} := \mathsf{fbe.crs}$.

KGen**.** Sample $(\mathsf{fbe.pk}, \mathsf{fbe.sk}) \leftarrow \mathsf{EqFBE.KGen}$. Output $(\mathsf{pk}, \mathsf{sk}) := (\mathsf{fbe.pk}, \mathsf{fbe.sk})$.

$\mathsf{Enc}((\mathsf{pk}_1, \ldots, \mathsf{pk}_N), \mu)$. Parse pk_i as $\mathsf{fbe.pk}_i$ for $i \in [N]$.

Sample $\mathsf{fbe.ct} \leftarrow \mathsf{EqFBE.Enc}((\mathsf{fbe.pk}_i)_{i\in[N]}, \mu)$. Output $\mathsf{ct} := \mathsf{fbe.ct}$.

$\mathsf{Dec}((\mathsf{pk}_1, \ldots, \mathsf{pk}_N), (i, \mathsf{sk}_i), \mathsf{ct})$. Parse pk_j as $\mathsf{fbe.pk}_j$ for $j \in [N]$, sk_i as $\mathsf{fbe.sk}_i$, and ct as $\mathsf{fbe.ct}$.

Output $\mu' := \mathsf{EqFBE.Dec}((\mathsf{fbe.pk}_j)_{j\in[N]}, (i, \mathsf{fbe.sk}_i), \mathsf{fbe.ct})$.

Correctness and Compactness. This follows from correctness and compactness of EqFBE.

Theorem 5.6. If EqFBE is an EqFBE scheme (Definition 5.4), then Construction 5.5 is a semi-statically secure FBE scheme (Definition 5.2).

Proof Sketch. We prove this theorem using the following hybrids for an admissible adversary.

Hybrid 0. This is the semi-static security experiment from Definition 5.2 with $b = 0$.

Hybrid 1. In this hybrid, we will use $\mathsf{FakeSetup}(1^\lambda, 0)$ to generate $(\widetilde{\mathsf{crs}}, \widetilde{\mathsf{ct}}, \mathsf{aux})$. We will also use aux and $\mathsf{SimKGen}$ to generate pk_h for any key generation queries from $\mathcal{A}$. Finally, when $\mathcal{A}$ sends $S \subseteq [h]$, we use $\mathsf{Equivocate}(\mathsf{aux}, \mathsf{st}, S)$ to generate the ciphertext ct. Hybrids 0 and 1 are indistinguishable because of the mode indistinguishability property of EqFBE.

Hybrid 2. In this hybrid, we will encrypt message 1 using $\mathsf{EqFBE.FakeSetup}$ algorithm. The hybrids remain indistinguishable due to selective security for fake setup property of EqFBE.

Hybrid 3. This is the experiment from Definition 5.2 with $b = 1$. Indistinguishability follows by mode indistinguishability property of EqFBE.

We provide the full proof in Appendix C. □

6 Equivocal Flexible Broadcast Encryption from Decomposed LWE

Parameters. We provide the parameters used by our construction below.

- $n = \mathsf{poly}(\lambda)$ (Decomposed LWE security)
- $m = \Omega(n \log q)$ (Leftover hash lemma)
- $\chi = \mathsf{poly}(\lambda)$ (Decomposed LWE security)
- $\widetilde{\chi} > \lambda^{\omega(1)} m^{11/2} \chi$ (Smudging)
- $\sigma = \omega(\log m)$ (Decomposed LWE security)
- Odd prime $q > O(\lambda \cdot \chi \cdot \widetilde{\chi} \cdot \sigma \cdot \widetilde{\sigma} \cdot m^{10} \cdot \log q)$ (Correctness)

One way to instantiate these parameters is for constant $0 < \epsilon < 1/2$, $n = (\lambda)^{\Theta(1/\epsilon)} \cdot \mathsf{poly}(\log \lambda)$, $m = 4n \log q$, $\chi = \mathsf{poly}(n)$, $\sigma = \log^2 m$, and $q = \mathsf{poly}(n, 2^\lambda)$. This would require sub-exponential modulus-to-noise ratio.

Construction 6.1. We construct an EqFBE scheme assuming the hardness of $2m^2$-decomposed LWE using an Equivocal Matrix commitment scheme EqMCOM (Definition 4.1) as follows.

$\mathsf{GSetup}(1^\lambda)$. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{EqMCOM.Setup}(1^n, 1^m, q, \sigma, 0)$. Sample $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. Output $\mathsf{crs} := (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p})$.

$\mathsf{KGen}(\mathsf{crs})$. Sample $\mathbf{r} \leftarrow \{\pm 1\}^{4m^5}$ and compute $\mathbf{t} := \mathbf{Br} + \mathbf{p}$.

Output $\mathsf{pk} := \mathbf{t}$ and $\mathsf{sk} := \mathbf{r}$.

$\mathsf{Enc}((\mathsf{pk}_1, \ldots, \mathsf{pk}_N), \mu)$. Parse pk_i as $\mathbf{t}_i$ for each $i \in [N]$. Do the following.

1. Compute matrix $\mathbf{M} = [\mathbf{t}_1 \mid \ldots \mid \mathbf{t}_N] \in \mathbb{Z}_q^{n \times N}$.
2. Sample $(\mathbf{C}, \mathsf{aux}) \leftarrow \mathsf{EqMCOM.Com}(\mathbf{M})$.
3. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$.
4. Compute $\mathbf{c}_1^\top := \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$, $\mathbf{c}_2^\top := \mathbf{s}^\top \mathbf{C} + \widetilde{\mathbf{e}}^\top$, and $c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu$.

Output $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{aux}) \in \mathbb{Z}_q^{1 \times (4m^5 + m + 1)} \times \{0,1\}^{\mathsf{poly}(\lambda, \log N)}$.

$\mathsf{Dec}((\mathsf{pk}_1, \ldots, \mathsf{pk}_N), (i, \mathsf{sk}_i), \mathsf{ct})$. Parse pk_j as $\mathbf{t}_j$ for each $j \in [N]$, sk_i for $i \in [N]$ as $\mathbf{r}_i$, and ct as $(\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{aux})$. Do the following.

1. Compute $\mathbf{Z} = \mathsf{EqMCOM.Open}(\mathbf{M}, \mathsf{aux})$ where $\mathbf{M}$ is computed like in step 1 of Enc.
2. Also, compute $\mathbf{V}_N = \mathsf{EqMCOM.Verify}(1^N)$. Let $\mathbf{v}_i, \mathbf{z}_i$ denote the i-th columns of $\mathbf{V}_N, \mathbf{Z}$ respectively.
3. Compute $c_i = \mathbf{c}_2^\top \mathbf{v}_i + \mathbf{c}_1^\top (\mathbf{z}_i - \mathbf{r}_i)$. Compute $y = c_3 - c_i$.

Output $\mu' := 0$ if $-q/4 < y < q/4$ and $\mu' := 1$ otherwise.

Correctness. We have that $\mathsf{pk}_j = \mathbf{t}_j \in \mathbb{Z}_q^n$ and $\mathsf{sk}_j = \mathbf{r}_j \in \{\pm 1\}^{4m^5}$ for $j \in [N]$ where $\mathbf{t}_j = \mathbf{B}\mathbf{r}_j + \mathbf{p}$. By correctness of EqMCOM, we have that $\mathbf{C}\mathbf{V}_N = \mathbf{M} - \mathbf{B}\mathbf{Z}$ where $\mathbf{Z} = \mathsf{EqMCOM.Open}(\mathbf{M}, \mathsf{aux})$. So, $\mathbf{C}\mathbf{v}_i = \mathbf{t}_i - \mathbf{B}\mathbf{z}_i$.

We have that

$$\begin{aligned} c_i &= \mathbf{c}_2^\top \mathbf{v}_i + \mathbf{c}_1^\top (\mathbf{z}_i - \mathbf{r}_i) \\ &= \mathbf{s}^\top (\mathbf{C}\mathbf{v}_i + \mathbf{B}\mathbf{z}_i - \mathbf{B}\mathbf{r}_i) + \underbrace{\widetilde{\mathbf{e}}^\top (\mathbf{v}_i) + \mathbf{e}^\top \mathbf{z}_i - \mathbf{r}_i)}_{\widetilde{e}} \\ &= \mathbf{s}^\top (\mathbf{t}_i - \mathbf{B}\mathbf{r}_i) + \widetilde{e} \\ &= \mathbf{s}^\top \mathbf{p} + \widetilde{e} \end{aligned}$$

Thus, $y = \lfloor q/2 \rfloor \mu + \mathbf{e}^\top \mathbf{r_p} - \widetilde{e}$ and $\mu' = \mu$ as long as $\left\| \mathbf{e}^\top \mathbf{r_p} - \widetilde{e} \right\| < q/4$. We will now calculate the norm of $\mathbf{e}^\top \mathbf{r_p} - \widetilde{e}$. Note that $\mathbf{e}^\top \mathbf{r_p} - \widetilde{e} = \mathbf{e}^\top (\mathbf{r_p} - \mathbf{z}_i + \mathbf{r}_i) + \widetilde{\mathbf{e}}^T \mathbf{v}_i$. By construction, $\|\mathbf{r_p}\| = \|\mathbf{r}_i\| = 1$. With high probability, $\|\mathbf{D}\| \le \sigma\sqrt{m}$ and $\|\mathbf{e}\| \le \chi\sqrt{m}$. By compactness property of EqMCOM, we have $\|\mathbf{v}_i\| \le \|\mathbf{V}_N\| = O(\sigma m^{9/2} \log q)$ and $\|\mathbf{z}_i\| \le \|\mathbf{Z}\| \le O(\widetilde{\sigma}\sqrt{m})$.

Putting all of this together, $|\mathbf{e}^\top \mathbf{r_p} - \widetilde{e}| \le O(\chi\widetilde{\chi}\sigma\widetilde{\sigma}m^{10} \log q \log N)$. As $\log N \le \lambda$ (very loosely), as long as $q > O(\lambda\chi\widetilde{\chi}\sigma\widetilde{\sigma}m^{10} \log q)$, correctness holds.

Compactness. Size of ct is $(4m^5 + m + 1)\lceil \log q \rceil + \mathsf{poly}(\lambda, \log N) = O(m^5 \log q) + \mathsf{poly}(\lambda, \log N) = \mathsf{poly}(\lambda, \log N) \le \mathsf{poly}(\lambda)$ by our parameters. Size of pk is $n\lceil \log q \rceil$ and sk is $4m^5$ both of which are polynomially bounded in λ. Size of crs is $(2m^2(nm + m^2) + n)\lceil \log q \rceil + \mathsf{poly}(\lambda)$. We need not include $\mathbf{B}$ in this calculation as it is efficiently derivable from $\mathbf{W}$ and $\mathbf{D}$. This is a polynomial in $n, \log q$ which is in turn a polynomial in λ. Moreover, since crs is public coin, we can replace it with a random oracle to get even more compact crs.

6.1 Fake Setup, Key Generation Simulation, and Equivocation

We now describe our $\mathsf{FakeSetup}, \mathsf{SimKGen}$, and $\mathsf{Equivocate}$ algorithms.

$\mathsf{FakeSetup}(1^\lambda, \mu)$**.** Given $\mu \in \{0,1\}$, do the following.

1. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{EqMCOM.Setup}(1^n, 1^m, q, \sigma, 0)$. Sample $\mathbf{p} \leftarrow \mathbb{Z}_q^n$.
2. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$.
3. Set $\widetilde{\mathsf{crs}} := (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p})$, $\widetilde{\mathsf{ct}} := (\mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top, \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu)$, and $\mathsf{aux} := (\widetilde{\mathsf{ct}}, \mathbf{R_C})$.

Output $(\widetilde{\mathsf{crs}}, \widetilde{\mathsf{ct}}, \mathsf{aux})$.

$\mathsf{SimKGen}(\mathsf{aux}, \mathsf{st})$**.** Parse st as $(h, \mathcal{K})$. Sample $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma_{\mathsf{er}}}^{4m^5}$ and compute $\mathbf{t} := \mathbf{B}\mathbf{y}$. Set $\mathsf{pk} := \mathbf{t}$. Set $\mathsf{st}' := (h+1, \mathcal{K} \cup \{(h, \mathsf{pk}, \mathbf{y})\})$.

Output $(\mathsf{pk}, \mathsf{st}')$.

$\mathsf{Equivocate}(\mathsf{aux}, \mathsf{st}, S)$**.** Parse aux as $(\mathbf{c}_1^\top, c_3, \mathbf{R_C})$. Parse st as $(h, \mathcal{K})$. Let $|S| = N$. If $S \not\subseteq [h]$, abort and output $\perp$. Otherwise, retrieve $(\mathsf{pk}_{h^*})_{h^* \in S}$ and order them as $\mathbf{t}_1, \ldots, \mathbf{t}_N$ based on h^*. Use $S, \mathcal{K}$ to get $\mathbf{y}_1, \ldots, \mathbf{y}_N$ for these public keys. Then, do the following.

1. Compute $\mathbf{Y} = [\mathbf{y}_1 \mid \ldots \mid \mathbf{y}_N]$ and $\mathbf{M} = \mathbf{B}\mathbf{Y}$.
2. Sample $\mathsf{aux}' \leftarrow \mathsf{EqMCOM.EqCom}(\mathbf{Y}, \mathbf{R_C})$.
3. Sample $\widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m$.

Output $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_1^\top \mathbf{R_C} + \widetilde{\mathbf{e}}^\top, c_3, \mathsf{aux}')$.

6.2 Mode Indistinguishability Analysis

Theorem 6.2 (Mode Indistinguishability.)**.** If EqMCOM satisfies commitment indistinguishability, then Construction 6.1 satisfies mode indistinguishability property.

Proof Sketch. We define the following hybrids.

Hybrid 0. This is the mode indistinguishability experiment from Definition 5.4 with $b = 0$.

Hybrid 1. In this hybrid, we sample the public key $\mathbf{t} \leftarrow \mathbb{Z}_q^n$ for every key generation query made by $\mathcal{A}$. Hybrids 0 and 1 are statistically indistinguishable by leftover hash lemma (Lemma A.3).

Hybrid 2. In this hybrid, we sample $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and set $\mathbf{t} := \mathbf{B}\mathbf{y}$ for every key generation query. Hybrids 1 and 2 are statistically indistinguishable by the gaussian pre-image distribution (Lemma A.6).

Hybrid 3. In this hybrid, during the encryption phase, we sample $\mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$, set $\mathbf{C} = \mathbf{B}\mathbf{R_C}$, and sample $\mathsf{aux}' \leftarrow \mathsf{EqMCOM.EqCom}(\mathbf{Y}, \mathbf{R_C})$. Hybrids 2 and 3 are computationally indistinguishable by the commitment indistinguishability property of EqMCOM, with $\mathbf{R} = \mathbf{0} \in \mathbb{Z}_q^{4m^5 \times N}$ and no corruption queries.

Hybrid 4. In this hybrid, we set $\mathbf{c}_2^\top := \mathbf{c}_1^\top \mathbf{R_C} + \widetilde{\mathbf{e}}^\top$ during the encryption phase. Hybrids 3 and 4 are statistically close by the gaussian smudging lemma (Lemma 3.5).

Hybrid 5. This is the mode indistinguishability experiment from Definition 5.4 with $b = 1$. The changes between hybrids 4 and 5 are simply semantic.

We provide full proof in Appendix D.1. □

6.3 Selective Security for Fake Setup Analysis

Theorem 6.3 (Selective Security for Fake Setup.)**.** If $2m^2$-decomposed LWE assumption (Assumption 3.7) holds, then Construction 6.1 satisfies selective security for fake setup property.

Proof Sketch. We define the following hybrids.

Hybrid 0. This is the selective security for fake setup experiment from Definition 5.4 for any $\mu \in \{0,1\}$.

Hybrid 1. In this hybrid, we set $\mathbf{p} := \mathbf{B}\mathbf{r}_{\mathbf{p}}$, so that $\widetilde{\mathsf{ct}} = (\mathbf{c}_1^\top, \mathbf{c}_1^\top \mathbf{r}_{\mathbf{p}} + \lfloor q/2 \rfloor \mu)$ where $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$. Hybrids 0 and 1 are statistically close by leftover hash lemma (Lemma A.3).

Hybrid 2. In this hybrid, we sample $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1\times 4m^5}$. Hybrids 1 and 2 are computationally indistinguishable by the $2m^2$-decomposed LWE assumption (Assumption 3.7).

Hybrid 3. In this hybrid, we sample $\widetilde{\mathsf{ct}} \leftarrow \mathbb{Z}_q^{1\times(4m^5+1)}$, which is independent of μ. Hybrids 2 and 3 are statistically indistinguishable by leftover hash lemma (Lemma A.3), using the fact that $[\mathbf{B}^\top \mid \mathbf{c}_1^\top]^\top \mathbf{r}_{\mathbf{p}} \approx_s [\mathbf{p}^\top \mid c_3]$ given random $[\mathbf{B}^\top \mid \mathbf{c}_1^\top]^\top$.

We provide full proof in Appendix D.2. □

Corollary 6.4. Assuming the hardness of $2m^2$-decomposed LWE assumption (Assumption 3.7), there exists an adaptively secure flexible broadcast encryption scheme in the random oracle model.

Proof. By Theorem 6.2 and Theorem 6.3 we have an EqFBE scheme and by Theorem 5.6, we have a semi-static FBE scheme in random oracle mode. Then, Theorem 5.3 gives an adaptively secure FBE scheme in random oracle model. □

7 (Equivocal) Identity-Based Broadcast Encryption

7.1 Identity-Based Broadcast Encryption

Syntax. An Identity-Based Broadcast Encryption scheme (IBBE) for identity space $\mathcal{I} = \{\{0,1\}^\kappa\}_{\kappa\in\mathbb{N}}$ consists of the following polynomial time algorithms.

$\mathsf{Setup}(1^\lambda, 1^\kappa) \to (\mathsf{mpk}, \mathsf{msk})$**.** The randomized setup algorithm takes as input security parameter λ, identity space index κ, outputs master public key mpk and master secret key msk.

$\mathsf{KGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{id}) \to \mathsf{sk}_{\mathsf{id}}$**.** The possibly randomized key generation algorithm takes as input master public and secret keys $\mathsf{mpk}, \mathsf{msk}$, identity $\mathsf{id} \in \{0,1\}^\kappa$, and outputs secret key $\mathsf{sk}_{\mathsf{id}}$.

$\mathsf{Enc}(\mathsf{mpk}, S, \mu) \to \mathsf{ct}$**.** The randomized encryption algorithm takes as input master public key mpk, a set $S \subseteq \{0,1\}^\kappa$, message $\mu \in \{0,1\}$, and outputs ciphertext ct.

$\mathsf{Dec}(\mathsf{mpk}, S, \mathsf{sk}_{\mathsf{id}}, \mathsf{ct}) \to \mu'$**.** The deterministic decryption algorithm takes as input master public key mpk, set $S \subseteq \{0,1\}^\kappa$, secret key for id, $\mathsf{sk}_{\mathsf{id}}$, ciphertext ct, and outputs $\mu' \in \{0,1\} \cup \{\bot\}$.

Definition 7.1 (IBBE)**.** An IBBE scheme $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is said to be an IBBE scheme for $\mathcal{I} = \{\{0,1\}^\kappa\}_{\kappa \in \mathbb{N}}$ if it satisfies the following properties.

Correctness. For any $\lambda \in \mathbb{N}, \kappa = \mathsf{poly}(\lambda), S \subseteq \{0,1\}^\kappa, \mathsf{id} \in S, \mu \in \{0,1\}$,

$$\Pr\left[\mu = \mathsf{Dec}(\mathsf{mpk}, S, \mathsf{sk}_{\mathsf{id}}, \mathsf{ct}) \quad : \quad \begin{array}{l}(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\kappa), \\ \mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{KGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{id}), \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, S, \mu)\end{array}\right] = 1$$

Compactness. There exist polynomials $\mathsf{poly}_{\mathsf{mpk}}(\cdot,\cdot), \mathsf{poly}_{\mathsf{sk}}(\cdot,\cdot), \mathsf{poly}_{\mathsf{ct}}(\cdot,\cdot)$ such that for any $\lambda \in \mathbb{N}, \kappa = \mathsf{poly}(\lambda), \mathsf{id} \in \{0,1\}^\kappa, S \subseteq \{0,1\}^\kappa$, $|\mathsf{mpk}| = \mathsf{poly}_{\mathsf{mpk}}(\lambda, \kappa), |\mathsf{sk}_{\mathsf{id}}| = \mathsf{poly}_{\mathsf{sk}}(\lambda, \kappa), |\mathsf{ct}| = \mathsf{poly}_{\mathsf{ct}}(\lambda, \kappa)$.

Adaptive Security. For any stateful admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$, the following probability is at most $1/2 + \mathsf{negl}(\lambda)$.

$$\Pr\left[b \leftarrow \mathcal{A}^{\mathsf{KGen}(\mathsf{mpk},\mathsf{msk},\cdot)}(\mathsf{ct}) \quad : \quad \begin{array}{l}1^\kappa \leftarrow \mathcal{A}(1^\lambda), \\ (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\kappa), \\ S \leftarrow \mathcal{A}^{\mathsf{KGen}(\mathsf{mpk},\mathsf{msk},\cdot)}(\mathsf{mpk}), \\ b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, S, b)\end{array}\right]$$

A stateful PPT machine $\mathcal{A}$ is said to be admissible if it never queries $\mathsf{KGen}(\mathsf{mpk}, \mathsf{msk}, \cdot)$ with id^* for any $\mathsf{id}^* \in S \subseteq \{0,1\}^\kappa$.

7.2 Equivocal Identity-Based Broadcast Encryption

Syntax. An Equivocal Identity-Based Broadcast Encryption scheme (EqBE) for identity space $\mathcal{I} = \{\{0,1\}^\kappa\}_{\kappa \in \mathbb{N}}$ consists of the following polynomial time algorithms.

$\mathsf{Setup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}$**.** These are defined similarly to an IBBE scheme for $\mathcal{I}$ (Definition 7.1).

$\mathsf{FakeSetup}(1^\lambda, 1^\kappa, \mu) \to (\widetilde{\mathsf{mpk}}, \widetilde{\mathsf{ct}}, \mathsf{aux})$**.** The randomized fake setup algorithm takes as input security parameter λ, identity space index κ, message $\mu \in \{0,1\}$, outputs fake master public key, $\widetilde{\mathsf{mpk}}$, fake ciphertext $\widetilde{\mathsf{ct}}$, and auxiliary information aux.

$\mathsf{SimKGen}(\mathsf{aux}, \mathsf{st}, \mathsf{id}) \to (\mathsf{sk}_{\mathsf{id}}, \mathsf{st}')$ The randomized *stateful* key generation simulation algorithm takes as input auxiliary information aux, state st, identity $\mathsf{id} \in \{0,1\}^\kappa$, and outputs secret key for id, $\mathsf{sk}_{\mathsf{id}}$, and updated state st'.

$\mathsf{Equivocate}(\mathsf{aux}, \mathsf{st}, S) \to (\mathsf{ct}, \mathsf{st}')$**.** The possibly randomized equivocation algorithm takes as input auxiliary information aux, state st, set of identities $S \subseteq \{0,1\}^\kappa$, outputs ciphertext ct and updated state st'.

Definition 7.2. An EqBE scheme $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FakeSetup}, \mathsf{SimKGen}, \mathsf{Equivocate})$ for identity space $\mathcal{I} = \{\{0,1\}^\kappa\}_{\kappa \in \mathbb{N}}$ should satisfy the following properties.

Correctness and Compactness. These are defined for $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ exactly as correctness and compactness for an IBBE scheme (Definition 7.1).

Mode Indistinguishability. We define the following game between a stateful PPT adversary $\mathcal{A}$ and a challenger Chal.

1. $\mathcal{A}$ sends $(1^\kappa, \mu \in \{0,1\})$ to Chal. Chal flips a coin $b \leftarrow \{0,1\}$. Chal samples state st.
2. If $b = 0$, Chal samples $(\mathsf{mpk}^{(0)}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\kappa)$, and sends mpk to $\mathcal{A}$.
   Otherwise, if $b = 1$, Chal samples $(\mathsf{mpk}^{(1)}, \widetilde{\mathsf{ct}}, \mathsf{aux}) \leftarrow \mathsf{FakeSetup}(1^\lambda, 1^\kappa, \mu)$ and sends $\mathsf{mpk}^{(1)}$ to $\mathcal{A}$.
3. $\mathcal{A}$ (polynomially-many times) queries $\mathsf{id} \in \{0,1\}^\kappa$. If $b = 0$, Chal computes $\mathsf{sk}_{\mathsf{id}}^{(0)} \leftarrow \mathsf{KGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{id})$ and sends $\mathsf{sk}_{\mathsf{id}}^{(0)}$ to $\mathcal{A}$.
   Otherwise, if $b = 1$, Chal computes $(\mathsf{sk}_{\mathsf{id}}^{(1)}, \mathsf{st}') \leftarrow \mathsf{SimKGen}(\mathsf{aux}, \mathsf{st}, \mathsf{id})$ and sends $\mathsf{sk}_{\mathsf{id}}^{(1)}$ to $\mathcal{A}$. Chal also updates $\mathsf{st} := \mathsf{st}'$.
4. $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$. If any $\mathsf{id}^* \in S$ were queried for secret key before, abort and output $b' \leftarrow \{0,1\}$.
   Otherwise, if $b = 0$, Chal samples $\mathsf{ct}^{(0)} \leftarrow \mathsf{Enc}(\mathsf{mpk}, S, \mu)$ and sends $\mathsf{ct}^{(0)}$ to $\mathcal{A}$.
   Otherwise, if $b = 1$, Chal samples $(\mathsf{ct}^{(1)}, \mathsf{st}') \leftarrow \mathsf{Equivocate}(\mathsf{aux}, \mathsf{st}, S)$ and sends $\mathsf{ct}^{(1)}$ to $\mathcal{A}$. Chal also updates $\mathsf{st} := \mathsf{st}'$.
5. $\mathcal{A}$ can repeat previous secret key query phase and Chal responds similarly to previous phase.
6. $\mathcal{A}$ guess b as b'.

