Equivocal Broadcast Encryption: Adaptively-Secure Optimal Distributed Broadcast Encryption from Lattices

Rishab Goyal
UW-Madison*

Saikumar Yadugiri
UW-Madison†

Abstract

We present the first Distributed Broadcast Encryption (DBE) scheme from falsifiable lattice assumptions that achieves adaptive security with optimal parameters (short public/secret keys and ciphertexts). Our construction enjoys transparent setup and offers flexible instantiation: we achieve a succinct CRS in the Random Oracle Model, or a long CRS in the standard model. Previously, no lattice-based DBE simultaneously achieved adaptivity and optimal parameters in either setting.

To achieve this, we introduce a new methodology for proving adaptive security: *Equivocal Encryption Systems*. This framework operates in two indistinguishable modes: a 'real' mode utilizing standard algorithms, and a 'fake' mode where keys and ciphertexts are jointly sampled with auxiliary trapdoors, enabling the dynamic equivocation of ciphertexts to arbitrary challenge values. While our approach is technically distinct from the celebrated Dual System Encryption (Waters, CRYPTO'09), we believe it could serve as a similarly powerful paradigm for realizing adaptive security across a broad class of lattice-based encryption systems.

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

Contents

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

2 Technical Overview **3**
- 2.1 Equivocal Encryption: A New Methodology to Prove Adaptive Security . . . 4
- 2.2 Equivocal DBE in Random Oracle Model . . . 5
- 2.3 Equivocal DBE in Standard Model . . . 8
- 2.4 Programmable Pseudorandom Set Sampler Constructions . . . 12
- 2.5 Related, Concurrent, and Subsequent Works . . . 15

3 Preliminaries **16**
- 3.1 Lattice Preliminaries . . . 16
- 3.2 ℓ-Decomposed LWE Assumption . . . 17
- 3.3 Matrix Commitment Scheme . . . 18
- 3.4 Distributed Broadcast Encryption . . . 18

4 Equivocal DBE: A New Methodology for Adaptive Security **20**
- 4.1 Semi-Statically Secure DBE from Equivocal DBE . . . 21

5 Equivocal DBE in Random Oracle Model **21**
- 5.1 Fake Setup and Equivocation . . . 23
- 5.2 Security Analysis . . . 24

6 Programmable Pseudorandom Set Sampler **25**

7 Equivocal DBE in Standard Model from Programmable Pseudorandom Set Sampler **27**
- 7.1 Fake Setup and Equivocation . . . 28
- 7.2 Security Analysis . . . 29

8 Programmable Pseudorandom Set Sampler from LWE **30**
- 8.1 Security Analysis . . . 32

9 Programmable Pseudorandom Set Sampler from 1-key LWE **33**
- 9.1 1-key LWE . . . 33
- 9.2 Construction . . . 34
- 9.3 Security Analysis . . . 35

A Additional Lattice Preliminaries **43**

B Accept LWE Assumption **44**
- B.1 d-Accept LWE . . . 45
- B.2 Discussion on Hardness of Accept LWE . . . 45

C Proof of Theorem 4.3 **46**

D Proofs from Section 5 **48**
D.1 Proof of Theorem 5.2 48
D.2 Proof of Theorem 5.3 52

E Proofs from Section 7 **55**
E.1 Proof of Theorem 7.2 55
E.2 Proof of Theorem 7.3 64

F Proofs from Section 8 **67**
F.1 Proof of Theorem 8.3 67
F.2 Proof of Theorem 8.4 73

G Proofs from Section 9 **76**
G.1 Proof of Theorem 9.4 76
G.2 Proof of Theorem 9.5 78

H Overview of the Piecewise Guessing Strategy **81**

1 Introduction

Broadcast Encryption (BE) [FN94] is a fundamental primitive that allows a sender to securely transmit data to a dynamically chosen subset of users. While traditional BE relies on a central authority, Distributed Broadcast Encryption (DBE) [WQZD10, BZ14] eliminates this single point of failure and trust, allowing users to generate their own keys and join the system permissionlessly. As post-quantum security becomes imperative, constructing efficient DBE from lattice assumptions has become a central goal in cryptography.

The State of the Art: The "Selectivity" Barrier. Recent breakthroughs have brought lattice-based DBE closer to reality. Under the hardness of a new falsifiable lattice assumption (ℓ-succinct/decomposed-LWE) [Wee24, AMR25], a line of recent works [CW24, CHW25, WW25, AMR25] have successfully constructed DBE schemes[1] with varying efficiency profiles. However, these constructions face a fundamental dichotomy:

Efficiency without Adaptivity: The most recent work by Wee and Wu [WW25] achieves fully optimal parameters—where public keys, secret keys, and ciphertexts are all small—but is limited to *selective security*. The adversary must declare the target set before seeing the public parameters, a restriction that is often unrealistic in dynamic distributed systems.

Adaptivity with Inefficiency: Conversely, the only available DBE scheme that achieves adaptive security, due to the recent work of Champion-Hsieh-Wu [CHW25], incurs a significant overhead. For N users, they require prohibitively large public keys ($|\mathsf{pk}| \propto N$) and large public parameters ($|\mathsf{crs}| \propto N^2$), and rely on the Random Oracle Model (ROM) [BR93] to achieve even these trade-offs. Moreover, their scheme does not achieve "transparent setup" which goes against the main motivation of decentralization in DBE.

Essentially, if we want a post-quantum DBE with adaptive security, then the only existing approach, by Champion-Hsieh-Wu [CHW25], comes with large public keys, large CRS *and* relies on ROM.

The Complexity Leveraging Bottleneck. In many cryptographic settings, one can generically lift selective security to adaptive security via complexity leveraging. While this standard approach works for identity-based or attribute-based encryption, it incurs an exponential loss in the reduction's advantage. Due to this exponential loss, complexity leveraging is insufficient in the context of (distributed) broadcast encryption. The reason is, in DBE, the reduction loss scales exponentially with the number of users N, requiring parameters to grow by at least a factor of N. DBE with such parameters trivially follows from vanilla public-key encryption, and not useful for applications.

The problem. In light of the current state-of-the-art, designing a DBE scheme that is (simultaneously) *adaptively secure and fully-succinct* (optimal parameters) from falsifiable assumptions remains a fascinating open problem. Going one step further, the problem of designing an adaptively-secure post-quantum DBE in the plain model (without ROs) is also completely open.

1.1 Our Contributions

Our techniques: A new methodology to realize adaptive security. We introduce a new methodology, that we call *'Equivocal Encryption System'*, to bridge the gap between selective and adaptive security in DBE and related encryption systems.

[1] [AMR25] constructed a registered ABE scheme [GHMR18, GV20, HLWW23] with *very selective* security which generically yields a selectively secure DBE scheme [FWW23].

In an Equivocal Encryption System, one can set up the system in two modes – 'real' mode and 'fake' mode. In the real mode, the secret keys and ciphertexts are generated using the actual system's key generation and encryption procedure. While in the fake mode, the secret keys and ciphertexts are jointly sampled along with auxiliary trapdoor information with one very special feature. By using the auxiliary trapdoor, ciphertexts sampled in the fake mode can be "equivocated" dynamically to any challenge value later. The special property of an Equivocal Encryption System is that an adversary cannot distinguish a 'real' ciphertext from an 'equivocated' ciphertext, while a 'fake' ciphertext (before equivocation) still provides a meaningful notion of semantic security.

Equivocal Encryption Systems opens up a new way to prove adaptive security in DBE. While Dual System Encryption [Wat09] relies on the algebraic indistinguishability of 'semi-functional' keys and ciphertexts which reside in hidden subspaces and functionally fail against specific ciphertexts, Equivocal Encryption operates via a trapdoor-governed fake mode. Instead of managing the orthogonality of semi-functional components, our framework generates 'empty' ciphertexts that allow the reduction to adaptively equivocate (or program) them to explain any choice of the adversary's challenge. Although our methodology is technically distinct from the celebrated Dual System Encryption, we believe it offers a similarly powerful template for realizing adaptive security across a broad class of lattice-based encryption systems.

Our results. Using our methodology, we resolve the aforementioned open problems in DBE. We present the first DBE scheme that achieves adaptive security and fully optimal parameters, while proving security under a falsifiable assumption. Our schemes are plausibly post-quantum secure, and rely on the same lattice assumptions as prior state-of-the-art, requiring no new hardness assumptions. We summarize our results along two axes below:

1. Fully-Optimal in ROM: We construct an adaptively-secure DBE scheme in the Random Oracle Model where all parameters—ciphertext size, user secret/public keys, and global public parameters—are optimal (polylogarithmic in the number of users, N). This is the *first* lattice-based scheme to achieve full succinctness and adaptive security simultaneously.

2. Standard Model with Long CRS: We also construct an adaptively-secure DBE scheme in the standard model. In this construction, user secret/public keys and ciphertexts remain fully-succinct, while only the global public parameters (i.e., CRS) grows with N. This is the *first* adaptively-secure DBE in the standard model from any falsifiable lattice assumption.

Programmable Pseudorandom Set Sampler. Our standard model DBE construction crucially relies on a new tool that we call Programmable Pseudorandom Set Samplers (PPS). PPS captures the essential ability to adaptively program a subset of sampled values while keeping the remaining values pseudorandom. We provide two instantiations of PPS. One based on standard LWE [Reg05] and another on a new falsifiable LWE assumption, demonstrating the flexibility of this tool for constructing adaptively secure systems. These constructions may be of independent interest, especially in the context of removing random oracles.

All our DBE schemes enjoy *transparent setup*, and are thus in line with the main motivation of decentralization behind DBE. We summarize our results and compare them with the state of the art in Table 1.

Reference	Assumption	$\|\mathsf{crs}\|$	$\|\mathsf{pk}\|$	$\|\mathsf{sk}\|$	$\|\mathsf{ct}\|$	TP	PQ	AD	No RO
[BZ14]	$i\mathcal{O}$ + OWF	–	1	1	1	✓	–	✓	✓
[GVW19]	Positional WE	1	1	1	1	✓	–	✓	✓
[FWW23]	WE + LWE	1	1	1	1	✓	–	✓	✓
[WQZD10]	Bilinear Diffie-Hellman Exponent	N	N^2	N	1	✓	✗	✗	✓
[KMW23]	Bilinear Diffie-Hellman Exponent	N	N	1	1	✗	✗	✓	✓
[GKPW24]	Generic Bilinear Group	N	N	1	1	✗	✗	✗	✓
[CW24]	succinct LWE	N^2	N	1	1	✗	✓	✗	✓
[CHW25]	succinct LWE	N^2	N	1	1	✗	✓	✓	✗
[AMR25]	decomposed LWE	N	1	1	1	✓	✓	✗	✓
[WW25]	decomposed LWE	1	1	1	1	✓	✓	✗	✓
Corollary 5.4	decomposed LWE	1	1	1	1	✓	✓	✓	✗
Corollary 9.6	decomposed LWE + 1-key LWE	N	1	1	1	✓	✓	✓	✓
Corollary 8.5	decomposed LWE	N^2	1	1	1	✓	✓	✓	✓

Table 1: Comparison of our results with prior works on Distributed Broadcast Encryption. $|\mathsf{crs}|, |\mathsf{pk}|, |\mathsf{sk}|$, and $|\mathsf{ct}|$ denotes the sizes of the common random string, user public keys, user secret keys, and ciphertext sizes respectively for N users. The values reported in these columns hide $\mathsf{poly}(\lambda, \log N)$ terms. For instance, if $|\mathsf{crs}|$ is reported as X, it signifies $|\mathsf{crs}| = X\,\mathsf{poly}(\lambda, \log N)$. **TP**, **PQ**, **AD**, and **No RO** denote transparent setup, post-quantumness, adaptive security, and absence of random oracles in the scheme respectively. Technically, [GVW19] constructed a standard broadcast encryption but the techniques used can be adapted to DBE.

2 Technical Overview

A Distributed Broadcast Encryption (DBE) [WQZD10, BZ14] scheme is a generalization of Broadcast Encryption [FN94] in which each user can generate their own public-secret key pair. Encryption uses public keys for a subset $S \subseteq [N]$ of users to encrypt a message μ to generate a compact ciphertext ct such that any one of these users can decrypt ct.

DBE Security. We start by recalling various security notions commonly studied in the DBE literature. The most general and powerful security notion is that of (full) adaptive security, described as follows:

Setup Phase. Attacker $\mathcal{A}$ sends the number of users 1^N. Generate $\mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^N)$ and send it to $\mathcal{A}$. Also initiate set $\mathcal{I} \leftarrow \emptyset$.

Query Phase. $\mathcal{A}$ can now make two types of queries.

- **Key Generation.** $\mathcal{A}$ sends $i \in [N]$ and receives pk_i where $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KGen}(i)$. Add i to $\mathcal{I}$.
- **Key Corruption.** $\mathcal{A}$ sends $i \in \mathcal{I}$ and receives sk_i. Remove i from $\mathcal{I}$.

Challenge Phase. $\mathcal{A}$ submits a challenge set $S^* \subseteq \mathcal{I}$, to which the challenger responds with $\mathsf{ct} \leftarrow \mathsf{Enc}(\{\mathsf{pk}_i\}_{i \in S^*}, \mu)$ for a random $\mu \leftarrow \{0,1\}$.

A DBE scheme is said to be adaptively secure if there doesn't exist any poly-time attacker $\mathcal{A}$ that can correctly predict μ with non-negligible advantage in the above game. Due to the technical difficulty in proving adaptive security directly for any DBE scheme, many works typically prove security under one of two alternate security notions – *semi-static and selective security.*

In a semi-static security game, $\mathcal{A}$ submits an 'honest' set $S' \subseteq [N]$ at the beginning and receives only the public keys corresponding to S'. That is, $\mathcal{A}$ chooses S', and receives $\{\mathsf{pk}_i\}_{i \in S'}$ along with crs. During the challenge phase, $\mathcal{A}$ can *adaptively* select any subset S^* as the challenge set as long as $S^* \subseteq S'$. Basically, unlike the (full) adaptive game, $\mathcal{A}$ can not corrupt any secret keys within S', but it still can choose challenge set adaptively.

In a selective security game, the adversary is maximally constrained as it must submit the challenge set S^* during setup phase and it receives $(\mathsf{crs}, \{\mathsf{pk}_i\}_{i\in S^*}, \mathsf{ct})$ in one shot. In short, unlike adaptive and semi-static games, adversary must choose everything at the beginning, and is not permitted to make any adaptive choices.

Semi-static is as good as Adaptive. Starting with the beautiful work of Gentry and Waters [GW09], prior works [GW09, KMW23, HWW25] have shown that a semi-statically secure DBE can be generically boosted into an adaptively secure DBE. Despite these generic compilers, most prior works have only been successful in proving selective security for DBE, highlighting the large gap between selective and semi-static security.

The Matrix Programming Barrier: *Why is it difficult to prove semi-static security from lattices?* Proving that lattice-based DBE is semi-statically/adaptively secure is difficult for the same fundamental reason the seminal $\mathsf{BGG+}$ paradigm [BGG+14] is restricted to selective security. Both face the same underlying barrier in their security reductions. Let us elaborate. In the $\mathsf{BGG+}$ paradigm for designing attribute-based encryption, a reduction algorithm needs to program the LWE matrix $\mathbf{A}$ as $\mathbf{A} := \mathbf{BR} - \mathbf{x}^\top \otimes \mathbf{G}$, where $\mathbf{x}$ is the attribute string and $\mathbf{G}$ is a gadget matrix. This is essential for two reasons: (a) this lets the reduction answer key queries for any unsatisfying predicate, and (b) the challenge ciphertext can be simulated given an LWE challenge.

Prior lattice-based DBE schemes [CW24, AMR25, WW25] directly builds on the $\mathsf{BGG+}$ paradigm, thus also inherit the same 'matrix programming' barrier. In particular, in all these schemes, they need to program the global LWE matrix $\mathbf{A}$ as $\mathbf{A} := \mathbf{BR} - \mathsf{Com}(S^*)$, where $\mathsf{Com}(S^*)$ is a structured *succinct* commitment of the challenge set S^*. Since in, both, semi-static and adaptive security games, the adversary can adaptively choose S^* after receiving the public keys and global parameters, thus one cannot design a reduction algorithm that can correctly simulate the challenge ciphertext from an LWE challenge. Essentially, the reduction must program the matrix $\mathbf{A}$ to encode $\mathsf{Com}(S^*)$; thus, prior works could only prove selective security.

2.1 Equivocal Encryption: A New Methodology to Prove Adaptive Security

As discussed in the introduction, one of our key technical contributions is the introduction of a new methodology that we call the *Equivocal Encryption System*. The central goal of our methodology is to get around the aforementioned 'Matrix Programming Barrier' in lattices.

At a very high level, our intuition is to establish a 'fake' mode in our encryption system such that, while we are in the fake mode, we can sample a fake ciphertext (say an encryption of dummy challenge set in the case of DBE) such that a fake ciphertext can later be *equivocated to a 'real-looking' ciphertext for any adaptively selected challenge* (i.e., an adaptive set S^* in the case of DBE). The advantage of defining a 'fake' mode is that we can bypass the *Matrix Programming Barrier* as the LWE matrices can be programmed w.r.t. a dummy challenge, and this dummy challenge can be selectively fixed (before the adversary even begins). Thus, our insight is to use the fake mode while proving security to ensure we do not need to embed S^* into the global parameters and/or public keys, yet we can design a reduction algorithm that can simulate a challenge ciphertext given an LWE challenge. In short, we bypass the matrix programming issue by using the power of "indistinguishable" ciphertext equivocation.

Although we explain our intuition in the context of DBE, our methodology is far more general and could be useful in many other areas of lattice-based cryptography to prove adaptive security. Next, we explain our methodology in more detail while focussing primarily on DBE.

Defining Equivocal Distributed Broadcast Encryption (EqDBE). To demonstrate our equivocal encryption framework for DBE, we define a strengthening of DBE, Equivocal DBE (EqDBE). An EqDBE scheme consists of $(\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ whose functionality is similar to a DBE scheme. In addition, in an EqDBE scheme there are two new algorithms $\mathsf{FakeSetup}, \mathsf{Equivocate}$. The $\mathsf{FakeSetup}$ algorithm uses the set $S' \subseteq [N], \mu \in \{0,1\}$ to generate a fake crs, $\widetilde{\mathsf{crs}}$, fake public keys $\{\widetilde{\mathsf{pk}}_i\}_{i\in S'}$, and more importantly a "fake ciphertext" $\widetilde{\mathsf{ct}}$. It also outputs aux which can be used to equivocate $\widetilde{\mathsf{ct}}$ during the challenge phase. The equivocation algorithm simply uses this aux and S^* to generate ciphertext ct for any $S^* \subseteq S'$. EqDBE's $(\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ algorithms satisfy correctness and compactness conditions like any DBE scheme. In addition, EqDBE also satisfies the following properties:

Mode Indistinguishability. The distribution of $(\mathsf{crs}, \{\mathsf{pk}_i\}_{i\in S'}, \mathsf{ct})$ generated via $(\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc})$ algorithms (respectively) is indistinguishable from $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \mathsf{ct}_{\mathsf{eq}})$ generated using $(\mathsf{FakeSetup}, \mathsf{Equivocate})$ algorithms. Here the equivocated ciphertext $\mathsf{ct}_{\mathsf{eq}}$ is generated by taking the fake ciphertext $\widetilde{\mathsf{ct}}$ (generated by $\mathsf{FakeSetup}$), and equivocating to the adaptively chosen S^* supplied by the attacker $\mathcal{A}$.

Selective Security for Fake Setup. For any selectively declared set S', the distribution of $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux}) \leftarrow \mathsf{FakeSetup}(S', \mu)$ should hide μ from any poly-time attacker $\mathcal{A}$. Crucially, we want μ to be hidden even when $\mathcal{A}$ receives the auxiliary trapdoor aux that can be used to equivocate $\widetilde{\mathsf{ct}}$ arbitrarily.

We show that an EqDBE scheme is tremendously useful in constructing a semi-statically secure DBE. Let us explain.

Semi-static DBE from EqDBE. To construct a DBE scheme from EqDBE, we simply use the $(\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ algorithms of the EqDBE scheme as is. As these algorithms satisfy correctness and compactness properties, so will the resulting DBE scheme. EqDBE's $\mathsf{FakeSetup}$ and $\mathsf{Equivocate}$ algorithms are only used in the security proof as we explain below:

(1) Start with the semi-static security game described earlier.
(2) Next, instead of using $\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}$, use $\mathsf{FakeSetup}, \mathsf{Equivocate}$ algorithms to sample the challenge ciphertext. Note that a semi-static attacker declares S' at the beginning, thus we could run $\mathsf{FakeSetup}$ w.r.t. S'.

We use *mode indistinguishability* to argue indistinguishability of (1)-(2), and then rely on *selective security for fake setup* to show that no polytime attacker can win with non-negligible probability in (2). Coupling this with known generic compilers [GW09, KMW23, HWW25], we obtain an adaptively secure DBE scheme. More details are provided in Section 4.

Taking a step back, the central advantage in using EqDBE is that it breaks the difficulty of proving semi-static security into two tractable pieces – mode indistinguishability and selective security. Since the seminal $\mathsf{BGG+}$ paradigm is typically sufficient to prove selective security, thus our intuition is that all we need to do is come up with an approach to equivocate ciphertexts and prove mode indistinguishability. Once we achieve this, then an adaptively secure DBE scheme should follow naturally by applying our methodology.

2.2 Equivocal DBE in Random Oracle Model

Our construction of EqDBE uses the matrix commitment scheme introduced in [Wee25]. We provide a brief description here. A reader familiar with this scheme can safely skip the next paragraph.