We say that EqBE satisfies mode indistinguishability property if for any stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$,

$$\left| \Pr[b' = b] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda)$$

Semantic Security. For any stateful admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$, the following probability is at most $1/2 + \mathsf{negl}(\lambda)$.

$$\Pr\left[\mu \leftarrow \mathcal{A}^{\mathsf{SimKGen}(\mathsf{aux},\mathsf{st},\cdot)}(\mathsf{ct}) \quad : \quad \begin{array}{l} 1^\kappa \leftarrow \mathcal{A}(1^\lambda), \mu \leftarrow \{0,1\}, \\ (\widetilde{\mathsf{mpk}}, \widetilde{\mathsf{ct}}, \mathsf{aux}) \leftarrow \mathsf{FakeSetup}(1^\lambda, 1^\kappa, \mu), \\ S \leftarrow \mathcal{A}^{\mathsf{SimKGen}(\mathsf{aux},\mathsf{st},\cdot)}(\widetilde{\mathsf{mpk}}), \\ (\mathsf{ct}, \mathsf{st}') \leftarrow \mathsf{Equivocate}(\mathsf{aux}, \mathsf{st}, S), \\ \mathsf{st} := \mathsf{st}' \end{array} \right]$$

A stateful PPT machine $\mathcal{A}$ is said to be admissible if $S \subseteq \{0,1\}^\kappa$ and $\mathcal{A}$ never queries the stateful $\mathsf{SimKGen}(\mathsf{aux}, \mathsf{st}, \cdot)$ oracle with id^* for any $\mathsf{id}^* \in S$.

7.3 IBBE from EqBE

Construction 7.3. We construct an IBBE scheme (Definition 7.1) for identity space $\mathcal{I} = \{\{0,1\}^\kappa\}_{\kappa \in \mathbb{N}}$ using an EqBE scheme (Definition 7.2) for identity space $\mathcal{I} = \{\{0,1\}^\kappa\}_{\kappa \in \mathbb{N}}$ as follows.

$\mathsf{Setup}(1^\lambda, 1^\kappa)$. Sample $(\mathsf{be.mpk}, \mathsf{be.msk}) \leftarrow \mathsf{EqBE.Setup}(1^\lambda, 1^\kappa)$. Output $\mathsf{mpk} := \mathsf{be.mpk}$ and $\mathsf{msk} := \mathsf{be.msk}$.

$\mathsf{KGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{id})$. Parse msk as $\mathsf{be.msk}$ and sample $\mathsf{be.sk_{id}} \leftarrow \mathsf{EqBE.KGen}(\mathsf{be.mpk}, \mathsf{be.msk}, \mathsf{id})$.

Output $\mathsf{sk_{id}} := \mathsf{be.sk_{id}}$.

$\mathsf{Enc}(\mathsf{mpk}, S, \mu)$. Sample $\mathsf{be.ct} \leftarrow \mathsf{EqBE.Enc}(\mathsf{be.mpk}, S, \mu)$. Output $\mathsf{ct} := \mathsf{be.ct}$.

$\mathsf{Dec}(\mathsf{mpk}, S, \mathsf{sk_{id}}, \mathsf{ct})$. Parse $\mathsf{sk_{id}}$ as $\mathsf{be.sk_{id}}$ and ct as $\mathsf{be.ct}$.

Output $\mu' := \mathsf{EqBE.Dec}(\mathsf{be.mpk}, S, \mathsf{be.sk_{id}}, \mathsf{be.ct})$.

Correctness and Compactness. This follows from correctness and compactness of EqBE.

Theorem 7.4. If EqBE is an EqBE scheme for $\mathcal{I} = \{\{0,1\}^\kappa\}_{\kappa\in\mathbb{N}}$ (Definition 7.2), then Construction 7.3 is an IBBE scheme for $\mathcal{I} = \{\{0,1\}^\kappa\}_{\kappa\in\mathbb{N}}$ (Definition 7.1).

Proof Sketch. We prove this theorem using the following hybrids for an admissible adversary.

Hybrid 0. This is the adaptive security experiment from Definition 7.1 with $b = 0$.

Hybrid 1. In this hybrid, we will use $\mathsf{FakeSetup}(1^\lambda, 1^\kappa, 0)$ to generate $(\widetilde{\mathsf{mpk}}, \widetilde{\mathsf{ct}}, \mathsf{aux})$. We will also use aux and $\mathsf{SimKGen}$ to generate $\mathsf{sk_{id}}$ for any id that $\mathcal{A}$ queries. Also, when $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$, we will use $\mathsf{Equivocate}(\mathsf{aux}, \mathsf{st}, S)$ to generate the ciphertext ct. Hybrids 0 and 1 are indistinguishable because of the mode indistinguishability property of EqBE.

Hybrid 2. In this hybrid, we will encrypt message 1 using $\mathsf{EqBE.FakeSetup}$ algorithm. The hybrids remain indistinguishable due to semantic security property of EqBE.

Hybrid 3. This is the adaptive security experiment from Definition 7.1 with $b = 1$. Indistinguishability follows by mode indistinguishability property of EqBE.

We provide full proof in Appendix E. □

8 Equivocal Identity-Based Broadcast Encryption from Decomposed LWE

In this section, we construct an EqBE scheme from $2m^2$-decomposed LWE assumption.

8.1 Tools Required: Trapdoored Equivocal Matrix Commitment Schemes and Back-Samplable EqFBE

Trapdoored EqCOM. For our construction of EqBE, we require a strengthening of EqMCOM from Section 4 where the Setup algorithm should output trapdoor $\mathbf{T_B}$ for $\mathbf{B}$ such that $\mathbf{BT_B} = \mathbf{G}$.

Syntax. A Trapdoored Equivocal Matrix Commitment scheme (TEqMCOM) consists of the following polynomial-time algorithms.

$\mathsf{Setup}, \mathsf{Com}, \mathsf{Verify}, \mathsf{Open}, \mathsf{EqCom}$. These are defined similarly to EqMCOM (Definition 4.1).

$\widetilde{\mathsf{Setup}}(1^n, 1^m, q, \sigma, B) \to (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{T_B})$. The randomized alternative setup algorithm takes as input LWE parameters (n, m, q, σ) where $\sigma = \mathsf{poly}(\lambda, m)$, bound $B \in \mathbb{N}$, outputs public parameters pp, $\mathbf{W} \in \mathbb{Z}_q^{n \times m_1}$, $\mathbf{D} \in \mathcal{D}_{\mathbb{Z},\sigma}^{m \times m_1}$, $\mathbf{B}, \mathbf{T_B}$ such that $\mathbf{B} = \mathbf{W}(\mathbf{I}_{m_2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{m_2})^\top \otimes \mathbf{G}$ and $\mathbf{B}\mathbf{T_B} = \mathbf{G}$.

Definition 8.1 (TEqMCOM)**.** A TEqMCOM scheme $(\mathsf{Setup}, \mathsf{Com}, \mathsf{Verify}, \mathsf{Open}, \mathsf{EqCom}, \widetilde{\mathsf{Setup}})$ should satisfy the following properties.

Correctness, Compactness, Commitment Indistinguishability. These are defined for (Setup, $\mathsf{Com}, \mathsf{Verify}, \mathsf{Open}, \mathsf{EqCom}$) exactly same as Definition 4.1.

Setup Indistinguishability. For any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$, such that for any $\lambda \in \mathbb{N}$, the following probability is at most $1/2 + \mathsf{negl}(\lambda)$.

$$\Pr\left[\begin{array}{l} b \leftarrow \mathcal{A}(\mathsf{pp}^{(b)}, \mathbf{W}^{(b)}, \mathbf{D}^{(b)}, \mathbf{B}^{(b)}) \end{array} : \begin{array}{l} (1^n, 1^m, q, \sigma, B) \leftarrow \mathcal{A}(1^\lambda), \\ b \leftarrow \{0,1\}, \\ (\mathsf{pp}^{(0)}, \mathbf{W}^{(0)}, \mathbf{D}^{(0)}, \mathbf{B}^{(0)}) \leftarrow \\ \mathsf{Setup}(1^n, 1^m, q, \sigma, B), \\ (\mathsf{pp}^{(1)}, \mathbf{W}^{(1)}, \mathbf{D}^{(1)}, \mathbf{B}^{(1)}, \mathbf{T_B}) \\ \leftarrow \widetilde{\mathsf{Setup}}(1^n, 1^m, q, \sigma, B) \end{array}\right]$$

Remark 8.2. Note that this can be constructed from EqMCOM using the alternative way of sampling $\mathbf{B}$ with a trapdoor as explained in Section 3. This in fact gives stronger statistical closeness of Setup and $\widetilde{\mathsf{Setup}}$. We omit full construction for brevity.

Back-Sampleable EqFBE. We require an EqFBE scheme where key generation can be performed in two indistinguishable modes:

Forward Mode: Sample $\mathsf{sk} = \mathbf{r} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$. Set $\mathsf{pk} = \mathbf{t} := \mathbf{B}\mathbf{r} + \mathbf{p}$. Output $(\mathsf{pk}, \mathsf{sk})$.

Backward Mode: Sample $\mathsf{pk} = \mathbf{t} \leftarrow \mathbb{Z}_q^n$ and $\mathsf{sk} = \mathbf{r} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, \mathbf{t} - \mathbf{p}, \widetilde{\sigma})$. Output $(\mathsf{pk}, \mathsf{sk})$.

Although this increases the secret key size in EqFBE of Section 5.2, it doesn't adversely affect correctness or parameters too much. It is easy to check that EqFBE of Section 5.2 satisfies this property readily.

8.2 Construction

Parameters. We provide the parameters used by our construction here.

- $n = \mathsf{poly}(\lambda)$ (Decomposed LWE security)
- $m = \Omega(n \log q)$ (Leftover hash lemma)
- $\chi = \mathsf{poly}(\lambda)$ (Decomposed LWE security)
- $\widetilde{\chi} > \lambda^{\omega(1)} m^{11/2} \chi$ (Smudging)
- $\sigma = \omega(\log m)$ (Decomposed LWE security)
- $q > O(\lambda \cdot \chi \cdot \widetilde{\chi} \cdot \sigma \cdot \widetilde{\sigma} \cdot m^{10} \cdot \log q)$ (Correctness)

One way to instantiate these parameters is for constant $0 < \epsilon < 1/2$, $n = (\lambda\kappa)^{1/\epsilon} \cdot \mathsf{poly}(\log \lambda, \kappa)$, $m = 4n \log q$, $\chi = \mathsf{poly}(n)$, $\sigma = \log^2 m$, and $q = 2^{\lambda} \cdot \mathsf{poly}(n, 2^{\kappa})$. This would require sub-exponential modulus-to-noise ratio.

Construction 8.3. We construct an EqBE scheme for identity space $\mathcal{I} = \{\{0,1\}^{\kappa}\}_{\kappa \in \mathbb{N}}$ assuming the hardness of $2m^2$-decomposed LWE in random oracle model using

- Trapdoored Equivocal Matrix commitment scheme EqMCOM (Definition 8.1)
- Hash function $\mathcal{H}_{\mathsf{pk}} : \{0,1\}^{\kappa} \to \mathbb{Z}_q^n$ which will be modeled as random oracle in security proof.

$\mathsf{Setup}(1^{\lambda}, 1^{\kappa})$**.** Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{T_B}) \leftarrow \mathsf{TEqMCOM}.\widetilde{\mathsf{Setup}}(1^n, 1^m, q, \sigma, 1)$.

Sample $\mathbf{p} \leftarrow \mathbb{Z}_q^n$ and hash function $\mathcal{H}_{\mathsf{pk}} : \{0,1\}^{\kappa} \to \mathbb{Z}_q^n$.

Output $\mathsf{mpk} := (\kappa, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}})$ and $\mathsf{msk} := \mathbf{T_B}$.

$\mathsf{KGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{id})$**.** Given $\mathsf{id} \in \{0,1\}^{\kappa}$, compute $\mathbf{t}_{\mathsf{id}} = \mathcal{H}_{\mathsf{pk}}(\mathsf{id})$. Now perform the backward sampling of EqFBE. That is,

Sample $\mathbf{y}_{\mathsf{id}} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, \mathbf{t}_{\mathsf{id}} - \mathbf{p}, \widetilde{\sigma})$[2]. Output $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}_{\mathsf{id}})$.

$\mathsf{Enc}(\mathsf{mpk}, S, \mu)$**.** Let $L = |S|$ and $(\mathsf{id}_1, \ldots, \mathsf{id}_L)$ be the lexicographical ordering of S. Compute $\mathbf{t}_{\mathsf{id}_i} = \mathcal{H}_{\mathsf{pk}}(\mathsf{id}_i)$ for each $i \in [L]$ and run the Enc algorithm from Section 6 with $(\mathbf{t}_{\mathsf{id}_1}, \ldots, \mathbf{t}_{\mathsf{id}_L}, \mu)$ as input. Let $(\mathbf{c}_1^{\top}, \mathbf{c}_2^{\top}, c_3, \mathsf{aux}) \in \mathbb{Z}_q^{1 \times (4m^5 + m + 1)} \times \{0,1\}^{\mathsf{poly}(\lambda, \log L)}$ be the resulting ciphertext. Output $\mathsf{ct} := (\mathbf{c}_1^{\top}, \mathbf{c}_2^{\top}, c_3, \mathsf{aux})$.

$\mathsf{Dec}(\mathsf{mpk}, S, \mathsf{sk}_{\mathsf{id}}, \mathsf{ct})$**.** Parse $\mathsf{sk}_{\mathsf{id}}$ as $(\mathsf{id}, \mathbf{y}_{\mathsf{id}})$. If $\mathsf{id} \notin S$, abort and output $\perp$. Otherwise, parse ct as $(\mathbf{c}_1^{\top}, \mathbf{c}_2^{\top}, c_3, \mathsf{aux})$. Let $|S| = L$ and $\mathsf{id}_1, \ldots, \mathsf{id}_L$ be the lexicographical ordering of S. Find $i^* \in [L]$ such that $\mathsf{id}_{i^*} = \mathsf{id}$. Compute $\mathbf{t}_{\mathsf{id}_i} = \mathcal{H}_{\mathsf{pk}}(\mathsf{id}_i)$ for each $i \in [L]$ and run the Dec algorithm from Section 6 with $(\mathbf{t}_{\mathsf{id}_1}, \ldots, \mathbf{t}_{\mathsf{id}_L}, (i^*, \mathbf{y}_{\mathsf{id}}), \mathsf{ct})$ as input and let μ' be the output. Output μ'.

Correctness and Compactness. This is similar to correctness analysis of Section 6. As $\log L \le \lambda$ (very loosely), as long as $q > O(\lambda \chi \widetilde{\chi} \sigma \widetilde{\sigma} m^{10} \log q)$, correctness holds. Similar to compactness analysis of Section 6, we have that $|\mathsf{mpk}|, |\mathsf{sk}_{\mathsf{id}}|$, and $|\mathsf{ct}|$ are polynomially bounded in λ, κ.

Remark 8.4 (Removing $\mathcal{H}_{\mathsf{pk}}$ in secret-key IBBE.)**.** Observe that the random oracle $\mathcal{H}_{\mathsf{pk}} : \{0,1\}^{\kappa} \to \mathbb{Z}_q^n$ was only used to generate a random vector $\mathbf{t}_{\mathsf{id}}$ consistently between Enc and KGen. In the secret-key IBBE setting where both Enc and KGen use msk, we can simply use a PRF key K for $\mathsf{PRF} : \{0,1\}^{\lambda} \times \{0,1\}^{\kappa} \to \mathbb{Z}_q^n$ as part of msk to sample the same.

Remark 8.5 (Using Succinct LWE for even shorter ciphertext)**.** Note that as we are in a centralized encryption setting, we can use $2m^2$-succinct LWE [Wee24, Wee25] which is a stronger assumption than $2m^2$-decomposed LWE [AMR25, Wee25] to obtain a ciphertext of size $(2m+1)\lceil \log q \rceil$. Analysis would be similar to [WW25, Wee25].

[2]Recall that $\widetilde{\sigma} \in \mathbb{N}$ is implicitly defined in pp.

8.3 Fake Setup, Key Generation Simulation, and Equivocation

We now describe our $\mathsf{FakeSetup}, \mathsf{SimKGen}$, and $\mathsf{Equivocate}$ algorithms. These are defined with respect to a programmable random oracle which is treated as a dictionary H_{pk}.

$\mathsf{FakeSetup}(1^\lambda, 1^\kappa, \mu)$**.** Given $\mu \in \{0,1\}$, do the following.

1. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{TEqMCOM.Setup}(1^n, 1^m, q, \sigma, 1)$.
2. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}, \mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$. Compute $\mathbf{p} := \mathbf{B}\mathbf{r_p}$.
3. Initiate dictionaries $\mathsf{H}_{\mathsf{pk}} := \emptyset$.
4. Set $\widetilde{\mathsf{ct}} := \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$, $\widetilde{\mathsf{mpk}} := (\kappa, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}})$, $\mathsf{aux} := (\mu, \widetilde{\mathsf{ct}}, \widetilde{\mathsf{mpk}}, \mathbf{r_p}, \mathbf{R_C})$.

Here, $\mathcal{H}_{\mathsf{pk}}$ is a handle for random oracle queries. Output $(\widetilde{\mathsf{mpk}}, \widetilde{\mathsf{ct}}, \mathsf{aux})$.

$\mathsf{SimKGen}^{\mathsf{H}_{\mathsf{pk}}(\cdot)}(\mathsf{aux}, \mathsf{st}, \mathsf{id})$**.** Parse st as $\mathcal{K}$. Do the following.

1. If $\mathsf{id} \in \mathsf{H}_{\mathsf{pk}}$, retrieve $\mathbf{y}'_{\mathsf{id}}$ from $\mathcal{K}[\mathsf{id}]$. Set $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$ and $\mathsf{st}' := \mathsf{st}$.
2. Otherwise, sample $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$, compute $\mathbf{t}' = \mathbf{B}\mathbf{y}'_{\mathsf{id}}$ and $\mathbf{t}_{\mathsf{id}} = \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$. Set $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$, set $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}] = \mathbf{t}_{\mathsf{id}}$ and $\mathcal{K}[\mathsf{id}] = \mathbf{y}'_{\mathsf{id}}$. Set $\mathsf{st}' := \mathcal{K}$.

Output $(\mathsf{sk}_{\mathsf{id}}, \mathsf{st}')$.

$\mathsf{Equivocate}^{\mathsf{H}_{\mathsf{pk}}(\cdot)}(\mathsf{aux}, \mathsf{st}, S)$**.** Let $L = |S|$ and $\mathsf{id}_1, \ldots \mathsf{id}_L$ be the lexicographical ordering of S. Parse st as $\mathcal{K}$, aux as $(\mu, \widetilde{\mathsf{ct}}, \kappa, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}}, \mathbf{r_p}, \mathbf{R_C})$, and $\widetilde{\mathsf{ct}}$ as $\mathbf{c}_1^\top$. Do the following.

1. For each $i \in [L]$, if $\mathsf{id}_i \in \mathsf{H}_{\mathsf{pk}}$, get $\mathbf{t}_{\mathsf{id}} = \mathsf{H}_{\mathsf{pk}}[\mathsf{id}_i]$, $\mathbf{t}'_{\mathsf{id}} = \mathbf{t}_{\mathsf{id}} - \mathbf{p}$, and $\mathbf{y}'_{\mathsf{id}_i} = \mathcal{K}[\mathsf{id}_i]$. Otherwise, sample $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$, compute $\mathbf{t}' = \mathbf{B}\mathbf{y}'_{\mathsf{id}}$ and $\mathbf{t}_{\mathsf{id}} = \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$. Set $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$, set $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}] = \mathbf{t}_{\mathsf{id}}$ and $\mathcal{K}[\mathsf{id}] = \mathbf{y}'_{\mathsf{id}}$.
2. Compute matrix $\mathbf{Y} = [\mathbf{y}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{y}'_{\mathsf{id}_L}]$. Compute $\mathbf{C} = \mathbf{B}\mathbf{R_C}$.
3. Sample $\mathsf{aux}' \leftarrow \mathsf{TEqMCOM.EqCom}(\mathbf{Y} + \mathbf{1}_L^\top \otimes \mathbf{r_p}, \mathbf{R_C})$.
4. Sample $\widetilde{\mathbf{e}}^\top \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m$.
5. Set $\mathsf{st}' := \mathcal{K}$.

Output $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_1^\top \mathbf{R_C} + \widetilde{\mathbf{e}}^\top, \mathbf{c}_1^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu, \mathsf{aux}')$ and st'.

8.4 Mode Indistinguishability Analysis

Theorem 8.6 (Mode Indistinguishability.)**.** If $\mathsf{TEqMCOM}$ satisfies setup indistinguishability and commitment indistinguishability properties (Definition 8.1), then Construction 8.3 satisfies mode indistinguishability property (Definition 7.2).

Proof Sketch. We define the following hybrids.

Hybrid 0. This is the mode indistinguishability game from Definition 7.2 with $b = 0$.

Hybrid 1. In this hybrid, we answer a random oracle query for id by sampling $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}^{4m^5}_{\mathbb{Z},\widetilde{\sigma}}$ and setting $\mathcal{H}_{\mathsf{pk}}(\mathsf{id}) := \mathbf{B}\mathbf{y}'_{\mathsf{id}} + \mathbf{p}$, and we answer a secret key query for id by handing out the same $\mathbf{y}'_{\mathsf{id}}$. Writing $\mathbf{t}_{\mathsf{id}}$ as $\mathbf{t}'_{\mathsf{id}} + \mathbf{p}$ for $\mathbf{t}'_{\mathsf{id}} \leftarrow \mathbb{Z}_q^n$ is only a semantic change, and replacing a uniform $\mathbf{t}'_{\mathsf{id}}$ together with its pre-image by $\mathbf{B}\mathbf{y}'_{\mathsf{id}}$ is statistically close by the gaussian pre-image distribution (Lemma A.6 and Lemma A.1).

Hybrid 2. In this hybrid, we sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{TEqMCOM.Setup}(1^n, 1^m, q, \sigma, 1)$ during setup, that is, without a trapdoor. Hybrids 1 and 2 are statistically close by the setup indistinguishability property of $\mathsf{TEqMCOM}$ (Definition 8.1).

Hybrid 3. In this hybrid, we set $\mathbf{p} := \mathbf{B}\mathbf{r}_{\mathbf{p}}$ for $\mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}$ during setup. Hybrids 2 and 3 are statistically close by leftover hash lemma (Lemma A.3).

Hybrid 4. In this hybrid, during the challenge encryption phase, we sample $\mathbf{R}_{\mathbf{C}} \leftarrow \{\pm 1\}^{4m^5 \times m}$, set $\mathbf{C} = \mathbf{B}\mathbf{R}_{\mathbf{C}}$, and sample $\mathsf{aux}' \leftarrow \mathsf{TEqMCOM.EqCom}([\mathbf{y}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{y}'_{\mathsf{id}_L}] + \mathbf{1}_L^\top \otimes \mathbf{r}_{\mathbf{p}}, \mathbf{R}_{\mathbf{C}})$. Hybrids 3 and 4 are computationally indistinguishable by the commitment indistinguishability property of $\mathsf{TEqMCOM}$, with $\mathbf{R} = \mathbf{1}_L^\top \otimes \mathbf{r}_{\mathbf{p}}$, for which $\|\mathbf{R}\| = 1$.

Hybrid 5. In this hybrid, we set $\mathbf{c}_2^\top := \mathbf{c}_1^\top \mathbf{R}_{\mathbf{C}} + \widetilde{\mathbf{e}}^\top$ during the challenge encryption phase. Hybrids 4 and 5 are statistically close by the gaussian smudging lemma (Lemma 3.5). This hybrid is the mode indistinguishability game from Definition 7.2 with $b = 1$, up to semantic changes.

We provide full proof in Appendix F.1. □

8.5 Semantic Security Analysis

Theorem 8.7 (Semantic Security.)**.** If $2m^2$-decomposed LWE assumption (Assumption 3.7) holds and $\mathsf{TEqMCOM}$ satisfies setup indistinguishability property (Definition 8.1), then Construction 8.3 satisfies semantic security property (Definition 7.2).

Proof Sketch. We define the following hybrids. Throughout, $\mu \leftarrow \{0,1\}$ denotes the message sampled by the semantic security game (Definition 7.2).

Hybrid 0. This is the semantic security experiment from Definition 7.2.

Hybrid 1. In this hybrid, we sample $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1 \times 4m^5}$ during the fake setup phase, rather than computing it as $\mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$. Hybrids 0 and 1 are computationally indistinguishable by the $2m^2$-decomposed LWE assumption (Assumption 3.7).

Hybrid 2. In this hybrid, we sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{T}_{\mathbf{B}}) \leftarrow \mathsf{TEqMCOM.}\widetilde{\mathsf{Setup}}(1^n, 1^m, q, \sigma, 1)$ during the fake setup phase, that is, together with a trapdoor. Hybrids 1 and 2 are statistically close by the setup indistinguishability property of $\mathsf{TEqMCOM}$ (Definition 8.1).

Hybrid 3. In this hybrid, we sample $\mathbf{t}'_{\mathsf{id}} \leftarrow \mathbb{Z}_q^n$ and set $\mathbf{t}_{\mathsf{id}} := \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$, answering a key generation simulation query for id with $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, \mathbf{t}'_{\mathsf{id}}, \widetilde{\sigma})$ and sampling $\mathbf{Y} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, [\mathbf{t}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{t}'_{\mathsf{id}_L}], \widetilde{\sigma})$ during equivocation. Hybrids 2 and 3 are statistically close by the pre-image distribution property of gadget trapdoors (Lemma A.6 and Lemma A.1).

Hybrid 4. In this hybrid, during the equivocation phase we sample $\mathbf{Y}' \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, [\mathbf{t}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{t}'_{\mathsf{id}_L}] + \mathbf{1}_L^\top \otimes \mathbf{p}, \widetilde{\sigma})$ and set $\mathsf{aux}' \leftarrow \mathsf{TEqMCOM.EqCom}(\mathbf{Y}', \mathbf{R}_{\mathbf{C}})$. Hybrids 3 and 4 are statistically close by the gaussian pre-image smudging lemma (Lemma A.7 and Lemma A.2). Note that $\mathbf{r}_{\mathbf{p}}$ is no longer used in the equivocation phase.

Hybrid 5. In this hybrid, we sample $\mathbf{p} \leftarrow \mathbb{Z}_q^n$ and $c_3 \leftarrow \mathbb{Z}_q$. Hybrids 4 and 5 are statistically close by leftover hash lemma (Lemma A.5), which holds for $\mathbf{B}$ sampled with a trapdoor. This hybrid is independent of μ, and so any $\mathcal{A}$ has 0 advantage.

We provide full proof in Appendix F.2. □

Corollary 8.8. Assuming the hardness of $2m^2$-decomposed LWE assumption (Assumption 3.7), there exists an adaptively secure identity-based broadcast encryption scheme for $\mathcal{I} = \{\{0,1\}^\kappa\}_{\kappa \in \mathbb{N}}$ in the random oracle model.

Proof. By Theorem 8.6 and Theorem 8.7 we have an EqBE scheme in random oracle model. Then, Theorem 7.4 gives us an adaptively secure IBBE scheme in random oracle model. □

References

[ABB10] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 553–572, French Riviera, May 30 – June 3, 2010. Springer Berlin Heidelberg, Germany.

[AGY26] Abtin Afshar, Rishab Goyal, and Saikumar Yadugiri. Distributed monotone-policy encryption with silent setup from lattices. Cryptology ePrint Archive, Paper 2026/372, 2026.

[AJL+12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 483–501, Cambridge, UK, April 15–19, 2012. Springer Berlin Heidelberg, Germany.

[AMR25] Damiano Abram, Giulio Malavolta, and Lawrence Roy. Key-homomorphic computations for RAM: Fully succinct randomised encodings and more. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology – CRYPTO 2025, Part III*, volume 16002 of *Lecture Notes in Computer Science*, pages 236–268, Santa Barbara, CA, USA, August 17–21, 2025. Springer, Cham, Switzerland.

[AMR26a] Damiano Abram, Giulio Malavolta, and Lawrence Roy. How to authenticate a non-deterministic computation: Shift-hiding functions, compressed LWE sampling, broadcast encryption, and obfuscation. Cryptology ePrint Archive, Paper 2026/741, 2026. Version dated June 11, 2026, https://eprint.iacr.org/archive/2026/741/20260611:204614.

[AMR26b] Damiano Abram, Giulio Malavolta, and Lawrence Roy. Tree encodings i: How to authenticate a truth table. Cryptology ePrint Archive, Paper 2026/741, 2026. Version dated September 18, 2026.

[AY20] Shweta Agrawal and Shota Yamada. Optimal broadcast encryption from pairings and LWE. In *Advances in Cryptology – EUROCRYPT 2020, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 13–43. Springer, Cham, Switzerland, 2020.

[BBG05] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer Berlin Heidelberg, Germany, 2005.

[BDJ+25] Pedro Branco, Nico Döttling, Abhishek Jain, Giulio Malavolta, Surya Mathialagan, Spencer Peters, and Vinod Vaikuntanathan. Pseudorandom obfuscation and applications. In *Advances in Cryptology – CRYPTO 2025, Part V*, volume 16004 of *Lecture Notes in Computer Science*, pages 663–698. Springer, Cham, Switzerland, 2025.

[BGW05] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275, Santa Barbara, CA, USA, August 14–18, 2005. Springer Berlin Heidelberg, Germany.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[BW06] Dan Boneh and Brent Waters. A fully collusion resistant broadcast, trace, and revoke system. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006: 13th Conference on Computer and Communications Security*, pages 211–220, Alexandria, Virginia, USA, October 30 – November 3, 2006. ACM Press.

[BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In *34rd Annual International Cryptology Conference, CRYPTO 2014*, pages 480–499. Springer Verlag, 2014.

[CHW25] Jeffrey Champion, Yao-Ching Hsieh, and David J. Wu. Registered ABE and adaptively-secure broadcast encryption from succinct LWE. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology – CRYPTO 2025, Part III*, volume 16002 of *Lecture Notes in Computer Science*, pages 3–34, Santa Barbara, CA, USA, August 17–21, 2025. Springer, Cham, Switzerland.

[Clo20] Cloudflare. Geo Key Manager: How it works. Cloudflare Blog, 2020. Available at https://blog.cloudflare.com/geo-key-manager-how-it-works/.

[CW24] Jeffrey Champion and David J. Wu. Distributed broadcast encryption from lattices. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024: 22nd Theory of Cryptography Conference, Part III*, volume 15366 of *Lecture Notes in Computer Science*, pages 156–189, Milan, Italy, December 2–6, 2024. Springer, Cham, Switzerland.

[Del07] Cécile Delerablée. Identity-based broadcast encryption with constant size ciphertexts and private keys. In Kaoru Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 200–215, Kuching, Malaysia, December 2–6, 2007. Springer Berlin Heidelberg, Germany.

[DGI+19] Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. Cryptology ePrint Archive, Paper 2019/639, 2019.

[DKW21] Pratish Datta, Ilan Komargodski, and Brent Waters. Decentralized multi-authority abe for dnf s from lwe. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 177–209. Springer, 2021.

[DRS04] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 523–540, Interlaken, Switzerland, May 2–6, 2004. Springer Berlin Heidelberg, Germany.

[FN94] Amos Fiat and Moni Naor. Broadcast encryption. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO'93*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491, Santa Barbara, CA, USA, August 22–26, 1994. Springer Berlin Heidelberg, Germany.

[FWW23] Cody Freitag, Brent Waters, and David J. Wu. How to use (plain) witness encryption: Registered ABE, flexible broadcast, and more. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part IV*, volume 14084 of *Lecture Notes in Computer Science*, pages 498–531, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 467–476, Palo Alto, CA, USA, June 1–4, 2013. ACM Press.

[GKPW24] Sanjam Garg, Dimitris Kolonelos, Guru-Vamsi Policharla, and Mingyuan Wang. Threshold encryption with silent setup. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part VII*, volume 14926 of *Lecture Notes in Computer Science*, pages 352–386, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland.

[GLWW23] Rachit Garg, George Lu, Brent Waters, and David J. Wu. Realizing flexible broadcast encryption: How to broadcast to a public-key directory. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *ACM CCS 2023:*

30th Conference on Computer and Communications Security, pages 1093–1107, Copenhagen, Denmark, November 26–30, 2023. ACM Press.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 197–206, Victoria, BC, Canada, May 17–20, 2008. ACM Press.

[GV20] Rishab Goyal and Satyanarayana Vusirikala. Verifiable registration-based encryption. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part I*, volume 12170 of *Lecture Notes in Computer Science*, pages 621–651, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Cham, Switzerland.

[GVW19] Rishab Goyal, Satyanarayana Vusirikala, and Brent Waters. Collusion resistant broadcast and trace from positional witness encryption. In Dongdai Lin and Kazue Sako, editors, *PKC 2019: 22nd International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 11443 of *Lecture Notes in Computer Science*, pages 3–33, Beijing, China, April 14–17, 2019. Springer, Cham, Switzerland.

[GW09] Craig Gentry and Brent Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 171–188, Cologne, Germany, April 26–30, 2009. Springer Berlin Heidelberg, Germany.

[GY26a] Rishab Goyal and Saikumar Yadugiri. Distributed monotone policy encryption with stronger security for DNFs and threshold policies from lattices. Cryptology ePrint Archive, Paper 2026/1609, 2026.

[GY26b] Rishab Goyal and Saikumar Yadugiri. Equivocal broadcast encryption: Adaptively-secure optimal distributed broadcast encryption from lattices. Cryptology ePrint Archive, Paper 2026/792, 2026.

[HLWW23] Susan Hohenberger, George Lu, Brent Waters, and David J. Wu. Registered attribute-based encryption. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part III*, volume 14006 of *Lecture Notes in Computer Science*, pages 511–542, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

[HWW25] Yao-Ching Hsieh, Brent Waters, and David J. Wu. A generic approach to adaptively-secure broadcast encryption in the plain model. In Serge Fehr and Pierre-Alain Fouque, editors, *Advances in Cryptology – EUROCRYPT 2025, Part III*, volume 15603 of *Lecture Notes in Computer Science*, pages 336–365, Madrid, Spain, May 4–8, 2025. Springer, Cham, Switzerland.

[ILL89] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions (extended abstracts). In *21st Annual ACM Symposium on Theory of Computing*, pages 12–24, Seattle, WA, USA, May 15–17, 1989. ACM Press.

[KMW23] Dimitris Kolonelos, Giulio Malavolta, and Hoeteck Wee. Distributed broadcast encryption from bilinear groups. In Jian Guo and Ron Steinfeld, editors, *Advances in*

Cryptology – ASIACRYPT 2023, Part V, volume 14442 of *Lecture Notes in Computer Science*, pages 407–441, Guangzhou, China, December 4–8, 2023. Springer, Singapore, Singapore.

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 738–755, Cambridge, UK, April 15–19, 2012. Springer Berlin Heidelberg, Germany.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 700–718, Cambridge, UK, April 15–19, 2012. Springer Berlin Heidelberg, Germany.

[PS18] Chris Peikert and Sina Shiehian. Privately constraining and programming PRFs, the LWE way. In *PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 10770 of *Lecture Notes in Computer Science*, pages 675–701. Springer, Cham, Switzerland, 2018.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 84–93, Baltimore, MA, USA, May 22–24, 2005. ACM Press.

[SF07] Ryuichi Sakai and Jun Furukawa. Identity-based broadcast encryption. Cryptology ePrint Archive, Report 2007/217, 2007.

[Wat11] Brent Waters. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011: 14th International Conference on Theory and Practice of Public Key Cryptography*, volume 6571 of *Lecture Notes in Computer Science*, pages 53–70, Taormina, Italy, March 6–9, 2011. Springer Berlin Heidelberg, Germany.

[Wee24] Hoeteck Wee. Circuit ABE with $\mathsf{poly}(\mathsf{depth}, \lambda)$-sized ciphertexts and keys from lattices. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part III*, volume 14922 of *Lecture Notes in Computer Science*, pages 178–209, Santa Barbara, CA, USA, August 18–22, 2024. Springer, Cham, Switzerland.

[Wee25] Hoeteck Wee. Almost optimal KP and CP-ABE for circuits from succinct LWE. In Serge Fehr and Pierre-Alain Fouque, editors, *Advances in Cryptology – EUROCRYPT 2025, Part III*, volume 15603 of *Lecture Notes in Computer Science*, pages 34–62, Madrid, Spain, May 4–8, 2025. Springer, Cham, Switzerland.

[WQZD10] Qianhong Wu, Bo Qin, Lei Zhang, and Josep Domingo-Ferrer. Ad hoc broadcast encryption (poster presentation). In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 2010: 17th Conference on Computer and Communications Security*, pages 741–743, Chicago, Illinois, USA, October 4–8, 2010. ACM Press.

[WW25] Hoeteck Wee and David J. Wu. Unbounded distributed broadcast encryption and registered ABE from succinct LWE. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology – CRYPTO 2025, Part III*, volume 16002 of *Lecture Notes in Computer Science*, pages 204–235, Santa Barbara, CA, USA, August 17–21, 2025. Springer, Cham, Switzerland.

[WW26] Brent Waters and David J. Wu. Silent threshold cryptography from pairings: Expressive policies in the plain model. To appear at Eurocrypt, 2026.

A Additional Proofs

In this section, we provide proofs of lattice-based lemmas useful in proving our main theorems in the context of decomposed LWE assumption. We begin with the useful lemma statements.

Lemma A.1 ([GPV08, MP12])**.** Let n, m, q be lattice parameters such that $m = \Omega(n \log q)$, q is an odd prime. Then, for $\sigma > \log m$, all but negligible fraction of matrices $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$,

$$\{(\mathbf{B}, \mathbf{B}_\sigma^{-1}(\mathbf{y}), \mathbf{y}) : \mathbf{y} \leftarrow \mathbb{Z}_q^n\} \approx_s \{(\mathbf{B}, \mathbf{z}, \mathbf{Bz}) : \mathbf{z} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^m\}$$

This can also be extended to matrices naturally by sampling column-wise.

Lemma A.2 ([CHW25, WW25])**.** Let n, m, q be lattice parameters such that $m = \Omega(n \log q)$, q is an odd prime. Then, for any $\mathbf{y} \in \mathbb{Z}_q^n, \mathbf{z} \in \mathbb{Z}_q^m$, all but negligible fraction of matrices $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$,

$$\{(\mathbf{B}, \mathbf{B}_\sigma^{-1}(\mathbf{y} + \mathbf{Bz}))\} \approx_s \{(\mathbf{B}, \mathbf{B}_\sigma^{-1}(\mathbf{y}) + \mathbf{z})\}$$

where $\sigma > \lambda^{\omega(1)} \sqrt{m} \, \|\mathbf{z}\|$. This can also be extended to matrices naturally by sampling column-wise.

Essentially, in this section, we prove that these lemmas hold with respect to the structured matrix $\mathbf{B}$ from decomposed LWE assumption.

Lemma A.3. For positive integers n, m, and odd prime q, such that $m = \Omega(n \log q)$, for any $\ell > 2, \ell = \mathsf{poly}(\lambda), L = \mathsf{poly}(\lambda)$, $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times \ell m}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times \ell m}, \mathbf{B} = \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + (\mathsf{vec}(\mathbf{I}_\ell))^\top \otimes \mathbf{G})$,

$$\{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{BR}, \mathbf{e}^\top \mathbf{R})\} \approx_s \{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{C}, \mathbf{e}^\top \mathbf{R})\}$$

where $\mathbf{R} \leftarrow \{\pm 1\}^{\ell^2 m \times L}, \sigma = \omega(\sqrt{\log m}), \mathbf{C} \leftarrow \mathbb{Z}_q^{n \times L}$, $\mathbf{e} \in \mathbb{Z}_q^{\ell^2 m}$ is any fixed vector.

Proof. Parse $\mathbf{W}$ as $[\mathbf{W}_1 \mid \ldots \mid \mathbf{W}_\ell] \in \mathbb{Z}_q^{n\times \ell m}$, $\mathbf{D}$ as $[\mathbf{D}_1 \mid \ldots \mid \mathbf{D}_\ell] \in \mathbb{Z}_q^{m\times \ell m}$.

$$\left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{BR}, \mathbf{e}^\top \mathbf{R}) : \begin{array}{l} \mathbf{W} \leftarrow \mathbb{Z}_q^{n\times \ell m}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times \ell m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_\ell)^\top \otimes \mathbf{G}, \\ \mathbf{R} \leftarrow \{\pm 1\}^{\ell^2 m \times L} \end{array} \right\}$$

$$\approx_s \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{BR}, \mathbf{e}^\top \mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_\ell] \leftarrow \mathbb{Z}_q^{n\times (\ell-1)m}, \\ \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times \ell m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_\ell)^\top \otimes \mathbf{G}, \\ \mathbf{R} \leftarrow \{\pm 1\}^{\ell^2 m \times L} \end{array} \right\} \left(\begin{array}{l} \text{By matrix distribution of} \\ \text{gadget trapdoors} \end{array} \right)$$

$$\approx_s \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{BR}, \mathbf{e}^\top \mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_\ell] \leftarrow \mathbb{Z}_q^{n\times (\ell-1)m}, \\ \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_\ell] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times (\ell-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_\ell)^\top \otimes \mathbf{G}, \\ \mathbf{R} = \left[\mathbf{R}_1^\top \mid \mathbf{R}_2^\top \mid \ldots \mathbf{R}_{\ell^2}^\top\right]^\top \leftarrow \{\pm 1\}^{\ell^2 m \times L} \end{array} \right\} \text{(By Lemma A.1)}$$

$$\equiv \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{AR}_2 + \mathbf{X}, \mathbf{e}^\top \mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_\ell] \leftarrow \mathbb{Z}_q^{n\times (\ell-1)m}, \\ \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_\ell] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times (\ell-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_\ell)^\top \otimes \mathbf{G}, \\ \mathbf{R} = \left[\mathbf{R}_1^\top \mid \mathbf{R}_2^\top \mid \ldots \mid \mathbf{R}_{\ell^2}^\top\right]^\top \leftarrow \{\pm 1\}^{\ell^2 m \times L} \end{array} \right\}$$

where $\mathbf{X} = \sum_{(i,j)\in[\ell]\times[\ell]\setminus\{(1,2)\}} (\mathbf{W}_i \mathbf{D}_j + \delta_{ij}\mathbf{G})\mathbf{R}_{(i-1)\ell+j}$ and $\delta_{ij} = 1$ if and only if $i = j$.

$$\approx_s \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{C}, \mathbf{e}^\top \mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_\ell] \leftarrow \mathbb{Z}_q^{n\times (\ell-1)m}, \\ \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_\ell] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times (\ell-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_\ell)^\top \otimes \mathbf{G}, \\ \mathbf{C} \leftarrow \mathbb{Z}_q^{n\times L}, \mathbf{R} \leftarrow \{\pm 1\}^{\ell^2 m \times L} \end{array} \right\} \text{(By Lemma 3.2)}$$

$$\approx_s \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{C}, \mathbf{e}^\top \mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_\ell] \leftarrow \mathbb{Z}_q^{n\times (\ell-1)m}, \\ \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times \ell m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_\ell)^\top \otimes \mathbf{G}, \\ \mathbf{C} \leftarrow \mathbb{Z}_q^{n\times L}, \mathbf{R} \leftarrow \{\pm 1\}^{\ell^2 m \times L} \end{array} \right\} \text{(By Lemma A.1)}$$

$$\approx_s \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{C}, \mathbf{e}^\top \mathbf{R}) : \begin{array}{l} \mathbf{W} \leftarrow \mathbb{Z}_q^{n\times \ell m}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times \ell m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_\ell)^\top \otimes \mathbf{G}, \\ \mathbf{C} \leftarrow \mathbb{Z}_q^{n\times L}, \mathbf{R} \leftarrow \{\pm 1\}^{\ell^2 m \times L} \end{array} \right\} \left(\begin{array}{l} \text{By matrix distribution of} \\ \text{gadget trapdoors} \end{array} \right)$$

□

Lemma A.4. Let n, m, q be lattice parameters such that $m = \Omega(n \log q)$, q is an odd prime. Then, for $\sigma > \log m$,

$$\left\{(\mathbf{W}, \mathbf{D}, \mathbf{B}) : \mathbf{W} \leftarrow \mathbb{Z}_q^{n \times 2m^3}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times 2m^3}, \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}\right\} \approx_s$$
$$\left\{(\mathbf{W}, \mathbf{D}, \mathbf{B}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G} \end{array}\right\}$$

Proof. Parse $\mathbf{W}$ as $[\mathbf{W}_1 \mid \ldots \mid \mathbf{W}_{2m^2}] \in \mathbb{Z}_q^{n \times 2m^3}$, $\mathbf{D}$ as $[\mathbf{D}_1 \mid \ldots \mid \mathbf{D}_{2m^2}] \in \mathbb{Z}_q^{m \times 2m^3}$.

$$\left\{(\mathbf{W}, \mathbf{D}, \mathbf{B}) : \mathbf{W} \leftarrow \mathbb{Z}_q^{n \times 2m^3}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times 2m^3}, \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}\right\}$$
$$\approx_s \left\{(\mathbf{W}, \mathbf{D}, \mathbf{B}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times (2m^2-1)m}, \\ \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times 2m^3}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G} \end{array}\right\} \quad \left(\begin{array}{l}\text{By matrix distribution of} \\ \text{gadget trapdoors}\end{array}\right)$$
$$\approx_s \left\{(\mathbf{W}, \mathbf{D}, \mathbf{B}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G} \end{array}\right\} \quad \text{(By Lemma A.1)}$$

$$\approx_s \left\{(\mathbf{W}, \mathbf{D}, \mathbf{B}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G} \end{array}\right\} \quad \left(\begin{array}{l}\text{By matrix distribution of} \\ \text{gadget trapdoors}\end{array}\right)$$

□

Lemma A.5. Let n, m, q be lattice parameters such that $m = \Omega(n \log q)$, q is an odd prime. Then,

for $\sigma > \log m$, $\ell > 3, \ell = \mathsf{poly}(\lambda)$, any $L = \mathsf{poly}(\lambda)$,

$$\left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{BR}, \mathbf{e}^\top \mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{R} \leftarrow \{\pm 1\}^{4m^5 \times L} \end{array} \right\} \approx_s$$

$$\left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{C}, \mathbf{e}^\top \mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{C} \leftarrow \mathbb{Z}_q^{n \times L}, \mathbf{R} \leftarrow \{\pm 1\}^{4m^5 \times L} \end{array} \right\}$$

where $\mathbf{e} \in \mathbb{Z}_q^{4m^5}$ is any fixed vector.

Proof. Parse $\mathbf{W}$ as $[\mathbf{W}_1 \mid \ldots \mid \mathbf{W}_{2m^2}] \in \mathbb{Z}_q^{n \times 2m^3}$, $\mathbf{D}$ as $[\mathbf{D}_1 \mid \ldots \mid \mathbf{D}_{2m^2}] \in \mathbb{Z}_q^{m \times 2m^3}$.

$$\left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{BR}, \mathbf{e}^\top \mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{T}_{\mathbf{B}} = \mathbb{1}_{4m^4,2} \otimes \mathbf{T}_{\mathbf{A}}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{R} \leftarrow \{\pm 1\}^{4m^5 \times L} \end{array} \right\}$$

$$\approx_s \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{BR}, \mathbf{e}^\top \mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times (2m^2-1)m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ \mathbf{A}' \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{D}_3 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}', \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_4 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-2)m}, \\ \mathbf{T}_{\mathbf{B}} = \mathbb{1}_{4m^4,2} \otimes \mathbf{T}_{\mathbf{A}}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{R} = \left[\mathbf{R}_1^\top \mid \mathbf{R}_2^\top \mid \mathbf{R}_3^\top \mid \ldots \mid \mathbf{R}_{4m^4}^\top\right]^\top \leftarrow \{\pm 1\}^{4m^5 \times L} \end{array} \right\} \text{(By Lemma A.1)}$$

$$\equiv \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}'\mathbf{R}_3 + \mathbf{X}, \mathbf{e}^\top\mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n\times(2m^2-1)m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ \mathbf{A}' \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{D}_3 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}', \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_4 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times(2m^2-2)m}, \\ \mathbf{T}_{\mathbf{B}} = \mathbb{1}_{4m^4,2} \otimes \mathbf{T}_{\mathbf{A}}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{R} = \left[\mathbf{R}_1^\top \mid \mathbf{R}_2^\top \mid \mathbf{R}_3^\top \mid \ldots \mid \mathbf{R}_{4m^4}^\top\right]^\top \leftarrow \{\pm 1\}^{4m^5\times L} \end{array} \right\}$$

where $\mathbf{X} = \sum_{(i,j)\in[2m^2]\times[2m^2]\setminus\{(1,3)\}} (\mathbf{W}_i\mathbf{D}_j + \delta_{ij}\mathbf{G})\mathbf{R}_{(i-1)2m^2+j}$ and $\delta_{ij} = 1$ if and only if $i = j$.

$$\approx_s \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{C}, \mathbf{e}^\top\mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n\times(2m^2-1)m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ \mathbf{A}' \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{D}_3 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}', \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_4 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times(2m^2-2)m}, \\ \mathbf{T}_{\mathbf{B}} = \mathbb{1}_{4m^4,2} \otimes \mathbf{T}_{\mathbf{A}}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{C} \leftarrow \mathbb{Z}_q^{n\times L}, \mathbf{R} \leftarrow \{\pm 1\}^{4m^5\times L} \end{array} \right\} \text{(By Lemma 3.2)}$$

$$\approx_s \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{C}, \mathbf{e}^\top\mathbf{R}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n\times(2m^2-1)m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \mathbf{D}_4 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times(2m^2-1)m}, \\ \mathbf{T}_{\mathbf{B}} = \mathbb{1}_{4m^4,2} \otimes \mathbf{T}_{\mathbf{A}}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{C} \leftarrow \mathbb{Z}_q^{n\times L}, \mathbf{R} \leftarrow \{\pm 1\}^{4m^5\times L} \end{array} \right\} \text{(By Lemma A.1)}$$

Note that in the above argument, $\mathbf{T}_{\mathbf{B}}$ is independent of $\mathbf{A}'$ and $\mathbf{T}_{\mathbf{W}_1}$. So, this property holds even if we use $\mathbf{T}_{\mathbf{B}}$ to generate pre-images for any random $\mathbf{y} \leftarrow \mathbb{Z}_q^n$ similar to [DKW21, Lemma 3.4]. □

Lemma A.6. Let n, m, q be lattice parameters such that $m = \Omega(n \log q)$, q is an odd prime. Then,

for $\sigma > \log m$,

$$\left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{d}, \mathbf{y}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, , \\ \mathbf{y} \leftarrow \mathbb{Z}_q^n, \mathbf{d} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, \mathbf{y}, \sigma) \end{array} \right\} \approx_s$$

$$\left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{d}, \mathbf{B}\mathbf{d}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{d} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{4m^5} \end{array} \right\}$$

This can also be extended to matrices naturally by sampling column-wise

Proof. Parse $\mathbf{W}$ as $[\mathbf{W}_1 \mid \ldots \mid \mathbf{W}_{2m^2}] \in \mathbb{Z}_q^{n \times 2m^3}$, $\mathbf{D}$ as $[\mathbf{D}_1 \mid \ldots \mid \mathbf{D}_{2m^2}] \in \mathbb{Z}_q^{m \times 2m^3}$. Note that by construction, there exists a matrix $\mathbf{A}, \mathbf{T}_{\mathbf{A}}$ inside $\mathbf{B}, \mathbf{T}_{\mathbf{B}}$. Moreover, $\mathbf{A} = \mathbf{W}_1\mathbf{D}_2$. Parse $\mathbf{d} \in \mathbb{Z}_q^{4m^5}$ as $\mathbf{d} = [\mathbf{d}_1^\top \mid \ldots \mid \mathbf{d}_{4m^4}^\top]^\top$. Let $\widetilde{\mathbf{y}} = \mathbf{B}(\mathbf{d} - \mathbb{1}_{4m^4,2} \otimes \mathbf{d}_2) \in \mathbb{Z}_q^n$ in each of these hybrids below.

$$\left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{d}, \mathbf{B}\mathbf{d}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{d} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{4m^5} \end{array} \right\}$$

$$\equiv \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{d}, \mathbf{A}\mathbf{d}_2 + \widetilde{\mathbf{y}}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{d} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{4m^5} \end{array} \right\}$$

$$
\approx_s \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{d}, \mathbf{y}' + \widetilde{\mathbf{y}}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{y}' \leftarrow \mathbb{Z}_q^n, \mathbf{d}_1 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^m, \\ \left[\mathbf{d}_3^\top \mid \ldots \mid \mathbf{d}_{4m^4}^\top\right]^\top \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{(4m^4-2)m}, \\ \mathbf{d}_2 \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{y}', \sigma) \end{array} \right\} \quad \text{(By Lemma A.1)}
$$

$$
\equiv \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{d}, \mathbf{y}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{y} \leftarrow \mathbb{Z}_q^n, \mathbf{d}_1 \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^m, [\mathbf{d}_3^\top \mid \ldots \mid \mathbf{d}_{4m^4}^\top]^\top \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{(4m^4-2)m}, \\ \mathbf{d}_2 \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{y} - \widetilde{\mathbf{y}}, \sigma) \end{array} \right\}
$$

$$
\approx_s \left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{d}, \mathbf{y}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{y} \leftarrow \mathbb{Z}_q^n, \mathbf{d} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, \mathbf{y}, \sigma) \end{array} \right\} \quad \text{(By [ABB10])}
$$

□

Lemma A.7. Let n, m, q be lattice parameters such that $m = \Omega(n \log q)$, q is an odd prime. Then, for any $\mathbf{y} \in \mathbb{Z}_q^n, \mathbf{z} \in \mathbb{Z}_q^{4m^5}$,

$$
\left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{d}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{d} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, \mathbf{y} + \mathbf{B}\mathbf{z}, \sigma) \end{array} \right\} \approx_s
$$

$$
\left\{ (\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{d}) : \begin{array}{l} (\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ [\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times m}, \\ (\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \\ \mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma), \\ [\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^2-1)m}, \\ \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}, \\ \mathbf{d} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, \mathbf{y}, \sigma) + \mathbf{z} \end{array} \right\}
$$

where $\sigma > \lambda^{\omega(1)}\sqrt{m}\,\|\mathbf{z}\|$. This can also be extended to matrices naturally by sampling column-wise.