Matrix Commitments. Let (n, m, q, χ, σ) be Learning with Errors (LWE) [Reg05] parameters and $\mathsf{com.pp} := (\mathbf{W}, \mathbf{D}, \mathbf{B})$ be from the decomposed LWE assumption [AMR25]. Then, there exist three *deterministic* algorithms $(\mathsf{Com}, \mathsf{Verify}, \mathsf{Open})$ which take $\mathsf{com.pp}$ implicitly such that $\forall\ N \in \mathbb{N}$, any matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$,

$$\mathbf{C}\mathbf{V}_N = \mathbf{M} - \mathbf{B}\mathbf{Z}$$

where $\mathbf{C} = \mathsf{Com}(\mathbf{M}), \mathbf{V}_N = \mathsf{Verify}(1^N), \mathbf{Z} = \mathsf{Open}(\mathbf{M}), \mathbf{C} \in \mathbb{Z}_q^{n \times m}, \mathbf{V}_N \in \mathbb{Z}_q^{m \times N}, \mathbf{Z} \in \mathbb{Z}_q^{m \times N}$. Moreover, $\mathbf{V}_N, \mathbf{Z}$ have bounded norms independent of N. Full definition is provided in Section 3.3. Decomposed LWE assumption[2] [AMR25] conjectures that given $(\mathbf{W}, \mathbf{D}, \mathbf{B})$[3], $\{\mathbf{s}^\top \mathbf{B} + \mathsf{err}\} \approx_c \{\mathbf{c}^\top\}$ where $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{c} \leftarrow \mathbb{Z}_q^m$. By err we mean bounded norm discrete gaussian noise. The exact structure of $(\mathbf{W}, \mathbf{D}, \mathbf{B})$ is not relevant to the discussion. We refer the reader to Assumption 3.4 for full definition.

Starting Point: A simplified view of selective DBE from [WW25]. We describe a *selectively-secure* DBE scheme from the decomposed LWE assumption inspired by [WW25] (WW from henceforth). Our presentation slightly deviates the original WW selective DBE. We find that our selective DBE offers a much better starting point in understanding our EqDBE constructions. In this selective DBE scheme for N users,

$$\mathsf{crs} = (\mathsf{com.pp} := (\mathbf{W}, \mathbf{D}, \mathbf{B}), \mathbf{A}, \mathbf{p}, N), \text{ where } \mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$$
$$(\mathsf{pk}_i, \mathsf{sk}_i) = (\mathbf{t}_i, \mathbf{r}_i), \text{ such that } \mathbf{t}_i := \mathbf{B}\mathbf{r}_i + \mathbf{p} - \mathbf{A}\mathbf{v}_i, \mathbf{r}_i \leftarrow \{\pm 1\}^m$$

where $\mathbf{v}_i$ is i-th column of $\mathbf{V}_N = \mathsf{Verify}(1^N)$ from matrix commitments. The Enc algorithm uses public keys $\{\mathbf{t}_i\}_{i \in S}$ to construct a matrix $\mathbf{M} = [\mathbf{m}_1 \mid \ldots \mid \mathbf{m}_N] \in \mathbb{Z}_q^{n \times N}$ whose j-th column $\mathbf{m}_j$ is

$$\mathbf{m}_j = \begin{cases} \mathbf{t}_j & \text{if } j \in S \\ \mathbf{0} & \text{otherwise} \end{cases} \tag{1}$$

Then, Enc computes $\mathbf{C} = \mathsf{Com}(\mathbf{M})$, samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, and outputs

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

In order to decrypt given $\mathbf{r}_j$ for j-th user, Dec computes $\mathbf{V}_N = \mathsf{Verify}(1^N)$ and also $\mathbf{M}$ using the public keys $\{\mathbf{t}_i\}_{i \in S}$ (by Equation (1)). Using $\mathbf{M}$, Dec can compute $\mathbf{Z} = \mathsf{Open}(\mathbf{M})$. Then, Dec computes the expanded vector $\mathbf{c}_{exp}^\top = \mathbf{c}_2^\top \mathbf{V}_N + \mathbf{c}_1^\top \mathbf{Z} \approx \mathbf{s}^\top(\mathbf{A}\mathbf{V}_N + \mathbf{M}) \in \mathbb{Z}_q^{1 \times N}$. Note that the j-th element of $\mathbf{c}_{exp}^\top$ is roughly $\mathbf{s}^\top(\mathbf{t}_j + \mathbf{A}\mathbf{v}_j) = \mathbf{s}^\top(\mathbf{B}\mathbf{r}_j + \mathbf{p})$ if $j \in S$. Thus, using $\mathbf{r}_j$ and $\mathbf{c}_1^\top$, Dec roughly computes $\mathbf{s}^\top \mathbf{p}$ which can be used to retrieve μ with high probability.

Selective security of this DBE. We provide a sketch of the selective security proof below.

(1) This is the selective security game where poly-time attacker $\mathcal{A}$ selectively submits the challenge set $S^* \subseteq [N]$. It receives $(\mathsf{crs}, \{\mathbf{t}_i\}_{i \in S^*}, \mathsf{ct})$ generated as above for any μ.
(2) Using leftover hash lemma, we sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$ for each $i \in S^*$.
(3) Once again, using leftover hash lemma, we set $\mathbf{A}$ and $\mathbf{p}$ in crs as $\mathbf{B}\mathbf{R}_\mathbf{A} - \mathsf{Com}(\mathbf{M}^*)$ and $\mathbf{B}\mathbf{r}_\mathbf{p}$ respectively. Here, $\mathbf{M}^*$ is the matrix constructed using Equation (1) for set S^*. This lets us set ct as $(\mathbf{c}_1^\top, \mathbf{c}_2^\top \approx \mathbf{c}_1^\top \mathbf{R}_\mathbf{A}, c_3 \approx \mathbf{c}_1^\top \mathbf{r}_\mathbf{p} + \lfloor q/2 \rfloor \mu)$.

[2] ℓ-decomposed LWE is implied by ℓ-succinct LWE [Wee24] as shown in [AMR25].
[3] We are imprecise in our description about the size of $\mathbf{B}$. Throughout overview, we assume $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$ but it's column length is much larger (Assumption 3.4). Nevertheless, our design can be adapted as we demonstrate in main body.

(4) Now, using decomposed LWE security we can change $\mathbf{c}_1^\top$ to a random string, which then would mean c_3 loses all information about μ.

Note that in step (3), we program crs with information about $\mathbf{M}^*$ and thus inherit the matrix programming barrier as explained above. If we consider the semi-static security game, where $\mathcal{A}$ adaptively chooses challenge set S^*, it is unclear if we can set $\mathbf{c}_2^\top$ using $\mathbf{c}_1^\top$ like in step (3).

Equivocating DBE ciphertexts. To construct an EqDBE scheme, we should be able to take a ct' for the set S' and explain it for any set $S^* \subseteq S'$ under the same crs. To do so, we add random masks to the ciphertext which will allow us to equivocate ct' for S^*. We expand on this below.

In our EqDBE scheme, $\mathsf{crs}, (\mathsf{pk}_i, \mathsf{sk}_i)$ for any $i \in [N]$ remain unchanged from the selective DBE scheme described above. Enc on the other hand, now generates $\mathbf{M} = [\mathbf{m}_1 \mid \ldots \mid \mathbf{m}_N] \in \mathbb{Z}_q^{n \times N}$ as

$$\mathbf{m}_j = \begin{cases} \mathbf{t}_j + \boxed{\mathbf{B}\mathbf{x}_j} & \text{if } j \in S \\ \boxed{\mathbf{d}_j} & \text{otherwise} \end{cases} \tag{2}$$

where $\mathbf{x}_j \leftarrow \{\pm 1\}^m$ and $\mathbf{d}_j \leftarrow \mathbb{Z}_q^n$. Enc now computes $\mathbf{C} = \mathsf{Com}(\mathbf{M})$, samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, and outputs

$$\mathsf{ct} := \left(\mathbf{s}^\top \mathbf{B} + \mathsf{err}, \mathbf{s}^\top(\mathbf{A} + \mathbf{C}) + \mathsf{err}, \mathbf{s}^\top \mathbf{p} + \mathsf{err} + \lfloor q/2 \rfloor \mu, \boxed{\{\mathbf{x}_j\}_{j \in S}}, \boxed{\{\mathbf{d}_j\}_{j \notin S}}\right)$$

Here, $\boxed{X}$ represents the random masks we add to the ciphertext. This process helps in transforming the matrix commitment scheme to be lossy / imperfectly binding while still preserving all the properties provided by this scheme. In order to decrypt given $\mathbf{r}_j$ for j-th user, Dec can still compute $\mathbf{c}_{exp}^\top = \mathbf{c}_2^\top \mathbf{V}_N + \mathbf{c}_1^\top \mathbf{Z} \approx \mathbf{s}^\top(\mathbf{A}\mathbf{V}_N + \mathbf{M})$. The j-th element of $\mathbf{c}_{exp}^\top$ is $\mathbf{s}^\top(\mathbf{t}_j + \mathbf{A}\mathbf{v}_j + \mathbf{B}\mathbf{x}_j) = \mathbf{s}^\top(\mathbf{B}(\mathbf{r}_j + \mathbf{x}_j) + \mathbf{p})$ if $j \in S$. Thus, using $\mathbf{x}_j, \mathbf{c}_1^\top, \mathbf{r}_j$, user can compute μ. We point out that it is essential to give out the random masks $\boxed{\{\mathbf{x}_j\}_{j \in S}}$ and $\boxed{\{\mathbf{d}_j\}_{j \notin S}}$ as part of ct. Only then Dec can compute $\mathbf{M}$ fully and $\mathbf{Z} = \mathsf{Open}(\mathbf{M})$. Dec also needs $\mathbf{x}_j$ to properly compute μ.

Fake Setup and Equivocation. The $\mathsf{FakeSetup}$ algorithm uses $S' \subseteq [N], \mu \in \{0,1\}$ to sample public keys $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$ for each $i \in S'$. It also samples $\{\mathbf{x}'_j\}_{j \in S'}, \{\mathbf{d}'_j\}_{j \notin S'}$ and computes matrix $\mathbf{M}'$ (for set S' using Equation (2)). It then sets $\mathbf{A} = \mathbf{A}' - \mathsf{Com}(\mathbf{M}')$ after sampling random $\mathbf{A}'$ and $\mathbf{p}$. $\widetilde{\mathsf{ct}}$ is set as $(\mathbf{s}^\top \mathbf{B} + \mathsf{err}, \mathbf{s}^\top(\mathbf{A} + \mathbf{C}') + \mathsf{err}, \mathbf{s}^\top \mathbf{p} + \mathsf{err} + \lfloor q/2 \rfloor \mu)$ after sampling $\mathbf{s} \leftarrow \mathbb{Z}_q^n$. aux information contains $(S', \{\mathbf{x}_j\}_{j \in S'}, \{\mathbf{d}_j\}_{j \notin S'}, \widetilde{\mathsf{ct}})$ among other things.

The $\mathsf{Equivocate}$ algorithm given a set $S^* \subseteq S'$, sets $\mathbf{x}_j^*$ and $\mathbf{d}_j^*$ as follows:

$$\mathbf{x}_j^* = \mathbf{x}'_j \text{ for } j \in S^* \text{ and } \mathbf{d}_j^* = \begin{cases} \mathbf{d}'_j & \text{if } j \notin S' \\ \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j & \text{if } j \in S' \setminus S^* \end{cases}$$

It then outputs $\mathsf{ct} := \left(\widetilde{\mathsf{ct}}, \boxed{\{\mathbf{x}_j^*\}_{j \in S^*}}, \boxed{\{\mathbf{d}_j^*\}_{j \notin S^*}}\right)$.

Crucially, observe that matrix $\mathbf{M}^* = [\mathbf{m}_1^* \mid \ldots \mid \mathbf{m}_N^*]$ computed according to Equation (2) for the set $S^* \subseteq S'$ using the random masks $\{\mathbf{x}_j^*\}_j, \{\mathbf{d}_j^*\}_j$ given out by $\mathsf{Equivocate}$, and $\mathbf{M}' = [\mathbf{m}'_1 \mid \ldots \mid \mathbf{m}'_N]$ computed according to Equation (2) for S' using $\{\mathbf{x}'_j\}_j, \{\mathbf{d}'_j\}_j$ sampled by $\mathsf{FakeSetup}$ are exactly the same. The only difference in these matrices is for j-th columns when $j \in S' \setminus S^*$. In these columns, $\mathbf{m}'_j$ is $\mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ and $\mathbf{m}_j^* = \mathbf{d}_j^* = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$. Thus, they are both identical.

Security Analysis. We prove that the above construction satisfies mode indistinguishability and selective security for fake setup properties. For simplicity assume that $N = 3, S' = \{1, 2\}, S^* = \{1\}$.

Mode Indistinguishability. We provide a sketch of the proof below.

(1) This is the semi-static game where we generate $(\mathsf{crs}, \mathsf{pk}_1, \mathsf{pk}_2, \mathsf{ct})$ using $\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}$ described above for any $\mu \in \{0,1\}$.
(2) Using leftover hash lemma, we sample $\mathsf{pk}_1 = \mathbf{t}_1 \leftarrow \mathbb{Z}_q^n$ and $\mathsf{pk}_2 = \mathbf{t}_2 \leftarrow \mathbb{Z}_q^n$.
(3) Now, we will sample $\mathbf{x}'_1, \mathbf{x}'_2 \leftarrow \{\pm 1\}^m, \mathbf{d}'_3 \leftarrow \mathbb{Z}_q^n$, compute $\mathbf{M}' = [\mathbf{t}_1 + \mathbf{B}\mathbf{x}'_1 \mid \mathbf{t}_2 + \mathbf{B}\mathbf{x}'_2 \mid \mathbf{d}'_3]$ and $\mathbf{C}' = \mathsf{Com}(\mathbf{M}')$. We still compute ct similar to Enc, i.e., by sampling fresh $\mathbf{x}_1^* \leftarrow \{\pm 1\}^m, \mathbf{d}_2^*, \mathbf{d}_3^* \leftarrow \mathbb{Z}_q^n$.
Till now, the view of $\mathcal{A}$ only changed a little. Moreover, $\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{d}'_3$ do not appear anywhere in $\mathsf{crs}, \{\mathbf{t}_1, \mathbf{t}_2\}$, or ct.
(4) Using leftover hash lemma, we switch $\mathbf{d}_2^* := \mathbf{t}_2 + \mathbf{B}\mathbf{x}'_2$ $(2 \in S' \setminus S^*)$. Moreover, we will also set $\mathbf{x}_1^* := \mathbf{x}'_1, \mathbf{d}_3^* := \mathbf{d}'_3$ as they are from the same distribution.
(5) This is the semi-static game where we generate $(\mathsf{crs}, \mathsf{pk}_1, \mathsf{pk}_2, \mathsf{ct})$ using $\mathsf{FakeSetup}, \mathsf{Equivocate}$ described above for any $\mu \in \{0,1\}$.

Steps (4)-(5) are simply semantic changes. Thus, mode indistinguishability holds statistically.

Selective Security for Fake Setup. Note that $\widetilde{\mathsf{ct}}$ is a ciphertext for S' distributed similarly to the ciphertext in our selectively secure DBE scheme. The auxiliary information $\mathsf{aux} := (S', \mathbf{x}_1, \mathbf{x}_2, \mathbf{d}_3, \widetilde{\mathsf{ct}})$ does not reveal much information about a $\widetilde{\mathsf{ct}}$ for S' and can only be used to recreate $\mathbf{M}'$ or equivocation. Thus, selective security for fake setup holds by a similar argument to that of our selectively secure DBE scheme.

Final Step: Using ROM to make ciphertext compact. Unfortunately, the ciphertext ct in our scheme is no longer compact and grows with N. This is because we have N random masks $(\{\mathbf{x}_j\}_{j\in S}, \{\mathbf{d}_j\}_{j\notin S})$ which we have to provide for correctness. Our idea is to use a random function to make ct compact. More specifically, we consider two hash functions $\mathcal{H}_1 : \{0,1\}^\lambda \times [N] \to \{\pm 1\}^m$ and $\mathcal{H}_2 : \{0,1\}^\lambda \times [N] \to \mathbb{Z}_q^n$ which will be part of the crs. During encryption, we sample two handles $\mathsf{sd}_1, \mathsf{sd}_2 \leftarrow \{0,1\}^\lambda$ and compute $\mathbf{x}_j = \mathcal{H}_1(\mathsf{sd}_1, j)$ for $j \in S$ and $\mathbf{d}_j = \mathcal{H}_2(\mathsf{sd}_2, j)$ for $j \notin S$ and send $\mathsf{sd}_1, \mathsf{sd}_2$ along with ct. This way Dec can recreate $\mathbf{M}$ efficiently. We show that by viewing these hash functions as random oracles [BR93], we can prove mode indistinguishability readily. As handles $\mathsf{sd}_1, \mathsf{sd}_2$ are short and their size is independent of N, this yields a fully compact ciphertext. This concludes the description of our EqDBE scheme in the random oracle model. Full construction is provided in Section 5.

2.3 Equivocal DBE in Standard Model

Our next goal is to remove random oracles from previous construction and attain adaptively secure DBE scheme in the standard model. To the best of our knowledge, there does not exist an adaptively secure DBE scheme from lattices without random oracles.

Our observation is that we need random oracles in the previous EqDBE construction to sample the random masks in such a way that they can be "adaptively" programmed during the challenge phase. In a bit more detail, we highlight two important properties that random oracles $\mathcal{H}_1, \mathcal{H}_2$ present in crs of previous EqDBE construction yield.

1. **Compactness.** The crs remains short and allows to compute N random masks given the handles $\mathsf{sd}_1, \mathsf{sd}_2 \in \{0,1\}^\lambda$. Moreover, given $\mathsf{sd}_1, \mathsf{sd}_2 \in \{0,1\}^\lambda$, we can sample any N random masks deterministically.

2. **Adaptive Correlation Hiding.** For $j \in S' \setminus S^*$, the random mask sampled during $\mathsf{FakeSetup}$, $\mathbf{x}'_j$ and the random mask revealed by $\mathsf{Equivocate}\ \mathbf{d}^*_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ are correlated. Random oracles allow us to hide this correlation by adaptively, during equivocation, program $\mathcal{H}_1(\mathsf{sd}_1, j), \mathcal{H}_2(\mathsf{sd}_2, j)$ with $\mathbf{x}^*_j, \mathbf{d}^*_j$ respectively where $\mathbf{x}^*_j$ is completely independent of $\mathbf{x}'_j$.

We remark that achieving property 2 in standard model requires great care. This is because while working in the standard model, crs sampled by $\mathsf{FakeSetup}$ is computed without the knowledge of S^*. Still, this crs should reveal either $\mathbf{x}'_j$ or $\mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ (only one of these) based on whether $j \stackrel{?}{\in} S^*$ all while keeping the handles short. To address these challenges in the standard model, we introduce a new tool called, *Programmable Pseudorandom Set Sampler* (PPS). This tool may be of independent interest in the context of removing (programmable) random oracles. We provide definition of PPS below.

Programmable Pseudorandom Set Sampler. A PPS scheme for message space $\mathcal{M}$ consists of 6 main algorithms $(\mathsf{PPGen}, \mathsf{MSKGen}, \mathsf{KGen}, \mathsf{pkEnc}, \mathsf{skEnc}, \mathsf{Dec})$. PPGen takes 1^N as input and generates crs that the rest of the algorithms use implicitly. MSKGen generates a master secret key msk. KGen uses a msk and a set $S \subseteq [N]$ to compute a short secret key sk_S. The skEnc algorithm uses a msk, and index $i \in [N]$, message $\mu \in \mathcal{M}$ to generate ciphertext ct. The pkEnc algorithm simply uses index i to generate a garbage ciphertext ct. Decode algorithm uses $(i, \mathsf{ct}, S, \mathsf{sk}_S)$ to output message $\mu' \in \mathcal{M} \cup \{\perp\}$. A PPS scheme needs to satisfy the following properties.

Correctness. We say PPS is correct if $i \in S \implies \mu = \mathsf{Decode}(i, \mathsf{ct}, S, \mathsf{sk}_S)$, where $\mathsf{sk}_S \leftarrow \mathsf{KGen}(\mathsf{msk}, S)$, $\mathsf{ct} \leftarrow \mathsf{skEnc}(\mathsf{msk}, i, \mu)$.

Compactness. $|\mathsf{sk}_S| = \mathsf{poly}(\lambda, \log N)$.

Payload Hiding. For any i, μ (even adversarially chosen), $\mathsf{ct} \leftarrow \mathsf{skEnc}(\mathsf{msk}, i, \mu)$ and $\mathsf{ct} \leftarrow \mathsf{pkEnc}(i)$ should be indistinguishable to an attacker $\mathcal{A}$ that adaptively queries for a single sk_{S^*} for S^* such that $i \notin S^*$. This property should hold even if $\mathcal{A}$ sees multiple skEnc ciphertexts for indices in set P such that $i \notin P$.

Mode Indistinguishability. For any i, $\mathsf{ct} \leftarrow \mathsf{skEnc}(\mathsf{msk}, i, \mu)$ and $\mathsf{ct} \leftarrow \mathsf{pkEnc}(i)$ should be indistinguishable to an attacker $\mathcal{A}$ that adaptively queries for a single sk_{S^*} for S^* such that $i \in S^*$. More importantly, μ is *chosen randomly by the challenger*[4]. This property should hold even if $\mathcal{A}$ sees multiple skEnc ciphertexts for indices in set P such that $i \notin P$.

Full definition of a PPS scheme is provided in Section 6.