Proof. This is similar to proof of Lemma A.6 except using Lemma A.2. □

B Proof of Theorem 4.5

Correctness. By correctness of MCOM, we have that $\mathbf{C_M V}_N = \mathbf{M} - \mathbf{B Z_M}$ for the values computed in Com. Moreover, if sd is same in Com and Open, we have that same $\mathbf{C}_0, \mathbf{Z}_0$ are computed in Com and Open. Then, by correctness of ER, we have that $\mathbf{C}_0\mathbf{V}_N = -\mathbf{B Z}_0$. Hence, $\mathbf{C V}_N = (\mathbf{C}_0 + \mathbf{C_M})\mathbf{V}_N = \mathbf{M} - \mathbf{B}(\mathbf{Z}_0 + \mathbf{Z_M}) = \mathbf{M} - \mathbf{BZ}$. Thus, correctness holds.

Compactness. As $\mathsf{aux} \in \{0,1\}^\lambda$, we have that compactness for aux holds and $p_1(m) = 2m^3$ and $p_2(m) = 2m^2$. Moreover, $\|\mathbf{V}_N\| \le O(\|\mathbf{D}\|\, m^4 \log q)$ by compactness of MCOM. Similarly, $\|\mathbf{Z}\| \le \|\mathbf{Z}_0\| + \|\mathbf{Z_M}\| \le \sqrt{m}\widetilde{\sigma} + O(m^{15/2}\sigma \log q \log N) = O(\sqrt{m}\widetilde{\sigma})$ by our parameter choices.

Proof of Theorem 4.5. We define the following hybrids for an admissible adversary $\mathcal{A}$.

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the commitment indistinguishability game from Definition 4.1 with $b = 0$.

- **Setup Phase.** $\mathcal{A}$ sends $(1^n, 1^m, q, \sigma, B)$. Sample $\widetilde{\sigma} > \max\{2^{3\lambda} \cdot \sigma \cdot O(m^{25/2}), \lambda^{\omega(1)} \cdot (B + m^{15/2} \cdot \sigma \cdot \log q \cdot \log N)\}$. Set $\mathsf{pp} := (n, m, q, \sigma, B, \widetilde{\sigma}, \mathcal{H})$ where $\mathcal{H}$ is a handle to the random oracle. Initiate empty dictionaries $\mathsf{H} := \emptyset, \mathcal{Y} := \emptyset$ and handle $h := 1$.

  Sample $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times 2m^3}$ and $\mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times 2m^3}$. Compute $\mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}$.

  Send $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$ to $\mathcal{A}$.

- **Pre-Challenge Queries.** $\mathcal{A}$ is allowed to make the following queries.

  - **Random Oracle Queries.** Given $(\mathsf{sd}, j) \in \{0,1\}^\lambda \times \{0,1\}^\lambda$, if there is an entry for (sd, j) in H, retrieve and send $\mathsf{H}[(\mathsf{sd}, j)]$. Otherwise, sample $\beta \leftarrow \{0,1\}$, set $\mathsf{H}[(\mathsf{sd}, j)] = \beta$ and send β to $\mathcal{A}$.
  - **Generation Queries.** Sample $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and compute $\mathbf{t} = \mathbf{B}\mathbf{y}$. Send $(\mathbf{t}, h)$ to $\mathcal{A}$. Store $\mathcal{Y}[h] := \mathbf{y}$, and update $h := h + 1$.
  - **Corruption Queries.** $\mathcal{A}$ sends $h' \in [h]$. If there is no entry for h' in $\mathcal{Y}$, send $\perp$ to $\mathcal{A}$. Otherwise, retrieve and send $\mathcal{Y}[h']$. Remove the entry corresponding to h' from $\mathcal{Y}$.

- **Challenge Query.** $\mathcal{A}$ sends $S \subseteq [h]$ and $\mathbf{R} \in \mathbb{Z}_q^{4m^5 \times N}$ such that $\|\mathbf{R}\| \le B$. Let $N = |S|$ and $(h_1, \ldots, h_N)$ be the lexicographical ordering of S. Retrieve and compute $\mathbf{Y} = [\mathcal{Y}[h_1] \mid \ldots \mid \mathcal{Y}[h_N]]$. Compute $\mathbf{M} = \mathbf{B}(\mathbf{Y} + \mathbf{R})$.

  1. Compute $\mathbf{C_M} = \mathsf{MCOM.Com}(\mathbf{M})$.
  2. Sample $\mathsf{sd} \leftarrow \{0,1\}^\lambda$.
  3. Let $\rho = \mathsf{poly}(\lambda, N)$ be the length of the random coins used by ER for column length N. Sample $r_i = \mathcal{H}(\mathsf{sd}, i)$ for each $i \in [\rho]$. Set random coins $r = (r_i)_{i \in [\rho]}$.
  4. Compute $(\mathbf{C}_0, \mathbf{Z}_0) = \mathsf{ER.Sample}(1^\lambda, 1^N, \widetilde{\sigma}; r)$.

  Send $\mathbf{C} := \mathbf{C}_0 + \mathbf{C_M}$ and $\mathsf{aux} := \mathsf{sd}$ to $\mathcal{A}$.

- **Post-Challenge Queries.** Handle these similarly to the pre-challenge queries. Recall that an admissible $\mathcal{A}$ does not corrupt any $h^* \in S$ in post-challenge queries.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ except that we abort the experiment if the seed sampled during the challenge phase already appears in H. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.
- **Pre-Challenge Queries.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.
- **Challenge Query.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$, except that step 2 is replaced by the following.

  2. Sample $\mathsf{sd} \leftarrow \{0,1\}^\lambda$. If H already contains an entry for (sd, j) for some $j \in \{0,1\}^\lambda$, abort the experiment and output a random bit. Otherwise, proceed as in $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.

- **Post-Challenge Queries.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ except that we program H with uniformly sampled coins during the challenge phase, instead of reading the coins off the random oracle. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Pre-Challenge Queries.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Challenge Query.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$, except that step 3 is replaced by the following.

  3. Let $\rho = \mathsf{poly}(\lambda, N)$ be the length of the random coins used by ER for column length N. Sample $(r_1, \ldots, r_\rho) \leftarrow \{0,1\}^\rho$ and set $\mathsf{H}[(\mathsf{sd}, i)] = r_i$ for each $i \in [\rho]$. Set random coins $r = (r_i)_{i \in [\rho]}$.

- **Post-Challenge Queries.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ except that the setup phase samples $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ together with a trapdoor. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $(1^n, 1^m, q, \sigma, B)$. Sample $\widetilde{\sigma} > \max\{2^{3\lambda} \cdot \sigma \cdot O(m^{25/2}), \lambda^{\omega(1)} \cdot (B + m^{15/2} \cdot \sigma \cdot \log q \cdot \log N)\}$. Set $\mathsf{pp} := (n, m, q, \sigma, B, \widetilde{\sigma}, \mathcal{H})$ where $\mathcal{H}$ is a handle to the random oracle. Initiate empty dictionaries $\mathsf{H} := \emptyset, \mathcal{Y} := \emptyset$ and handle $h := 1$.

  Sample $(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, $[\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times (2m^3 - m)}$, $(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, $\mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma)$, and $[\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^3 - m)}$.

Set $\mathbf{W} := [\mathbf{W}_1 \mid \ldots \mid \mathbf{W}_{2m^2}]$ and $\mathbf{D} := [\mathbf{D}_1 \mid \ldots \mid \mathbf{D}_{2m^2}]$. Compute $\mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}$ along with the resulting trapdoor $\mathbf{T_B}$ satisfying $\mathbf{BT_B} = \mathbf{G}$ and $\|\mathbf{T_B}\| = 1$.

Send $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$ to $\mathcal{A}$. Note that $\mathbf{T_B}$ is retained by the experiment and is never given to $\mathcal{A}$.

- **Pre-Challenge Queries.** Same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$.
- **Challenge Query.** Same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$.
- **Post-Challenge Queries.** Same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ except that generation queries return a uniformly random $\mathbf{t}$ and the corresponding $\mathbf{y}$ is sampled on-the-fly, only if and when the handle is corrupted. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $(1^n, 1^m, q, \sigma, B)$. Sample $\widetilde{\sigma} > \max\{2^{3\lambda} \cdot \sigma \cdot O(m^{25/2}), \lambda^{\omega(1)} \cdot (B + m^{15/2} \cdot \sigma \cdot \log q \cdot \log N)\}$. Set $\mathsf{pp} := (n, m, q, \sigma, B, \widetilde{\sigma}, \mathcal{H})$ where $\mathcal{H}$ is a handle to the random oracle. Initiate empty dictionaries $\mathsf{H} := \emptyset, \mathcal{Y} := \emptyset$ and handle $h := 1$.

  Sample $(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, $[\mathbf{W}_2 \mid \ldots \mid \mathbf{W}_{2m^2}] \leftarrow \mathbb{Z}_q^{n \times (2m^3 - m)}$, $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, $\mathbf{D}_2 \leftarrow \mathsf{SamplePre}(\mathbf{W}_1, \mathbf{T}_{\mathbf{W}_1}, \mathbf{A}, \sigma)$, and $[\mathbf{D}_1 \mid \mathbf{D}_3 \mid \ldots \mid \mathbf{D}_{2m^2}] \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times (2m^3 - m)}$. Set $\mathbf{W} := [\mathbf{W}_1 \mid \ldots \mid \mathbf{W}_{2m^2}]$, $\mathbf{D} := [\mathbf{D}_1 \mid \ldots \mid \mathbf{D}_{2m^2}]$, and compute $\mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}$ along with its trapdoor $\mathbf{T_B}$.

  Send $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$ to $\mathcal{A}$.

- **Pre-Challenge Queries.** $\mathcal{A}$ is allowed to make the following queries.
  - **Random Oracle Queries.** Given $(\mathsf{sd}, j) \in \{0,1\}^\lambda \times \{0,1\}^\lambda$, if there is an entry for (sd, j) in H, retrieve and send $\mathsf{H}[(\mathsf{sd}, j)]$. Otherwise, sample $\beta \leftarrow \{0,1\}$, set $\mathsf{H}[(\mathsf{sd}, j)] = \beta$ and send β to $\mathcal{A}$.
  - **Generation Queries.** Sample $\mathbf{t} \leftarrow \mathbb{Z}_q^n$. Send $(\mathbf{t}, h)$ to $\mathcal{A}$. Store $\mathcal{Y}[h] := \mathbf{t}$, and update $h := h + 1$.
  - **Corruption Queries.** $\mathcal{A}$ sends $h' \in [h]$. If there is no entry for h' in $\mathcal{Y}$, send $\perp$ to $\mathcal{A}$. Otherwise, retrieve $\mathbf{t} := \mathcal{Y}[h']$, sample $\mathbf{y} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, \mathbf{t}, \widetilde{\sigma})$, and send $\mathbf{y}$ to $\mathcal{A}$. Remove the entry corresponding to h' from $\mathcal{Y}$.
- **Challenge Query.** $\mathcal{A}$ sends $S \subseteq [h]$ and $\mathbf{R} \in \mathbb{Z}_q^{4m^5 \times N}$ such that $\|\mathbf{R}\| \leq B$. Let $N = |S|$ and $(h_1, \ldots, h_N)$ be the lexicographical ordering of S. Retrieve $\mathbf{t}_{h_1}, \ldots, \mathbf{t}_{h_N}$ from $\mathcal{Y}$ and compute $\mathbf{M} = [\mathbf{t}_{h_1} \mid \ldots \mid \mathbf{t}_{h_N}] + \mathbf{BR}$.
  1. Compute $\mathbf{C_M} = \mathsf{MCOM.Com}(\mathbf{M})$.
  2. Sample $\mathsf{sd} \leftarrow \{0,1\}^\lambda$. If H already contains an entry for (sd, j) for some $j \in \{0,1\}^\lambda$, abort the experiment and output a random bit.
  3. Let $\rho = \mathsf{poly}(\lambda, N)$ be the length of the random coins used by ER for column length N. Sample $(r_1, \ldots, r_\rho) \leftarrow \{0,1\}^\rho$ and set $\mathsf{H}[(\mathsf{sd}, i)] = r_i$ for each $i \in [\rho]$. Set random coins $r = (r_i)_{i \in [\rho]}$.

4. Compute $(\mathbf{C}_0, \mathbf{Z}_0) = \mathsf{ER.Sample}(1^\lambda, 1^N, \widetilde{\sigma}; r)$.

Send $\mathbf{C} := \mathbf{C}_0 + \mathbf{C}_\mathbf{M}$ and $\mathsf{aux} := \mathsf{sd}$ to $\mathcal{A}$.

- **Post-Challenge Queries.** Handle these similarly to the pre-challenge queries. Recall that an admissible $\mathcal{A}$ does not corrupt any $h^* \in S$ in post-challenge queries.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_5^\mathcal{A}(1^\lambda)$. This is the same as $\mathbf{Hyb}_4^\mathcal{A}(1^\lambda)$ except that we program H with coins that *explain* the sampled pair $(\mathbf{C}_0, \mathbf{Z}_0)$, rather than with the coins that generated it. As this regroups the steps of the challenge query, we write that phase out in full. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_4^\mathcal{A}(1^\lambda)$.
- **Pre-Challenge Queries.** Same as $\mathbf{Hyb}_4^\mathcal{A}(1^\lambda)$.
- **Challenge Query.** $\mathcal{A}$ sends $S \subseteq [h]$ and $\mathbf{R} \in \mathbb{Z}_q^{4m^5 \times N}$ such that $\|\mathbf{R}\| \leq B$. Let $N = |S|$ and $(h_1, \ldots, h_N)$ be the lexicographical ordering of S. Retrieve $\mathbf{t}_{h_1}, \ldots, \mathbf{t}_{h_N}$ from $\mathcal{Y}$ and compute $\mathbf{M} = [\mathbf{t}_{h_1} \mid \ldots \mid \mathbf{t}_{h_N}] + \mathbf{BR}$.

  1. Compute $\mathbf{C}_\mathbf{M} = \mathsf{MCOM.Com}(\mathbf{M})$.
  2. Sample $\mathsf{sd} \leftarrow \{0,1\}^\lambda$. If H already contains an entry for (sd, j) for some $j \in \{0,1\}^\lambda$, abort the experiment and output a random bit.
  3. Let $\rho = \mathsf{poly}(\lambda, N)$ be the length of the random coins used by ER for column length N. Sample $r \leftarrow \{0,1\}^\rho$ and compute $(\mathbf{C}_0, \mathbf{Z}_0) = \mathsf{ER.Sample}(1^\lambda, 1^N, \widetilde{\sigma}; r)$. We no longer program H with r.
  4. Sample $r' \leftarrow \mathsf{ER.Explain}(1^\lambda, 1^T, \mathbf{C}_0, \mathbf{Z}_0, \widetilde{\sigma})$.
  5. Set $\mathsf{H}[(\mathsf{sd}, i)] = r'_i$ for each $i \in [\rho]$.

  Send $\mathbf{C} := \mathbf{C}_0 + \mathbf{C}_\mathbf{M}$ and $\mathsf{aux} := \mathsf{sd}$ to $\mathcal{A}$.
- **Post-Challenge Queries.** Same as $\mathbf{Hyb}_4^\mathcal{A}(1^\lambda)$.
- **Guess Phase.** Same as $\mathbf{Hyb}_4^\mathcal{A}(1^\lambda)$.

$\mathbf{Hyb}_6^\mathcal{A}(1^\lambda)$. This is the same as $\mathbf{Hyb}_5^\mathcal{A}(1^\lambda)$ except that $(\mathbf{C}_0, \mathbf{Z}_0)$ is sampled directly rather than through $\mathsf{ER.Sample}$. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_5^\mathcal{A}(1^\lambda)$.
- **Pre-Challenge Queries.** Same as $\mathbf{Hyb}_5^\mathcal{A}(1^\lambda)$.
- **Challenge Query.** Same as $\mathbf{Hyb}_5^\mathcal{A}(1^\lambda)$, except that step 3 is replaced by the following.

  3. Sample $\mathbf{C}_0 \leftarrow \mathbb{Z}_q^{n \times m}$ and $\mathbf{Z}_0 \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_\mathbf{B}, -\mathbf{C}_0\mathbf{V}_N, \widetilde{\sigma})$, where $\mathbf{V}_N = \mathsf{MCOM.Verify}(1^N)$.
- **Post-Challenge Queries.** Same as $\mathbf{Hyb}_5^\mathcal{A}(1^\lambda)$.
- **Guess Phase.** Same as $\mathbf{Hyb}_5^\mathcal{A}(1^\lambda)$.

$\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$ except that $\mathbf{C}_0$ is set so that the commitment sent to $\mathcal{A}$ is $\mathbf{BR_C}$ for a low-norm $\mathbf{R_C}$. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$.
- **Pre-Challenge Queries.** Same as $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$.
- **Challenge Query.** Same as $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$, except that step 3 is replaced by the following.

  3. Sample $\mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$ and set $\mathbf{C}_0 = \mathbf{BR_C} - \mathbf{C_M}$. Sample $\mathbf{Z}_0 \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, -\mathbf{C}_0\mathbf{V}_N, \widetilde{\sigma})$, where $\mathbf{V}_N = \mathsf{MCOM.Verify}(1^N)$.

- **Post-Challenge Queries.** Same as $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Same as $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_8^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$ except that $\mathbf{Z}_0$ is obtained by pre-image sampling on $[\mathbf{t}_{h_1} \mid \ldots \mid \mathbf{t}_{h_N}]$ and then shifting by a low-norm term, rather than by pre-image sampling on $-\mathbf{C}_0\mathbf{V}_N$. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$.
- **Pre-Challenge Queries.** Same as $\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$.
- **Challenge Query.** Same as $\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$, except that step 3 is replaced by the following.

  3. Sample $\mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$ and set $\mathbf{C}_0 = \mathbf{BR_C} - \mathbf{C_M}$. Set $\mathbf{Z}_0 := \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, [\mathbf{t}_{h_1} \mid \ldots \mid \mathbf{t}_{h_N}], \widetilde{\sigma}) + \mathbf{R} - \mathbf{Z_M} - \mathbf{R_C}\mathbf{V}_N$, where $\mathbf{Z_M} = \mathsf{MCOM.Open}(\mathbf{M})$, $\mathbf{V}_N = \mathsf{MCOM.Verify}(1^N)$.

- **Post-Challenge Queries.** Same as $\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Same as $\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_9^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_8^{\mathcal{A}}(1^\lambda)$ except that generation queries again sample a gaussian $\mathbf{y}$ and return $\mathbf{By}$, and $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ is once again sampled without a trapdoor. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $(1^n, 1^m, q, \sigma, B)$. Sample $\widetilde{\sigma} > \max\{2^{3\lambda} \cdot \sigma \cdot O(m^{25/2}), \lambda^{\omega(1)} \cdot (B + m^{15/2} \cdot \sigma \cdot \log q \cdot \log N)\}$. Set $\mathsf{pp} := (n, m, q, \sigma, B, \widetilde{\sigma}, \mathcal{H})$ where $\mathcal{H}$ is a handle to the random oracle. Initiate empty dictionaries $\mathsf{H} := \emptyset, \mathcal{Y} := \emptyset$ and handle $h := 1$.

  Sample $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times 2m^3}$ and $\mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times 2m^3}$. Compute $\mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}$.

  Send $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$ to $\mathcal{A}$.

- **Pre-Challenge Queries.** $\mathcal{A}$ is allowed to make the following queries.

  - **Random Oracle Queries.** Given $(\mathsf{sd}, j) \in \{0,1\}^\lambda \times \{0,1\}^\lambda$, if there is an entry for (sd, j) in H, retrieve and send $\mathsf{H}[(\mathsf{sd}, j)]$. Otherwise, sample $\beta \leftarrow \{0,1\}$, set $\mathsf{H}[(\mathsf{sd}, j)] = \beta$ and send β to $\mathcal{A}$.

- **Generation Queries.** Sample $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and compute $\mathbf{t} = \mathbf{B}\mathbf{y}$. Send $(\mathbf{t}, h)$ to $\mathcal{A}$. Store $\mathcal{Y}[h] := \mathbf{y}$, and update $h := h + 1$.
- **Corruption Queries.** $\mathcal{A}$ sends $h' \in [h]$. If there is no entry for h' in $\mathcal{Y}$, send $\perp$ to $\mathcal{A}$. Otherwise, retrieve and send $\mathcal{Y}[h']$. Remove the entry corresponding to h' from $\mathcal{Y}$.

- **Challenge Query.** $\mathcal{A}$ sends $S \subseteq [h]$ and $\mathbf{R} \in \mathbb{Z}_q^{4m^5 \times N}$ such that $\|\mathbf{R}\| \leq B$. Let $N = |S|$ and $(h_1, \ldots, h_N)$ be the lexicographical ordering of S. Retrieve and compute $\mathbf{Y} = [\mathcal{Y}[h_1] \mid \ldots \mid \mathcal{Y}[h_N]]$. Compute $\mathbf{M} = \mathbf{B}(\mathbf{Y} + \mathbf{R})$.

  1. Compute $\mathbf{C_M} = \mathsf{MCOM.Com}(\mathbf{M})$, $\mathbf{V}_N = \mathsf{MCOM.Verify}(1^N)$, $\mathbf{Z_M} = \mathsf{MCOM.Open}(\mathbf{M})$.
  2. Sample $\mathsf{sd} \leftarrow \{0,1\}^\lambda$. If H already contains an entry for (sd, j) for some $j \in \{0,1\}^\lambda$, abort the experiment and output a random bit.
  3. Sample $\mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$ and set $\mathbf{C}_0 = \mathbf{B}\mathbf{R_C} - \mathbf{C_M}$.
  4. Set $\mathbf{Z}_0 = (\mathbf{Y} + \mathbf{R}) - \mathbf{Z_M} - \mathbf{R_C}\mathbf{V}_N$.
  5. Sample $r' \leftarrow \mathsf{ER.Explain}(1^\lambda, 1^T, \mathbf{C}_0, \mathbf{Z}_0, \widetilde{\sigma})$ and set $\mathsf{H}[(\mathsf{sd}, i)] = r'_i$ for each $i \in [\rho]$, where $\rho = |r'|$.

  Send $\mathbf{C} := \mathbf{C}_0 + \mathbf{C_M} = \mathbf{B}\mathbf{R_C}$ and $\mathsf{aux} := \mathsf{sd}$ to $\mathcal{A}$.

- **Post-Challenge Queries.** Handle these similarly to the pre-challenge queries. Recall that an admissible $\mathcal{A}$ does not corrupt any $h^* \in S$ in post-challenge queries.

- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

Lemma B.1. For any PPT adversary, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids are identical unless $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ aborts, so it suffices to bound the probability of aborting. As $\mathcal{A}$ is a polynomial-time adversary, there exists a polynomial $Q = Q(\lambda)$ such that $\mathcal{A}$ makes at most Q queries to $\mathcal{H}$ before the challenge phase. The seed sd is sampled uniformly from $\{0,1\}^\lambda$ during the challenge phase, independently of everything $\mathcal{A}$ has seen so far, and H contains an entry with first component sd only if $\mathcal{A}$ queried $\mathcal{H}$ on (sd, j) for some j. By a standard union bound argument, the probability that sd coincides with the first component of one of these at most Q queries is at most $Q/2^\lambda$. Thus the abort in $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ occurs with probability at most $Q/2^\lambda$, and the lemma holds. □

Lemma B.2. $\Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)]$.