EqDBE in Standard Model via PPS. At a very high level, the idea is to simply replace hash functions $\mathcal{H}_1, \mathcal{H}_2$ with two instantiation of PPS, one for message space $\{\pm 1\}^m$ and other for $\mathbb{Z}_q^n$. crs will now contain $2N$ PPS ciphertexts, N from each instantiation. The handles $\mathsf{sd}_1, \mathsf{sd}_2$ we sample during Enc will be replaced with secret keys of PPS. We substitute $\mathcal{H}(\mathsf{sd}, i)$ evaluation with PPS Decode.

Compact secret keys of PPS allow us to achieve short ciphertexts while still allowing to reveal N random masks (property 1). Although it isn't immediately clear how PPS helps us adaptively reveal at most one of $\mathbf{x}'_j, \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ for $j \in S' \setminus S^*$, we show that by a careful combination of payload hiding and mode indistinguishability, this feature can be achieved. We provide our construction of EqDBE in standard model using PPS below.

[4]Indeed if $\mathcal{A}$ can choose μ, it can trivially break this property by using Decode algorithm as pkEnc doesn't use μ.

GSetup: Along with $\mathsf{com.pp}, \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$, we sample two PPS instantiations $\mathsf{PPS}^{(1)}, \mathsf{PPS}^{(2)}$. We then sample $\{\mathsf{ct}_i^{(1)} \leftarrow \mathsf{PPS}^{(1)}.\mathsf{pkEnc}(i)\}_{i\in[N]}$ and $\{\mathsf{ct}_i^{(2)} \leftarrow \mathsf{PPS}^{(2)}.\mathsf{pkEnc}(i)\}_{i\in[N]}$.
$\mathsf{crs} := (\mathsf{com.pp}, \mathbf{A}, \mathbf{p}, \{\mathsf{ct}_i^{(1)}\}_{i\in[N]}, \{\mathsf{ct}_i^{(2)}\}_{i\in[N]})$

Enc: Sample $\mathsf{msk}^{(1)} \leftarrow \mathsf{PPS}^{(1)}.\mathsf{MSKGen}, \mathsf{msk}^{(2)} \leftarrow \mathsf{PPS}^{(2)}.\mathsf{MSKGen}$. Further, $\mathsf{sk}_S \leftarrow \mathsf{PPS}^{(1)}.\mathsf{KGen}(\mathsf{msk}^{(1)}, S)$ and $\mathsf{sk}_{\overline{S}} \leftarrow \mathsf{PPS}^{(2)}.\mathsf{KGen}(\mathsf{msk}^{(2)}, [N] \setminus S)$.
For each $j \in [N]$, if $j \in S$, compute $\mathbf{x}_j = \mathsf{PPS}^{(1)}.\mathsf{Decode}(j, \mathsf{ct}_j^{(1)}, S, \mathsf{sk}_S)$ and if $j \notin S$, $\mathbf{d}_j = \mathsf{PPS}^{(2)}.\mathsf{Decode}(j, \mathsf{ct}_j^{(2)}, [N] \setminus S, \mathsf{sk}_{\overline{S}})$. Then compute matrix $\mathbf{M} = [\mathbf{m}_1 \mid \ldots \mid \mathbf{m}_N] \in \mathbb{Z}_q^{n\times N}$ using these random masks according to Equation (2) and $\mathbf{C} = \mathsf{Com}(\mathbf{M})$.
Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, output $\mathsf{ct} := (\mathbf{s}^\top\mathbf{B} + \mathsf{err}, \mathbf{s}^\top(\mathbf{A} + \mathbf{C}) + \mathsf{err}, \mathbf{s}^\top\mathbf{p} + \mathsf{err} + \lfloor q/2 \rfloor \mu, \mathsf{sk}_S, \mathsf{sk}_{\overline{S}})$.

KGen remains unchanged from Section 2.2. For Dec, we can use $\mathsf{sk}_S, \mathsf{sk}_{\overline{S}}$ which are provided as part of EqDBE ciphertext and crs to reconstruct matrix $\mathbf{M}$ via PPS Decode algorithm. From there we proceed similarly to EqDBE from Section 2.2. Compactness holds because PPS secret keys are independent of N. We will now describe FakeSetup and Equivocate algorithms.

FakeSetup : We leverage the correctness of PPS and program $\{\mathbf{x}'_j\}_{j\in S'}$ and $\{\mathbf{d}'_j\}_{j\notin S'}$ using PPS skEnc algorithm as follows.
Sample $\widetilde{\mathsf{pk}_i} = \mathbf{t}_i \leftarrow \mathbb{Z}_q^n$ for each $i \in S'$. Sample $\mathsf{msk}^{(1)} \leftarrow \mathsf{PPS}^{(1)}.\mathsf{MSKGen}, \mathsf{msk}^{(2)} \leftarrow \mathsf{PPS}^{(2)}.\mathsf{MSKGen}$.
Sample random masks $\mathbf{x}'_j \leftarrow \{\pm 1\}^m$ if $j \in S'$ and $\mathbf{d}'_j \leftarrow \mathbb{Z}_q^n$ if $j \notin S'$. Compute $\mathbf{M}'$ for set S' according to Equation (2) with these random masks.
Compute $\mathbf{C}' = \mathsf{Com}(\mathbf{M}')$ and $\mathbf{A} = \mathbf{A}' - \mathbf{C}'$ where $\mathbf{A}' \leftarrow \mathbb{Z}_q^{n\times m}$. Also, $\mathbf{p} \leftarrow \mathbb{Z}_q^n$.
Sample $\mathsf{ct}_j^{(1)}, \mathsf{ct}_j^{(2)}$ for $j \in [N]$ as follows

$$\mathsf{ct}_j^{(1)} \leftarrow \begin{cases} \mathsf{PPS}^{(1)}.\mathsf{skEnc}(j, \mathbf{x}_j) & \text{if } j \in S' \\ \mathsf{PPS}^{(1)}.\mathsf{pkEnc}(j) & \text{if } j \notin S' \end{cases} \text{ and } \mathsf{ct}_j^{(2)} \leftarrow \begin{cases} \mathsf{PPS}^{(2)}.\mathsf{skEnc}(j, \mathbf{t}_j + \mathbf{B}\mathbf{x}_j) & \text{if } j \in S' \\ \mathsf{PPS}^{(2)}.\mathsf{skEnc}(j, \mathbf{d}'_j) & \text{if } j \notin S' \end{cases}$$

Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ and compute $\widetilde{\mathsf{ct}} := (\mathbf{s}^\top\mathbf{B} + \mathsf{err}, \mathbf{s}^\top(\mathbf{A} + \mathbf{C}') + \mathsf{err}, \mathbf{s}^\top\mathbf{p} + \mathsf{err} + \lfloor q/2 \rfloor \mu)$. Set $\mathsf{aux} := (S', \mathsf{msk}^{(1)}, \mathsf{msk}^{(2)}, \widetilde{\mathsf{ct}})$ among other things.
Output $\left(\widetilde{\mathsf{crs}} := \left(\mathsf{com.pp}, \mathbf{A}, \mathbf{p}, \{\mathsf{ct}_i^{(1)}\}_{i\in[N]}, \{\mathsf{ct}_i^{(2)}\}_{i\in[N]}\right), \{\widetilde{\mathsf{pk}_i}\}_{i\in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux}\right)$

Equivocate: Given $S^* \subseteq S'$, sample $\mathsf{sk}_{S^*} \leftarrow \mathsf{PPS}^{(1)}.\mathsf{KGen}(\mathsf{msk}^{(1)}, S^*), \mathsf{sk}_{\overline{S^*}} \leftarrow \mathsf{PPS}^{(2)}.\mathsf{KGen}(\mathsf{msk}^{(2)}, [N] \setminus S^*)$ using $\mathsf{msk}^{(1)}, \mathsf{msk}^{(2)}$ that are present in aux.
Output $\mathsf{ct} := \left(\widetilde{\mathsf{ct}}, \mathsf{sk}_{S^*}, \mathsf{sk}_{\overline{S^*}}\right)$.

Observe that using $\mathsf{sk}_{S^*}, \mathsf{sk}_{\overline{S^*}}$ from EqDBE ciphertext and $\{\mathsf{ct}_j^{(1)}\}_{j\in[N]}, \{\mathsf{ct}_j^{(2)}\}_{j\in[N]}$ present in crs, by correctness of $\mathsf{PPS}^{(1)}, \mathsf{PPS}^{(2)}$ one can only learn the values of $\mathbf{M}'$. Take $\{\mathsf{ct}_j^{(1)}\}_{j\in[N]}$ and sk_{S^*}, using these only if $j \in S^*$, $\mathsf{PPS}^{(1)}.\mathsf{Decode}$ reveals $\mathbf{x}'_j$. Similarly, using $\{\mathsf{ct}_j^{(2)}\}_{j\in[N]}$ and $\mathsf{sk}_{\overline{S^*}}$, only if $j \notin S^*$, we can learn $\mathbf{d}'_j$ if $j \notin S'$ and $\mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S' \setminus S^*$. These are the exact same values that $\mathbf{M}'$ uses in its columns. Thus, $\mathbf{M}'$ computed for set S' and $\mathbf{M}^*$ computed for set S^* according to Equation (2) are exactly the same. For illustrative purposes, we provide this for the simple case of $N = 3, S' = \{1, 2\}, S^* = \{1\}$ in Figure 1.

Security Analysis. We prove that the above construction satisfies mode indistinguishability and selective security for fake setup properties.

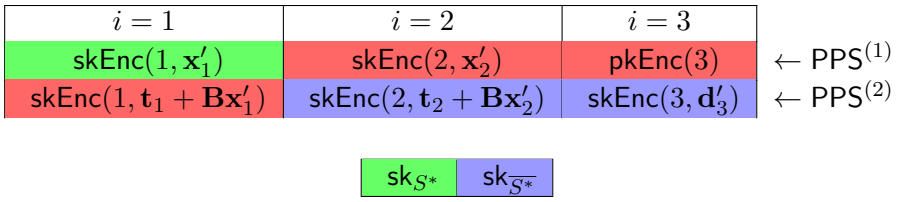


Figure 1: Structure of crs and values that can be learned from crs and secret keys yielded by FakeSetup, Equivocate algorithms. x means that there is either no value programmed in this slot or with the secret keys present, this value cannot be learned. x can be decoded using sk_{S^*} and x can be decoded using $\mathsf{sk}_{\overline{S^*}}$ respectively.

Selective Security for Fake Setup. Note that only useful part of aux is $\mathsf{msk}^{(1)}$, $\mathsf{msk}^{(2)}$ of PPS instantiations. Using these an adversary can either recreate $\mathbf{M}'$ from crs or equivocate. Thus, selective security is almost identical to Section 2.2 but in the standard model with PPS.

Mode Indistinguishability. This is tricky. Firstly, we can set $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$ for each $i \in S'$ by leftover hash lemma. Then for each $i \notin S'$, we can set $\mathsf{ct}_i^{(2)} \leftarrow \mathsf{PPS}^{(2)}.\mathsf{skEnc}(i, \mathbf{d}'_i)$ for randomly sampled $\mathbf{d}'_i \leftarrow \mathbb{Z}_q^n$ using mode indistinguishability property of $\mathsf{PPS}^{(2)}$.

Next, we would like to program for each $i \in S'$, $\mathsf{ct}_i^{(1)}, \mathsf{ct}_i^{(2)}$ such that they decode to $\mathbf{x}'_i, \mathbf{t}_i + \mathbf{B}\mathbf{x}'_i$. We do this via a hybrid argument, iterating through each user $i \in S'$. The main hurdle is that the correct programming strategy for user i depends on the bit $b_i := (i \in S^*)$, which is unknown during crs generation. If we guess full set S^* we suffer exponential loss. To bypass this, we utilize the 'Piecewise Guessing Framework' [HW09, FKPR14, FJP15, HJO+16, JW16, JKK+17, KW20]. In the i-th hybrid transition, we only need to correctly guess the single bit b_i. This localized guessing strategy keeps the security loss minimal.

In a nutshell, here is the proof strategy: for each $i \in S'$, we will guess one bit of information corresponding to $(i \in S^*)$. If yes: firstly, we will use the mode indistinguishability property of $\mathsf{PPS}^{(1)}$ to change $\mathsf{ct}_i^{(1)} \leftarrow \mathsf{PPS}^{(1)}.\mathsf{skEnc}(i, \mathbf{x}'_i)$ and then use the payload hiding property of $\mathsf{PPS}^{(2)}$ to change $\mathsf{ct}_i^{(2)} \leftarrow \mathsf{PPS}^{(2)}.\mathsf{skEnc}(i, \mathbf{t}_i + \mathbf{B}\mathbf{x}'_i)$. If no: then we first use the mode indistinguishability property of $\mathsf{PPS}^{(2)}$ to change $\mathsf{ct}_i^{(2)} \leftarrow \mathsf{PPS}^{(2)}.\mathsf{skEnc}(\mathsf{msk}^{(2)}, i, \mathbf{d}'_i)$ for uniformly random $\mathbf{d}'_i \leftarrow \mathbb{Z}_q^n$. At this point, $\mathbf{x}'_i$ is not appearing anywhere in the view of the attacker. Thus, we can use leftover hash lemma to switch $\mathbf{d}'_i$ to be $\mathbf{t}_i + \mathbf{B}\mathbf{x}'_i$. Then, we will use payload hiding property of $\mathsf{PPS}^{(1)}$ to change $\mathsf{ct}_i^{(1)} \leftarrow \mathsf{PPS}^{(1)}.\mathsf{skEnc}(i, \mathbf{x}'_i)$.

Crucially, observe that in both yes and no cases, we arrive at the same hybrid where both $\mathsf{ct}_i^{(1)}$ and $\mathsf{ct}_i^{(2)}$ are sampled using skEnc. Thus, by the piecewise guessing strategy from [JKK+17, KW20], the difference between the hybrids where $\mathsf{ct}_i^{(1)}, \mathsf{ct}_i^{(2)}$ are sampled using pkEnc and the one where they are sampled using skEnc is at most $2\max\{\epsilon_{yes}, \epsilon_{no}\}$ where ϵ_{yes} and ϵ_{no} are the advantages of an attacker in distinguishing the yes and no strands of hybrids respectively (also check Appendix H). Similar techniques were used in the context of registered attribute-based encryption in [HLWW23, ZZGQ23]. An illustrative flow of hybrids in mode indistinguishability proof can be found in Figure 2 below.

This way, using the piecewise guessing strategy, after $|S'|$-many hybrids, we can set the crs to be exactly as we do in FakeSetup. The rest of the changes are just semantic and mode indistinguishability property of the design holds.

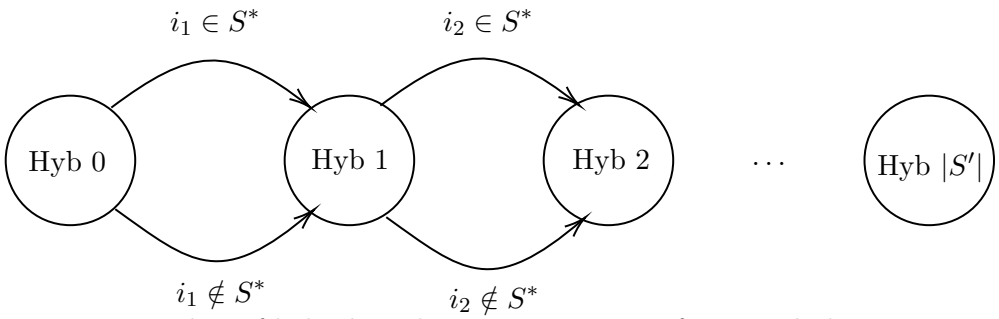


Figure 2: Flow of hybrids and piecewise guessing framework description

On the size of crs. Interestingly, crs in our EqDBE in standard model construction grows with $\Omega(N)$ due to presence of $2N$ PPS ciphertexts (N per each instantiation). If our crs doesn't grow with N, sampling and programming N random masks in the crs with short handles $\mathsf{sk}_S, \mathsf{sk}_{\overline{S}}$ might incur certain incompressibility barriers similar to non-committing encryption [Nie02]. We leave it as an interesting open problem to either prove a barrier to reducing the crs size or to construct an adaptively secure DBE with short crs in standard model.

This concludes the description of our EqDBE scheme in standard model construction from decomposed LWE and PPS. Full construction is provided in Section 7.

2.4 Programmable Pseudorandom Set Sampler Constructions

We now provide two constructions of PPS scheme with varying efficiencies. The first is from standard LWE and the second is from a new falsifiable assumption that we call 1-key LWE. Full definition of PPS is provided in Section 6. A sanitized version is provided in Section 2.3.

PPS from LWE. Our construction for message space $\mathcal{M} = \mathbb{Z}_2$ is as follows:

PPGen : Sample a matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$. Output $\mathsf{crs} := (N, \mathbf{A})$ along with other LWE parameters.

MSKGen : For each $i \in [N]$ sample $\mathbf{s}_{i,0}, \mathbf{s}_{i,1} \leftarrow \mathbb{Z}_q^n$. Output $\mathsf{msk} := (\mathbf{s}_{i,b})_{i \in [N], b \in \{0,1\}}$.

$\mathsf{KGen}(S)$: Let $\mathbf{r} = (r_1, \ldots, r_N) \in \{0,1\}^N$ denote the characteristic vector of S, i.e., $r_i = 1$ if and only $i \in S$. Output $\mathsf{sk}_S := \sum_{i \in [N]} \mathbf{s}_{i,r_i} \in \mathbb{Z}_q^n$.

$\mathsf{skEnc}(i, \mu)$: Sample $\mathbf{x} \leftarrow \{\pm 1\}^m$ and set $\mathbf{c}_0 := \mathbf{A}\mathbf{x}$. Also sample $\{z_j\}_{j \in [N]}$ such that $\sum_{j \in [N]} z_j = 0$. For each $j \in [N], b \in \{0,1\}$,

$$c_{j,b} = \begin{cases} (\mathbf{s}_{j,b}^\top \mathbf{A} + \mathsf{err})\mathbf{x} + z_j & \text{if } j \neq i \\ (\mathbf{s}_{i,1}^\top \mathbf{A} + \mathsf{err})\mathbf{x} + z_i + t & \text{if } j = i, b = 1 \end{cases} \quad \text{and } c_{i,0} \leftarrow \mathbb{Z}_q$$

Output $\mathsf{ct}_i := (\mathbf{c}_0, (c_{j,b})_{j \in [N], b \in \{0,1\}}, \lceil t \rfloor_2 + \mu) \in \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$.

$\mathsf{pkEnc}(i)$: Sample $\mathsf{ct}_i \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$.

$\mathsf{Decode}(i, \mathsf{ct}, S, \mathsf{sk}_S)$: Parse ct as $(\mathbf{c}_0, (c_{j,b})_{j \in [N], b \in \{0,1\}}, c_T)$. If $i \notin S$, output $\perp$. Otherwise, using characteristic vector $(r_1, \ldots, r_N)$ for S do the following.

Compute $c' = \left(\sum_{j \in [N]} c_{j,r_j}\right) - \mathsf{sk}_S^\top \mathbf{c}_0$. Then, output $c_T - \lceil c' \rfloor_2$.

Note that compactness is immediate as $\mathsf{sk}_S \in \mathbb{Z}_q^n$ which is independent of number of users N. Correctness is argued as follows: as $c_{j,r_j} = (\mathbf{s}_{j,r_j}^\top \mathbf{A} + \mathsf{err})\mathbf{x} + z_j + \delta_{ji} t$[5], if $i \in S, r_i = 1$, then $c_{i,1}$

[5] δ_{ji} denotes the Kronecker delta function. That is, $\delta_{ji} = 1$ if and only if $j = i$.

contains t. So, $\sum_{j\in[N]} c_{j,r_j} = \left(\sum_{j\in[N]} \mathbf{s}_{j,r_j}^\top \mathbf{A} + \mathsf{err}\right)\mathbf{x} + t$ as $\sum_{j\in[N]} z_j = 0$. Hence, $c' \approx t$ and thus $c_T - \lceil c' \rfloor_2 \approx \mu$.

Security Analysis. We prove that above design for PPS satisfies payload hiding and mode indistinguishability. Essentially we show that for an index $i \in [N]$, $\mathsf{skEnc}(i, \mu)$ is indistinguishable from a random string of $\mathbb{Z}_q$, $\mathbb{Z}_2$ elements even when an attacker $\mathcal{A}$ is given a secret key sk_S for $i \in S$ and $i \notin S$.

Payload Hiding. In this game, attacker queries an encryption for $(i, \mu) \in [N] \times \mathbb{Z}_2$ and later adaptively queries for a secret key for S such that $i \notin S$. Note that as $i \notin S$, $\mathbf{s}_{i,1}$ is not used anywhere in the game. Particularly, skEnc queries for other indices $k \neq i$ can be constructed given $\mathbf{s}_{i,1}^\top \mathbf{A} + \mathsf{err}$ as opposed to $\mathbf{s}_{i,1}$. Thus, we can use LWE and leftover hash lemma to switch $c_{i,1}$ in the challenge ciphertext to a uniformly random element in $\mathbb{Z}_q$. As a consequence, both z_i and t are hidden from $\mathcal{A}$'s view. Hence, we can set c_T in the challenge ciphertext for index i to be a uniformly random element of $\mathbb{Z}_2$ and also compute $\{z_j\}_{j\neq i}$ as uniformly random elements from $\mathbb{Z}_q$.