Proof. The only difference between the two hybrids is how the entries $\mathsf{H}[(\mathsf{sd}, i)]$ for $i \in [\rho]$ come into being. In $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ they are created on-the-fly. Step 3 queries $\mathcal{H}(\mathsf{sd}, i)$, and since these entries are unset, each query samples a fresh $\beta \leftarrow \{0,1\}$ and records it. In $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ they are created early. Step 3 samples $(r_1, \ldots, r_\rho) \leftarrow \{0,1\}^\rho$ and writes them into H directly. Conditioned on not aborting, no query with first component sd was made before the challenge phase, so in both hybrids these ρ entries are uniform and independent of $\mathcal{A}$'s view up to that point. Any post-challenge query to $\mathcal{H}$ on (sd, i) then returns the recorded value in both hybrids alike. Hence the two hybrids induce the same distribution on $\mathcal{A}$'s view and are identically distributed. □

Lemma B.3. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The only difference between the two hybrids is how $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ is sampled in the setup phase. In $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ we sample $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times 2m^3}$ and $\mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times 2m^3}$ and derive $\mathbf{B}$; in $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ we instead generate $\mathbf{W}_1$ with a trapdoor, sample $\mathbf{D}_2$ as a pre-image of $\mathbf{A}$ under $\mathbf{W}_1$, and sample the remaining blocks as before. By the matrix distribution and pre-image distribution properties of gadget trapdoors (Lemma A.4), the tuple $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ produced in $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ is statistically close to the one produced in $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. As $\mathbf{B}$ is a deterministic function of $(\mathbf{W}, \mathbf{D})$ in both hybrids, and as the trapdoor $\mathbf{T_B}$ is retained by the experiment and never given to $\mathcal{A}$, the rest of $\mathcal{A}$'s view is unaffected. Thus the two hybrids are statistically close. □

Lemma B.4. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ in how a generation query and its (possible) later corruption are answered. In $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ we sample $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ eagerly, answer the generation query with $\mathbf{t} = \mathbf{By}$, and return the stored $\mathbf{y}$ on corruption. In $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ we answer the generation query with a uniform $\mathbf{t} \leftarrow \mathbb{Z}_q^n$ and only on corruption sample $\mathbf{y} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, \mathbf{t}, \widetilde{\sigma})$. By the pre-image distribution property of gadget trapdoors (Lemma A.6 and Lemma A.1), the joint distributions

$$\{(\mathbf{B}, \mathbf{y}, \mathbf{By}) : \mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}\} \approx_s \{(\mathbf{B}, \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, \mathbf{t}, \widetilde{\sigma}), \mathbf{t}) : \mathbf{t} \leftarrow \mathbb{Z}_q^n\}$$

are statistically close, so the pair $(\mathbf{t}, \mathbf{y})$ that $\mathcal{A}$ observes for any single handle is distributed almost identically in the two hybrids. As $\mathcal{A}$ is a PPT machine it makes at most $Q = Q(\lambda)$ generation queries, each answered with independently sampled randomness, so a standard hybrid argument over these Q handles incurs a loss of at most $Q \cdot \mathsf{negl}(\lambda)$, which remains negligible. Note also that the challenge query is unaffected as it reads $\mathbf{t}_{h_1}, \ldots, \mathbf{t}_{h_N}$, and $\mathbf{M} = [\mathbf{t}_{h_1} \mid \ldots \mid \mathbf{t}_{h_N}] + \mathbf{BR}$ agrees with $\mathbf{B}(\mathbf{Y} + \mathbf{R})$ in $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. □

Lemma B.5. $|\Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)]| \leq 1/T + \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ only in the coins written into H at the challenge query. In $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$, the pair $(\mathbf{C}_0, \mathbf{Z}_0)$ is generated from coins $r \leftarrow \{0,1\}^\rho$ and those same coins are recorded in H. In $\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$, the pair is generated from r as before, but the coins recorded are instead $r' \leftarrow \mathsf{ER.Explain}(1^\lambda, 1^T, \mathbf{C}_0, \mathbf{Z}_0, \widetilde{\sigma})$. Everything else, including $\mathbf{C} = \mathbf{C}_0 + \mathbf{C_M}$ and $\mathsf{aux} = \mathsf{sd}$, is unchanged, so $\mathcal{A}$'s view differs only through the ρ entries $\mathsf{H}[(\mathsf{sd}, i)]$, which it can read via post-challenge random oracle queries. By the explainability property of ER (Definition 4.3), the distribution of $(\mathbf{C}_0, \mathbf{Z}_0, r)$ and that of $(\mathbf{C}_0, \mathbf{Z}_0, r')$ are within statistical distance $1/T + \mathsf{negl}(\lambda)$. We show how to set T below. □

Lemma B.6. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ only in how $(\mathbf{C}_0, \mathbf{Z}_0)$ is produced in step 3 of the challenge query. In $\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$ it is the output of $\mathsf{ER.Sample}(1^\lambda, 1^N, \widetilde{\sigma}; r)$ on uniform coins r. In $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$ we sample $\mathbf{C}_0 \leftarrow \mathbb{Z}_q^{n \times m}$ uniformly and set $\mathbf{Z}_0 \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, -\mathbf{C}_0 \mathbf{V}_N, \widetilde{\sigma})$. By the randomness property of ER (Definition 4.3), these two distributions on $(\mathbf{C}_0, \mathbf{Z}_0)$ are statistically close, provided $\widetilde{\sigma}$ lies in

the admissible range, which holds by our parameter choices since $\widetilde{\sigma} > 2^{3\lambda} \cdot \sigma \cdot O(m^{25/2})$. The pair is then explained and programmed into H in exactly the same way in both hybrids, so $\mathcal{A}$'s view is a function of $(\mathbf{C}_0, \mathbf{Z}_0)$ alone and the two hybrids are statistically close. Note that from $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$ onwards the coins r play no role, as $\mathbf{C}_0$ and $\mathbf{Z}_0$ are sampled directly. □

Lemma B.7. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ only in how $\mathbf{C}_0$ is sampled. In $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$ we sample $\mathbf{C}_0 \leftarrow \mathbb{Z}_q^{n \times m}$; in $\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$ we sample $\mathbf{R}_\mathbf{C} \leftarrow \{\pm 1\}^{4m^5 \times m}$ and set $\mathbf{C}_0 = \mathbf{B}\mathbf{R}_\mathbf{C} - \mathbf{C}_\mathbf{M}$. By the leftover hash lemma for structured $\mathbf{B}$ (Lemma A.5), we have

$$\{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{B}\mathbf{R}_\mathbf{C})\} \approx_s \{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{C})\}$$

for $\mathbf{R}_\mathbf{C} \leftarrow \{\pm 1\}^{4m^5 \times m}$ and $\mathbf{C} \leftarrow \mathbb{Z}_q^{n \times m}$, and this holds even though $\mathbf{B}$ is sampled with a trapdoor in these hybrids. The matrix $\mathbf{C}_\mathbf{M}$ is determined by $\mathbf{M}$, which in turn is fixed by $\mathcal{A}$'s choice of S and $\mathbf{R}$ before $\mathbf{R}_\mathbf{C}$ is sampled, so subtracting $\mathbf{C}_\mathbf{M}$ is a fixed shift and does not affect the argument. Hence $\mathbf{B}\mathbf{R}_\mathbf{C} - \mathbf{C}_\mathbf{M}$ is statistically close to uniform over $\mathbb{Z}_q^{n \times m}$. As $\mathbf{Z}_0$ is computed from $\mathbf{C}_0$ in the same way in both hybrids, $\mathcal{A}$'s view is statistically close. □

Lemma B.8. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_8^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ only in how $\mathbf{Z}_0$ is sampled. We first rewrite the pre-image target used in $\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$. Since $\mathbf{C}_0 = \mathbf{B}\mathbf{R}_\mathbf{C} - \mathbf{C}_\mathbf{M}$, we have $-\mathbf{C}_0\mathbf{V}_N = \mathbf{C}_\mathbf{M}\mathbf{V}_N - \mathbf{B}\mathbf{R}_\mathbf{C}\mathbf{V}_N$, and by correctness of MCOM we have $\mathbf{C}_\mathbf{M}\mathbf{V}_N = \mathbf{M} - \mathbf{B}\mathbf{Z}_\mathbf{M}$ where $\mathbf{Z}_\mathbf{M} = \mathsf{MCOM.Open}(\mathbf{M})$. Substituting $\mathbf{M} = [\mathbf{t}_{h_1} \mid \ldots \mid \mathbf{t}_{h_N}] + \mathbf{B}\mathbf{R}$ gives

$$-\mathbf{C}_0\mathbf{V}_N = [\mathbf{t}_{h_1} \mid \ldots \mid \mathbf{t}_{h_N}] + \mathbf{B}\,(\mathbf{R} - \mathbf{Z}_\mathbf{M} - \mathbf{R}_\mathbf{C}\mathbf{V}_N)$$

So $\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$ samples $\mathbf{Z}_0$ as a pre-image of $[\mathbf{t}_{h_1} \mid \ldots \mid \mathbf{t}_{h_N}] + \mathbf{B}\mathbf{E}$ for the low-norm matrix $\mathbf{E} := \mathbf{R} - \mathbf{Z}_\mathbf{M} - \mathbf{R}_\mathbf{C}\mathbf{V}_N$, whereas $\mathbf{Hyb}_8^{\mathcal{A}}(1^\lambda)$ samples a pre-image of $[\mathbf{t}_{h_1} \mid \ldots \mid \mathbf{t}_{h_N}]$ and adds $\mathbf{E}$ afterwards. By the gaussian pre-image smudging lemma (Lemma A.7 and Lemma A.2), these two distributions are statistically close as long as $\widetilde{\sigma} > \lambda^{\omega(1)}\sqrt{m}\,\|\mathbf{E}\|$. This holds by our parameter choices as $\|\mathbf{R}\| \leq B$ by admissibility of $\mathcal{A}$, $\|\mathbf{Z}_\mathbf{M}\| \leq O(m^{15/2}\sigma \log q \log N)$ and $\|\mathbf{R}_\mathbf{C}\mathbf{V}_N\| \leq O(\|\mathbf{D}\|\, m^4 \log q)$ by compactness of MCOM together with $\|\mathbf{R}_\mathbf{C}\| = 1$, and $\widetilde{\sigma} > \lambda^{\omega(1)} \cdot (B + m^{15/2} \cdot \sigma \cdot \log q \cdot \log N)$ by the choice made in the setup phase. Thus the two hybrids are statistically close. □

Lemma B.9. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_8^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_9^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. First, generation queries revert to sampling $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and returning $\mathbf{t} = \mathbf{B}\mathbf{y}$, with corruption queries returning the stored $\mathbf{y}$. This is statistically close by the pre-image distribution property of gadget trapdoors (Lemma A.6 and Lemma A.1), by exactly the argument given in Lemma B.4 read in the other direction, again with a hybrid over the at most $Q = Q(\lambda)$ generation queries. Under this change, the term $\mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_\mathbf{B}, [\mathbf{t}_{h_1} \mid \ldots \mid \mathbf{t}_{h_N}], \widetilde{\sigma})$ appearing in $\mathbf{Z}_0$ is replaced by the matrix $\mathbf{Y} = [\mathbf{y}_{h_1} \mid \ldots \mid \mathbf{y}_{h_N}]$ of the gaussians recorded for the handles in S, so that $\mathbf{Z}_0 = (\mathbf{Y} + \mathbf{R}) - \mathbf{Z}_\mathbf{M} - \mathbf{R}_\mathbf{C}\mathbf{V}_N$.

Second, after this change the trapdoor $\mathbf{T_B}$ is no longer used anywhere in the experiment: $\mathbf{Z}_0$ is given by the explicit algebraic expression above, and corruption queries return the stored $\mathbf{y}$ rather than sampling a pre-image. The setup phase may therefore revert to sampling $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times 2m^3}$ and $\mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times 2m^3}$ without a trapdoor, which is statistically close by Lemma A.4, by exactly the argument given in Lemma B.3 read in the other direction. Combining the two, the hybrids are statistically close. □

Deriving T. One aspect that we left out in the above proof is deriving $T = T(\lambda)$ from $\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$ to $\mathbf{Hyb}_9^{\mathcal{A}}(1^\lambda)$. Similar to [CHW25, WW25], we can set $T = 2T'$ where $T' = \mathsf{poly}(\lambda)$ such that for any adversary with advantage $\epsilon(\lambda) \geq 1/T'$ for infinitely many $\lambda \in \mathbb{N}$.

□

C Proof of Theorem 5.6

We define the following hybrids.

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the semi-static security experiment from Definition 5.2 with $b = 0$.

- **Setup Phase.** Initiate $\mathcal{K} := \emptyset$ and handle $h := 1$. Sample $\mathsf{fbe.crs} \leftarrow \mathsf{EqFBE.GSetup}(1^\lambda)$. Set $\mathsf{crs} := \mathsf{fbe.crs}$. Send crs to $\mathcal{A}$.
- **Key Generation Queries.** Sample $(\mathsf{fbe.pk}_h, \mathsf{fbe.sk}_h) \leftarrow \mathsf{EqFBE.KGen}$ and set $(\mathsf{pk}_h, \mathsf{sk}_h) := (\mathsf{fbe.pk}_h, \mathsf{fbe.sk}_h)$. Update $\mathcal{K} := \mathcal{K} \cup \{(h, \mathsf{pk}_h, \mathsf{sk}_h)\}$ and send (h, pk_h) to $\mathcal{A}$. Update $h := h + 1$.
- **Encryption Phase.** $\mathcal{A}$ sends $S \subseteq [h]$. Let $|S| = N$ and $(h_1, \ldots, h_N)$ be the lexicographical ordering of S. Set $\mathsf{pk}_i := \mathsf{pk}_{h_i}$ for each $i \in [N]$. Sample $\mathsf{fbe.ct} \leftarrow \mathsf{EqFBE.Enc}((\mathsf{pk}_i)_{i \in [N]}, 0)$. Send $\mathsf{ct} := \mathsf{fbe.ct}$ to $\mathcal{A}$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. In this hybrid, we will use $\mathsf{EqFBE.FakeSetup}$, $\mathsf{EqFBE.SimKGen}$, and $\mathsf{EqFBE.Equivocate}$ to encrypt the bit 0. The changes are highlighted in red.

- **Setup Phase.** Initiate $\mathcal{K} := \emptyset$ and handle $h := 1$. Sample $(\mathsf{fbe.}\widetilde{\mathsf{crs}}, \mathsf{fbe.}\widetilde{\mathsf{ct}}, \mathsf{fbe.aux}) \leftarrow \mathsf{EqFBE.FakeSetup}(1^\lambda, 0)$. Set $\mathsf{crs} := \mathsf{fbe.}\widetilde{\mathsf{crs}}$ and initiate $\mathsf{st} := (h, \mathcal{K})$. Send crs to $\mathcal{A}$.
- **Key Generation Queries.** Sample $(\mathsf{fbe.pk}_h, \mathsf{st}') \leftarrow \mathsf{EqFBE.SimKGen}(\mathsf{fbe.aux}, \mathsf{st})$, set $\mathsf{pk}_h := \mathsf{fbe.pk}_h$, and update $\mathsf{st} := \mathsf{st}'$. Send (h, pk_h) to $\mathcal{A}$. Update $h := h + 1$.
- **Encryption Phase.** $\mathcal{A}$ sends $S \subseteq [h]$. Sample $\mathsf{fbe.ct} \leftarrow \mathsf{EqFBE.Equivocate}(\mathsf{fbe.aux}, \mathsf{st}, S)$. Send $\mathsf{ct} := \mathsf{fbe.ct}$ to $\mathcal{A}$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ except that we encrypt the bit 1 using $\mathsf{EqFBE.FakeSetup}$. The changes are highlighted in red.

- **Setup Phase.** Initiate $\mathcal{K} := \emptyset$ and handle $h := 1$. Sample $(\mathsf{fbe.}\widetilde{\mathsf{crs}}, \mathsf{fbe.}\widetilde{\mathsf{ct}}, \mathsf{fbe.aux}) \leftarrow \mathsf{EqFBE.FakeSetup}(1^\lambda, 1)$. Set $\mathsf{crs} := \mathsf{fbe.}\widetilde{\mathsf{crs}}$ and initiate $\mathsf{st} := (h, \mathcal{K})$. Send crs to $\mathcal{A}$.
- **Key Generation Queries.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Encryption Phase.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. This is the semi-static security experiment from Definition 5.2 with $b = 1$. That is, this is the same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ except that the encryption phase samples $\mathsf{fbe.ct} \leftarrow \mathsf{EqFBE.Enc}((\mathsf{pk}_i)_{i\in[N]}, 1)$.

Lemma C.1. Assuming that EqFBE satisfies mode indistinguishability property, for any admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$, $\left|\Pr[1 \leftarrow \mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)]\right| \leq \mathsf{negl}(\lambda)$.

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $\left|\Pr[1 \leftarrow \mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)]\right| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. Then, we will construct an adversary $\mathcal{B}$ that can break the mode indistinguishability property of EqFBE with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ sends $\mu := 0$ to the challenger Chal in the EqFBE mode indistinguishability game and receives $\mathsf{fbe.crs}$. $\mathcal{B}$ sets $\mathsf{crs} := \mathsf{fbe.crs}$ and sends crs to $\mathcal{A}(1^\lambda)$. Whenever $\mathcal{A}$ makes a key generation query, $\mathcal{B}$ forwards it to Chal, receives $(h, \mathsf{fbe.pk}_h)$, and sends $(h, \mathsf{pk}_h := \mathsf{fbe.pk}_h)$ to $\mathcal{A}$. When $\mathcal{A}$ sends $S \subseteq [h]$, $\mathcal{B}$ forwards S to Chal to receive $\mathsf{fbe.ct}$ and passes it to $\mathcal{A}$ as ct. $\mathcal{B}$ outputs the guess of $\mathcal{A}$ as its guess.

If Chal's coin toss $b = 0$, then Chal answers using $\mathsf{EqFBE.GSetup}$, $\mathsf{EqFBE.KGen}$, $\mathsf{EqFBE.Enc}$ and $\mathcal{B}$ behaves like $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. If $b = 1$, then Chal answers using $\mathsf{EqFBE.FakeSetup}$, $\mathsf{EqFBE.SimKGen}$, $\mathsf{EqFBE.Equivocate}$ with $\mu = 0$ and $\mathcal{B}$ behaves like $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. Thus, advantage of $\mathcal{B}$ is also $\epsilon(\lambda)$. So, it must be the case that if EqFBE satisfies mode indistinguishability, any PPT adversary only has negligible advantage in distinguishing these hybrids. □

Lemma C.2. Assuming that EqFBE satisfies selective security for fake setup property, for any admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$, $\left|\Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)]\right| \leq \mathsf{negl}(\lambda)$.

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $\left|\Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)]\right| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. Then, we will construct an adversary $\mathcal{B}$ that can break the selective security for fake setup property of EqFBE with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ receives $(\mathsf{fbe.}\widetilde{\mathsf{crs}}, \mathsf{fbe.}\widetilde{\mathsf{ct}}, \mathsf{fbe.aux})$ from the challenger Chal in the EqFBE selective security for fake setup game. $\mathcal{B}$ initiates $\mathcal{K} := \emptyset, h := 1, \mathsf{st} := (h, \mathcal{K})$, sets $\mathsf{crs} := \mathsf{fbe.}\widetilde{\mathsf{crs}}$, and sends crs to $\mathcal{A}(1^\lambda)$. Whenever $\mathcal{A}$ makes a key generation query, $\mathcal{B}$ samples $(\mathsf{fbe.pk}_h, \mathsf{st}') \leftarrow \mathsf{EqFBE.SimKGen}(\mathsf{fbe.aux}, \mathsf{st})$, updates $\mathsf{st} := \mathsf{st}'$, sends $(h, \mathsf{pk}_h := \mathsf{fbe.pk}_h)$ to $\mathcal{A}$, and updates $h := h + 1$. When $\mathcal{A}$ sends $S \subseteq [h]$, $\mathcal{B}$ samples $\mathsf{fbe.ct} \leftarrow \mathsf{EqFBE.Equivocate}(\mathsf{fbe.aux}, \mathsf{st}, S)$ and sends it to $\mathcal{A}$ as ct. $\mathcal{B}$ outputs the guess of $\mathcal{A}$ as its guess.

Note that $\mathcal{B}$ needs only $\mathsf{fbe.aux}$ to run $\mathsf{EqFBE.SimKGen}$ and $\mathsf{EqFBE.Equivocate}$, and $\mathsf{fbe.aux}$ is given to it by Chal. If Chal encrypts $b = 0$, $\mathcal{B}$ behaves like $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ and if Chal encrypts $b = 1$,

$\mathcal{B}$ behaves like $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. Thus, advantage of $\mathcal{B}$ is also $\epsilon(\lambda)$. So, it must be the case that if EqFBE satisfies selective security for fake setup, any PPT adversary only has negligible advantage in distinguishing these hybrids. □

Lemma C.3. Assuming that EqFBE satisfies mode indistinguishability property, for any admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\, \lambda \in \mathbb{N}$, $\left|\Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)]\right| \leq \mathsf{negl}(\lambda)$.

Proof. This is similar to the proof of Lemma C.1, with $\mathcal{B}$ sending $\mu := 1$ to Chal instead of $\mu := 0$. In this case, Chal's coin toss $b = 1$ makes $\mathcal{B}$ behave like $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ and $b = 0$ makes $\mathcal{B}$ behave like $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. □

D Proofs from Section 6

Correctness. We have that $\mathsf{pk}_j = \mathbf{t}_j \in \mathbb{Z}_q^n$ and $\mathsf{sk}_j = \mathbf{r}_j \in \{\pm 1\}^{4m^5}$ for $j \in [N]$ where $\mathbf{t}_j = \mathbf{B}\mathbf{r}_j + \mathbf{p}$. By correctness of EqMCOM, we have that $\mathbf{CV}_N = \mathbf{M} - \mathbf{BZ}$ where $\mathbf{Z} = \mathsf{EqMCOM.Open}(\mathbf{M}, \mathsf{aux})$. So, $\mathbf{C}\mathbf{v}_i = \mathbf{t}_i - \mathbf{B}\mathbf{z}_i$.

We have that

$$\begin{aligned} c_i &= \mathbf{c}_2^\top \mathbf{v}_i + \mathbf{c}_1^\top(\mathbf{z}_i - \mathbf{r}_i) \\ &= \mathbf{s}^\top(\mathbf{C}\mathbf{v}_i + \mathbf{B}\mathbf{z}_i - \mathbf{B}\mathbf{r}_i) + \underbrace{\widetilde{\mathbf{e}}^\top(\mathbf{v}_i) + \mathbf{e}^\top \mathbf{z}_i - \mathbf{r}_i)}_{\widetilde{e}} \\ &= \mathbf{s}^\top(\mathbf{t}_i - \mathbf{B}\mathbf{r}_i) + \widetilde{e} \\ &= \mathbf{s}^\top \mathbf{p} + \widetilde{e} \end{aligned}$$

Thus, $y = \lfloor q/2 \rfloor \mu + \mathbf{e}^\top \mathbf{r}_{\mathbf{p}} - \widetilde{e}$ and $\mu' = \mu$ as long as $\left\|\mathbf{e}^\top \mathbf{r}_{\mathbf{p}} - \widetilde{e}\right\| < q/4$. We will now calculate the norm of $\mathbf{e}^\top \mathbf{r}_{\mathbf{p}} - \widetilde{e}$. Note that $\mathbf{e}^\top \mathbf{r}_{\mathbf{p}} - \widetilde{e} = \mathbf{e}^\top(\mathbf{r}_{\mathbf{p}} - \mathbf{z}_i + \mathbf{r}_i) + \widetilde{\mathbf{e}}^T \mathbf{v}_i$. By construction, $\|\mathbf{r}_{\mathbf{p}}\| = \|\mathbf{r}_i\| = 1$. With high probability, $\|\mathbf{D}\| \leq \sigma\sqrt{m}$ and $\|\mathbf{e}\| \leq \chi\sqrt{m}$. By compactness property of EqMCOM, we have $\|\mathbf{v}_i\| \leq \|\mathbf{V}_N\| = O(\sigma m^{9/2} \log q)$ and $\|\mathbf{z}_i\| \leq \|\mathbf{Z}\| \leq O(\widetilde{\sigma}\sqrt{m})$.

Putting all of this together, $|\mathbf{e}^\top \mathbf{r}_{\mathbf{p}} - \widetilde{e}| \leq O(\chi\widetilde{\chi}\sigma\widetilde{\sigma} m^{10} \log q \log N)$. As $\log N \leq \lambda$ (very loosely), as long as $q > O(\lambda\chi\widetilde{\chi}\sigma\widetilde{\sigma} m^{10} \log q)$, correctness holds.

Compactness. Size of ct is $(4m^5 + m + 1)\lceil \log q \rceil + \mathsf{poly}(\lambda, \log N) = O(m^5 \log q) + \mathsf{poly}(\lambda, \log N) = \mathsf{poly}(\lambda, \log N) \leq \mathsf{poly}(\lambda)$ by our parameters. Size of pk is $n\lceil \log q \rceil$ and sk is $4m^5$ both of which are polynomially bounded in λ. Size of crs is $(2m^2(nm + m^2) + n)\lceil \log q \rceil + \mathsf{poly}(\lambda)$. We need not include $\mathbf{B}$ in this calculation as it is efficiently derivable from $\mathbf{W}$ and $\mathbf{D}$. This is a polynomial in $n, \log q$ which is in turn a polynomial in λ. Moreover, since crs is public coin, we can replace it with a random oracle to get even more compact crs.

D.1 Proof of Theorem 6.2

We define the following hybrids for an admissible adversary $\mathcal{A}$. Throughout, $\mu \in \{0, 1\}$ denotes the message $\mathcal{A}$ sends in the setup phase.

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the mode indistinguishability game from Definition 5.4 with $b = 0$.

- **Setup Phase.** $\mathcal{A}$ sends $\mu \in \{0,1\}$. Initiate $\mathcal{K} := \emptyset$ and handle $h := 1$. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{EqMCOM.Setup}(1^n, 1^m, q, \sigma, 0)$ and $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{crs} := (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p})$. Send crs to $\mathcal{A}$.
- **Key Generation Queries.** Sample $\mathbf{r} \leftarrow \{\pm 1\}^{4m^5}$ and compute $\mathbf{t} := \mathbf{B}\mathbf{r} + \mathbf{p}$. Set $\mathsf{pk} := \mathbf{t}$ and $\mathsf{sk} := \mathbf{r}$. Send pk to $\mathcal{A}$. Update $\mathcal{K} := \mathcal{K} \cup \{(h, \mathsf{pk}, \mathsf{sk})\}$ and $h := h + 1$.
- **Encryption Phase.** $\mathcal{A}$ sends $S \subseteq [h]$. Let $|S| = N$. Retrieve from $\mathcal{K}$, $(\mathsf{pk}_{h^*})_{h^* \in S}$ and order them as $\mathbf{t}_1, \ldots, \mathbf{t}_N$ based on h^*. Then, do the following.
  1. Compute matrix $\mathbf{M} = [\mathbf{t}_1 \mid \ldots \mid \mathbf{t}_N] \in \mathbb{Z}_q^{n \times N}$.
  2. Sample $(\mathbf{C}, \mathsf{aux}) \leftarrow \mathsf{EqMCOM.Com}(\mathbf{M})$.
  3. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$.
  4. Compute $\mathbf{c}_1^\top := \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$, $\mathbf{c}_2^\top := \mathbf{s}^\top \mathbf{C} + \widetilde{\mathbf{e}}^\top$, and $c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu$.

  Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{aux})$ to $\mathcal{A}$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ except we sample the public key uniformly at random for every key generation query. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.
- **Key Generation Queries.** Sample $\mathbf{t} \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{pk} := \mathbf{t}$. Send pk to $\mathcal{A}$. Update $\mathcal{K} := \mathcal{K} \cup \{(h, \mathsf{pk}, \perp)\}$ and $h := h + 1$.
- **Encryption Phase.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ except we sample a wide discrete gaussian $\mathbf{y}$ and set the public key to $\mathbf{By}$ for every key generation query. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Key Generation Queries.** Sample $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and compute $\mathbf{t} := \mathbf{By}$. Set $\mathsf{pk} := \mathbf{t}$. Send pk to $\mathcal{A}$. Update $\mathcal{K} := \mathcal{K} \cup \{(h, \mathsf{pk}, \mathbf{y})\}$ and $h := h + 1$.
- **Encryption Phase.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ except we produce the commitment in equivocal mode during the encryption phase. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$.
- **Key Generation Queries.** Same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$.
- **Encryption Phase.** $\mathcal{A}$ sends $S \subseteq [h]$. Let $|S| = N$. Retrieve from $\mathcal{K}$, $(\mathsf{pk}_{h^*})_{h^* \in S}$ and order them as $\mathbf{t}_1, \ldots, \mathbf{t}_N$ based on h^*. Then, do the following.

  1. Compute $\mathbf{Y} = [\mathbf{y}_1 \mid \ldots \mid \mathbf{y}_N]$ and $\mathbf{M} = \mathbf{BY}$.
  2. Sample $\mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$, set $\mathbf{C} := \mathbf{BR_C}$, and sample $\mathsf{aux} \leftarrow \mathsf{EqMCOM.EqCom}(\mathbf{Y}, \mathbf{R_C})$.
  3. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$.
  4. Compute $\mathbf{c}_1^\top := \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$, $\mathbf{c}_2^\top := \mathbf{s}^\top \mathbf{C} + \widetilde{\mathbf{e}}^\top$, and $c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu$.

  Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{aux})$ to $\mathcal{A}$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ except we compute $\mathbf{c}_2^\top$ from $\mathbf{c}_1^\top$ rather than from $\mathbf{s}$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $\mu \in \{0,1\}$. Initiate $\mathcal{K} := \emptyset$ and handle $h := 1$. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{EqMCOM.Setup}(1^n, 1^m, q, \sigma, 0)$ and $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{crs} := (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p})$. Send crs to $\mathcal{A}$.
- **Key Generation Queries.** Sample $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and compute $\mathbf{t} := \mathbf{By}$. Set $\mathsf{pk} := \mathbf{t}$. Send pk to $\mathcal{A}$. Update $\mathcal{K} := \mathcal{K} \cup \{(h, \mathsf{pk}, \mathbf{y})\}$ and $h := h + 1$.
- **Encryption Phase.** $\mathcal{A}$ sends $S \subseteq [h]$. Let $|S| = N$. Retrieve from $\mathcal{K}$, $(\mathsf{pk}_{h^*})_{h^* \in S}$ and order them as $\mathbf{t}_1, \ldots, \mathbf{t}_N$ based on h^*. Let $\mathbf{y}_1, \ldots, \mathbf{y}_N$ be the gaussians recorded in $\mathcal{K}$ for the same handles, in the same order. Then, do the following.

  1. Compute $\mathbf{Y} = [\mathbf{y}_1 \mid \ldots \mid \mathbf{y}_N]$ and $\mathbf{M} = \mathbf{BY}$.
  2. Sample $\mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$, set $\mathbf{C} := \mathbf{BR_C}$, and sample $\mathsf{aux} \leftarrow \mathsf{EqMCOM.EqCom}(\mathbf{Y}, \mathbf{R_C})$.
  3. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$.
  4. Compute $\mathbf{c}_1^\top := \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$, $\mathbf{c}_2^\top := \mathbf{c}_1^\top \mathbf{R_C} + \widetilde{\mathbf{e}}^\top$, and $c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu$.

  Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{aux})$ to $\mathcal{A}$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$. This is the mode indistinguishability game from Definition 5.4 with $b = 1$.

Lemma D.1. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. This follows from the leftover hash lemma for structured $\mathbf{B}$ (Lemma A.3), which gives $\{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{Br})\} \approx_s \{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{t})\}$ for $\mathbf{r} \leftarrow \{\pm 1\}^{4m^5}, \mathbf{t} \leftarrow \mathbb{Z}_q^n$, and hence $\{\mathbf{Br} + \mathbf{p}\} \equiv \{\mathbf{t}\}$ for any fixed $\mathbf{p}$. As $\mathcal{A}$ is a PPT machine, it makes at most $Q = Q(\lambda)$ key generation queries, each answered with a freshly sampled $\mathbf{r}$, so a standard hybrid argument over these Q queries incurs a loss of at most $Q \cdot \mathsf{negl}(\lambda)$, which is negligible. □

Lemma D.2. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. This follows from the gaussian pre-image distribution (Lemma A.6), which gives $\{(\mathbf{B}, \mathbf{By})\} \approx_s \{(\mathbf{B}, \mathbf{t})\}$ for $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}, \mathbf{t} \leftarrow \mathbb{Z}_q^n$. As above, a standard hybrid argument over the at most $Q = Q(\lambda)$ key generation queries incurs only a negligible loss. Note that $\mathbf{y}$ is recorded but never revealed to $\mathcal{A}$, so the two hybrids differ only in how each $\mathbf{t}$ is sampled. □

Lemma D.3. Assuming that EqMCOM satisfies commitment indistinguishability (Definition 4.1), for any admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)]| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. We will construct an adversary $\mathcal{B}$ that can break the commitment indistinguishability property of EqMCOM with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ sends $(1^n, 1^m, q, \sigma, 0)$ to the challenger Chal in the EqMCOM commitment indistinguishability game and receives $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$. $\mathcal{B}$ samples $\mathbf{p} \leftarrow \mathbb{Z}_q^n$, sets $\mathsf{crs} := (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p})$, and sends crs to $\mathcal{A}(1^\lambda)$. $\mathcal{B}$ receives $\mu \in \{0, 1\}$ from $\mathcal{A}$.

Whenever $\mathcal{A}$ makes a key generation query, $\mathcal{B}$ makes a Gen query to its oracle $\mathcal{O}$ and receives $(\mathbf{t}, h)$. $\mathcal{B}$ sets $\mathsf{pk}_h := \mathbf{t}$ and sends it to $\mathcal{A}$. Recall that $\mathcal{O}$ samples $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$, sets $\mathbf{t} = \mathbf{By}$, and stores $\mathcal{Y}[h] := \mathbf{y}$. Thus $\mathcal{B}$ never learns $\mathbf{y}$ and the challenger retains it.

When $\mathcal{A}$ sends $S \subseteq [h]$, $\mathcal{B}$ sends $(S, \mathbf{R} := \mathbf{0} \in \mathbb{Z}_q^{4m^5 \times N})$ to Chal and receives $(\mathbf{C}^{(b)}, \mathsf{aux}^{(b)})$. $\mathcal{B}$ then samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$ and computes $\mathbf{c}_1^\top := \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$, $\mathbf{c}_2^\top := \mathbf{s}^\top \mathbf{C}^{(b)} + \widetilde{\mathbf{e}}^\top$, and $c_3 := \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu$. $\mathcal{B}$ sends $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{aux}^{(b)})$ to $\mathcal{A}$ and outputs the guess of $\mathcal{A}$ as its guess.

Note that $\mathcal{B}$ needs neither $\mathbf{Y}$ nor $\mathbf{R_C}$. It uses the commitment only through $\mathbf{C}^{(b)}$, and the opening information only through $\mathsf{aux}^{(b)}$, both of which Chal supplies. Moreover, $\mathcal{B}$ is admissible as $S \subseteq [h]$ and every $h^* \in S$ has an entry in $\mathcal{Y}$ because $\mathcal{A}$ is admissible, $\|\mathbf{R}\| = 0 \leq 0$ matches the bound $\mathcal{B}$ declared to Chal, and $\mathcal{B}$ makes no corruption queries at all.

If Chal's coin toss $b = 0$, then $(\mathbf{C}^{(0)}, \mathsf{aux}^{(0)}) \leftarrow \mathsf{EqMCOM.Com}(\mathbf{M})$ for $\mathbf{M} = \mathbf{B}(\mathbf{Y} + \mathbf{R}) = \mathbf{BY}$, which is exactly the matrix committed in $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$, and so $\mathcal{B}$ behaves like $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. If $b = 1$, then $\mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$, $\mathbf{C}^{(1)} = \mathbf{BR_C}$, and $\mathsf{aux}^{(1)} \leftarrow \mathsf{EqMCOM.EqCom}(\mathbf{Y} + \mathbf{R}, \mathbf{R_C}) = \mathsf{EqMCOM.EqCom}(\mathbf{Y}, \mathbf{R_C})$, and so $\mathcal{B}$ behaves like $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. Thus, advantage of $\mathcal{B}$ is also $\epsilon(\lambda)$. So, if EqMCOM satisfies commitment indistinguishability, then it must be that any PPT adversary's advantage in distinguishing these hybrids is negligible. □

Lemma D.4. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. This follows from the smudging lemma (Lemma 3.5). In $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$, $\mathbf{c}_2^\top = \mathbf{s}^\top \mathbf{B}\mathbf{R}_\mathbf{C} + \widetilde{\mathbf{e}}^\top$, whereas in $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$, $\mathbf{c}_2^\top = \mathbf{c}_1^\top \mathbf{R}_\mathbf{C} + \widetilde{\mathbf{e}}^\top = \mathbf{s}^\top \mathbf{B}\mathbf{R}_\mathbf{C} + \mathbf{e}^\top \mathbf{R}_\mathbf{C} + \widetilde{\mathbf{e}}^\top$. The two differ by the additive term $\mathbf{e}^\top \mathbf{R}_\mathbf{C}$, which has norm at most $O(\sqrt{m}\chi \cdot m^5)$ with high probability, and this is smudged by $\widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m$ by our choice of $\widetilde{\chi} > \lambda^{\omega(1)} m^{11/2} \chi$. □

Lemma D.5. For any adversary $\mathcal{A}$, $\Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)]$.

Proof. The changes between $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ and $\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$ are only semantic. In $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$, the setup phase samples $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{EqMCOM.Setup}(1^n, 1^m, q, \sigma, 0)$ and $\mathbf{p} \leftarrow \mathbb{Z}_q^n$, and the ciphertext carries $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$ and $c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r}_\mathbf{p} + \lfloor q/2 \rfloor \mu$; together with $\mathbf{R}_\mathbf{C} \leftarrow \{\pm 1\}^{4m^5 \times m}$ this is exactly the output of $\mathsf{FakeSetup}(1^\lambda, \mu)$, with $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, c_3)$ and $\mathsf{aux} := (\widetilde{\mathsf{ct}}, \mathbf{R}_\mathbf{C})$. Each key generation query samples $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and returns $\mathbf{B}\mathbf{y}$ while recording $\mathbf{y}$, which is exactly $\mathsf{SimKGen}(\mathsf{aux}, \mathsf{st})$. Finally, the encryption phase computes $\mathbf{Y}$ from the recorded gaussians, sets $\mathbf{C} := \mathbf{B}\mathbf{R}_\mathbf{C}$, samples $\mathsf{aux} \leftarrow \mathsf{EqMCOM.EqCom}(\mathbf{Y}, \mathbf{R}_\mathbf{C})$, and outputs $(\mathbf{c}_1^\top, \mathbf{c}_1^\top \mathbf{R}_\mathbf{C} + \widetilde{\mathbf{e}}^\top, c_3, \mathsf{aux})$, which is exactly $\mathsf{Equivocate}(\mathsf{aux}, \mathsf{st}, S)$. Thus the two hybrids are identically distributed. □

D.2 Proof of Theorem 6.3

We define the following hybrids. Throughout, $b \leftarrow \{0,1\}$ denotes the challenge bit sampled by the selective security for fake setup game (Definition 5.4).

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the selective security for fake setup game from Definition 5.4.

- **Setup Phase.** Perform these steps.
  1. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{EqMCOM.Setup}(1^n, 1^m, q, \sigma, 0)$. Recall that $\mathsf{pp} := (n, m, q, \sigma, 0, \widetilde{\sigma}, \mathcal{H})$ and $\mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}$.
  2. Sample $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. Set $\widetilde{\mathsf{crs}} := (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p})$.
  3. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R}_\mathbf{C} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$.
  4. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$ and $c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r}_\mathbf{p} + \lfloor q/2 \rfloor b$. Set $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, c_3)$.
  5. Set $\mathsf{aux} := (\widetilde{\mathsf{ct}}, \mathbf{R}_\mathbf{C})$.

  Send $(\widetilde{\mathsf{crs}}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ to $\mathcal{A}$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ except we set $\mathbf{p} = \mathbf{B}\mathbf{r}_\mathbf{p}$ by sampling $\mathbf{r}_\mathbf{p}$ early. The changes are highlighted in red.

- **Setup Phase.** Perform these steps.
  1. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{EqMCOM.Setup}(1^n, 1^m, q, \sigma, 0)$. Sample $\mathbf{R}_\mathbf{C} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$.

2. Set $\mathbf{p} := \mathbf{B}\mathbf{r}_{\mathbf{p}}$. Set $\widetilde{\mathsf{crs}} := (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p})$.
3. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}$.
4. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$ and $c_3 = \mathbf{c}_1^\top \mathbf{r}_{\mathbf{p}} + \lfloor q/2 \rfloor b$. Set $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, c_3)$.
5. Set $\mathsf{aux} := (\widetilde{\mathsf{ct}}, \mathbf{R}_{\mathbf{C}})$.

Send $(\widetilde{\mathsf{crs}}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ to $\mathcal{A}$.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ except we sample $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1\times 4m^5}$ instead of computing it as $\mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$. The changes are highlighted in red.

- **Setup Phase.** Perform these steps.

1. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{EqMCOM.Setup}(1^n, 1^m, q, \sigma, 0)$. Sample $\mathbf{R}_{\mathbf{C}} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}$.
2. Set $\mathbf{p} := \mathbf{B}\mathbf{r}_{\mathbf{p}}$. Set $\widetilde{\mathsf{crs}} := (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p})$.
3. Sample $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1\times 4m^5}$.
4. Compute $c_3 = \mathbf{c}_1^\top \mathbf{r}_{\mathbf{p}} + \lfloor q/2 \rfloor b$. Set $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, c_3)$.
5. Set $\mathsf{aux} := (\widetilde{\mathsf{ct}}, \mathbf{R}_{\mathbf{C}})$.

Send $(\widetilde{\mathsf{crs}}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ to $\mathcal{A}$.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ except we set $\widetilde{\mathsf{ct}} \leftarrow \mathbb{Z}_q^{1\times(4m^5+1)}$ and $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. In this hybrid, c_3 is a uniformly random element of $\mathbb{Z}_q$ and as such any dependence on b is eliminated.

Lemma D.6. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)]| \le \mathsf{negl}(\lambda)$.

Proof. This follows from the leftover hash lemma for structured $\mathbf{B}$ (Lemma A.3), which gives $\{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{B}\mathbf{r}_{\mathbf{p}}, \mathbf{e}^\top \mathbf{r}_{\mathbf{p}})\} \approx_s \{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathbf{e}^\top \mathbf{r}_{\mathbf{p}})\}$ for $\mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$. □

Lemma D.7. Assuming the hardness of $2m^2$-decomposed LWE assumption (Assumption 3.7), for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)]| \le \mathsf{negl}(\lambda)$.

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)]| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. We will construct an adversary $\mathcal{B}$ that can break the $2m^2$-decomposed LWE assumption with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ receives $(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{c}_1^\top)$ from the $2m^2$-decomposed LWE challenger Chal, where $\mathbf{W} \leftarrow \mathbb{Z}_q^{n\times 2m^3}$, $\mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times 2m^3}$, and $\mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}$. Note that this is exactly the distribution of $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ produced by $\mathsf{EqMCOM.Setup}(1^n, 1^m, q, \sigma, 0)$, and that the remaining components of pp, namely $\widetilde{\sigma}$ and $\mathcal{H}$, are sampled independently of $(\mathbf{W}, \mathbf{D}, \mathbf{B})$. Thus $\mathcal{B}$ samples $\widetilde{\sigma}$ and $\mathcal{H}$ exactly as $\mathsf{EqMCOM.Setup}(1^n, 1^m, q, \sigma, 0)$ does and sets $\mathsf{pp} := (n, m, q, \sigma, 0, \widetilde{\sigma}, \mathcal{H}, \mathbf{W}, \mathbf{D}, \mathbf{B})$.

$\mathcal{B}$ then samples $b \leftarrow \{0,1\}, \mathbf{R}_{\mathbf{C}} \leftarrow \{\pm 1\}^{4m^5\times m}, \mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}$, and sets $\mathbf{p} := \mathbf{B}\mathbf{r}_{\mathbf{p}}$ and $\widetilde{\mathsf{crs}} := (\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p})$. Using the challenge $\mathbf{c}_1^\top$ it received from Chal, $\mathcal{B}$ computes $c_3 := \mathbf{c}_1^\top \mathbf{r}_{\mathbf{p}} + \lfloor q/2 \rfloor b$

and sets $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, c_3)$ and $\mathsf{aux} := (\widetilde{\mathsf{ct}}, \mathbf{R}_\mathbf{C})$. $\mathcal{B}$ sends $(\widetilde{\mathsf{crs}}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ to $\mathcal{A}(1^\lambda)$ and outputs 1 if $\mathcal{A}$ outputs b, and 0 otherwise.

If Chal's coin toss is 0, then $\mathbf{c}_1^\top = \mathbf{s}^\top\mathbf{B} + \mathbf{e}^\top$ for $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}$ and every value $\mathcal{B}$ hands to $\mathcal{A}$ is distributed exactly as in $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. If Chal's coin toss is 1, then $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1\times 4m^5}$ and $\mathcal{B}$ behaves exactly like $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. Thus, $\mathcal{B}$ has an advantage of $\epsilon(\lambda)$ in the $2m^2$-decomposed LWE game. So, if the $2m^2$-decomposed LWE assumption is hard, then it must be that any PPT adversary's advantage in distinguishing these hybrids is negligible. □

Lemma D.8. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)]| \le \mathsf{negl}(\lambda)$.

Proof. This follows from the leftover hash lemma for structured $\mathbf{B}$ (Lemma A.3). Specifically, the distributions

$$\left\{ \begin{bmatrix}\mathbf{B}\\ \mathbf{c}_1^\top\end{bmatrix}, \begin{bmatrix}\mathbf{B}\\ \mathbf{c}_1^\top\end{bmatrix}\mathbf{r}_\mathbf{p} \right\} \approx_s \left\{ \begin{bmatrix}\mathbf{B}\\ \mathbf{c}_1^\top\end{bmatrix}, \begin{bmatrix}\mathbf{p}\\ c_3\end{bmatrix} \right\}$$

where $\mathbf{W} \leftarrow \mathbb{Z}_q^{n\times 2m^3}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times 2m^3}, \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}$, $\mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$, $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1\times 4m^5}, \mathbf{p} \leftarrow \mathbb{Z}_q^n, c_3 \leftarrow \mathbb{Z}_q$. Note that $\mathbf{r}_\mathbf{p}$ is used nowhere else in $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. In particular, $\mathbf{R}_\mathbf{C}$, the only other component of aux, is sampled independently of $\mathbf{r}_\mathbf{p}$. Hence the masking of $\lfloor q/2 \rfloor b$ by c_3 holds even given aux. □

E Proof of Theorem 7.4

We define the following hybrids for an admissible adversary $\mathcal{A}$.

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the adaptive security experiment from Definition 7.1 with $b = 0$.

- **Setup Phase.** $\mathcal{A}$ sends 1^κ. Sample $(\mathsf{be.mpk}, \mathsf{be.msk}) \leftarrow \mathsf{EqBE.Setup}(1^\lambda, 1^\kappa)$. Set $\mathsf{mpk} := \mathsf{be.mpk}$ and $\mathsf{msk} := \mathsf{be.msk}$. Send mpk to $\mathcal{A}$.
- **Secret Key Queries.** $\mathcal{A}$ sends $\mathsf{id} \in \{0,1\}^\kappa$. Sample $\mathsf{be.sk_{id}} \leftarrow \mathsf{EqBE.KGen}(\mathsf{be.mpk}, \mathsf{be.msk}, \mathsf{id})$. Send $\mathsf{sk_{id}} := \mathsf{be.sk_{id}}$ to $\mathcal{A}$.
- **Challenge Encryption Phase.** $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$. Sample $\mathsf{be.ct} \leftarrow \mathsf{EqBE.Enc}(\mathsf{be.mpk}, S, 0)$. Send $\mathsf{ct} := \mathsf{be.ct}$ to $\mathcal{A}$.
- **Secret Key Queries.** Handle these similarly to the pre-challenge secret key queries.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. In this hybrid, we will use $\mathsf{EqBE.FakeSetup}$, $\mathsf{EqBE.SimKGen}$, and $\mathsf{EqBE.Equivocate}$ to encrypt the bit 0. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends 1^κ. Sample a state st and $(\mathsf{be.}\widetilde{\mathsf{mpk}}, \mathsf{be.}\widetilde{\mathsf{ct}}, \mathsf{be.aux}) \leftarrow \mathsf{EqBE.FakeSetup}(1^\lambda, 1^\kappa, 0)$. Set $\mathsf{mpk} := \mathsf{be.}\widetilde{\mathsf{mpk}}$. Send mpk to $\mathcal{A}$.
- **Secret Key Queries.** $\mathcal{A}$ sends $\mathsf{id} \in \{0,1\}^\kappa$. Sample $(\mathsf{be.sk_{id}}, \mathsf{st}') \leftarrow \mathsf{EqBE.SimKGen}(\mathsf{be.aux}, \mathsf{st}, \mathsf{id})$ and update $\mathsf{st} := \mathsf{st}'$. Send $\mathsf{sk_{id}} := \mathsf{be.sk_{id}}$ to $\mathcal{A}$.

- **Challenge Encryption Phase.** $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$. Sample $(\mathsf{be.ct}, \mathsf{st}') \leftarrow \mathsf{EqBE.Equivocate}(\mathsf{be.aux}, \mathsf{st}, S)$ and update $\mathsf{st} := \mathsf{st}'$. Send $\mathsf{ct} := \mathsf{be.ct}$ to $\mathcal{A}$.
- **Secret Key Queries.** Handle these similarly to the pre-challenge secret key queries.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ except that we encrypt the bit 1 using $\mathsf{EqBE.FakeSetup}$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends 1^κ. Sample a state st and $(\mathsf{be.}\widetilde{\mathsf{mpk}}, \mathsf{be.}\widetilde{\mathsf{ct}}, \mathsf{be.aux}) \leftarrow \mathsf{EqBE.FakeSetup}(1^\lambda, 1^\kappa, 1)$. Set $\mathsf{mpk} := \mathsf{be.}\widetilde{\mathsf{mpk}}$. Send mpk to $\mathcal{A}$.
- **Secret Key Queries.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Challenge Encryption Phase.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Secret Key Queries.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. This is the adaptive security experiment from Definition 7.1 with $b = 1$. That is, this is the same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ except that the challenge encryption phase samples $\mathsf{be.ct} \leftarrow \mathsf{EqBE.Enc}(\mathsf{be.mpk}, S, 1)$.

Lemma E.1. Assuming that EqBE satisfies mode indistinguishability property, for any admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$, $\left|\Pr[1 \leftarrow \mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)]\right| \leq \mathsf{negl}(\lambda)$.

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $\left|\Pr[1 \leftarrow \mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)]\right| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. Then, we will construct an adversary $\mathcal{B}$ that can break the mode indistinguishability property of EqBE with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ interacts with $\mathcal{A}(1^\lambda)$ and receives 1^κ from $\mathcal{A}$. $\mathcal{B}$ sends $(1^\kappa, \mu := 0)$ to the challenger Chal in the EqBE mode indistinguishability game and receives $\mathsf{be.mpk}$. $\mathcal{B}$ sets $\mathsf{mpk} := \mathsf{be.mpk}$ and sends it to $\mathcal{A}$. Whenever $\mathcal{A}$ queries a secret key for $\mathsf{id} \in \{0,1\}^\kappa$, $\mathcal{B}$ forwards id to Chal, receives $\mathsf{be.sk}_{\mathsf{id}}$, and sends $\mathsf{sk}_{\mathsf{id}} := \mathsf{be.sk}_{\mathsf{id}}$ to $\mathcal{A}$. When $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$, $\mathcal{B}$ forwards S to Chal to receive $\mathsf{be.ct}$ and passes it to $\mathcal{A}$ as ct. Post-challenge secret key queries are answered in the same way as the pre-challenge ones. $\mathcal{B}$ outputs the guess of $\mathcal{A}$ as its guess.

As $\mathcal{A}$ is admissible, it never queries a secret key for any $\mathsf{id}^* \in S$, and hence Chal never aborts. If Chal's coin toss $b = 0$, then Chal answers using $\mathsf{EqBE.Setup}, \mathsf{EqBE.KGen}, \mathsf{EqBE.Enc}$ and $\mathcal{B}$ behaves like $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. If $b = 1$, then Chal answers using $\mathsf{EqBE.FakeSetup}, \mathsf{EqBE.SimKGen}, \mathsf{EqBE.Equivocate}$ with $\mu = 0$ and $\mathcal{B}$ behaves like $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. Thus, advantage of $\mathcal{B}$ is also $\epsilon(\lambda)$. So, it must be the case that if EqBE satisfies mode indistinguishability, any PPT adversary only has negligible advantage in distinguishing these hybrids. □

Lemma E.2. Assuming that EqBE satisfies semantic security property, for any admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in \mathbb{N}$, $\left|\Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)]\right| \leq \mathsf{negl}(\lambda)$.

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $\left|\Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)]\right| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. Then, we will construct an adversary $\mathcal{B}$ that can break the semantic security property of EqBE with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ interacts with $\mathcal{A}(1^\lambda)$ and receives 1^κ from $\mathcal{A}$. $\mathcal{B}$ forwards 1^κ to the challenger Chal in the EqBE semantic security game and receives $\mathsf{be.}\widetilde{\mathsf{mpk}}$. $\mathcal{B}$ sets $\mathsf{mpk} := \mathsf{be.}\widetilde{\mathsf{mpk}}$ and sends it to $\mathcal{A}$. Whenever $\mathcal{A}$ queries a secret key for $\mathsf{id} \in \{0,1\}^\kappa$, $\mathcal{B}$ queries its own $\mathsf{SimKGen}(\mathsf{be.aux}, \mathsf{st}, \cdot)$ oracle on id, receives $\mathsf{be.sk_{id}}$, and sends $\mathsf{sk_{id}} := \mathsf{be.sk_{id}}$ to $\mathcal{A}$. When $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$, $\mathcal{B}$ forwards S to Chal to receive $\mathsf{be.ct}$ and passes it to $\mathcal{A}$ as ct. Post-challenge secret key queries are answered using the same oracle. $\mathcal{B}$ outputs the guess of $\mathcal{A}$ as its guess.