Next, we define N hybrids, where in each Hybrid j, we would like to set $c_{\leq j,0}$ and $c_{\leq j,1}$ as random elements in $\mathbb{Z}_q$. However, there is an issue with this. Both $c_{j,0}$ and $c_{j,1}$ share a common random element z_j. To resolve this, let us guess if $j \in S$ for the single key query S. If yes: we can use LWE security and leftover hash lemma to set $c_{j,0}$ to be a uniformly random element in $\mathbb{Z}_q$ as $\mathbf{s}_{j,0}$ is revealed as part of key query (we can use $\mathbf{s}_{j,0}^\top \mathbf{A} + \mathsf{err}$ to construct other skEnc ciphertexts). Now, z_j only appears in $c_{j,1}$ and is a uniformly random element form $\mathbb{Z}_q$. This allows us to set $c_{j,1}$ as a uniformly random element and move to the next hybrid. If no: we proceed in a flipped manner, now $\mathbf{s}_{j,1}$ is not used anywhere which lets us set $c_{j,1}$ as a random element using LWE security and leftover hash lemma. Then as z_j only appears in $c_{j,0}$, we can set this as a random element as well. As both paths result in the same hybrid, we can use the piecewise guess framework [JKK+17, KW20] (similar to proof of EqDBE from PPS) to argue that $\mathsf{skEnc}(i, \mu)$ is indistinguishable from a uniformly random string (also check Appendix H). Thus, payload hiding holds.

Mode Indistinguishability. In this game, attacker queries an encryption for $i \in [N]$ and later adaptively queries for a secret key for S such that $i \notin S$. More importantly $\mu \leftarrow \mathbb{Z}_2$ is chosen by the challenger. Thus, we can readily set c_T to be a random element from $\mathbb{Z}_2$ in the challenge ciphertext. As a consequence, t only appears in $c_{i,1}$ of the challenge ciphertext and thus, $c_{i,1}$ can be sampled as a random element in $\mathbb{Z}_q$. Now, we can do the same piecewise guessing argument that we did in payload hiding analysis and set the entire challenge ciphertext to be a random string of $\mathbb{Z}_q$ and $\mathbb{Z}_2$ elements.

We remark that it is important to set $c_{i,0}$ in $\mathsf{skEnc}(i, \mu)$ to be a random element in the construction. For instance, if we set $c_{i,0}$ as a structured element akin to $(\mathbf{s}_{i,0}^\top \mathbf{A} + \mathsf{err})\mathbf{x} + z_i$, then we cannot argue security. Specifically, $\mathcal{A}$ in possession of sk_S for $i \notin S$ can compute $\sum_{j\neq i} c_{j,r_j} + c_{i,0} - \mathsf{sk}_S^\top \mathbf{c}_0$ which will be a low norm element in $\mathbb{Z}_q$ and thus distinguishable from a uniformly random element. This concludes the description of our PPS construction from LWE.

PPS from 1-key LWE. Note that the definition of our PPS scheme is very much close to a private-key ABE scheme [SW05, GPSW06] with pseudorandom ciphertexts and compact secret keys. The celebrated work of Boneh et al. [BGG+14] does provide the same. So, it looks straight-forward to use [BGG+14] to construct a PPS scheme. Using this insight, we give a construction of PPS for message space $\mathbb{Z}_2$. One benefit is that $|\mathsf{ct}|$ in this scheme is virtually independent of N.

PPGen : Simply output LWE parameters.

MSKGen : Sample $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ with a trapdoor $\mathbf{T_A}$, $\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times \lceil \log N \rceil m}, \mathbf{d} \leftarrow \mathbb{Z}_q^n$.

$\mathsf{KGen}(S)$: Let mem-S denote the membership verification circuit for S, i.e., mem-$S(i) = 0$ if and only if $i \in S$[6]. Sample $\mathbf{r}_S \leftarrow [\mathbf{A} \mid \mathbf{B}_S]^{-1}(\mathbf{d})$ where $\mathbf{B}_S = \mathsf{EvalF}(\mathbf{B}, \mathsf{mem}\text{-}S)$. Output $\mathsf{sk}_S := (\mathbf{r}_S, \mathbf{B}) \in \mathbb{Z}_q^{2m} \times \mathbb{Z}_q^{n \times \lceil \log N \rceil m}$.

$\mathsf{skEnc}(i, \mu)$: Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n$. Output $\mathsf{ct}_i := (\mathbf{s}^\top \mathbf{A} + \mathsf{err}, \mathbf{s}^\top (\mathbf{B} + \mathbf{i}^\top \otimes \mathbf{G}) + \mathsf{err}, \lceil \mathbf{s}^\top \mathbf{d} \rfloor_2 + \mu)$ where $\mathbf{i} \in \{0,1\}^{\lceil \log N \rceil}$ is the bit-vector representation of i and $\mathbf{G}$ is the gadget matrix [MP12].

$\mathsf{pkEnc}(i)$: Sample $\mathsf{ct}_i \leftarrow \mathbb{Z}_q^{1 \times (1 + \lceil \log N \rceil)m} \times \mathbb{Z}_2$.

$\mathsf{Decode}(i, \mathsf{ct}, S, \mathsf{sk}_S)$: Parse ct as $(\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_T)$. If $i \notin S$, output $\perp$. Otherwise, compute $c' = [\mathbf{c}_1^\top \mid \mathbf{c}_2^\top \mathbf{H}_{S,\mathbf{i}}] \mathbf{r}_S$ where $\mathbf{H}_{S,\mathbf{i}} = \mathsf{EvalFX}(\mathbf{B}, \mathsf{mem}\text{-}S, \mathbf{i})$. Then, output $c_T - \lceil c' \rfloor_2$.

Here, $\mathsf{EvalF}, \mathsf{EvalFX}$ are the lattice matrix homomorphic computation functions [GSW13, BGG+14] (check Appendix A for more details). Compactness is immediate as sk_S only grows with $\log N$. Correctness follows as $\mathbf{c}_2^\top \mathbf{H}_{S,\mathbf{i}} \approx \mathbf{s}^\top (\mathbf{B} + \mathbf{i}^\top \otimes \mathbf{G}) \mathbf{H}_{S,\mathbf{i}} = \mathbf{s}^\top \mathbf{B}_S$. Thus, $c' \approx \mathbf{s}^\top \mathbf{d} \implies c_T - \lceil c' \rfloor_2 \approx \mu$. In fact, we can also argue payload hiding readily. If $i \notin S$, the challenge ciphertext can be set to a random string by hardness of LWE and leftover hash lemma similar to the selective security argument of [BGG+14]. All that is left is to argue mode indistinguishability. Before that, we describe our falsifiable lattice-based assumption that we call '1-key LWE'.

1-key LWE. We describe this assumption using the following game:

1. $\mathcal{A}$ sends $i \in [N]$. Challenger samples $\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times \lceil \log N \rceil m}, \mathbf{d} \leftarrow \mathbb{Z}_q^n$, $\mathbf{A}$ sampled with trapdoor $\mathbf{T_A}$, and samples a bit $\beta \leftarrow \{0,1\}$.
   If $\beta = 0$, challenger sets $\mathbf{c}_1^\top := \mathbf{s}^\top \mathbf{A} + \mathsf{err}, \mathbf{c}_2^\top := \mathbf{s}^\top (\mathbf{B} + \mathbf{i}^\top \otimes \mathbf{G}) + \mathsf{err}$ for $\mathbf{s} \leftarrow \mathbb{Z}_q^n$. Otherwise, if $\beta = 1$, challenger samples $\mathbf{c}_1 \leftarrow \mathbb{Z}_q^m, \mathbf{c}_2 \leftarrow \mathbb{Z}_q^{\lceil \log N \rceil m}$.
   Challenger sends $(\mathbf{A}, \mathbf{B}, \mathbf{d}, \mathbf{c}_1^\top, \mathbf{c}_2^\top)$ to $\mathcal{A}$.
2. $\mathcal{A}$ sends $S \subseteq [N]$ and receives $\mathbf{r}_S \in \mathbb{Z}_q^{2m}$ such that $[\mathbf{A} | \mathbf{B}_S] \mathbf{r}_S = \mathbf{d}$. $\mathcal{A}$ declares its guess β'.

We say the assumption holds if $|\Pr[\beta' = \beta] - 1/2| \leq \mathsf{negl}(\lambda)$. Clearly, this is a falsifiable assumption. Also, observe that if $i \notin S$, this assumption follows from LWE. Intuitively, hardness of this assumption holds as the most an attacker can do is take the vectors $\mathbf{s}^\top \mathbf{A} + \mathsf{err}, \mathbf{s}^\top (\mathbf{B} + \mathbf{i}^\top \otimes \mathbf{G}) + \mathsf{err}, \mathbf{r}_S$ and get a new LWE sample for $\mathbf{d}$[7]. This does not reveal too much information about the prior vectors as this new LWE sample itself is pseudorandom. We provide a more detailed preliminary cryptanalysis of this assumption in Appendix B.

Mode Indistinguishability from 1-key LWE. We need to show that $\mathsf{skEnc}(i, \mu)$ is a pseudorandom string even if $\mathcal{A}$ *adaptively* queries a secret key for set S such that $i \in S$ and $\mu \leftarrow \mathbb{Z}_2$ sampled by challenger. $\mathsf{skEnc}(i, \mu)$ has the form $(\mathbf{s}^\top \mathbf{A} + \mathsf{err}, \mathbf{s}^\top (\mathbf{B} + \mathbf{i}^\top \otimes \mathbf{G}) + \mathsf{err}, \lceil \mathbf{s}^\top \mathbf{d} \rfloor_2 + \mu)$. Thus, we can use the hardness 1-key LWE assumption to set the first two components as uniformly random strings. Any skEnc queries for indices $k \neq i$ can be constructed using $\mathbf{A}, \mathbf{B}, \mathbf{d}$ provided by the 1-key LWE challenger. Then, as $\mu \leftarrow \mathbb{Z}_2$ is unknown to the $\mathcal{A}$, we can also set the third component as a uniformly random element from $\mathbb{Z}_2$.

Mode Indistinguishability from LWE? We remark that mode indistinguishability of above design might not be possible from standard LWE. If $\mathcal{A}$ *selectively* declares set S, then we can sample $\mathbf{r}_S$ from appropriate gaussian distribution and set $\mathbf{d} = [\mathbf{A} \mid \mathbf{B}_S] \mathbf{r}_S$. Even in this case, $\mathbf{d}$ is close to random by pre-image distribution properties of lattice trapdoors [GPV08, ABB10,

[6] Following the convention for lattices, 0 denotes accept.

[7] If challenger also provides $c_3 = \mathbf{s}^\top \mathbf{d} + \mathsf{err}$, $\mathcal{A}$ can trivially win. It is important that this is not provided.

CHKP10, MP12]. However, if $\mathcal{A}$ adaptively declares S such that $i \in S$, without a trapdoor for $\mathbf{A}$, it doesn't seem possible to sample $\mathbf{r}_S$ such that $[\mathbf{A} \mid \mathbf{B}_S]\mathbf{r}_S = \mathbf{d}$ for any S. Thus, we rely on 1-key LWE assumption for mode indistinguishability. We leave it as an interesting open problem to prove mode indistinguishability of above design from standard LWE.

This concludes the description of our PPS constructions. Full construction of PPS from LWE is provided in Section 8 and PPS from 1-key LWE is provided in Section 9.

Further Extensions. We conclude with a few extensions possible for our equivocal framework in the context of DBE. One can alternatively define a strongly-secure EqDBE scheme which readily yields an adaptively secure DBE. This way we can skip the $2\times$ ciphertext blowup incurred by the semi-static to adaptively secure transformations of [GW09, KMW23, HWW25]. Alternatively, we can also consider the following: our EqDBE scheme can equivocate at most one ciphertext. One can define an EqDBE scheme with a $\mathsf{FakeSetup}$ algorithm that can produce a fake ciphertext for two disjoint sets and size of the fake ciphertext is less than two ciphertexts of regular EqDBE combined. This way we can equivocate two subsets and rely on the two-key transformation of [GW09, KMW23] to construct an adaptively secure DBE scheme with even shorter ciphertexts.

2.5 Related, Concurrent, and Subsequent Works

Related work for PPS. At a high level, our PPS abstraction can be viewed as a one-key, secret-key analogue to aggregate IBE [GV22] and its derivatives including batched identity-based encryption [CGPP24, AFP25, BLT25]. However, the core distinguishing feature of PPS is its mode indistinguishability property, which ensures that a ciphertext remains pseudorandom even to an evaluator holding an accepting secret key as long as the encrypted message is sampled randomly. In this regard, mode indistinguishability conceptually mirrors the pseudorandomness guarantees of projective PRGs [ABI+23, HWW25]. Our PPS primitive fundamentally diverges from projective PRGs due to its requirement of simultaneously achieving both mode indistinguishability and payload hiding.

Syntactically, our PPS framework also shares similarities with the mixed FE scheme introduced in [GKW18]. Despite these parallels, the underlying requirements are different. Rather than demanding the full expressiveness of FE, PPS is much closer in scope to a mixed attribute-based encryption scheme for inner product predicates. This targeted functionality allows us to instantiate PPS efficiently from standard LWE.

Concurrent Work. Independently and concurrently, Abram, Malavolta, and Roy [AMR26a, ePrint 2026/741] presented an adaptively secure DBE scheme in the standard model whose public-key, secret-key, and ciphertext sizes are independent of N and whose CRS size grows proportional to N, matching Corollary 9.6 from decomposed LWE and 1-key LWE. They also claimed a semi-statically secure *flexible* BE scheme (FBE) [FWW23] in the standard model with optimal parameters. However, that version contained an error in its parameter analysis. Its FBE CRS was not compact, and the claimed linear CRS bound for its standard-model DBE did not hold.

Subsequent Work. In a follow-up work, Goyal and Yadugiri [GY26] construct an adaptively secure FBE and identity-based BE scheme in the random oracle model with optimal parameters from decomposed LWE. The latest revision of Abram, Malavolta, and Roy [AMR26b] repairs their earlier result using trapdoor hashing [DGI+19] and also obtains a FBE scheme with semi-static security in the standard model with optimal parameters from 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 $\lambda \in \mathbb{N}$, $\sum_{\alpha \in \Omega} |\Pr[\alpha \leftarrow \mathcal{X}_\lambda] - \Pr[\alpha \leftarrow \mathcal{Y}_\lambda]| \leq \mathsf{negl}(\lambda)$.

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 sometimes alternately write $\mathbf{x}[i] := x_i$ and $\mathbf{A}[i, j] := a_{i,j}$. 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 $\mathbf{A}, \mathbf{B}$. The rounding function $\lceil \cdot \rfloor_p : \mathbb{Z}_q \to \mathbb{Z}_p$ is given by $\lceil x \rfloor_p = \lceil \frac{p}{q} x \rfloor \in \mathbb{Z}_p$. That is, if $x = y \cdot q/p + z$ where $z \in [-q/2p, q/2p)$, we have $\lceil x \rfloor_p = y$.

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 $\mathbf{G} = (\mathbf{I}_n \otimes \mathbf{g}^\top)$, $\mathbf{g} = [1\ 2\ 2^2\ \ldots\ 2^{\lfloor \log q \rfloor}\ 0\ \ldots\ 0]^\top \in \mathbb{Z}_q^m$.

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 $\sigma = \omega(\sqrt{\log m})$.

Assumption 3.3 ([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 Assumption

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

Assumption 3.4 ([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 provide the following lemmas which show that leftover hash lemma holds even with respect to $\mathbf{B}$.

Lemma 3.5. For positive integers n, m, and odd prime q, such that $m = \Omega(n\log q)$, for any $\ell = \mathsf{poly}(\lambda)$ $\mathbf{W} \leftarrow \mathbb{Z}_q^{n\times\ell m}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times\ell m}, \mathbf{B} = \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + (\mathsf{vec}(\mathbf{I}_\ell)^\top \otimes \mathbf{G})$, $\{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{BR})\} \approx_s \{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A})\}$ where $\mathbf{R} \leftarrow \{\pm 1\}^{\ell^2 m\times L}, \sigma = \omega(\sqrt{\log m}), \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times L}$.

Proof. It suffices to prove that the distribution of $\mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D})\mathbf{R}$ is statistically close to uniformly random distribution given $\mathbf{W}, \mathbf{D}$. It is known that linear combination of independent discrete Gaussian samples over a lattice are statistically close to a discrete Gaussian with appropriately increased width, under standard smoothing conditions [Pei10, AGHS13, AR13]. Thus, $(\mathbf{I}_\ell \otimes \mathbf{D})\mathbf{R}$ is a convolution of discrete Gaussian samples and as such is a sufficiently wide Gaussian. As $\mathbf{W} \leftarrow \mathbb{Z}_q^{n\times m}$ is a universal hash function, this follows from Lemma 3.2, Item 3. □

Lemma 3.6. For positive integers n, m, and odd prime q, such that $m = \Omega(n\log q)$, for any $\ell = \mathsf{poly}(\lambda)$ $\mathbf{W} \leftarrow \mathbb{Z}_q^{n\times\ell m}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times\ell m}, \mathbf{B} = \mathbf{W}(\mathbf{I}_\ell \otimes \mathbf{D}) + (\mathsf{vec}(\mathbf{I}_\ell)^\top \otimes \mathbf{G})$, $\{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{BR}, \mathbf{e}^\top\mathbf{R})\} \approx_s \{(\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{e}^\top\mathbf{R})\}$ where $\mathbf{R} \leftarrow \{\pm 1\}^{\ell^2 m\times L}, \sigma = \omega(\sqrt{\log m}), \mathbf{A} \leftarrow \mathbb{Z}_q^{n\times L}$, $\mathbf{e}^\top \in \mathbb{Z}_q^{\ell^2 m}$ is any fixed vector.

Proof. This follows by a similar argument as Lemma 3.5 and [ABB10, Lemma 13]. □

3.3 Matrix Commitment Scheme

We provide the definition of a matrix commitment scheme (MCOM) from decomposed LWE adapted from [Wee25, Appendix C].

Syntax. An MCOM scheme 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 3.7 (MCOM)**.** 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 $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$, $\|\mathbf{V}_N\| \le O(\|\mathbf{D}\| m^4 \log q)$ and $\|\mathbf{Z}\| \le O(\|\mathbf{D}\| m^7 \log q \log N)$ where $\mathbf{V}_N = \mathsf{Verify}(1^N)$ and $\mathbf{Z} = \mathsf{Open}(\mathbf{M})$.

3.4 Distributed Broadcast Encryption

Syntax. A distributed broadcast encryption scheme (DBE) consists of the following polynomial-time algorithms.

$\mathsf{GSetup}(1^\lambda, 1^N) \to \mathsf{crs}$**.** The possibly randomized global setup algorithm takes as input number of users N and outputs common reference string crs. The following algorithms take crs implicitly.

$\mathsf{KGen}(i) \to (\mathsf{pk}_i, \mathsf{sk}_i)$**.** The randomized key generation algorithm takes as input index $i \in [N]$ and outputs public-secret key pair $(\mathsf{pk}_i, \mathsf{sk}_i)$.

$\mathsf{Enc}(\{\mathsf{pk}_i\}_{i \in S}, \mu) \to \mathsf{ct}$**.** The randomized encryption algorithm takes as input public keys for users in set S, message $\mu \in \{0, 1\}$, and outputs a ciphertext ct.

$\mathsf{Dec}(\{\mathsf{pk}_i\}_{i \in S}, (j, \mathsf{sk}_j), \mathsf{ct}) \to \mu'$ The deterministic decryption algorithm takes as input public keys for users in set S, secret key for user j, sk_j, ciphertext ct, and outputs $\mu' \in \{0, 1\} \cup \{\perp\}$.

Definition 3.8. A DBE scheme $(\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is said to be a DBE scheme if it satisfies the following properties.

Correctness. For any $\lambda \in \mathbb{N}$, positive polynomial $N \in \mathbb{N}$, $S \subseteq [N]$, $j \in S, \mu \in \{0,1\}$,

$$\Pr\left[\mu = \mathsf{Dec}(\{\mathsf{pk}_i\}_{i\in S}, (j, \mathsf{sk}_j), \mathsf{ct}) \quad : \quad \begin{array}{l} \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^N), \\ \forall\ i \in S, (\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KGen}(i), \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\{\mathsf{pk}_i\}_{i\in S}, \mu) \end{array}\right] = 1$$

Compactness. There exists three polynomials $\mathsf{poly}_{\mathsf{pk}}(\cdot,\cdot), \mathsf{poly}_{\mathsf{sk}}(\cdot,\cdot), \mathsf{poly}_{\mathsf{ct}}(\cdot,\cdot)$ such that for any $\lambda, N \in \mathbb{N}, S \subseteq [N], \forall\, i \in [N], |\mathsf{pk}_i| = \mathsf{poly}_{\mathsf{pk}}(\lambda, \log N), |\mathsf{sk}_i| = \mathsf{poly}_{\mathsf{sk}}(\lambda, \log N), |\mathsf{ct}| = \mathsf{poly}_{\mathsf{ct}}(\lambda, \log N)$.

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

$$\left|\Pr\left[b \leftarrow \mathcal{A}(\mathsf{ct}) \quad : \quad \begin{array}{l} 1^N \leftarrow \mathcal{A}(1^\lambda), \\ \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^N), \\ S \leftarrow \mathcal{A}^{\mathcal{O}(\cdot)}(\mathsf{crs}), \\ b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}(\{\mathsf{pk}_i\}_{i\in S}, b) \end{array}\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)$$

where $\mathcal{O}$ is a stateful oracle that responds to the following queries.

- (KGen, i). $\mathcal{A}$ sends index $i \in [N]$. $\mathcal{O}$ checks if i was already queried for KGen. If so, it returns $\perp$. Otherwise, $\mathcal{O}$ performs $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KGen}(i)$ and sends pk_i to $\mathcal{A}$.
- (Corr, i) $\mathcal{A}$ sends index i. $\mathcal{O}$ checks if i was already queried for KGen. If so, it responds to $\mathcal{A}$ with sk_i. Otherwise, it outputs $\perp$.

A stateful PPT machine $\mathcal{A}$ is said to be admissible if $S \subseteq [N]$ and $\forall\ i^* \in S$, $\mathcal{A}$ made (KGen, i^*) query but not (Corr, i^*) query.

Definition 3.9 (Semi-Static Security)**.** A DBE scheme $(\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ is said to be semi-statically secure if satisfies correctness and compactness of Definition 3.8 and semi-static security which is defined as follows.

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

$$\left|\Pr\left[b \leftarrow \mathcal{A}(\mathsf{ct}) \quad : \quad \begin{array}{l} (1^N, S') \leftarrow \mathcal{A}(1^\lambda), \\ \mathsf{crs} \leftarrow \mathsf{GSetup}(1^\lambda, 1^N), \\ \forall\ i \in S', (\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KGen}(i), \\ S^* \leftarrow \mathcal{A}(\mathsf{crs}, \{\mathsf{pk}_i\}_{i\in S'}) \\ b \leftarrow \{0,1\}, \mathsf{ct} \leftarrow \mathsf{Enc}(\{\mathsf{pk}_i\}_{i\in S^*}, b) \end{array}\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)$$

A stateful PPT machine $\mathcal{A}$ is said to be admissible if $S^* \subseteq S' \subseteq [N]$.

Theorem 3.10 ([GW09, KMW23])**.** Assuming the existence of a semi-statically secure DBE scheme, there exists an adaptively-secure DBE scheme in the random oracle model.

Theorem 3.11 ([HWW25])**.** Assuming the existence of a semi-statically secure DBE scheme and hardness of LWE with subexponential modulus-to-noise ratio, there exists an adaptively-secure DBE scheme in the standard model.

Proof. This follows from [HWW25, Appendix A]. However, naïvely using this semi-static to adaptive transformation would result in a biased crs for the resultant adaptive DBE scheme. The PSPRG construction from LWE of [HWW25, Section 6.3] has pseudorandom pp (via [BV15], [HWW25, Theorem 6.20]). Thus, we can use a random string of sufficient length in the construction to get a transparent crs. □

4 Equivocal DBE: A New Methodology for Adaptive Security

In this section, we introduce our notion of equivocal DBE scheme (EqDBE) and show that this generically implies semi-statically secure DBE. In turn this yields an adaptively secure DBE in the random oracle model [GW09, KMW23] with optimal parameters or in the standard model in conjunction with the hardness of $\mathsf{LWE}/\mathsf{CDH}/\mathsf{CBDH}/\mathsf{RSA}$ assumptions.

Syntax. An EqDBE consists of the following polynomial-time algorithms.

$\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}$**.** These algorithms are defined similar to a DBE scheme (Section 3.4).

$\mathsf{FakeSetup}(1^\lambda, 1^N, S', \mu) \to (\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$**.** The randomized fake setup algorithm takes as input number of users N, set $S' \subseteq [N]$, message $\mu \in \{0,1\}$, outputs fake CRS, $\widetilde{\mathsf{crs}}$, fake public keys for all users in $i \in S'$, $\{\widetilde{\mathsf{pk}}_i\}_{i\in S'}$, fake ciphertext $\widetilde{\mathsf{ct}}$, and auxiliary information aux.

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

Definition 4.1. An EqDBE scheme $(\mathsf{GSetup}, \mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FakeSetup}, \mathsf{Equivocate})$ 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 a DBE scheme (Definition 3.8).

Mode Indistinguishability. For any PPT admissible adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $\forall\ \lambda \in N$,

$$\left|\Pr\left[\, b \leftarrow \mathcal{A}(\mathsf{ct}^{(b)}) \;:\; \begin{array}{l} b \leftarrow \{0,1\}, (1^N, S', \mu) \leftarrow \mathcal{A}(1^\lambda), \\ \text{if } b = 0, \mathsf{crs}^{(0)} \leftarrow \mathsf{GSetup}(1^\lambda, 1^N), \\ \forall\ i \in S', \{\mathsf{pk}_i^{(0)}, \mathsf{sk}_i^{(0)}\} \leftarrow \mathsf{KGen}(i), \\ \text{if } b = 1, (\mathsf{crs}^{(1)}, \{\mathsf{pk}_i^{(1)}\}_{i\in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux}) \leftarrow \\ \mathsf{FakeSetup}(1^\lambda, 1^N, S', \mu) \\ S^* \leftarrow \mathcal{A}(\mathsf{crs}^{(b)}, \{\mathsf{pk}_i^{(b)}\}_{i\in S'}), \\ \text{if } b = 0, \mathsf{ct}^{(0)} \leftarrow \mathsf{Enc}(\{\mathsf{pk}_i^{(0)}\}_{i\in S^*}, \mu), \\ \text{if } b = 1, \mathsf{ct}^{(1)} \leftarrow \mathsf{Equivocate}(\mathsf{aux}, S^*) \end{array} \right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)$$

A stateful PPT machine $\mathcal{A}$ is said to be admissible if $S^* \subseteq S' \subseteq [N]$.

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

$$\left|\Pr\left[\, b \leftarrow \mathcal{A}(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux}) \;:\; \begin{array}{l} (1^N, S') \leftarrow \mathcal{A}(1^\lambda), b \leftarrow \{0,1\}, \\ (\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux}) \leftarrow \\ \mathsf{FakeSetup}(1^\lambda, 1^N, S', b) \end{array} \right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda)$$

4.1 Semi-Statically Secure DBE from Equivocal DBE

Construction 4.2. We provide the construction of a DBE scheme that is semi-statically secure (Definition 3.9) from EqDBE (Definition 4.1) as follows.

$\mathsf{GSetup}(1^\lambda, 1^N)$: Sample $\mathsf{dbe.crs} \leftarrow \mathsf{EqDBE.GSetup}(1^\lambda, 1^N)$. Output $\mathsf{crs} := \mathsf{dbe.crs}$.

$\mathsf{KGen}(i)$: Sample $(\mathsf{dbe.pk}, \mathsf{dbe.sk}) \leftarrow \mathsf{EqDBE.KGen}(i)$. Output $(\mathsf{pk}, \mathsf{sk}) := (\mathsf{dbe.pk}, \mathsf{dbe.sk})$.

$\mathsf{Enc}(\{\mathsf{pk}_i\}_{i\in S}, \mu)$: Parse pk_i as $\mathsf{dbe.pk}_i$ for each $i \in S$ and sample $\mathsf{dbe.ct} \leftarrow \mathsf{EqDBE.Enc}(\{\mathsf{dbe.pk}_i\}_{i\in S}, \mu)$. Output $\mathsf{ct} := \mathsf{dbe.ct}$.

$\mathsf{Dec}(\{\mathsf{pk}_i\}_{i\in S}, (j, \mathsf{sk}_j), \mathsf{ct})$: Parse pk_i as $\mathsf{dbe.pk}_i$ for each $i \in S$, sk_j as $\mathsf{dbe.sk}_j$, and ct as $\mathsf{dbe.ct}$. Output $\mu' = \mathsf{EqDBE.Dec}(\{\mathsf{dbe.pk}\}_{i\in S}, (j, \mathsf{dbe.sk}_j), \mathsf{dbe.ct})$.

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

Theorem 4.3. If EqDBE is an equivocal DBE scheme (Definition 4.1), then Construction 4.2 is a semi-statically secure DBE scheme (Definition 3.9).

Proof Sketch. We prove this theorem using the following hybrids.

Hybrid 0. This is the experiment from Definition 3.9 with $b = 0$.

Hybrid 1. In this hybrid, we will use $\mathsf{EqDBE.FakeSetup}(1^\lambda, 1^N, S', 0)$ to generate $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ and send $\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i\in S'}$ to $\mathcal{A}$. In addition, we will generate ct using $\mathsf{EqDBE.Equivocate}$ algorithm once $\mathcal{A}$ sends S^*. Hybrids 0 and 1 are indistinguishable because of the mode indistinguishability property of EqDBE.

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

Hybrid 3. This is the experiment from Definition 3.9 with $b = 1$. Indistinguishability follows by mode indistinguishability property of EqDBE.

Full proof is provided in Appendix C. □

5 Equivocal DBE in Random Oracle Model

Parameters. We describe the LWE parameters that our construction uses for $N = N(\lambda)$ users.

- $n = \mathsf{poly}(\lambda)$ (Decomposed LWE security)
- $m = \Omega(n \log q)$ (Leftover hash lemma, Lemma 3.2)
- $\chi = \mathsf{poly}(\lambda)$ (Decomposed LWE security)
- $\sigma = \mathsf{poly}(\lambda, m)$ (Decomposed LWE security)
- $q > O(\sigma\chi m^{13} \log q \log N)$ (Correctness)

Construction 5.1. We provide the construction of an EqDBE scheme assuming the hardness of $2m^2$-decomposed LWE in the random oracle model. We will use the matrix commitment scheme MCOM from Section 3.3.

$\mathsf{GSetup}(1^\lambda, 1^N)$: Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$. Sample hash functions $\mathcal{H}_1 : \{0,1\}^\lambda \times [N] \to \{\pm 1\}^{4m^5}$ and $\mathcal{H}_2 : \{0,1\}^\lambda \times [N] \to \mathbb{Z}_q^n$. Output $\mathsf{crs} := (N, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, \mathcal{H}_1, \mathcal{H}_2)$.

$\mathsf{KGen}(i)$: Compute $\mathbf{V}_N = \mathsf{MCOM.Verify}(1^N)$. Let $\mathbf{v}_i$ denote the i-th column of $\mathbf{V}_N$. Sample $\mathbf{r} \leftarrow \{\pm 1\}^{4m^5}$. Compute $\mathbf{t}_i = \mathbf{Br} + \mathbf{p} - \mathbf{Av}_i \in \mathbb{Z}_q^n$. Output $\mathsf{pk}_i := (i, \mathbf{t})$ and $\mathsf{sk}_i := \mathbf{r}$.

$\mathsf{Enc}(\{\mathsf{pk}_i\}_{i \in S}, \mu)$: Parse pk_i as $(i, \mathbf{t}_i)$ for each $i \in S$.

1. Sample $\mathsf{sd}_1, \mathsf{sd}_2 \leftarrow \{0,1\}^\lambda$.
2. For each $j \in [N]$, if $j \in S$, let $\mathbf{x}_j = \mathcal{H}_1(\mathsf{sd}_1, j)$ and if $j \notin S$, $\mathbf{d}_j = \mathcal{H}_2(\mathsf{sd}_2, j)$.
3. Construct an $n \times N$ matrix $\mathbf{M}$ where the j-th column $\mathbf{m}_j$ is as follows

$$\mathbf{m}_j = \begin{cases} \mathbf{t}_j + \mathbf{B}\mathbf{x}_j & \text{if } j \in S \\ \mathbf{d}_j & \text{if } j \notin S \end{cases}$$

4. Compute $\mathbf{C} = \mathsf{MCOM.Com}(\mathbf{M})$.
5. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R_A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$.
6. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top(\mathbf{A} + \mathbf{C}) + \mathbf{e}^\top \mathbf{R_A}, c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu$.

Output $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{sd}_1, \mathsf{sd}_2) \in \mathbb{Z}_q^{1 \times (4m^5 + m + 1)} \times \{0,1\}^{2\lambda}$.

$\mathsf{Dec}(\{\mathsf{pk}_i\}_{i \in S}, (j, \mathsf{sk}_j), \mathsf{ct})$: Parse pk_i as $(i, \mathbf{t}_i)$ for each $i \in S$, sk_j as $\mathbf{r}_j$, and ct as $(\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{sd}_1, \mathsf{sd}_2)$. If $j \notin S$, output $\perp$. Otherwise,

1. For each $k \in [N]$, if $k \in S$, let $\mathbf{x}_k = \mathcal{H}_1(\mathsf{sd}_1, k)$ and if $k \notin S$, $\mathbf{d}_k = \mathcal{H}_2(\mathsf{sd}_2, k)$.
2. Compute matrix $\mathbf{M}$ like in Enc. Compute $\mathbf{V}_N = \mathsf{MCOM.Verify}(1^N), \mathbf{Z} = \mathsf{MCOM.Open}(\mathbf{M})$.
3. Compute $\mathbf{c}_{exp}^\top = \mathbf{c}_2^\top \mathbf{V}_N + \mathbf{c}_1^\top \mathbf{Z} \in \mathbb{Z}_q^{1 \times N}$. Let c_j denote the j-th element of $\mathbf{c}_{exp}^\top$.
4. Compute $y = c_3 - c_j + \mathbf{c}_1^\top(\mathbf{x}_j + \mathbf{r}_j) \in \mathbb{Z}_q$.

Output 0 if $-q/4 < y < q/4$ and 1 otherwise.

Comparison to prior works. Before proceeding further, we want to highlight major differences of our constructions and prior works on DBE from decomposed LWE. Like in [CW24, CHW25, WW25][8], our KGen algorithm is also an Ajtai commitment to $\mathbf{p} - \mathbf{Av}_i$. However, the major difference is in our Enc and Dec algorithms. Specifically, in Items 1 and 2 of Enc algorithm, we commit a matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times N}$ that is not sparse and contains entries from random oracles $\mathcal{H}_1, \mathcal{H}_2$. In our Dec algorithm we expand the ciphertext to the entire matrix $\mathbf{M}$ and only use the j-th column (Item 3 in Dec). While this makes our DBE scheme bounded / non-flexible, we achieve adaptive security (via semi-static security in ROM).

Correctness. Note that if $j \notin S$, Dec outputs $\perp$. So, we will only focus on the case when $j \in S$. We have that pk_i for each $i \in S$ is $(i, \mathbf{t}_i) \in [N] \times \mathbb{Z}_q^n$ where $\mathbf{t}_i = \mathbf{Br}_i + \mathbf{p} - \mathbf{Av}_i$. Here, $\mathsf{sk}_i = \mathbf{r}_i \in \{\pm 1\}^{4m^5}$

[8] Although security of these constructions rely on ℓ-succinct LWE [Wee24], they may be modified to work with ℓ-decomposed LWE [AMR25] as shown in [WW25].

and $\mathbf{v}_i$ is the i-th column of $\mathsf{MCOM.Verify}(1^N)$. For some $\mathsf{sd}_1, \mathsf{sd}_2 \in \{0,1\}^\lambda$, in both Enc and Dec, $\mathbf{M} = [\mathbf{m}_1 \mid \ldots \mid \mathbf{m}_N] \in \mathbb{Z}_q^{n \times N}$ where $\mathbf{m}_k = \mathbf{t}_k + \mathbf{B}\mathcal{H}_1(\mathsf{sd}_1, k)$ if $k \in S$ and $\mathbf{m}_k = \mathcal{H}_2(\mathsf{sd}_2, k)$ if $k \notin S$. $\mathsf{ct} = (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{sd}_1, \mathsf{sd}_2) = \left(\mathbf{s}^\top\mathbf{B} + \mathbf{e}^\top, \mathbf{s}^\top(\mathbf{A}+\mathbf{C}) + \mathbf{e}^\top\mathbf{R}_\mathbf{A}, \mathbf{s}^\top\mathbf{p} + \mathbf{e}^\top\mathbf{r}_\mathbf{p} + \lfloor q/2 \rfloor \mu, \mathsf{sd}_1, \mathsf{sd}_2\right)$ for some $\mathbf{s} \in \mathbb{Z}_q^n, \mathbf{e} \in \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R}_\mathbf{A} \in \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_\mathbf{p} \in \{\pm 1\}^{4m^5}$. Let $\mathbf{Z} = \mathsf{MCOM.Open}(\mathbf{M})$ as computed in Dec. Then,

$$\begin{aligned}
\mathbf{c}_{exp}^\top &= \mathbf{c}_2^\top \mathbf{V}_N + \mathbf{c}_1^\top \mathbf{Z} \\
&= \mathbf{s}^\top (\mathbf{A}\mathbf{V}_N + \mathbf{C}\mathbf{V}_N + \mathbf{B}\mathbf{Z}) + \left(\mathbf{e}^\top\mathbf{Z} + \mathbf{e}^\top\mathbf{R}_\mathbf{A}\mathbf{V}_N\right) \\
&= \mathbf{s}^\top(\mathbf{M} + \mathbf{A}\mathbf{V}_N) + \left(\mathbf{e}^\top\mathbf{Z} + \mathbf{e}^\top\mathbf{R}_\mathbf{A}\mathbf{V}_N\right)
\end{aligned}$$

The last equality follows by correctness of MCOM. The j-the element in $\mathbf{c}_{exp}^\top$ is $c_j = \mathbf{s}^\top(\mathbf{A}\mathbf{v}_j + \mathbf{t}_j + \mathbf{B}\mathcal{H}_1(\mathsf{sd}_1, j)) + (\mathbf{e}^\top\mathbf{z}_j + \mathbf{e}^\top(\mathbf{R}_\mathbf{A}\mathbf{V}_N)_j) = \mathbf{s}^\top(\mathbf{p} + \mathbf{B}(\mathcal{H}_1(\mathsf{sd}_1, j) + \mathbf{r}_j)) + (\mathbf{e}^\top\mathbf{z}_j + \mathbf{e}^\top(\mathbf{R}_\mathbf{A}\mathbf{V}_N)_j)$. Here, $\mathbf{z}_j$ and $(\mathbf{R}_\mathbf{A}\mathbf{V}_N)_j$ are the j-th columns of $\mathbf{Z}$ and $\mathbf{R}_\mathbf{A}\mathbf{V}_N$ respectively. Thus,

$$\begin{aligned}
y &= c_3 - c_j + \mathbf{c}_1^\top(\mathcal{H}_1(\mathsf{sd}_1, j) + \mathbf{r}_j) \\
&= \mathbf{s}^\top(\mathbf{p} - \mathbf{p} - \mathbf{B}(\mathcal{H}_1(\mathsf{sd}_1, j) + \mathbf{r}_j) + \mathbf{B}(\mathcal{H}_1(\mathsf{sd}_1, j) + \mathbf{r}_j)) + \lfloor q/2 \rfloor \mu + \widetilde{e} \\
&= \lfloor q/2 \rfloor \mu + \widetilde{e}
\end{aligned}$$

Here, $\widetilde{e} = \mathbf{e}^\top(\mathbf{r}_\mathbf{p} - (\mathbf{z}_j + (\mathbf{R}_\mathbf{A}\mathbf{V}_N)_j) + (\mathcal{H}_1(\mathsf{sd}_1, j) + \mathbf{r}_j))$. We will now bound the absolute value of $\widetilde{e}$. With high probability, $\|\mathbf{e}\| \leq \sqrt{m}\chi$ and $\|\mathbf{D}\| \leq \sqrt{m}\sigma$. By compactness of MCOM, $\|\mathbf{z}_j\| = \|\mathbf{Z}\| \leq O(\sigma m^{15/2} \log q \log N)$ and $\|\mathbf{V}_N\| \leq O(\sigma m^{9/2} \log q)$. As $\|\mathbf{R}_\mathbf{A}\| = 1$, $\left\|(\mathbf{R}_\mathbf{A}\mathbf{V}_N)_j\right\| = \|\mathbf{R}_\mathbf{A}\mathbf{V}_N\| \leq O(\sigma m^{11/2} \log q)$. Similarly, $\|\mathbf{r}_\mathbf{p}\| = \|\mathbf{r}_j\| = \|\mathcal{H}_1(\mathsf{sd}_1, j)\| = 1$. Thus, $|\widetilde{e}| \leq O(\sigma\chi m^{13} \log q \log N)$. If $|\widetilde{e}| < q/4$, the most-significant bit of y reveals μ. Thus, for any sufficiently large prime $q > O(\sigma\chi m^{13} \log q \log N)$, correctness holds.

Compactness. As $\mathsf{pk}_i = (i, \mathbf{t}_i) \in [N] \times \mathbb{Z}_q^n$, $|\mathsf{pk}_i| = \lceil \log N \rceil + n\lceil \log q \rceil = \mathsf{poly}(\lambda, \log N)$. Similarly, $|\mathsf{sk}_i| = 4m^5\lceil \log q \rceil = \mathsf{poly}(n, \log q, \log N) = \mathsf{poly}(\lambda, \log N)$. As $\mathsf{ct} \in \mathbb{Z}_q^{1 \times (4m^5+m+1)} \times \{0,1\}^{2\lambda}, |\mathsf{ct}| = (4m^5 + m + 1)\lceil \log q \rceil + 2\lambda = \mathsf{poly}(\lambda, \log N)$. Thus, compactness holds. We also remark that $|\mathsf{crs}|$ is also $\mathsf{poly}(\lambda, \log N)$.

5.1 Fake Setup and Equivocation

Now, we describe our $\mathsf{FakeSetup}$ and $\mathsf{Equivocate}$ algorithms. These algorithms are defined with respect to a programmable random oracle. $\mathsf{FakeSetup}$ provides handles $\mathcal{H}_1, \mathcal{H}_2$. However, they are treated as a single programmable oracle H (by augmenting the input with an additional slot $\iota \in \{1,2\}$) by $\mathsf{Equivocate}$.

$\mathsf{FakeSetup}(1^\lambda, 1^N, S', \mu)$: Perform the following steps.

1. For each $i \in S' \subseteq [N]$, sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. Set $\widetilde{\mathsf{pk}_i} := (i, \mathbf{t}_i)$ for each $i \in S'$.
2. Sample $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$.
3. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}, \mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively.