Note that $\mathcal{B}$ is never given $\mathsf{be.aux}$ and does not need it as every invocation of $\mathsf{EqBE.SimKGen}$ and $\mathsf{EqBE.Equivocate}$ is performed by Chal on $\mathcal{B}$'s behalf, and Chal maintains the state st across these invocations exactly as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ and $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ do. Moreover, $\mathcal{B}$ is admissible in the EqBE semantic security game because $\mathcal{A}$ is admissible in the IBBE adaptive security game, so $\mathcal{B}$ never queries the $\mathsf{SimKGen}$ oracle with any $\mathsf{id}^* \in S$. If Chal samples $\mu = 0$, $\mathcal{B}$ behaves like $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ and if Chal samples $\mu = 1$, $\mathcal{B}$ behaves like $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. Thus, advantage of $\mathcal{B}$ is also $\epsilon(\lambda)$. So, it must be the case that if EqBE satisfies semantic security, any PPT adversary only has negligible advantage in distinguishing these hybrids. □

Lemma E.3. Assuming that EqBE satisfies mode indistinguishability property, for any admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\, \lambda \in \mathbb{N}$, $\left|\Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)]\right| \leq \mathsf{negl}(\lambda)$.

Proof. This is similar to the proof of Lemma E.1, with $\mathcal{B}$ sending $(1^\kappa, \mu := 1)$ to Chal instead of $(1^\kappa, \mu := 0)$. In this case, Chal's coin toss $b = 1$ makes $\mathcal{B}$ behave like $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ and $b = 0$ makes $\mathcal{B}$ behave like $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. □

F Proofs from Section 8

F.1 Proof of Theorem 8.6

We define the following hybrids for an admissible adversary $\mathcal{A}$. Without loss of generality we assume that $\mathcal{A}$ queries $\mathsf{id} \in \{0,1\}^\kappa$ at most once. Throughout, $\mu \in \{0,1\}$ denotes the message $\mathcal{A}$ sends in the setup phase.

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the mode indistinguishability game from Definition 7.2 with $b = 0$.

- **Setup Phase.** $\mathcal{A}$ sends $(1^\kappa, \mu \in \{0,1\})$. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{T_B}) \leftarrow \mathsf{TEqMCOM.}\widetilde{\mathsf{Setup}}(1^n, 1^m, q, \sigma, 1)$ and $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{mpk} := (\kappa, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}})$ and $\mathsf{msk} := \mathbf{T_B}$. Send mpk to $\mathcal{A}$.
  Here, $\mathcal{H}_{\mathsf{pk}}$ is a handle for random oracle queries. Initiate dictionary $\mathsf{H_{pk}} := \emptyset$.
- **Random Oracle Queries.** Given $\mathsf{id} \in \{0,1\}^\kappa$, if $\mathsf{H_{pk}}$ contains an entry for id, then retrieve and return $\mathsf{H_{pk}}[\mathsf{id}]$. Otherwise, sample $\mathbf{t_{id}} \leftarrow \mathbb{Z}_q^n$ and set $\mathsf{H_{pk}}[\mathsf{id}] = \mathbf{t_{id}}$. Send $\mathbf{t_{id}}$ to $\mathcal{A}$.
- **Secret Key Queries.** $\mathcal{A}$ sends $\mathsf{id} \in \{0,1\}^\kappa$. Compute $\mathbf{t_{id}} = \mathcal{H}_{\mathsf{pk}}(\mathsf{id})$. Sample $\mathbf{y_{id}} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, \mathbf{t_{id}} - \mathbf{p}, \widetilde{\sigma})$. Send $\mathsf{sk_{id}} := (\mathsf{id}, \mathbf{y_{id}})$ to $\mathcal{A}$.

- **Challenge Encryption Query.** $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$. Let $L = |S|$ and $\mathsf{id}_1, \ldots, \mathsf{id}_L$ be the lexicographical ordering of S. Do the following.

  1. Compute matrix $\mathbf{M} = [\mathbf{t}_{\mathsf{id}_1} \mid \ldots \mid \mathbf{t}_{\mathsf{id}_L}]$ where $\mathbf{t}_{\mathsf{id}_i} = \mathcal{H}_{\mathsf{pk}}(\mathsf{id}_i)$ for each $i \in [L]$.
  2. Sample $(\mathbf{C}, \mathsf{aux}) \leftarrow \mathsf{TEqMCOM.Com}(\mathbf{M})$.
  3. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m, \mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}$.
  4. Compute $\mathbf{c}_1^\top := \mathbf{s}^\top\mathbf{B} + \mathbf{e}^\top$, $\mathbf{c}_2^\top := \mathbf{s}^\top\mathbf{C} + \widetilde{\mathbf{e}}^\top$, and $c_3 = \mathbf{s}^\top\mathbf{p} + \mathbf{e}^\top\mathbf{r}_{\mathbf{p}} + \lfloor q/2 \rfloor \mu$.

  Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{aux})$ to $\mathcal{A}$.

- **Secret Key and Random Oracle Queries.** Handle these similarly to the pre-challenge queries. Recall that an admissible $\mathcal{A}$ never queries a secret key for any $\mathsf{id}^* \in S$.

- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ except that we write $\mathbf{t}_{\mathsf{id}}$ as $\mathbf{t}'_{\mathsf{id}} + \mathbf{p}$ for a uniformly sampled $\mathbf{t}'_{\mathsf{id}}$. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.

- **Random Oracle Queries.** Given $\mathsf{id} \in \{0,1\}^\kappa$, if H_{pk} contains an entry for id, then retrieve and return $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}]$. Otherwise, sample $\mathbf{t}'_{\mathsf{id}} \leftarrow \mathbb{Z}_q^n$ and set $\mathbf{t}_{\mathsf{id}} := \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$, and set $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}] = \mathbf{t}_{\mathsf{id}}$. Send $\mathbf{t}_{\mathsf{id}}$ to $\mathcal{A}$.

- **Secret Key Queries.** $\mathcal{A}$ sends $\mathsf{id} \in \{0,1\}^\kappa$. Sample $\mathbf{y}_{\mathsf{id}} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, \mathbf{t}'_{\mathsf{id}}, \widetilde{\sigma})$. Send $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}_{\mathsf{id}})$ to $\mathcal{A}$.

- **Challenge Encryption Query.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.

- **Secret Key and Random Oracle Queries.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.

- **Guess Phase.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ except that $\mathbf{t}'_{\mathsf{id}}$ is generated as $\mathbf{B}\mathbf{y}'_{\mathsf{id}}$ for a gaussian $\mathbf{y}'_{\mathsf{id}}$, which is then handed out as the secret key. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. In addition, initiate an empty dictionary $\mathcal{K} := \emptyset$.

- **Random Oracle Queries.** Given $\mathsf{id} \in \{0,1\}^\kappa$, if H_{pk} contains an entry for id, then retrieve and return $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}]$. Otherwise, sample $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and compute $\mathbf{t}'_{\mathsf{id}} := \mathbf{B}\mathbf{y}'_{\mathsf{id}}$, set $\mathbf{t}_{\mathsf{id}} := \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$, and set $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}] = \mathbf{t}_{\mathsf{id}}$ and $\mathcal{K}[\mathsf{id}] = \mathbf{y}'_{\mathsf{id}}$. Send $\mathbf{t}_{\mathsf{id}}$ to $\mathcal{A}$.

- **Secret Key Queries.** $\mathcal{A}$ sends $\mathsf{id} \in \{0,1\}^\kappa$. Retrieve $\mathbf{y}'_{\mathsf{id}} = \mathcal{K}[\mathsf{id}]$, computing $\mathcal{H}_{\mathsf{pk}}(\mathsf{id})$ first if $\mathsf{id} \notin \mathsf{H}_{\mathsf{pk}}$. Send $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$ to $\mathcal{A}$.

- **Challenge Encryption Query.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.

- **Secret Key and Random Oracle Queries.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.

- **Guess Phase.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ except that the setup phase samples $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ without a trapdoor. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $(1^\kappa, \mu \in \{0,1\})$. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{TEqMCOM.Setup}(1^n, 1^m, q, \sigma, 1)$ and $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{mpk} := (\kappa, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}})$. The experiment no longer holds a trapdoor, and correspondingly no longer sets msk. Send mpk to $\mathcal{A}$.

  Here, $\mathcal{H}_{\mathsf{pk}}$ is a handle for random oracle queries. Initiate dictionaries $\mathsf{H}_{\mathsf{pk}} := \emptyset$ and $\mathcal{K} := \emptyset$.

- **Random Oracle Queries.** Given $\mathsf{id} \in \{0,1\}^\kappa$, if H_{pk} contains an entry for id, then retrieve and return $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}]$. Otherwise, sample $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$, compute $\mathbf{t}'_{\mathsf{id}} := \mathbf{B}\mathbf{y}'_{\mathsf{id}}$ and $\mathbf{t}_{\mathsf{id}} := \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$, and set $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}] = \mathbf{t}_{\mathsf{id}}$ and $\mathcal{K}[\mathsf{id}] = \mathbf{y}'_{\mathsf{id}}$. Send $\mathbf{t}_{\mathsf{id}}$ to $\mathcal{A}$.

- **Secret Key Queries.** $\mathcal{A}$ sends $\mathsf{id} \in \{0,1\}^\kappa$. Retrieve $\mathbf{y}'_{\mathsf{id}} = \mathcal{K}[\mathsf{id}]$, computing $\mathcal{H}_{\mathsf{pk}}(\mathsf{id})$ first if $\mathsf{id} \notin \mathsf{H}_{\mathsf{pk}}$. Send $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$ to $\mathcal{A}$.

- **Challenge Encryption Query.** $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$. Let $L = |S|$ and $\mathsf{id}_1, \ldots, \mathsf{id}_L$ be the lexicographical ordering of S. Do the following.

  1. Compute matrix $\mathbf{M} = [\mathbf{t}_{\mathsf{id}_1} \mid \ldots \mid \mathbf{t}_{\mathsf{id}_L}]$ where $\mathbf{t}_{\mathsf{id}_i} = \mathcal{H}_{\mathsf{pk}}(\mathsf{id}_i)$ for each $i \in [L]$.
  2. Sample $(\mathbf{C}, \mathsf{aux}) \leftarrow \mathsf{TEqMCOM.Com}(\mathbf{M})$.
  3. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m, \mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}$.
  4. Compute $\mathbf{c}_1^\top := \mathbf{s}^\top\mathbf{B} + \mathbf{e}^\top$, $\mathbf{c}_2^\top := \mathbf{s}^\top\mathbf{C} + \widetilde{\mathbf{e}}^\top$, and $c_3 = \mathbf{s}^\top\mathbf{p} + \mathbf{e}^\top\mathbf{r}_{\mathbf{p}} + \lfloor q/2 \rfloor \mu$.

  Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{aux})$ to $\mathcal{A}$.

- **Secret Key and Random Oracle Queries.** Handle these similarly to the pre-challenge queries. Recall that an admissible $\mathcal{A}$ never queries a secret key for any $\mathsf{id}^* \in S$.

- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ except that $\mathbf{p}$ is set to $\mathbf{B}\mathbf{r}_{\mathbf{p}}$ during setup. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $(1^\kappa, \mu \in \{0,1\})$. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{TEqMCOM.Setup}(1^n, 1^m, q, \sigma, 1)$. Sample $\mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}$ and set $\mathbf{p} := \mathbf{B}\mathbf{r}_{\mathbf{p}}$. Set $\mathsf{mpk} := (\kappa, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}})$. Send mpk to $\mathcal{A}$. Initiate dictionaries $\mathsf{H}_{\mathsf{pk}} := \emptyset$ and $\mathcal{K} := \emptyset$.

- **Random Oracle Queries.** Same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$.

- **Secret Key Queries.** Same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$.

- **Challenge Encryption Query.** Same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$, except that step 3 no longer samples $\mathbf{r}_{\mathbf{p}}$, which is now fixed during setup.

- **Secret Key and Random Oracle Queries.** Same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$.

- **Guess Phase.** Same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$.

Note that c_3 is now $\mathbf{s}^\top\mathbf{p} + \mathbf{e}^\top\mathbf{r}_{\mathbf{p}} + \lfloor q/2 \rfloor \mu = (\mathbf{s}^\top\mathbf{B} + \mathbf{e}^\top)\mathbf{r}_{\mathbf{p}} + \lfloor q/2 \rfloor \mu = \mathbf{c}_1^\top\mathbf{r}_{\mathbf{p}} + \lfloor q/2 \rfloor \mu$, exactly as $\mathsf{Equivocate}$ computes it. This is an equality, not an approximation.

$\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ except that the commitment is produced in equivocal mode. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$.
- **Random Oracle Queries.** Same as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$.
- **Secret Key Queries.** Same as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$.
- **Challenge Encryption Query.** Same as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$, except that steps 1 and 2 are replaced by the following.
  1. Compute $\mathbf{Y} = [\mathbf{y}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{y}'_{\mathsf{id}_L}]$, retrieving $\mathbf{y}'_{\mathsf{id}_i} = \mathcal{K}[\mathsf{id}_i]$ and computing $\mathcal{H}_{\mathsf{pk}}(\mathsf{id}_i)$ first where needed.
  2. Sample $\mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$, set $\mathbf{C} := \mathbf{B}\mathbf{R_C}$, and sample $\mathsf{aux} \leftarrow \mathsf{TEqMCOM.EqCom}(\mathbf{Y} + \mathbf{1}_L^\top \otimes \mathbf{r_p}, \mathbf{R_C})$.
- **Secret Key and Random Oracle Queries.** Same as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Same as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$. This is the mode indistinguishability game from Definition 7.2 with $b = 1$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $(1^\kappa, \mu \in \{0,1\})$. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{TEqMCOM.Setup}(1^n, 1^m, q, \sigma, 1)$. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}, \mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$ and compute $\mathbf{p} := \mathbf{B}\mathbf{r_p}$ and $\mathbf{c}_1^\top := \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$. Set $\widetilde{\mathsf{ct}} := \mathbf{c}_1^\top$. Set $\mathsf{mpk} := (\kappa, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}})$ and $\mathsf{aux} := (\mu, \widetilde{\mathsf{ct}}, \mathsf{mpk}, \mathbf{r_p}, \mathbf{R_C})$. Send mpk to $\mathcal{A}$. Initiate dictionaries $\mathsf{H}_{\mathsf{pk}} := \emptyset$ and $\mathcal{K} := \emptyset$.
- **Random Oracle Queries.** Given $\mathsf{id} \in \{0,1\}^\kappa$, if H_{pk} contains an entry for id, then retrieve and return $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}]$. Otherwise, sample $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$, compute $\mathbf{t}'_{\mathsf{id}} := \mathbf{B}\mathbf{y}'_{\mathsf{id}}$ and $\mathbf{t}_{\mathsf{id}} := \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$, and set $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}] = \mathbf{t}_{\mathsf{id}}$ and $\mathcal{K}[\mathsf{id}] = \mathbf{y}'_{\mathsf{id}}$. Send $\mathbf{t}_{\mathsf{id}}$ to $\mathcal{A}$.
- **Secret Key Queries.** $\mathcal{A}$ sends $\mathsf{id} \in \{0,1\}^\kappa$. Retrieve $\mathbf{y}'_{\mathsf{id}} = \mathcal{K}[\mathsf{id}]$, computing $\mathcal{H}_{\mathsf{pk}}(\mathsf{id})$ first if $\mathsf{id} \notin \mathsf{H}_{\mathsf{pk}}$. Send $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$ to $\mathcal{A}$.
- **Challenge Encryption Query.** $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$. Let $L = |S|$ and $\mathsf{id}_1, \ldots, \mathsf{id}_L$ be the lexicographical ordering of S. Do the following.
  1. Compute $\mathbf{Y} = [\mathbf{y}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{y}'_{\mathsf{id}_L}]$, retrieving $\mathbf{y}'_{\mathsf{id}_i} = \mathcal{K}[\mathsf{id}_i]$ and computing $\mathcal{H}_{\mathsf{pk}}(\mathsf{id}_i)$ first where needed.
  2. Sample $\mathbf{R_C}$-based commitment $\mathbf{C} := \mathbf{B}\mathbf{R_C}$ and $\mathsf{aux} \leftarrow \mathsf{TEqMCOM.EqCom}(\mathbf{Y} + \mathbf{1}_L^\top \otimes \mathbf{r_p}, \mathbf{R_C})$.
  3. Sample $\widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m$.
  4. Compute $\mathbf{c}_2^\top := \mathbf{c}_1^\top \mathbf{R_C} + \widetilde{\mathbf{e}}^\top$ and $c_3 := \mathbf{c}_1^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu$.

  Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{aux})$ to $\mathcal{A}$.

- **Secret Key and Random Oracle Queries.** Handle these similarly to the pre-challenge queries. Recall that an admissible $\mathcal{A}$ never queries a secret key for any $\mathsf{id}^* \in S$.

- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

Lemma F.1. $\Pr[1 \leftarrow \mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)]$.

Proof. Sampling $\mathbf{t}_{\mathsf{id}} \leftarrow \mathbb{Z}_q^n$ and sampling $\mathbf{t}'_{\mathsf{id}} \leftarrow \mathbb{Z}_q^n$ and setting $\mathbf{t}_{\mathsf{id}} := \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$ induce the same distribution on $\mathbf{t}_{\mathsf{id}}$, for any fixed $\mathbf{p}$. Moreover, the secret key query in $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ samples a pre-image of $\mathbf{t}_{\mathsf{id}} - \mathbf{p}$, which equals $\mathbf{t}'_{\mathsf{id}}$, so the two hybrids sample $\mathbf{y}_{\mathsf{id}}$ from the same distribution as well. The change is therefore purely notational and the two hybrids are identically distributed. □

Lemma F.2. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ in the order in which $\mathbf{t}'_{\mathsf{id}}$ and the secret key are sampled. In $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ we sample $\mathbf{t}'_{\mathsf{id}} \leftarrow \mathbb{Z}_q^n$ and set $\mathbf{y}_{\mathsf{id}} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, \mathbf{t}'_{\mathsf{id}}, \widetilde{\sigma})$. In $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ we sample $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and set $\mathbf{t}'_{\mathsf{id}} := \mathbf{B}\mathbf{y}'_{\mathsf{id}}$. By the pre-image distribution property of gadget trapdoors (Lemma A.6 and Lemma A.1),

$$\{(\mathbf{B}, \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, \mathbf{t}'_{\mathsf{id}}, \widetilde{\sigma}), \mathbf{t}'_{\mathsf{id}}) : \mathbf{t}'_{\mathsf{id}} \leftarrow \mathbb{Z}_q^n\} \approx_s \{(\mathbf{B}, \mathbf{y}'_{\mathsf{id}}, \mathbf{B}\mathbf{y}'_{\mathsf{id}}) : \mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}\}$$

so the pair $(\mathbf{t}_{\mathsf{id}}, \mathbf{y}_{\mathsf{id}})$ that $\mathcal{A}$ observes for any single identity is distributed almost identically in the two hybrids. As $\mathcal{A}$ is a PPT machine it makes at most $Q = Q(\lambda)$ random oracle queries, each answered with independently sampled randomness, so a standard hybrid argument over these Q identities incurs a loss of at most $Q \cdot \mathsf{negl}(\lambda)$, which remains negligible. □

Lemma F.3. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ only in how $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$ is sampled in the setup phase. By the setup indistinguishability property of TEqMCOM (Definition 8.1), the output of $\mathsf{TEqMCOM.}\widetilde{\mathsf{Setup}}(1^n, 1^m, q, \sigma, 1)$ restricted to $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$ is statistically close to the output of $\mathsf{TEqMCOM.Setup}(1^n, 1^m, q, \sigma, 1)$. The point of $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ is that $\mathbf{T}_{\mathbf{B}}$ is no longer used anywhere in the experiment. Secret key queries are answered with the $\mathbf{y}'_{\mathsf{id}}$ recorded during random oracle queries, so SamplePre is never invoked. Dropping the trapdoor is therefore well defined, and the rest of $\mathcal{A}$'s view is unaffected. □

Lemma F.4. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ only in how $\mathbf{p}$ is sampled. This follows from the leftover hash lemma for structured $\mathbf{B}$ (Lemma A.3), which gives $\{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{B}\mathbf{r}_{\mathbf{p}}, \mathbf{e}^\top \mathbf{r}_{\mathbf{p}})\} \approx_s \{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathbf{e}^\top \mathbf{r}_{\mathbf{p}})\}$ for $\mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}$ and $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. Note that in $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ the vector $\mathbf{r}_{\mathbf{p}}$ is sampled fresh during the challenge encryption query and is used only in c_3, whereas in $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ the same $\mathbf{r}_{\mathbf{p}}$ is fixed during setup and used both to define $\mathbf{p}$ and in c_3. Leftover hash lemma is applied to this single $\mathbf{r}_{\mathbf{p}}$, and every other value in $\mathcal{A}$'s view is independent of it. □

Lemma F.5. Assuming that TEqMCOM satisfies commitment indistinguishability (Definition 8.1), for any admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)]| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. We will construct an adversary $\mathcal{B}$ that can break the commitment indistinguishability property of $\mathsf{TEqMCOM}$ with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ sends $(1^n, 1^m, q, \sigma, 1)$ to the challenger Chal in the $\mathsf{TEqMCOM}$ commitment indistinguishability game and receives $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$. Note that Chal samples these through $\mathsf{TEqMCOM.Setup}$, without a trapdoor, which is exactly how $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ samples them. $\mathcal{B}$ interacts with $\mathcal{A}(1^\lambda)$ and receives $(1^\kappa, \mu)$. $\mathcal{B}$ samples $\mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$, sets $\mathbf{p} := \mathbf{B}\mathbf{r_p}$ and $\mathsf{mpk} := (\kappa, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}})$, and sends mpk to $\mathcal{A}$. $\mathcal{B}$ initiates an empty dictionary mapping identities to handles.

Whenever $\mathcal{A}$ queries the random oracle on a fresh $\mathsf{id} \in \{0,1\}^\kappa$, $\mathcal{B}$ makes a Gen query to its oracle $\mathcal{O}$ and receives $(\mathbf{t}'_{\mathsf{id}}, h_{\mathsf{id}})$. $\mathcal{B}$ records the handle h_{id} for id, sets $\mathbf{t}_{\mathsf{id}} := \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$, and returns $\mathbf{t}_{\mathsf{id}}$ to $\mathcal{A}$. Recall that $\mathcal{O}$ samples $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and sets $\mathbf{t}'_{\mathsf{id}} = \mathbf{B}\mathbf{y}'_{\mathsf{id}}$, so this matches the random oracle phase of $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ exactly, and $\mathcal{B}$ never learns $\mathbf{y}'_{\mathsf{id}}$. Whenever $\mathcal{A}$ queries a secret key for id, $\mathcal{B}$ makes a $(\mathsf{Corr}, h_{\mathsf{id}})$ query to $\mathcal{O}$ (first making a Gen query if id is fresh), receives $\mathbf{y}'_{\mathsf{id}}$, and returns $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$.

When $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$ with $L = |S|$ and lexicographical ordering $\mathsf{id}_1, \ldots, \mathsf{id}_L$, $\mathcal{B}$ sends $(S_h, \mathbf{R} := \mathbf{1}_L^\top \otimes \mathbf{r_p})$ to Chal, where $S_h = \{h_{\mathsf{id}_1}, \ldots, h_{\mathsf{id}_L}\}$ is the set of recorded handles, and receives $(\mathbf{C}^{(b)}, \mathsf{aux}^{(b)})$. $\mathcal{B}$ then samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m$, computes $\mathbf{c}_1^\top := \mathbf{s}^\top\mathbf{B} + \mathbf{e}^\top$, $\mathbf{c}_2^\top := \mathbf{s}^\top\mathbf{C}^{(b)} + \widetilde{\mathbf{e}}^\top$, and $c_3 := \mathbf{s}^\top\mathbf{p} + \mathbf{e}^\top\mathbf{r_p} + \lfloor q/2 \rfloor \mu$, and sends $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{aux}^{(b)})$ to $\mathcal{A}$. Post-challenge random oracle and secret key queries are answered exactly as before. $\mathcal{B}$ outputs the guess of $\mathcal{A}$ as its guess.

Observe that $\mathcal{B}$ needs neither $\mathbf{Y}$ nor $\mathbf{R_C}$. It uses the commitment only through $\mathbf{C}^{(b)}$ and the opening information only through $\mathsf{aux}^{(b)}$, both supplied by Chal. Moreover, $\mathcal{B}$ is admissible: $S_h \subseteq [h]$ and every handle in S_h still has an entry in $\mathcal{Y}$, because an admissible $\mathcal{A}$ never queries a secret key for any $\mathsf{id}^* \in S$ and therefore $\mathcal{B}$ never corrupts the corresponding handles, either before or after the challenge. Moreover, $\|\mathbf{R}\| = \left\|\mathbf{1}_L^\top \otimes \mathbf{r_p}\right\| = 1$, which meets the bound 1 that $\mathcal{B}$ declared to Chal.