4. Compute matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$. Otherwise $\mathbf{m}'_j = \mathbf{d}'_j$. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Sample $\mathbf{A}' \leftarrow \mathbb{Z}_q^{n \times m}$ and set $\mathbf{A} = \mathbf{A}' - \mathbf{C}'$.
5. Sample $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{crs} := (N, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, \mathcal{H}_1, \mathcal{H}_2)$ where $\mathcal{H}_1 : \{0,1\}^\lambda \times [N] \to \{\pm 1\}^{4m^5}$ and $\mathcal{H}_2 : \{0,1\}^\lambda \times [N] \to \mathbb{Z}_q^n$ are handles to random oracles.
6. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R}_\mathbf{A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top \mathbf{R}_\mathbf{A}, c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r}_\mathbf{p} + \lfloor q/2 \rfloor \mu$. Set $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$.
7. Set $\mathsf{aux} := (\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S'}, \widetilde{\mathsf{ct}}, \mathbf{X}', \mathbf{D}', S')$.

Output $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$.

$\mathsf{Equivocate}^{\mathsf{H}(\cdot)}(\mathsf{aux}, S^*)$: Parse aux as $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S'}, \widetilde{\mathsf{ct}}, \mathbf{X}', \mathbf{D}', S')$.

1. Sample $\mathsf{sd}_1, \mathsf{sd}_2 \leftarrow \{0,1\}^\lambda$. Given access to a programmable oracle H, if H contains entries for $(\mathsf{sd}_1, i, 1)$ or $(\mathsf{sd}_2, j, 2)$ for any $i, j \in [N]$, output μ. Otherwise, proceed as follows.
2. For each $j \in [N]$, if $j \in S^*$, set $\mathsf{H}[(\mathsf{sd}_1, j, 1)] = \mathbf{x}'_j$. If $j \notin S'$, set $\mathsf{H}[(\mathsf{sd}_2, j, 2)] = \mathbf{d}'_j$. Otherwise, if $j \in S' \setminus S^*$, set $\mathsf{H}[(\mathsf{sd}_1, j, 1)] = \mathbf{x}^*_j$ and $\mathsf{H}[(\mathsf{sd}_2, j, 2)] = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ where $\mathbf{x}^*_j \leftarrow \{\pm 1\}^{4m^5}$.
3. Parse $\widetilde{\mathsf{ct}}$ as $(\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$. Set $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c'_3, \mathsf{sd}_1, \mathsf{sd}_2)$.

Output ct.

5.2 Security Analysis

Theorem 5.2 (Mode Indistinguishability)**.** Construction 5.1 satisfies mode indistinguishability property against any admissible PPT adversary $\mathcal{A}$.

Proof Sketch. We define the following hybrids.

Hybrid 0. This is the game from Definition 4.1 with $b = 0$ in the random oracle model.

Hybrid 1. In this hybrid, challenger aborts if sd_1 and sd_2 sampled during encryption are already queried to the random oracle. If $\mathcal{A}$ is a PPT machine, then hybrids 0 and 1 are negligibly close.

Hybrid 2. In this hybrid, we will set $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$ for each $i \in S'$. Hybrids 1 and 2 are indistinguishable by leftover hash lemma.

Hybrid 3. In this hybrid, we sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}, \mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$ during setup phase. Later during encryption phase, we will set $\mathcal{H}_1[\mathsf{sd}_1, j] = \mathbf{x}'_j$, $\mathcal{H}_2[\mathsf{sd}_2, j] = \mathbf{d}'_j$ where $\mathbf{x}'_j$ and $\mathbf{d}'_j$ are the j-th columns of $\mathbf{X}'$ and $\mathbf{D}'$ respectively. These hybrids are identically distributed by the programmability property of random oracles.

Hybrid 4. In this hybrid, we will compute the matrix $\mathbf{M}'$ where $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$ and $\mathbf{m}'_j = \mathbf{d}'_j$ if $j \notin S'$. Later in encryption phase, we will set $\mathcal{H}_2[\mathsf{sd}_2, j] = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ and $\mathcal{H}_1[\mathsf{sd}_1, j] = \mathbf{x}^*_j$ where $\mathbf{x}^*_j \leftarrow \{\pm 1\}^{4m^5}$ is sampled randomly for $j \in S' \setminus S^*$. Indistinguishability of hybrids holds by leftover hash lemma.

Hybrid 5. In this hybrid, we will commit the matrix $\mathbf{M}'$ calculated during the encryption phase of previous hybrid while performing setup. We will also set $\mathbf{A} = \mathbf{A}' - \mathsf{MCOM.Com}(\mathbf{M}')$ for $\mathbf{A}' \leftarrow \mathbb{Z}_q^{n \times m}$. The hybrids are identically distributed.

Hybrid 6. This is the game from Definition 4.1 with $b = 1$.

Full proof is provided in Appendix D.

□

Theorem 5.3 (Selective Security for Fake Setup)**.** If $2m^2$-decomposed LWE assumption (Assumption 3.4) holds, then Construction 5.1 satisfies selective security for fake setup property.

Proof Sketch. We define the following hybrids.

Hybrid 0. This is the experiment with $b = 0$ from Definition 4.1.

Hybrid 1. In this hybrid, we set $\mathbf{A}' = \mathbf{B}\mathbf{R}_{\mathbf{A}}, \mathbf{p} = \mathbf{B}\mathbf{r}_{\mathbf{p}}$. Indistinguishability of hybrids 0 and 1 follows from generalized leftover hash lemma.

Hybrid 2. In this hybrid, we set ct (output of $\mathsf{Equivocate}$) to be $(\mathbf{c}_1^\top, \mathbf{c}_1^\top \mathbf{R}_{\mathbf{A}}, \mathbf{c}_1^\top \mathbf{r}_{\mathbf{p}}, \mathsf{sd}_1, \mathsf{sd}_1)$ for random $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1 \times 4m^5}$. Hybrids 0 and 1 are indistinguishable by $2m^2$-decomposed LWE.

Hybrid 3. We set $\mathsf{ct} \leftarrow \mathbb{Z}_q^{1 \times (4m^5 + m + 1)} \times \{0,1\}^{2\lambda}$. Hybrids 2 and 3 are statistically indistinguishable by generalized leftover hash lemma.

Full proof is provided in Appendix D.

□

Corollary 5.4. Assuming the hardness of $2m^2$-decomposed LWE with polynomial modulus-to-noise ratio, there exists an adaptively secure DBE scheme with optimal parameters in the random oracle model.

Proof. From Theorem 5.2 and Theorem 5.3, we have a EqDBE scheme from $2m^2$-decomposed LWE assumption. This implies a semi-statically secure DBE scheme in the random oracle mode via Theorem 4.3. Then, by Theorem 3.10, we have an adaptively secure DBE scheme in the random oracle mode with optimal parameters from $2m^2$-decomposed LWE assumption. . □

6 Programmable Pseudorandom Set Sampler

In this section, we introduce a new tool that we call Programmable Pseudorandom Set Sampler (PPS). We use this tool to replace the random oracles from our EqDBE construction.

Syntax. A programmable pseudorandom set sampler (PPS) scheme for message space $\mathcal{M}$ consists of the following polynomial time algorithms.

$\mathsf{PPGen}(1^\lambda, 1^N) \to \mathsf{pp}$**.** The randomized public parameter generation algorithm takes as input security parameter λ, number of users N, and outputs public parameters pp. The following algorithms takes pp implicitly.

$\mathsf{MSKGen} \to \mathsf{msk}$ The randomized master secret key generation algorithm takes as input public parameters pp and outputs master secret key msk.

$\mathsf{pkEnc}(i) \to \mathsf{ct}$. The randomized *public-key* encryption algorithm takes as input index $i \in [N]$ and outputs ciphertext ct.

$\mathsf{KGen}(\mathsf{msk}, S) \to \mathsf{sk}_S$. The possibly randomized secret key generation algorithm takes as input master secret key msk, set $S \subseteq [N]$, and outputs secret key sk_S.

$\mathsf{Decode}(i, \mathsf{ct}, S, \mathsf{sk}_S) \to \mu'$. The deterministic decoding algorithm takes as input a ciphertext ct, secret key sk_S for set S, and outputs $\mu' \in \mathcal{M} \cup \{\perp\}$.

$\mathsf{skEnc}(\mathsf{msk}, \{i \to \mu_i\}_{i \in P}) \to (\mathsf{ct}_i)_{i \in P}$. The randomized *secret-key* encryption algorithm takes as input master secret key msk, a programming dictionary which contains $\{i \to \mu_i\}_{i \in P}$ for a set of indices $P \subseteq [N]$ such that $\mu_i \in \mathcal{M}$, and outputs ciphertexts $(\mathsf{ct}_i)_{i \in P}$.

Definition 6.1 (PPS)**.** A PPS scheme $(\mathsf{PPGen}, \mathsf{MSKGen}, \mathsf{pkEnc}, \mathsf{KGen}, \mathsf{Decode}, \mathsf{skEnc})$ is said to be a programmable pseudorandom set sampler scheme (PPS) if it satisfies the following properties.

Correctness. For any $\lambda \in \mathbb{N}$, positive polynomials $N = N(\lambda)$, sets $S, P \subseteq [N], i \in P \cap S$, for any $\mu_j \in \mathcal{M}$ for every $j \in P$,

$$\Pr\left[\begin{array}{l|ll} \mu_i = \mathsf{Decode}(i, \mathsf{ct}_i, S, \mathsf{sk}_S) & : & \mathsf{pp} \leftarrow \mathsf{PPGen}(1^\lambda, 1^N), \\ & & \mathsf{msk} \leftarrow \mathsf{MSKGen}(\mathsf{pp}), \\ & & (\mathsf{ct}_j)_{j \in P} \leftarrow \mathsf{skEnc}(\mathsf{msk}, \{j \to \mu_j\}_{j \in P}), \\ & & \mathsf{sk}_S \leftarrow \mathsf{KGen}(\mathsf{msk}, S) \end{array}\right] = 1$$

Compactness. There exists a polynomial $\mathsf{poly}(\cdot, \cdot)$ such that $\forall\ \lambda, N \in \mathbb{N},\ S \subseteq [N], |\mathsf{sk}_S| = \mathsf{poly}(\lambda, \log N)$.

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

1. $\mathcal{A}(1^\lambda)$ sends $1^N, \{j \to \mu_j\}_{j \in P}, i^*$ where $P \subseteq [N]$, $i^* \notin P$ to Chal.
2. Chal computes $\mathsf{pp} \leftarrow \mathsf{PPGen}(1^\lambda, 1^N)$ and $\mathsf{msk} \leftarrow \mathsf{MSKGen}$.
3. Chal flips a coin $b \leftarrow \{0, 1\}$. If $b = 0$, Chal sets $T^{(0)} = \{j \to \mu_j\}_{j \in P}$ and computes $(\mathsf{ct}_i^{(0)})_{i \in P} \leftarrow \mathsf{skEnc}(\mathsf{msk}, T^{(0)})$ and for each $i \notin P, \mathsf{ct}_i^{(0)} \leftarrow \mathsf{pkEnc}(i)$.
   Otherwise, if $b = 1$, Chal samples $\mu_{i^*}^* \leftarrow \mathcal{M}$ and sets $T^{(1)} = \{j \to \mu_j\}_{j \in P} \cup \{i^* \to \mu_{i^*}^*\}$. Chal then samples $(\mathsf{ct}_i^{(1)})_{i \in P \cup \{i^*\}} \leftarrow \mathsf{skEnc}(\mathsf{msk}, T^{(1)})$ and for each $i \notin P \cup \{i^*\}, \mathsf{ct}_i^{(1)} \leftarrow \mathsf{pkEnc}(i)$.
4. Chal sends $\mathsf{pp}, (\mathsf{ct}_i^{(b)})_{i \in [N]}$ to $\mathcal{A}$.
5. $\mathcal{A}$ then sends $S^* \subseteq [N]$ such that $i^* \in S^*$. Chal samples $\mathsf{sk}_{S^*} \leftarrow \mathsf{KGen}(\mathsf{msk}, S^*)$ and sends sk_{S^*} to $\mathcal{A}$.
6. $\mathcal{A}$ guesses b as b'.

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

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

Payload Hiding. We define the following game between a PPT adversary $\mathcal{A}$ and a challenger Chal.

1. $\mathcal{A}(1^\lambda)$ sends $1^N, \{j \to \mu_j\}_{j \in P}, i^*, \mu_{i^*}$ where $P \subseteq [N]$, $i^* \notin P, \mu_{i^*} \in \mathcal{M}$ to Chal.
2. Chal computes $\mathsf{pp} \leftarrow \mathsf{PPGen}(1^\lambda, 1^N)$ and $\mathsf{msk} \leftarrow \mathsf{MSKGen}$.
3. Chal flips a coin $b \leftarrow \{0,1\}$. If $b = 0$, Chal sets $T^{(0)} = \{j \to \mu_j\}_{j \in P}$ and computes $(\mathsf{ct}_i^{(0)})_{i \in P} \leftarrow \mathsf{skEnc}(\mathsf{msk}, T^{(0)})$ and for each $i \notin P, \mathsf{ct}_i^{(0)} \leftarrow \mathsf{pkEnc}(i)$.
   Otherwise, if $b = 1$, Chal sets $T^{(1)} = \{j \to \mu_j\}_{j \in P} \cup \{i^* \to \mu_{i^*}\}$. Chal then samples $(\mathsf{ct}_i^{(1)})_{i \in P \cup \{i^*\}} \leftarrow \mathsf{skEnc}(\mathsf{msk}, T^{(1)})$ and for each $i \notin P \cup \{i^*\}, \mathsf{ct}_i^{(1)} \leftarrow \mathsf{pkEnc}(i)$.
4. Chal sends $\mathsf{pp}, (\mathsf{ct}_i^{(b)})_{i \in [N]}$ to $\mathcal{A}$.
5. $\mathcal{A}$ then sends $S^* \subseteq [N]$ such that $i^* \notin S^*$. Chal samples $\mathsf{sk}_{S^*} \leftarrow \mathsf{KGen}(\mathsf{msk}, S^*)$ and sends sk_{S^*} to $\mathcal{A}$.
6. $\mathcal{A}$ guesses b as b'.

We say that the PPS scheme satisfies payload hiding if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$,

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

Remark 6.2. By a standard hybrid argument, we can extend the definition of mode indistinguishability from a single index i^* to a set of indices $I \subseteq [N]$ as long as $P \cap I = \emptyset$ and $I \subseteq S^*$.

7 Equivocal DBE in Standard Model from Programmable Pseudorandom Set Sampler

In this section, we construct an EqDBE scheme in the standard model by substituting the random oracles from Section 5 with programmable pseudorandom set samplers.

Parameters. We describe the LWE parameters that our construction uses for $N = N(\lambda)$ users.

- $n = \mathsf{poly}(\lambda)$ (Decomposed LWE security)
- $m = \Omega(n \log q)$ (Leftover hash lemma, Lemma 3.2)
- $\chi = \mathsf{poly}(\lambda)$ (Decomposed LWE security)
- $\sigma = \mathsf{poly}(\lambda, m)$ (Decomposed LWE security)
- $q > O(\sigma \chi m^{13} \log q \log N)$ (Correctness)

Construction 7.1. We provide the construction of EqDBE in standard model assuming the hardness of $2m^2$-decomposed LWE (Assumption 3.4). We will use the matrix commitment scheme MCOM (Section 3.3) and Programmable Pseudorandom Set Samplers (Definition 6.1) $\mathsf{PPS}^{(1)}, \mathsf{PPS}^{(2)}$, for message space $\{\pm 1\}^{4m^5}$ and $\mathbb{Z}_q^n$ respectively.

$\mathsf{GSetup}(1^\lambda, 1^N)$: Sample $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$, $\mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$.

1. For each $i \in [N]$, sample $\mathsf{samp.ct}_i^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, i)$.
2. For each $i \in [N]$, sample $\mathsf{samp.ct}_i^{(2)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(2)}, i)$.

Output $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_i^{(1)})_{i \in [N]}, (\mathsf{samp.ct}_i^{(2)})_{i \in [N]})$.

$\mathsf{KGen}(i)$: This is same as $\mathsf{KGen}(i)$ from Construction 5.1.

$\mathsf{Enc}(\{\mathsf{pk}_i\}_{i \in S}, \mu)$: Parse pk_i as $(i, \mathbf{t}_i)$ for each $i \in S$.

1. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)})$, $\mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Then sample $\mathsf{samp.sk}_S \leftarrow \mathsf{PPS.KGen}(\mathsf{samp.msk}^{(1)}, S)$ and $\mathsf{samp.sk}_{\overline{S}} \leftarrow \mathsf{KGen}(\mathsf{samp.msk}^{(2)}, [N] \setminus S)$.
2. For each $j \in [N]$, if $j \in S$, let $\mathbf{x}_j = \mathsf{PPS.Decode}(j, \mathsf{samp.ct}_j^{(1)}, S, \mathsf{samp.sk}_S)$ and if $j \notin S$, $\mathbf{d}_j = \mathsf{PPS.Decode}(j, \mathsf{samp.ct}_j^{(2)}, [N] \setminus S, \mathsf{samp.sk}_{\overline{S}})$.

3-6. Perform steps 2-5 from Construction 5.1.

Output $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{samp.sk}_S, \mathsf{samp.sk}_{\overline{S}}) \in \mathbb{Z}_q^{1 \times (4m^5 + m + 1)} \times \{0,1\}^{2\mathsf{poly}(\lambda, \log N)}$.

$\mathsf{Dec}(\{\mathsf{pk}_i\}_{i \in S}, (j, \mathsf{sk}_j), \mathsf{ct})$: Parse pk_i as $(i, \mathbf{t}_i)$ for each $i \in S$, sk_j as $\mathbf{r}_j$, and ct as $(\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{samp.sk}_S, \mathsf{samp.sk}_{\overline{S}})$. If $j \notin S$, output $\perp$. Otherwise,

1. For each $k \in [N]$, if $k \in S$, let $\mathbf{x}_k = \mathsf{PPS.Decode}(k, \mathsf{samp.ct}_k^{(1)}, S, \mathsf{samp.sk}_S)$ and if $k \notin S$, $\mathbf{d}_k = \mathsf{PPS.Decode}(k, \mathsf{samp.ct}_k^{(2)}, [N] \setminus S, \mathsf{samp.sk}_{\overline{S}})$.

2-4. Perform steps 2-4 from Construction 5.1.

Output 0 if $-q/4 < y < q/4$ and 1 otherwise.

Correctness. This is almost identical to the correctness analysis of Construction 5.1. In addition, we rely on deterministic nature of $\mathsf{PPS.Decode}$ instead of deterministic nature of random oracles. Thus, for any sufficiently large and prime $q > O(\sigma\chi m^{13} \log q \log N)$, correctness holds.

Compactness. As KGen is same as Construction 5.1, $|\mathsf{pk}_i|, |\mathsf{sk}_i|$ are the same as Construction 5.1 and satisfy the compactness property. As $\mathsf{ct} \in \mathbb{Z}_q^{1 \times (4m^5 + m + 1)} \times \{0,1\}^{2|\mathsf{samp.sk}_S|}$, $|\mathsf{ct}| = \mathsf{poly}(\lambda, m, \log q) = \mathsf{poly}(\lambda, \log N)$. This follows from the compactness property of PPS.

7.1 Fake Setup and Equivocation

$\mathsf{FakeSetup}(1^\lambda, 1^N, S', \mu)$: Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$, $\mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$, and perform the following steps.

1. For each $i \in S' \subseteq [N]$, sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. Set $\widetilde{\mathsf{pk}}_i := \mathbf{t}_i$ for each $i \in S'$.
2. Sample $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$.

3. Sample $\mathbf{X}' \leftarrow \{0,1\}^{4m^5 \times N}, \mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively.
4. Compute matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$. Otherwise $\mathbf{m}'_j = \mathbf{d}'_j$. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Sample $\mathbf{A}' \leftarrow \mathbb{Z}_q^{n \times m}$ and set $\mathbf{A} = \mathbf{A}' - \mathbf{C}'$.
5. Sample $\mathbf{p} \leftarrow \mathbb{Z}_q^n$.
6. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)}), \mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$.
7. Set $T^{(1)} = \{j \to \mathbf{x}'_j\}_{j \in S'}$. Also, set $T^{(2)} = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S'} \cup \{j \to \mathbf{d}'_j\}_{j \notin S'}$. For each $j \notin S'$, sample $\mathsf{samp.}\widetilde{\mathsf{ct}}_j^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, i)$.
8. Sample $(\mathsf{samp.}\widetilde{\mathsf{ct}}^{(1)})_{j \in S'} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, T^{(1)})$ and $(\mathsf{samp.}\widetilde{\mathsf{ct}}^{(2)})_{j \in [N]} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T^{(2)})$.
9. Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.}\widetilde{\mathsf{ct}}_i^{(1)})_{i \in [N]}, (\mathsf{samp.}\widetilde{\mathsf{ct}}_i^{(2)})_{i \in [N]})$.
10. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R}_\mathbf{A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top \mathbf{R}_\mathbf{A}, c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r}_\mathbf{p} + \lfloor q/2 \rfloor \mu$. Set $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$. Set $\mathsf{aux} := (S', \mathsf{samp.msk}^{(1)}, \mathsf{samp.msk}^{(2)}, \widetilde{\mathsf{ct}})$.

Output $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$.

$\mathsf{Equivocate}(\mathsf{aux}, S^*)$: Parse aux as $(S', \mathsf{samp.msk}^{(1)}, \mathsf{samp.msk}^{(2)})$.

1. Sample $\mathsf{samp.sk}_{S^*} \leftarrow \mathsf{PPS.KGen}(\mathsf{samp.msk}^{(1)}, S^*)$ and $\mathsf{samp.sk}_{\overline{S^*}} \leftarrow \mathsf{PPS.KGen}(\mathsf{samp.msk}^{(2)}, [N] \setminus S^*)$
2. Parse $\widetilde{\mathsf{ct}}$ as $(\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$. Set $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{samp.sk}_{S^*}, \mathsf{samp.sk}_{\overline{S^*}})$.

Output ct.