If Chal's coin toss $b = 0$, then $(\mathbf{C}^{(0)}, \mathsf{aux}^{(0)}) \leftarrow \mathsf{TEqMCOM.Com}(\mathbf{M})$ for

$$\mathbf{M} = \mathbf{B}(\mathbf{Y} + \mathbf{R}) = \mathbf{B}\mathbf{Y} + \mathbf{B}\left(\mathbf{1}_L^\top \otimes \mathbf{r_p}\right) = [\mathbf{t}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{t}'_{\mathsf{id}_L}] + \mathbf{1}_L^\top \otimes \mathbf{p} = [\mathbf{t}_{\mathsf{id}_1} \mid \ldots \mid \mathbf{t}_{\mathsf{id}_L}]$$

which is exactly the matrix committed in $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$, and so $\mathcal{B}$ behaves like $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$. If $b = 1$, then $\mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$, $\mathbf{C}^{(1)} = \mathbf{B}\mathbf{R_C}$, and $\mathsf{aux}^{(1)} \leftarrow \mathsf{TEqMCOM.EqCom}(\mathbf{Y} + \mathbf{R}, \mathbf{R_C}) = \mathsf{TEqMCOM.EqCom}(\mathbf{Y} + \mathbf{1}_L^\top \otimes \mathbf{r_p}, \mathbf{R_C})$, and so $\mathcal{B}$ behaves like $\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$. Thus, advantage of $\mathcal{B}$ is also $\epsilon(\lambda)$. So, if $\mathsf{TEqMCOM}$ satisfies commitment indistinguishability, then it must be that any PPT adversary's advantage in distinguishing these hybrids is negligible. □

Lemma F.6. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ only in how $\mathbf{c}_2^\top$ is computed. In $\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$ we have $\mathbf{c}_2^\top = \mathbf{s}^\top\mathbf{C} + \widetilde{\mathbf{e}}^\top = \mathbf{s}^\top\mathbf{B}\mathbf{R_C} + \widetilde{\mathbf{e}}^\top$, whereas in $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$ we have $\mathbf{c}_2^\top = \mathbf{c}_1^\top\mathbf{R_C} + \widetilde{\mathbf{e}}^\top = \mathbf{s}^\top\mathbf{B}\mathbf{R_C} + \mathbf{e}^\top\mathbf{R_C} + \widetilde{\mathbf{e}}^\top$. The two differ by the additive term $\mathbf{e}^\top\mathbf{R_C}$, which has norm at most $O(\sqrt{m}\chi \cdot m^5)$ with high probability since $\|\mathbf{e}\| \leq \sqrt{m}\chi$ and $\mathbf{R_C} \in \{\pm 1\}^{4m^5 \times m}$, and this is smudged by $\widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m$ by our choice of $\widetilde{\chi} > \lambda^{\omega(1)} m^{11/2}\chi$ (Lemma 3.5). All remaining differences are semantic, as detailed after $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$. □

F.2 Proof of Theorem 8.7

We define the following hybrids for an admissible adversary $\mathcal{A}$. Without loss of generality we assume that $\mathcal{A}$ queries $\mathsf{id} \in \{0,1\}^\kappa$ at most once. Throughout, $\mu \leftarrow \{0,1\}$ denotes the message sampled by the semantic security game (Definition 7.2).

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the semantic security experiment from Definition 7.2.

- **Fake Setup Phase.** $\mathcal{A}$ sends 1^κ. Do the following.

  1. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{TEqMCOM.Setup}(1^n, 1^m, q, \sigma, 1)$.
  2. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}, \mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$. Compute $\mathbf{p} := \mathbf{B}\mathbf{r_p}$.
  3. Initiate dictionaries $\mathsf{H_{pk}} := \emptyset, \mathcal{K} := \emptyset$.
  4. Compute $\mathbf{c}_1^\top := \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$. Set $\widetilde{\mathsf{ct}} := \mathbf{c}_1^\top$, $\widetilde{\mathsf{mpk}} := (\kappa, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}})$, and $\mathsf{aux} := (\mu, \widetilde{\mathsf{ct}}, \widetilde{\mathsf{mpk}}, \mathbf{r_p}, \mathbf{R_C})$.

  Here, $\mathcal{H}_{\mathsf{pk}}$ is a handle for random oracle queries. Send $\widetilde{\mathsf{mpk}}$ to $\mathcal{A}$. Note that aux is retained by the experiment and is never given to $\mathcal{A}$.

- **Key Generation Simulation.** $\mathcal{A}$ sends $\mathsf{id} \in \{0,1\}^\kappa$. Do the following.

  1. If $\mathsf{id} \in \mathsf{H_{pk}}$, retrieve $\mathbf{y}'_{\mathsf{id}}$ from $\mathcal{K}[\mathsf{id}]$. Set $\mathsf{sk_{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$.
  2. Otherwise, sample $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\tilde{\sigma}}^{4m^5}$, compute $\mathbf{t}'_{\mathsf{id}} = \mathbf{B}\mathbf{y}'_{\mathsf{id}}$ and $\mathbf{t}_{\mathsf{id}} = \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$. Set $\mathsf{sk_{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$, $\mathsf{H_{pk}}[\mathsf{id}] = \mathbf{t}_{\mathsf{id}}$, and $\mathcal{K}[\mathsf{id}] = \mathbf{y}'_{\mathsf{id}}$.

  Send $\mathsf{sk_{id}}$ to $\mathcal{A}$.

- **Equivocation Phase.** $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$. Let $L = |S|$ and $\mathsf{id}_1, \ldots, \mathsf{id}_L$ be the lexicographical ordering of S. Do the following.

  1. For each $i \in [L]$, if $\mathsf{id}_i \in \mathsf{H_{pk}}$, retrieve $\mathbf{t}_{\mathsf{id}_i} = \mathsf{H_{pk}}[\mathsf{id}_i]$, set $\mathbf{t}'_{\mathsf{id}_i} = \mathbf{t}_{\mathsf{id}_i} - \mathbf{p}$, and retrieve $\mathbf{y}'_{\mathsf{id}_i} = \mathcal{K}[\mathsf{id}_i]$. Otherwise, sample $\mathbf{y}'_{\mathsf{id}_i} \leftarrow \mathcal{D}_{\mathbb{Z},\tilde{\sigma}}^{4m^5}$, compute $\mathbf{t}'_{\mathsf{id}_i} = \mathbf{B}\mathbf{y}'_{\mathsf{id}_i}$ and $\mathbf{t}_{\mathsf{id}_i} = \mathbf{t}'_{\mathsf{id}_i} + \mathbf{p}$, and set $\mathsf{H_{pk}}[\mathsf{id}_i] = \mathbf{t}_{\mathsf{id}_i}$ and $\mathcal{K}[\mathsf{id}_i] = \mathbf{y}'_{\mathsf{id}_i}$.
  2. Compute $\mathbf{Y} = [\mathbf{y}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{y}'_{\mathsf{id}_L}]$ and $\mathbf{C} = \mathbf{B}\mathbf{R_C}$.
  3. Sample $\mathsf{aux}' \leftarrow \mathsf{TEqMCOM.EqCom}(\mathbf{Y} + \mathbf{1}_L^\top \otimes \mathbf{r_p}, \mathbf{R_C})$.
  4. Sample $\widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m$.

  Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_1^\top \mathbf{R_C} + \widetilde{\mathbf{e}}^\top, \mathbf{c}_1^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu, \mathsf{aux}')$ to $\mathcal{A}$.

- **Key Generation Simulation.** Proceed exactly like the key generation simulation phase before. Recall that an admissible $\mathcal{A}$ never queries $\mathsf{id}^* \in S$, either before or after the equivocation phase.

- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ except that we sample $\mathbf{c}_1^\top$ uniformly at random. The changes are highlighted in red.

- **Fake Setup Phase.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$, except that steps 2 and 4 are replaced by the following.

  2. Sample $\mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}, \mathbf{R_C} \leftarrow \{\pm 1\}^{4m^5 \times m}$; we no longer sample $\mathbf{s}$ or $\mathbf{e}$. Compute $\mathbf{p} := \mathbf{B}\mathbf{r_p}$.
  4. Sample $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1 \times 4m^5}$. Set $\widetilde{\mathsf{ct}} := \mathbf{c}_1^\top$, $\widetilde{\mathsf{mpk}} := (\kappa, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}})$, and $\mathsf{aux} := (\mu, \widetilde{\mathsf{ct}}, \widetilde{\mathsf{mpk}}, \mathbf{r_p}, \mathbf{R_C})$.

- **Key Generation Simulation.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.
- **Equivocation Phase.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.
- **Key Generation Simulation.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ except that the fake setup phase samples $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ together with a trapdoor. The changes are highlighted in red.

- **Fake Setup Phase.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$, except that step 1 is replaced by the following.

  1. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{T_B}) \leftarrow \widetilde{\mathsf{TEqMCOM.Setup}}(1^n, 1^m, q, \sigma, 1)$.

- **Key Generation Simulation.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Equivocation Phase.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Key Generation Simulation.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ except that we sample $\mathbf{t}'_{\mathsf{id}}$ uniformly and obtain $\mathbf{y}'_{\mathsf{id}}$ by pre-image sampling. The changes are highlighted in red.

- **Fake Setup Phase.** Same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$.
- **Key Generation Simulation.** $\mathcal{A}$ sends $\mathsf{id} \in \{0,1\}^\kappa$. Do the following.

  1. If $\mathsf{id} \in \mathsf{H}_{\mathsf{pk}}$, retrieve $\mathbf{y}'_{\mathsf{id}}$ from $\mathcal{K}[\mathsf{id}]$. Set $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$.
  2. Otherwise, sample $\mathbf{t}'_{\mathsf{id}} \leftarrow \mathbb{Z}_q^n$, set $\mathbf{t}_{\mathsf{id}} = \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$, and sample $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, \mathbf{t}'_{\mathsf{id}}, \widetilde{\sigma})$. Set $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$, $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}] = \mathbf{t}_{\mathsf{id}}$, and $\mathcal{K}[\mathsf{id}] = \mathbf{y}'_{\mathsf{id}}$.

  Send $\mathsf{sk}_{\mathsf{id}}$ to $\mathcal{A}$.

- **Equivocation Phase.** Same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$, except that steps 1 and 2 are replaced by the following.

1. For each $i \in [L]$, if $\mathsf{id}_i \in \mathsf{H}_{\mathsf{pk}}$, retrieve $\mathbf{t}_{\mathsf{id}_i} = \mathsf{H}_{\mathsf{pk}}[\mathsf{id}_i]$ and set $\mathbf{t}'_{\mathsf{id}_i} = \mathbf{t}_{\mathsf{id}_i} - \mathbf{p}$. Otherwise, sample $\mathbf{t}'_{\mathsf{id}_i} \leftarrow \mathbb{Z}_q^n$ and set $\mathbf{t}_{\mathsf{id}_i} = \mathbf{t}'_{\mathsf{id}_i} + \mathbf{p}$ and $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}_i] = \mathbf{t}_{\mathsf{id}_i}$.
2. Sample $\mathbf{Y} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, [\mathbf{t}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{t}'_{\mathsf{id}_L}], \widetilde{\sigma})$. Compute $\mathbf{C} = \mathbf{B}\mathbf{R}_{\mathbf{C}}$.

- **Key Generation Simulation.** Same as above.
- **Guess Phase.** Same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$**.** This is the same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ except that the pre-image target used during equivocation is shifted by $\mathbf{1}_L^\top \otimes \mathbf{p}$. The changes are highlighted in red.

- **Fake Setup Phase.** Same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$.
- **Key Generation Simulation.** Same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$.
- **Equivocation Phase.** Same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$, except that steps 2 and 3 are replaced by the following.

  2. Sample $\mathbf{Y}' \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, [\mathbf{t}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{t}'_{\mathsf{id}_L}] + \mathbf{1}_L^\top \otimes \mathbf{p}, \widetilde{\sigma})$. Compute $\mathbf{C} = \mathbf{B}\mathbf{R}_{\mathbf{C}}$.
  3. Sample $\mathsf{aux}' \leftarrow \mathsf{TEqMCOM.EqCom}(\mathbf{Y}', \mathbf{R}_{\mathbf{C}})$.

- **Key Generation Simulation.** Same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$.
- **Guess Phase.** Same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$**.** This is the same as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ except that $\mathbf{p}$ and c_3 are sampled uniformly and $\mathbf{r}_{\mathbf{p}}$ is no longer sampled at all. The changes are highlighted in red.

- **Fake Setup Phase.** $\mathcal{A}$ sends 1^κ. Do the following.

  1. Sample $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{T}_{\mathbf{B}}) \leftarrow \mathsf{TEqMCOM}.\widetilde{\mathsf{Setup}}(1^n, 1^m, q, \sigma, 1)$.
  2. Sample $\mathbf{R}_{\mathbf{C}} \leftarrow \{\pm 1\}^{4m^5 \times m}$ and $\mathbf{p} \leftarrow \mathbb{Z}_q^n$; we no longer sample $\mathbf{r}_{\mathbf{p}}$.
  3. Initiate dictionaries $\mathsf{H}_{\mathsf{pk}} := \emptyset, \mathcal{K} := \emptyset$.
  4. Sample $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1 \times 4m^5}$. Set $\widetilde{\mathsf{ct}} := \mathbf{c}_1^\top$, $\widetilde{\mathsf{mpk}} := (\kappa, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}})$, and $\mathsf{aux} := (\mu, \widetilde{\mathsf{ct}}, \widetilde{\mathsf{mpk}}, \mathbf{R}_{\mathbf{C}})$.

  Here, $\mathcal{H}_{\mathsf{pk}}$ is a handle for random oracle queries. Send $\widetilde{\mathsf{mpk}}$ to $\mathcal{A}$. Note that aux is retained by the experiment and is never given to $\mathcal{A}$.

- **Key Generation Simulation.** $\mathcal{A}$ sends $\mathsf{id} \in \{0,1\}^\kappa$. Do the following.

  1. If $\mathsf{id} \in \mathsf{H}_{\mathsf{pk}}$, retrieve $\mathbf{y}'_{\mathsf{id}}$ from $\mathcal{K}[\mathsf{id}]$. Set $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$.
  2. Otherwise, sample $\mathbf{t}'_{\mathsf{id}} \leftarrow \mathbb{Z}_q^n$, set $\mathbf{t}_{\mathsf{id}} = \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$, and sample $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, \mathbf{t}'_{\mathsf{id}}, \widetilde{\sigma})$. Set $\mathsf{sk}_{\mathsf{id}} := (\mathsf{id}, \mathbf{y}'_{\mathsf{id}})$, $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}] = \mathbf{t}_{\mathsf{id}}$, and $\mathcal{K}[\mathsf{id}] = \mathbf{y}'_{\mathsf{id}}$.

  Send $\mathsf{sk}_{\mathsf{id}}$ to $\mathcal{A}$.

- **Equivocation Phase.** $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$. Let $L = |S|$ and $\mathsf{id}_1, \ldots, \mathsf{id}_L$ be the lexicographical ordering of S. Do the following.

  1. For each $i \in [L]$, if $\mathsf{id}_i \in \mathsf{H}_{\mathsf{pk}}$, retrieve $\mathbf{t}_{\mathsf{id}_i} = \mathsf{H}_{\mathsf{pk}}[\mathsf{id}_i]$ and set $\mathbf{t}'_{\mathsf{id}_i} = \mathbf{t}_{\mathsf{id}_i} - \mathbf{p}$. Otherwise, sample $\mathbf{t}'_{\mathsf{id}_i} \leftarrow \mathbb{Z}_q^n$, set $\mathbf{t}_{\mathsf{id}_i} = \mathbf{t}'_{\mathsf{id}_i} + \mathbf{p}$, and set $\mathsf{H}_{\mathsf{pk}}[\mathsf{id}_i] = \mathbf{t}_{\mathsf{id}_i}$.
  2. Sample $\mathbf{Y}' \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T}_{\mathbf{B}}, [\mathbf{t}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{t}'_{\mathsf{id}_L}] + \mathbf{1}_L^\top \otimes \mathbf{p}, \widetilde{\sigma})$. Compute $\mathbf{C} = \mathbf{B}\mathbf{R}_{\mathbf{C}}$.
  3. Sample $\mathsf{aux}' \leftarrow \mathsf{TEqMCOM.EqCom}(\mathbf{Y}', \mathbf{R}_{\mathbf{C}})$.
  4. Sample $\widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m$ and $c_3 \leftarrow \mathbb{Z}_q$.

  Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_1^\top \mathbf{R}_{\mathbf{C}} + \widetilde{\mathbf{e}}^\top, c_3, \mathsf{aux}')$ to $\mathcal{A}$.

- **Key Generation Simulation.** Proceed exactly like the key generation simulation phase before. Recall that an admissible $\mathcal{A}$ never queries $\mathsf{id}^* \in S$, either before or after the equivocation phase.

- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

Lemma F.7. Assuming the hardness of the $2m^2$-decomposed LWE assumption (Assumption 3.7), for any admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)]| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. We will construct an adversary $\mathcal{B}$ that can break the $2m^2$-decomposed LWE assumption with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ receives $(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{c}_1^\top)$ from the $2m^2$-decomposed LWE challenger Chal, where $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times 2m^3}$, $\mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times 2m^3}$, and $\mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + \mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}$. This is exactly the distribution of $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ produced by $\mathsf{TEqMCOM.Setup}(1^n, 1^m, q, \sigma, 1)$, and the remaining components of pp are sampled independently of $(\mathbf{W}, \mathbf{D}, \mathbf{B})$. $\mathcal{B}$ samples them exactly as $\mathsf{TEqMCOM.Setup}$ does and assembles pp.

$\mathcal{B}$ interacts with $\mathcal{A}(1^\lambda)$ and receives 1^κ. $\mathcal{B}$ samples $\mu \leftarrow \{0,1\}$, $\mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}$, and $\mathbf{R}_{\mathbf{C}} \leftarrow \{\pm 1\}^{4m^5 \times m}$, computes $\mathbf{p} := \mathbf{B}\mathbf{r}_{\mathbf{p}}$, initiates $\mathsf{H}_{\mathsf{pk}} := \emptyset, \mathcal{K} := \emptyset$, and sets $\widetilde{\mathsf{mpk}} := (\kappa, \mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{p}, \mathcal{H}_{\mathsf{pk}})$. $\mathcal{B}$ sends $\widetilde{\mathsf{mpk}}$ to $\mathcal{A}$.

Whenever $\mathcal{A}$ queries a key for $\mathsf{id} \in \{0,1\}^\kappa$, $\mathcal{B}$ answers exactly as the key generation simulation phase does. If $\mathsf{id} \in \mathsf{H}_{\mathsf{pk}}$ it returns the recorded $\mathbf{y}'_{\mathsf{id}}$, and otherwise it samples $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$, sets $\mathbf{t}'_{\mathsf{id}} = \mathbf{B}\mathbf{y}'_{\mathsf{id}}$ and $\mathbf{t}_{\mathsf{id}} = \mathbf{t}'_{\mathsf{id}} + \mathbf{p}$, records both, and returns $\mathbf{y}'_{\mathsf{id}}$. When $\mathcal{A}$ sends $S \subseteq \{0,1\}^\kappa$ with $L = |S|$, $\mathcal{B}$ completes any missing $\mathbf{y}'_{\mathsf{id}_i}$ in the same way, forms $\mathbf{Y} = [\mathbf{y}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{y}'_{\mathsf{id}_L}]$, samples $\mathsf{aux}' \leftarrow \mathsf{TEqMCOM.EqCom}(\mathbf{Y} + \mathbf{1}_L^\top \otimes \mathbf{r}_{\mathbf{p}}, \mathbf{R}_{\mathbf{C}})$ and $\widetilde{\mathbf{e}} \leftarrow [-\sqrt{m}\widetilde{\chi}, \sqrt{m}\widetilde{\chi}]^m$, and sends

$$\mathsf{ct} := (\mathbf{c}_1^\top,\ \mathbf{c}_1^\top \mathbf{R}_{\mathbf{C}} + \widetilde{\mathbf{e}}^\top,\ \mathbf{c}_1^\top \mathbf{r}_{\mathbf{p}} + \lfloor q/2 \rfloor \mu,\ \mathsf{aux}')$$

to $\mathcal{A}$, using the challenge $\mathbf{c}_1^\top$ it received from Chal. Post-challenge key queries are answered as before. $\mathcal{B}$ outputs 1 if $\mathcal{A}$ outputs μ, and 0 otherwise.

Observe that $\mathcal{B}$ never needs $\mathbf{s}$, $\mathbf{e}$, or a trapdoor for $\mathbf{B}$. Every value it hands to $\mathcal{A}$ is either sampled by $\mathcal{B}$ itself or is one of $\mathbf{B}$ and $\mathbf{c}_1^\top$, both supplied by Chal. If Chal's coin toss is 0, then $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top$ for $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}$, and $\mathcal{B}$ behaves exactly like $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. If Chal's coin toss is 1, then $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1 \times 4m^5}$ and $\mathcal{B}$ behaves exactly like $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. Thus, advantage of $\mathcal{B}$ is also $\epsilon(\lambda)$. So, if the $2m^2$-decomposed LWE assumption is hard, then it must be that any PPT adversary's advantage in distinguishing these hybrids is negligible. □

Lemma F.8. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ only in how $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$ is sampled in the fake setup phase. By the setup indistinguishability property of $\mathsf{TEqMCOM}$ (Definition 8.1), the output of $\mathsf{TEqMCOM}.\widetilde{\mathsf{Setup}}(1^n, 1^m, q, \sigma, 1)$ restricted to $(\mathsf{pp}, \mathbf{W}, \mathbf{D}, \mathbf{B})$ is statistically close to the output of $\mathsf{TEqMCOM}.\mathsf{Setup}(1^n, 1^m, q, \sigma, 1)$. As $\mathbf{T_B}$ is retained by the experiment and never given to $\mathcal{A}$, and as no other step of $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ uses it, the rest of $\mathcal{A}$'s view is unaffected. Thus the two hybrids are statistically close. □

Lemma F.9. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ in the order in which $\mathbf{t}'_{\mathsf{id}}$ and $\mathbf{y}'_{\mathsf{id}}$ are sampled. In $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ we sample $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}$ and set $\mathbf{t}'_{\mathsf{id}} = \mathbf{B}\mathbf{y}'_{\mathsf{id}}$. In $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ we sample $\mathbf{t}'_{\mathsf{id}} \leftarrow \mathbb{Z}_q^n$ and set $\mathbf{y}'_{\mathsf{id}} \leftarrow \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, \mathbf{t}'_{\mathsf{id}}, \widetilde{\sigma})$. By the pre-image distribution property of gadget trapdoors (Lemma A.6 and Lemma A.1),

$$\{(\mathbf{B}, \mathbf{y}'_{\mathsf{id}}, \mathbf{B}\mathbf{y}'_{\mathsf{id}}) : \mathbf{y}'_{\mathsf{id}} \leftarrow \mathcal{D}_{\mathbb{Z},\widetilde{\sigma}}^{4m^5}\} \approx_s \{(\mathbf{B}, \mathsf{SamplePre}(\mathbf{B}, \mathbf{T_B}, \mathbf{t}'_{\mathsf{id}}, \widetilde{\sigma}), \mathbf{t}'_{\mathsf{id}}) : \mathbf{t}'_{\mathsf{id}} \leftarrow \mathbb{Z}_q^n\}$$

so the pair $(\mathbf{t}_{\mathsf{id}}, \mathbf{y}'_{\mathsf{id}})$ that $\mathcal{A}$ observes for any single identity is distributed almost identically in the two hybrids. The same argument applies column-wise to the matrix $\mathbf{Y}$ sampled during equivocation. As $\mathcal{A}$ is a PPT machine it makes at most $Q = Q(\lambda)$ key generation simulation queries, and together with the at most L identities in S a standard hybrid argument over these incurs a loss of at most $(Q + L) \cdot \mathsf{negl}(\lambda)$, which remains negligible. □

Lemma F.10. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. The two hybrids differ only in the equivocation phase. Since $\mathbf{p} = \mathbf{B}\mathbf{r_p}$, we have $\mathbf{1}_L^\top \otimes \mathbf{p} = \mathbf{B}(\mathbf{1}_L^\top \otimes \mathbf{r_p})$, so the pre-image target in $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ is

$$[\mathbf{t}_{\mathsf{id}_1} \mid \ldots \mid \mathbf{t}_{\mathsf{id}_L}] + \mathbf{1}_L^\top \otimes \mathbf{p} = \mathbf{B}[\mathbf{y}'_{\mathsf{id}_1} \mid \ldots \mid \mathbf{y}'_{\mathsf{id}_L}] + \mathbf{B}\left(\mathbf{1}_L^\top \otimes \mathbf{r_p}\right)$$

By the gaussian pre-image smudging lemma (Lemma A.7 and Lemma A.2), sampling a pre-image of $\mathbf{y} + \mathbf{B}\mathbf{z}$ is statistically close to sampling a pre-image of $\mathbf{y}$ and adding $\mathbf{z}$, provided $\widetilde{\sigma} > \lambda^{\omega(1)}\sqrt{m}\,\|\mathbf{z}\|$. Here $\mathbf{z} = \mathbf{1}_L^\top \otimes \mathbf{r_p}$ has $\|\mathbf{z}\| = 1$ since $\mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$, so the condition holds comfortably by our choice of $\widetilde{\sigma}$. Hence $\mathbf{Y}'$ in $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ is statistically close to $\mathbf{Y} + \mathbf{1}_L^\top \otimes \mathbf{r_p}$ in $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$, and consequently $\mathsf{TEqMCOM.EqCom}(\mathbf{Y}', \mathbf{R_C})$ is statistically close to $\mathsf{TEqMCOM.EqCom}(\mathbf{Y} + \mathbf{1}_L^\top \otimes \mathbf{r_p}, \mathbf{R_C})$. Every other value in $\mathcal{A}$'s view is unchanged, so the two hybrids are statistically close. □

Lemma F.11. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. By the remark following $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$, the only occurrences of $\mathbf{r_p}$ in $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ are $\mathbf{p} = \mathbf{B}\mathbf{r_p}$ in the fake setup phase and $c_3 = \mathbf{c}_1^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu$ in the ciphertext. In particular $\mathbf{R_C}$ is sampled independently of $\mathbf{r_p}$, and aux is never given to $\mathcal{A}$. Recall also that $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1 \times 4m^5}$ is uniform

from $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ onwards. Applying the leftover hash lemma for structured $\mathbf{B}$ with a trapdoor (Lemma A.5), we have

$$\left\{ \begin{bmatrix} \mathbf{B} \\ \mathbf{c}_1^\top \end{bmatrix}, \begin{bmatrix} \mathbf{B} \\ \mathbf{c}_1^\top \end{bmatrix} \mathbf{r}_\mathbf{p} \right\} \approx_s \left\{ \begin{bmatrix} \mathbf{B} \\ \mathbf{c}_1^\top \end{bmatrix}, \begin{bmatrix} \mathbf{p} \\ c_3 \end{bmatrix} \right\}$$

for $\mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$, $\mathbf{p} \leftarrow \mathbb{Z}_q^n$, and $c_3 \leftarrow \mathbb{Z}_q$. Hence replacing $(\mathbf{p}, \mathbf{c}_1^\top \mathbf{r}_\mathbf{p})$ by a uniform pair leaves $\mathcal{A}$'s view statistically close, and the additive term $\lfloor q/2 \rfloor \mu$ is then masked by the uniform c_3. □