7.2 Security Analysis

Theorem 7.2 (Mode Indistinguishability)**.** If PPS satisfies mode indistinguishability and payload hiding properties (Definition 6.1), then Construction 7.1 satisfies mode indistinguishability.

Proof Sketch. We argue mode indistinguishability using the following series of hybrids.

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

Hybrid 1. In this hybrid, we will sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$ for each $i \in S'$. Hybrids 0 and 1 are indistinguishable by leftover hash lemma.

Hybrid 2. In this hybrid, we will sample $\mathsf{samp.msk}^{(2)}$ early during setup and sample $(\mathsf{samp.}\widetilde{\mathsf{ct}}_j^{(2)})_{j \notin S'}$ using $\mathsf{PPS.skEnc}$ with values $\mathbf{d}'_j \leftarrow \mathbb{Z}_q^n$ for each $j \notin S'$. Later during encryption phase, we will use $\mathsf{samp.msk}^{(2)}$ to sample $\mathsf{samp.sk}_{\overline{S^*}}$. Indistinguishability of hybrids follows from mode indistinguishability property of PPS (specifically, Remark 6.2 with an empty programming dictionary).

Hybrid 3, i **for** $i \in [N+1]$**.** In this hybrid, we will set first i $\mathsf{samp}.\widetilde{\mathsf{ct}}_i^{(1)}$ and $\mathsf{samp}.\widetilde{\mathsf{ct}}_i^{(2)}$ such that they can be decoded to $\mathbf{x}'_i$ and $\mathbf{t}_i + \mathbf{B}\mathbf{x}'_i$ respectively if $i \in S'$. Indistinguishability follows by PPS mode indistinguishability and payload hiding properties.

Hybrid 4. At this point, the matrix $\mathbf{M}$ constructed during encryption is $\mathbf{M} = [\mathbf{m}_1 \mid \ldots \mid \mathbf{m}_N]$ where $\mathbf{m}_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$ and $\mathbf{m}_j = \mathbf{d}'_j$ if $j \notin S'$ irregardless of $S^* \subseteq S'$ provided by $\mathcal{A}$. This matrix can be sampled during setup. We will do so and set $\mathbf{A} = \mathbf{A}' - \mathsf{MCOM.Com}(\mathbf{M})$.

Hybrid 5. This is mode indistinguishability experiment with $b = 1$.

We remark that indistinguishability between hybrids $3, i$ and $3, i+1$ for any $i \in [N]$ requires to guess whether $i \in S^* \subseteq S'$. This is because $\mathsf{samp}.\widetilde{\mathsf{ct}}_i^{(1)}$ and $\mathsf{samp}.\widetilde{\mathsf{ct}}_i^{(2)}$ decode to related messages ($\mathbf{x}'_i$ and $\mathbf{t}_i + \mathbf{B}\mathbf{x}'_i$ respectively). As explained in Section 2, we avoid guessing whole S^* by using the 'piecewise guessing framework' of [JKK+17, KW20] where between two hybrids we will guess a single bit of information, namely, $i \in S^*$. We show that there are two paths ($i \in S^*$ and $i \notin S^*$), both of them take us from Hybrid $3, i$ to $3, i+1$, and these hybrids' setup phases themselves are independent of S^*. Thus, we can argue security from polynomially hard assumptions (refer Figure 2).
Full description of hybrids and proof is provided in Appendix E. □

Theorem 7.3 (Selective Security for Fake Setup)**.** If $2m^2$-decomposed LWE assumption (Assumption 3.4) holds, then Construction 7.1 satisfies selective security for fake setup property.

Proof Sketch. This is similar to proof of Theorem 5.3. Full proof is provided in Appendix E. □

8 Programmable Pseudorandom Set Sampler from LWE

In this section, we construct a PPS scheme from Learning with Errors assumption (Assumption 3.3). Ciphertexts in this scheme grow linearly with N.

Parameters. We provide the parameters that the construction uses below.

- $n = \mathsf{poly}(\lambda)$ (LWE security)
- $m > (n + 1 + 2N)\log q + \lambda$ (Leftover Hash Lemma, Lemma 3.2)
- $\chi = \mathsf{poly}(\lambda)$ (LWE security)
- $B \geq Nm^{3/2}\chi$ (Correctness)
- $q > B\lambda^{\omega(1)}$ (Security)

Construction 8.1. We provide the construction of a PPS scheme (Definition 6.1) for message space $\mathcal{M} = \mathbb{Z}_2$ from the LWE assumption (Assumption 3.3) below.

$\mathsf{PPGen}(1^\lambda, 1^N)$: Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ Output $\mathsf{pp} := (n, m, q, \chi, B, \mathbf{A}, N)$.

MSKGen : Sample $\mathbf{s}_{i,b} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{i,b} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m$ for each $i \in [N], b \in \{0,1\}$. Compute $\mathbf{b}_{i,b}^\top = \mathbf{s}_{i,b}^\top \mathbf{A} + \mathbf{e}_{i,b}^\top$.
Output $\mathsf{msk} := (\mathbf{s}_{i,b}, \mathbf{b}_{i,b})_{i \in [N], b \in \{0,1\}}$.

$\mathsf{KGen}(\mathsf{msk}, S)$: Let $\mathbf{r} \in \{0,1\}^N$ denote the characteristic vector of set $S \subseteq [N]$. That is $\mathbf{r} = (r_1, \ldots, r_N)$ such that $r_i = 1$ if and only if $i \in S$.
Output $\mathsf{sk}_S := \sum_{j \in [N]} \mathbf{s}_{j,r_j} \in \mathbb{Z}_q^n$.

$\mathsf{skEnc}(\mathsf{msk}, \{j \to \mu_j\}_{j\in P})$: Given $P \subseteq [N], \forall\ j \in P, \mu_j \in \mathbb{Z}_2$, parse msk as $(\mathbf{s}_{i,b}, \mathbf{b}_{i,b})_{i\in[N],b\in\{0,1\}}$. For each $j \in P$, perform the following steps.

1. Sample $\mathbf{x}_j \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{j,0} = \mathbf{A}\mathbf{x}_j$.
2. Sample for each $k \in [N]$, random $z_{j,k} \in \mathbb{Z}_q$ such that $\sum_{k\in[N]} z_{j,k} = 0$.
3. Sample $t_j \leftarrow \mathbb{Z}_q$ subject to the constraint that $t_j \notin [-\lfloor q/4\rfloor - B, -\lfloor q/4\rfloor + B] \cup [\lfloor q/4\rfloor - B, \lfloor q/4\rfloor + B]$. As $q > \lambda^{\omega(1)}B$, the set of valid t_j is non-empty.
4. Set $c_{j,j,1} = \mathbf{b}_{j,1}^\top \mathbf{x}_j + z_{j,j} + t_j$ and sample $c_{j,j,0} \leftarrow \mathbb{Z}_q$.
5. For $k \in [N], k \neq j$ for each $b \in \{0,1\}$, set $c_{j,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_j + z_{j,k}$.
6. Set $c_{j,T} = \lceil t_j \rfloor_2 + \mu_j$
7. Set $\mathsf{ct}_j := (\mathbf{c}_{j,0}, (c_{j,k,b})_{k,b}, c_{j,T}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$.

Output $(\mathsf{ct}_j)_{j\in P}$.

$\mathsf{pkEnc}(i)$: Output $\mathsf{ct} \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$.

$\mathsf{Decode}(i, \mathsf{ct}, S, \mathsf{sk}_S)$: Parse ct as $(\mathbf{c}_0, (c_{j,b})_{j\in[N],b\in\{0,1\}}, c_T)$. Let $\mathbf{r} = (r_1, \ldots, r_N)$ denote the characteristic vector of S similar to KGen. If $i \notin S$, abort and output $\perp$.

Otherwise, compute $c' = \sum_{j\in[N]} c_{j,r_j} - (\mathsf{sk}_S)^\top \mathbf{c}_0$. Output $\mu' = c_T - \lceil c' \rfloor_2$.

Remark 8.2 (Encrypting longer messages.)**.** By a standard hybrid encryption technique, we can run λ instances of Construction 8.1 and use $(\mu_1, \ldots, \mu_\lambda) \in \{0,1\}^\lambda$ as a PRF key to encrypt messages of any length. This parallel composition will not effect the compactness of the scheme.

Correctness. Note that if $i \notin S$, Dec outputs $\perp$. Thus, we will focus on $i \in S$. For any set S, $\mathsf{sk}_S = \sum_{j\in[N]} \mathbf{s}_{j,r_j}$ where $\mathbf{r} = (r_1, \ldots, r_N)$ is the characteristic vector of S. Now, secret-key encryption for index $i \in [N]$, $\mu \in \mathbb{Z}_2$ is $\mathsf{ct} = (\mathbf{A}\mathbf{x}, (c_{j,b})_{j,b}, \lceil t \rfloor_2 + \mu)$ for $t \in \mathbb{Z}_q$, $\mathbf{A}$ is from pp, $\mathbf{x} \in \{\pm 1\}^m$, and

$$c_{j,b} = \begin{cases} (\mathbf{s}_{j,b}^\top \mathbf{A} + \mathbf{e}_{j,b}^\top)\mathbf{x} + z_j & \text{if } j \neq i \\ (\mathbf{s}_{i,1}^\top \mathbf{A} + \mathbf{e}_{i,1}^\top)\mathbf{x} + z_i + t & \text{if } j = i, b = 1 \end{cases}$$

As $i \in S, r_i = 1$. So, $\sum_{j\in[N]} c_{j,r_j} = \left(\sum_{j\in[N]} \mathbf{s}_{j,r_j}^\top \mathbf{A} + \mathbf{e}_{j,r_j}^\top\right)\mathbf{x} + \sum_{j\in[N]} z_j + t$. This means that

$$\begin{aligned} c' &= \sum_{j\in[N]} c_{j,r_j} - (\mathsf{sk}_S)^\top \mathbf{c}_0 \\ &= \left(\sum_{j\in[N]} \mathbf{s}_{j,r_j}^\top\right)\mathbf{A}\mathbf{x} + \left(\sum_{j\in[N]} \mathbf{e}_{j,r_j}^\top\right)\mathbf{x} + \sum_{j\in[N]} z_j + t - \left(\sum_{j\in[N]} \mathbf{s}_{j,r_j}^\top\right)\mathbf{A}\mathbf{x} \\ &= t + \left(\sum_{j\in[N]} \mathbf{e}_{j,r_j}^\top\right)\mathbf{x} \end{aligned}$$

We have that $\lceil c' \rfloor_2 = \lceil t \rfloor_2$ as long as $\left\| \left(\sum_{j \in [N]} \mathbf{e}_{j,r_j}^\top \right) \mathbf{x} \right\| \leq B$. Since $\left\| \mathbf{e}_{j,b}^\top \right\| \leq \sqrt{m}\chi$ for any j, b with high probability, $\left\| \left(\sum_{j \in [N]} \mathbf{e}_{j,r_j}^\top \right) \mathbf{x} \right\| \leq Nm^{3/2}\chi$ with high probability. Thus, if $B \geq Nm^{3/2}\chi$, $\mu' = c_T - \lceil c' \rfloor_2 = \lceil t \rfloor_2 + \mu - \lceil t \rfloor_2 = \mu$, and correctness holds.

Compactness. As $\mathsf{sk}_S \in \mathbb{Z}_q^n$, $|\mathsf{sk}_S| = n\lceil \log q \rceil = \mathsf{poly}(\lambda, \log N)$. Thus, compactness holds.

8.1 Security Analysis

Theorem 8.3 (Mode Indistinguishability)**.** Assuming the hardness of LWE (Assumption 3.3), Construction 8.1 satisfies mode indistinguishability.

Proof Sketch. We define the following hybrids.

Hybrid 0. This is the mode indistinguishability game from Definition 6.1 with $b = 1$.

Hybrid 1. In this hybrid, since $\mu \leftarrow \mathbb{Z}_p$ is chosen randomly by challenger, we can simply set $c_T \leftarrow \mathbb{Z}_2$ independent of t. Hybrids 0 and 1 are identically distributed.

Hybrid 2. We will now sample $t \leftarrow \mathbb{Z}_q$ uniformly randomly. This means that $c_{i^*,1}$ can be set as uniformly random element from $\mathbb{Z}_q$ as t does not appear anywhere else in the ciphertext. Hybrids 1 and 2 are statistically close as long as $(4B + 2)/q \leq \mathsf{negl}(\lambda)$ which is true by our parameter setting.

Hybrid 3. Now, as z_{i^*}, one of the zero shares in the challenge ciphertext, is nowhere in the view of the adversary as $c_{i^*,1}$ is randomly sampled. Thus, we will sample $\{z_j\}_{j \neq i^*}$ as random elements in $\mathbb{Z}_q$. Hybrids 2 and 3 are identically distributed.

Hybrid 4,κ for $\kappa \in [N+1]$. In this hybrid, we will set $c_{\kappa,0}$ and $c_{\kappa,1}$ of the challenge ciphertext to be random elements from $\mathbb{Z}_q$. Indistinguishability requires a piecewise guessing framework similar to proof of Theorem 7.2. At a high level, we will argue security as follows.

If $\kappa \in S^*$, then $\mathbf{s}_{\kappa,0}$ is not used in key generation. So, we can set $(\mathbf{s}_{\kappa,0}^\top \mathbf{A} + \mathbf{e}_{\kappa,0}^\top)$ to be random and thus $c_{\kappa,0}$ to be random by LWE security and leftover hash lemma. Then, as z_κ is random, we can set $c_{\kappa,1}$ to be random.

Otherwise, if $\kappa \notin S^*$, we proceed in a flipped manner.[Hybrid 5.] This is mode indistinguishability experiment with $b = 0$.

Full proof is provided in Appendix F. □

Theorem 8.4 (Payload Hiding)**.** Assuming the hardness of LWE (Assumption 3.3), Construction 8.1 satisfies payload hiding.

Proof Sketch. We define the following hybrids.

Hybrid 0. This is the payload hiding experiment from Definition 6.1 with $b = 1$.

Hybrid 1. In this hybrid, as $i^* \notin S^*$, we will set $(\mathbf{s}_{i^*,1}^\top \mathbf{A} + \mathbf{e}_{i^*,1}^\top)$ to be uniformly random as $\mathbf{s}_{i^*,1}$ is not used in secret key generation. Hybrids 0 and 1 are indistinguishable by LWE security. This allows us to sample $c_{i^*,1}$ to uniformly random string via leftover hash lemma.

Hybrid 2,κ for $\kappa \in [N+1]$. In this hybrid, we will set $c_{\kappa,0}$ and $c_{\kappa,1}$ of the challenge ciphertext to be random elements from $\mathbb{Z}_q$. Indistinguishability requires a piecewise guessing framework similar to proof of Theorem 8.3.

Hybrid 3. This is payload hiding experiment with $b = 0$.

Full proof is provided in Appendix F. □

Corollary 8.5. Assuming the hardness of $2m^2$-decomposed LWE, there exists an adaptively secure DBE scheme for N users in the standard model. Moreover, the public-key, secret-key and ciphertext sizes are independent of N and crs size grows proportional to N^2.

Proof. Theorems 8.3 and 8.4 give us a PPS scheme from LWE where ciphertexts grow proportional to N. Using this, theorems 7.2 and 7.3 give us an EqDBE scheme in the standard model with optimal public-secret keys and ciphertexts. The size of the crs in this scheme grows with N^2 as we sample N different ciphertexts. Theorems 4.3 and 3.11 then gives us an adaptively secure DBE scheme. □

9 Programmable Pseudorandom Set Sampler from 1-key LWE

In this section, we give the construction of a PPS scheme with $\mathsf{poly}(\lambda, \log N)$-size ct and sk_S from a new *falsifiable* lattice assumption, that we call 1-key LWE assumption. We begin by formally stating this assumption.

9.1 1-key LWE

We give our 1-key LWE assumption's definition below. Relevant preliminaries about lattice trapdoors and homomorphic computation is provided in Appendix A.

Assumption 9.1. Let (n, m, q, χ) be LWE parameters. σ be width parameter such that $\sigma > m^{O(\lambda)} \log n$. The 1-key LWE assumption is defined with respect to the following game between a challenger (Chal) and a PPT adversary ($\mathcal{A}$).

1. $\mathcal{A}(1^\lambda)$ sends 1^N, $i \in [N]$ to Chal.
2. Chal samples $b \leftarrow \{0,1\}$. Chal then samples $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \mathbf{B} \leftarrow \mathbb{Z}_q^{n \times \lceil \log N \rceil m}$, $\mathbf{d} \leftarrow \mathbb{Z}_q^n$. If $b = 0$, Chal samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_1 \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{X} \leftarrow \{\pm 1\}^{m \times \lceil \log N \rceil m}$ and sets $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{A} + \mathbf{e}_1^\top$ and $\mathbf{c}_2^\top = \mathbf{s}^\top (\mathbf{B} + \mathbf{i}^\top \otimes \mathbf{G}) + \mathbf{e}_1^\top \mathbf{X}$ where $\mathbf{i} \in \{0,1\}^{\lceil \log N \rceil}$ is the bit-vector representation of i.

   Otherwise, if $b = 1$, Chal samples $\mathbf{c}_1 \leftarrow \mathbb{Z}_q^m, \mathbf{c}_2 \leftarrow \mathbb{Z}_q^{\lceil \log N \rceil m}$. Chal sends $(\mathbf{A}, \mathbf{B}, \mathbf{d}, \mathbf{c}_1^\top, \mathbf{c}_2^\top)$ to $\mathcal{A}$.
3. $\mathcal{A}$ sends a set $S \subseteq [N]$. Chal constructs a function $\mathsf{mem}\text{-}S : [N] \to \{0,1\}$ defined as $\mathsf{mem}\text{-}S(i) = 0$ if and only if $i \in S$. That is, $\mathsf{mem}\text{-}S$ is the membership verification circuit for set S[9]. $\mathbf{B}_S = \mathsf{EvalF}(\mathbf{B}, \mathsf{mem}\text{-}S)$. Chal samples $\mathbf{r}_S \leftarrow \mathsf{SamplePre}\left([\mathbf{A} \mid \mathbf{B}_S], \begin{bmatrix} \mathbf{T_A} \\ \mathbf{0} \end{bmatrix}, \mathbf{d}, \sigma\right)$ and sends $\mathbf{r}_S$ to $\mathcal{A}$.

[9]Set membership for sets that are subsets of $[N]$ can be verified using a circuit of depth at most $O(\log N)$ (NC^1 circuit). This depth can be upper bounded by λ for large enough λ.

4. $\mathcal{A}$ sends a bit b'. Output b'.

We say that 1-key LWE assumption holds if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$,

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

This assumption can be generalized for any family of functions of depth d. We define this version formally and provide a discussion on its hardness in Appendix B.

Hardness of 1-key LWE. The most an attacker against 1-key LWE assumption can do is compute $[\mathbf{c}_1^\top \mid \mathbf{c}_2^\top \mathbf{H}_{S,\mathbf{i}}]\mathbf{r}_S$ to get a new LWE sample $\mathbf{s}^\top \mathbf{d} + \mathsf{err}$ for a randomly sampled $\mathbf{d}$. This might not yield any new information about $\mathbf{c}_1^\top, \mathbf{c}_2^\top$. Hence, we conjecture that the assumption holds. For an extended preliminary cryptanalysis of this assumption, please check Appendix B.

9.2 Construction

In this section, we provide the construction of a PPS scheme from 1-key LWE assumption (Assumption 9.1). Relevant preliminaries about lattice trapdoors and homomorphic computation are provided in Appendix A.

Parameters. We define the LWE parameters used in our construction below.

- $n = \mathsf{poly}(\lambda)$ (LWE, 1-key LWE security)
- $m = \Omega(n \log q)$ (Leftover hash lemma Lemma 3.2)
- $\chi = \mathsf{poly}(\lambda)$ (LWE, 1-key LWE security)
- $\sigma > m^{O(\lambda)} \log n$ (Preimage sampling indistinguishability)
- $q > \lambda^{\omega(1)} O(\lceil \log N \rceil \chi \sigma m^{O(\lambda)})$ (Correctness)

Construction 9.2. We provide the construction of a PPS scheme for $\mathcal{M} = \mathbb{Z}_2$ as follows.

$\mathsf{PPGen}(1^\lambda, 1^N)$: Output $\mathsf{pp} := (n, m, q, \chi, \sigma, N)$.

$\mathsf{MSKGen}(\mathsf{pp})$: Sample $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, $\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times \lceil \log N \rceil m}$, $\mathbf{d} \leftarrow \mathbb{Z}_q^n$. Output $\mathsf{msk} := (\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{B}, \mathbf{d})$.

$\mathsf{pkEnc}(i)$: Given $i \in [N]$, sample $\mathsf{ct}_i \leftarrow \mathbb{Z}_q^{1 \times (\lceil \log N \rceil + 1)m} \times \mathbb{Z}_2$. Output ct_i.

$\mathsf{KGen}(\mathsf{msk}, S)$: Parse msk as $(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{B}, \mathbf{d})$. Let $\mathsf{mem}\text{-}S$ be the membership verification circuit for set S, i.e, $\mathsf{mem}\text{-}S(j) = 0$ if and only if $j \in S$. Compute $\mathbf{B}_S = \mathsf{EvalF}(\mathbf{B}, \mathsf{mem}\text{-}S)$. Sample $\mathbf{r}_S \leftarrow \mathsf{SamplePre}\left([\mathbf{A} \mid \mathbf{B}_S], \begin{bmatrix} \mathbf{T}_\mathbf{A} \\ \mathbf{0} \end{bmatrix}, \mathbf{d}, \sigma\right)$. Output $\mathsf{sk}_S := (\mathbf{r}_S, \mathbf{B})$.

$\mathsf{Decode}(i, \mathsf{ct}, S, \mathsf{sk}_S)$: If $i \notin S$, output $\perp$. Otherwise, parse ct as $\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3 \in \mathbb{Z}_q^{1 \times (1 + \lceil \log N \rceil)m} \times \mathbb{Z}_2$ and sk_S as $(\mathbf{r}_S, \mathbf{B}) \in \mathbb{Z}_q^{2m} \times \mathbb{Z}_q^{n \times \lceil \log N \rceil)m}$. Compute $\mathbf{H}_{S,\mathbf{i}} = \mathsf{EvalFX}(\mathbf{B}, \mathsf{mem}\text{-}S, \mathbf{i})$. Compute $\widetilde{c} = \lceil [\mathbf{c}_1^\top \mid \mathbf{c}_2^\top \mathbf{H}_{S,\mathbf{i}}] \mathbf{r}_S \rfloor_p$. Compute $\mu' = c_3 - \lceil \widetilde{c} \rfloor_2$. Output μ'.

$\mathsf{skEnc}(\mathsf{msk}, \{j \to \mu_j\}_{j \in P})$: Parse msk as $(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{B}, \mathbf{d})$. If any $\mu_j \notin \mathbb{Z}_2$, output $\perp$. Otherwise, for each $j \in P$,

1. Sample $\mathbf{s}_j \leftarrow \mathbb{Z}_q^n$, $\mathbf{e}_{1,j} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m$, $\mathbf{X}_j \leftarrow \{\pm 1\}^{m \times \lceil \log N \rceil m}$.
2. Compute $\mathbf{c}_{1,j}^\top = \mathbf{s}_j^\top \mathbf{A} + \mathbf{e}_{1,j}^\top$, $\mathbf{c}_{2,j}^\top = \mathbf{s}_j^\top (\mathbf{B} + \mathbf{j}^\top \otimes \mathbf{G}) + \mathbf{e}_{1,j}^\top \mathbf{X}_j$, and $c_{3,j} = \lceil \mathbf{s}_j^\top \mathbf{d} \rfloor_2 + \mu_j$ where $\mathbf{j} \in \{0,1\}^{\lceil \log N \rceil}$ is the bit vector representation of $j \in P \subseteq [N]$.
3. Set $\mathsf{ct}_j = (\mathbf{c}_{1,j}^\top, \mathbf{c}_{2,j}^\top, c_{3,j}) \in \mathbb{Z}_q^{1 \times (1+\lceil \log N \rceil)m} \times \mathbb{Z}_2$.

Output $(\mathsf{ct}_j)_{j \in P}$.

Remark 9.3 (Encrypting longer messages.)**.** By a standard hybrid encryption technique, we can run λ instances of Construction 9.2 and use $(\mu_1, \ldots, \mu_\lambda) \in \{0,1\}^\lambda$ as a PRF key to encrypt messages of any length. This parallel composition will not effect the compactness of the scheme.

Correctness. If $i \notin S$, Decode outputs $\perp$. So, we will only focus on the case when $i \in S$. Let sk_S be $(\mathbf{r}_S, \mathbf{B})$ such that $[\mathbf{A} \mid \mathbf{B}_S]\mathbf{r}_S = \mathbf{d}$ where $\mathbf{A}, \mathbf{d}$ are part of msk and $\mathbf{B}_S = \mathsf{EvalF}(\mathbf{B}, \mathsf{mem}\text{-}S)$. Let ct be $(\mathbf{s}^\top \mathbf{A} + \mathbf{e}_1^\top, \mathbf{s}^\top(\mathbf{B} + \mathbf{i}^\top \otimes \mathbf{G}) + \mathbf{e}_1^\top \mathbf{X}, \lceil \mathbf{s}^\top \mathbf{d} \rfloor_2 + \mu)$ for some $\mathbf{s}^\top, \mathbf{e}_1^\top, \mathbf{X}$. As $\mathbf{H}_{S,\mathbf{i}} = \mathsf{EvalFX}(\mathbf{B}, \mathsf{mem}\text{-}S, \mathbf{i})$, it means that $(\mathbf{B} + \mathbf{i}^\top \otimes \mathbf{G})\mathbf{H}_{S,\mathbf{i}} = \mathbf{B}_S$. Thus,

$$\begin{aligned}
[\mathbf{c}_1^\top \mid \mathbf{c}_2^\top \mathbf{H}_{S,\mathbf{i}}]\mathbf{r}_S &= (\mathbf{s}^\top [\mathbf{A} \mid (\mathbf{B} + \mathbf{i}^\top \otimes \mathbf{G})\mathbf{H}_{S,\mathbf{i}}] + [\mathbf{e}_1^\top \mid \mathbf{e}_2^\top \mathbf{H}_{S,\mathbf{i}}])\mathbf{r}_S \\
&= \mathbf{s}^\top [\mathbf{A} \mid \mathbf{B}_S]\mathbf{r}_S + [\mathbf{e}_1^\top \mid \mathbf{e}_1^\top \mathbf{X}\mathbf{H}_{S,\mathbf{i}}]\mathbf{r}_S \\
&= \mathbf{s}^\top \mathbf{d} + \widetilde{\mathbf{e}}^\top
\end{aligned}$$

where $\widetilde{\mathbf{e}}^\top = [\mathbf{e}_1^\top \mid \mathbf{e}_1^\top \mathbf{X}\mathbf{H}_{S,\mathbf{i}}]\mathbf{r}_S$. Thus, $\lceil [\mathbf{c}_1^\top \mid \mathbf{c}_2^\top \mathbf{H}_{S,\mathbf{i}}]\mathbf{r}_S \rfloor_2 = \lceil \mathbf{s}^\top \mathbf{d} \rfloor_2$ as long as $\left\|\widetilde{\mathbf{e}}^\top\right\| < q/4$. We have that $c_3 = \lceil \mathbf{s}^\top \mathbf{d} \rfloor_2 + \mu$ with high probability if $q > \lambda^{\omega(1)} \left\|\widetilde{\mathbf{e}}^\top\right\|$. Thus, $\mu' = \mu$.

We will now bound the norm of $\widetilde{\mathbf{e}}^\top$. We have that $\|\mathbf{r}_S\| = O(\sqrt{m}\sigma)$. Also, $\left\|\mathbf{e}_1^\top\right\|$, with high probability, is at most $\sqrt{m}\chi$ and $\|\mathbf{H}_{S,\mathbf{i}}\| \leq m^{O(\lambda)}$, $\left\|\mathbf{e}_1^\top \mathbf{X}\mathbf{H}_{S,\mathbf{i}}\right\| \leq \lceil \log N \rceil \chi m^{O(\lambda)}$. Thus, $\left\|[\mathbf{e}_1^\top \mid \mathbf{e}_1^\top \mathbf{X}\mathbf{H}_{S,\mathbf{i}}]\right\| \leq \lceil \log N \rceil \chi m^{O(\lambda)}$. Hence, $\left\|\widetilde{\mathbf{e}}^\top\right\| \leq O(\lceil \log N \rceil \chi \sigma m^{O(\lambda)})$. Thus for any $q > \lambda^{\omega(1)} O(\lceil \log N \rceil \chi \sigma m^{O(\lambda)})$, correctness holds.

Compactness. Since $\mathsf{sk}_S = (\mathbf{r}_S, \mathbf{B}) \in \mathbb{Z}_q^{2m} \times \mathbb{Z}_q^{n \times \lceil \log N \rceil m}$, $|\mathsf{sk}_S| = \Theta(m \log q + nm \lceil \log N \rceil \log q) = \mathsf{poly}(\lambda, \log N)$. Thus, compactness holds.

9.3 Security Analysis

Theorem 9.4 (Mode Indistinguishability)**.** If 1-key LWE assumption (Assumption 9.1) holds then Construction 9.2 satisfies mode indistinguishability property.

Proof Sketch. We define the following hybrids.

Hybrid 0. This is the real experiment from Definition 6.1 with $b = 0$. That is, the $\mathcal{A}$ provides a programming dictionary and an index i^*.

Hybrid 1. In this hybrid, we will change ct_{i^*}'s $\mathbf{c}_1^\top, \mathbf{c}_2^\top$ to be $\mathbf{s}^\top \mathbf{A} + \mathbf{e}_1^\top$ and $\mathbf{s}^\top(\mathbf{B} + \mathbf{i}^* \otimes \mathbf{G}) + \mathbf{e}_2^\top$ respectively for some randomly sampled $\mathbf{s}^\top, \mathbf{e}_1^\top, \mathbf{e}_2^\top$ from appropriate distribution. Indistinguishability of hybrids follow by 1-key LWE assumption.

Hybrid 2. In this hybrid, we will change c_3 of ct_{i^*} to be $\lceil \mathbf{s}^\top \mathbf{d} \rfloor_2 + \mu^*_{i^*}$. As $\mu^*_{i^*}$ is chosen randomly and independently from other components, Hybrids 1 and 2 are identically distributed. Note that this is the mode indistinguishability game with $b = 1$.

Full proof is provided in Appendix G. □

Theorem 9.5 (Payload Hiding)**.** If LWE assumption (Assumption 3.3) holds then Construction 9.2 satisfies payload hiding property.

Proof Sketch. We define the following hybrids.

Hybrid 0. This is the real experiment from Definition 6.1 with $b = 0$.

Hybrid 1. In this hybrid, we will set $\mathbf{A}$ to be sampled uniformly randomly, $\mathbf{B} = \mathbf{AX} - (\mathbf{i}^*)^\top \otimes \mathbf{G}$ where $\mathbf{X} \leftarrow \{\pm 1\}^{m \times \lceil \log N \rceil m}$. As $i^* \notin S^*$, $\mathsf{mem}\text{-}S^*(i^*) = 1$. Thus, $(\mathbf{B} + (\mathbf{i}^*)^\top \otimes \mathbf{G})\mathbf{H}_{S^*,\mathbf{i}^*} = \mathbf{B}_{S^*} + \mathbf{G}$. Thus, $[(\mathbf{X}\mathbf{H}_{S^*,\mathbf{i}^*})^\top \mid -\mathbf{I}]^\top$ acts as a gadget trapdoor for $[\mathbf{A} \mid \mathbf{B}_{S^*}]$. So, we can efficiently sample pre-image $\mathbf{r}_{S^*}$ without a trapdoor for $\mathbf{A}$. Hybrids 0 and 1 are statistically close by properties of gadget trapdoors as long as $\sigma > m^{O(\lambda)} \log n$.

Hybrid 2. In this hybrid, we will set $\mathbf{c}^\top_{1,i^*} = \mathbf{s}^\top_{i^*}\mathbf{A} + \mathbf{e}^\top_{1,i^*}, \mathbf{c}^\top_{2,i^*} = \mathbf{c}^\top_{1,i^*}\mathbf{X}, \mathbf{c}^\top_{3,i^*} = \lceil \mathbf{s}^\top \mathbf{d} + \mathbf{e}^\top_{3,i^*} \rfloor_2 + \mu^*$ for appropriate $\mathbf{s}, \mathbf{e}_{1,i^*}, \mathbf{e}_{3,i^*}$. Hybrids 1 and 2 are indistinguishable by LWE assumption.

Full proof is provided in Appendix G. □

Corollary 9.6. Assuming the hardness of $2m^2$-decomposed LWE and 1-key LWE, there exists an adaptively secure DBE scheme for N users in the standard model. Moreover, the public-key, secret-key and ciphertext sizes are independent of N and crs size grows proportional to N.

Proof. Theorems 9.4 and 9.5 give us a PPS scheme from 1-key LWE where ciphertext sizes are independent of N. Using this, Theorems 7.2 and 7.3 give us an EqDBE scheme in the standard model with optimal public-secret keys and ciphertexts. The size of the crs in this scheme grows with N as we sample $2N$ different ciphertexts. Theorems 4.3 and 3.11 then gives us an adaptively secure DBE scheme. □

Acknowledgments. We would like to thank the anonymous reviewers of Crypto 2026 conference for their useful comments about the editorial quality and pointers towards related works on our PPS tool.

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.

[ABI+23] Benny Applebaum, Amos Beimel, Yuval Ishai, Eyal Kushilevitz, Tianren Liu, and Vinod Vaikuntanathan. Succinct computational secret sharing. In Barna Saha and Rocco A. Servedio, editors, *55th Annual ACM Symposium on Theory of Computing*, pages 1553–1566, Orlando, FL, USA, June 20–23, 2023. ACM Press.

[AFP25] Amit Agarwal, Rex Fernando, and Benny Pinkas. Efficiently-thresholdizable batched identity based encryption, with applications. 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 69–100, Santa Barbara, CA, USA, August 17–21, 2025. Springer, Cham, Switzerland.

[AGHS13] Shweta Agrawal, Craig Gentry, Shai Halevi, and Amit Sahai. Discrete Gaussian leftover hash lemma over infinite domains. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part I*, volume 8269 of *Lecture Notes in Computer Science*, pages 97–116, Bengalore, India, December 1–5, 2013. Springer Berlin Heidelberg, Germany.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th Annual ACM Symposium on Theory of Computing*, pages 99–108, Philadephia, PA, USA, May 22–24, 1996. ACM Press.

[AKY25] Shweta Agrawal, Simran Kumari, and Shota Yamada. Pseudorandom fe, io and applications. In *Theory of Cryptography: 23rd International Conference, TCC 2025, Aarhus, Denmark, December 1–5, 2025, Proceedings, Part II*, page 222–258, Berlin, Heidelberg, 2025. Springer-Verlag.

[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 nondeterministic computation: Shift-hiding functions, compressed LWE sampling, broadcast encryption, and obfuscation. Cryptology ePrint Archive, Paper 2026/741, 2026. Version dated June 11, 2026.

[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.

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

[AR13] Divesh Aggarwal and Oded Regev. A note on discrete gaussian combinations of lattice vectors. *arXiv preprint arXiv:1308.2405*, 2013.

[BGG+14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 533–556, Copenhagen, Denmark, May 11–15, 2014. Springer Berlin Heidelberg, Germany.

[BLT25] Dan Boneh, Evan Laufer, and Ertem Nusret Tas. Threshold batch identity-based encryption without epochs. Cryptology ePrint Archive, Paper 2025/1254, 2025.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press.

[BÜW24] Chris Brzuska, Akin Ünal, and Ivy K. Y. Woo. Evasive LWE assumptions: Definitions, classes, and counterexamples. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology – ASIACRYPT 2024, Part IV*, volume 15487 of *Lecture Notes in Computer Science*, pages 418–449, Kolkata, India, December 9–13, 2024. Springer, Singapore, Singapore.

[BV15] Zvika Brakerski and Vinod Vaikuntanathan. Constrained key-homomorphic PRFs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015: 12th Theory of Cryptography Conference, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 1–30, Warsaw, Poland, March 23–25, 2015. Springer Berlin Heidelberg, Germany.

[BZ14] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 480–499, Santa Barbara, CA, USA, August 17–21, 2014. Springer Berlin Heidelberg, Germany.

[CGPP24] Arka Rai Choudhuri, Sanjam Garg, Julien Piet, and Guru-Vamsi Policharla. Mempool privacy via batched threshold encryption: Attacks and defenses. In Davide Balzarotti and Wenyuan Xu, editors, *USENIX Security 2024: 33rd USENIX Security Symposium*, Philadelphia, PA, USA, August 14–16, 2024. USENIX Association.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 523–552, French Riviera, May 30 – June 3, 2010. Springer Berlin Heidelberg, Germany.

[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.

[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.

[DGI+19] Nico Döttling, Sanjam Garg, Yuval Ishai, Giulio Malavolta, Tamer Mour, and Rafail Ostrovsky. Trapdoor hash functions and their applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 3–32, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Cham, Switzerland.

[DJM+25] Nico Döttling, Abhishek Jain, Giulio Malavolta, Surya Mathialagan, and Vinod Vaikuntanathan. Simple and general counterexamples for private-coin evasive LWE. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology – CRYPTO 2025, Part VII*, volume 16006 of *Lecture Notes in Computer Science*, pages 73–92, Santa Barbara, CA, USA, August 17–21, 2025. Springer, Cham, Switzerland.

[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.

[FJP15] Georg Fuchsbauer, Zahra Jafargholi, and Krzysztof Pietrzak. A quasipolynomial reduction for generalized selective decryption on trees. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 601–620, Santa Barbara, CA, USA, August 16–20, 2015. Springer Berlin Heidelberg, Germany.

[FKPR14] Georg Fuchsbauer, Momchil Konstantinov, Krzysztof Pietrzak, and Vanishree Rao. Adaptive security of constrained PRFs. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 82–101, Kaoshiung, Taiwan, R.O.C., December 7–11, 2014. 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.

[GHMR18] Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ahmadreza Rahimi. Registration-based encryption: Removing private-key generator from IBE. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018: 16th Theory of Cryptography Conference, Part I*, volume 11239 of *Lecture Notes in Computer Science*, pages 689–718, Panaji, India, November 11–14, 2018. Springer, Cham, Switzerland.

[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.

[GKW18] Rishab Goyal, Venkata Koppula, and Brent Waters. Collusion resistant traitor tracing from learning with errors. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *50th Annual ACM Symposium on Theory of Computing*, pages 660–670, Los Angeles, CA, USA, June 25–29, 2018. ACM Press.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006: 13th Conference on Computer and Communications Security*, pages 89–98, Alexandria, Virginia, USA, October 30 – November 3, 2006. ACM Press. Available as Cryptology ePrint Archive Report 2006/309.

[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.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 75–92, Santa Barbara, CA, USA, August 18–22, 2013. Springer Berlin Heidelberg, Germany.

[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.

[GV22] Rishab Goyal and Vinod Vaikuntanathan. Locally verifiable signature and key aggregation. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part II*, volume 13508 of *Lecture Notes in Computer Science*, pages 761–791, Santa Barbara, CA, USA, August 15–18, 2022. 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.

[GY26] Rishab Goyal and Saikumar Yadugiri. Adaptively-secure flexible and identity-based broadcast encryption from decomposed LWE. Cryptology ePrint Archive, Paper 2026/862, 2026.

[HJL25] Yao-Ching Hsieh, Aayush Jain, and Huijia Lin. Lattice-based post-quantum iO from circular security with random opening assumption. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology – CRYPTO 2025, Part VII*, volume 16006 of *Lecture Notes in Computer Science*, pages 3–38, Santa Barbara, CA, USA, August 17–21, 2025. Springer, Cham, Switzerland.

[HJO+16] Brett Hemenway, Zahra Jafargholi, Rafail Ostrovsky, Alessandra Scafuro, and Daniel Wichs. Adaptively secure garbled circuits from one-way functions. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 149–178, Santa Barbara, CA, USA, August 14–18, 2016. Springer Berlin Heidelberg, Germany.

[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.

[HW09] Susan Hohenberger and Brent Waters. Short and stateless signatures from the RSA assumption. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 654–670, Santa Barbara, CA, USA, August 16–20, 2009. Springer Berlin Heidelberg, Germany.

[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.

[JKK+17] Zahra Jafargholi, Chethan Kamath, Karen Klein, Ilan Komargodski, Krzysztof Pietrzak, and Daniel Wichs. Be adaptive, avoid overcommitting. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 133–163, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Cham, Switzerland.

[JW16] Zahra Jafargholi and Daniel Wichs. Adaptive security of Yao's garbled circuits. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B: 14th Theory of Cryptography Conference, Part I*, volume 9985 of *Lecture Notes in Computer Science*, pages 433–458, Beijing, China, October 31 – November 3, 2016. Springer Berlin Heidelberg, Germany.

[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.

[KW20] Lucas Kowalczyk and Hoeteck Wee. Compact adaptively secure ABE for NC^1 from k-Lin. *Journal of Cryptology*, 33(3):954–1002, July 2020.

[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.

[Nao03] Moni Naor. On cryptographic assumptions and challenges (invited talk). In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 96–109, Santa Barbara, CA, USA, August 17–21, 2003. Springer Berlin Heidelberg, Germany.

[Nie02] Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 111–126, Santa Barbara, CA, USA, August 18–22, 2002. Springer Berlin Heidelberg, Germany.

[Pei10] Chris Peikert. An efficient and parallel Gaussian sampler for lattices. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 80–97, Santa Barbara, CA, USA, August 15–19, 2010. Springer Berlin Heidelberg, Germany.

[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.

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473, Aarhus, Denmark, May 22–26, 2005. Springer Berlin Heidelberg, Germany.

[Tsa22] Rotem Tsabary. Candidate witness encryption from lattice techniques. In *Annual International Cryptology Conference*, pages 535–559. Springer, 2022.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636, Santa Barbara, CA, USA, August 16–20, 2009. Springer Berlin Heidelberg, Germany.

[Wee22] Hoeteck Wee. Optimal broadcast encryption and cp-abe from evasive lattice assumptions. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 217–241. Springer, 2022.

[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.

[ZZGQ23] Ziqi Zhu, Kai Zhang, Junqing Gong, and Haifeng Qian. Registered ABE via predicate encodings. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology – ASIACRYPT 2023, Part V*, volume 14442 of *Lecture Notes in Computer Science*, pages 66–97, Guangzhou, China, December 4–8, 2023. Springer, Singapore, Singapore.

A Additional Lattice Preliminaries

In this section, we provide additional lattice preliminaries that are useful for our programmable pseudorandom set sampler construction. Specifically, we provide preliminaries on gadget trapdoors [Ajt96, GPV08, MP12], lattice homomorphic computation [GSW13, BGG+14], and other relevant lemmas.

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 $\Lambda_q^{\mathbf{y}}(\mathbf{A}) = \{\mathbf{x} : \mathbf{Ax} = \mathbf{y} \mod q, \mathbf{x} \in \mathbb{Z}^m\}$.

Lemma A.1 ([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 $\mathbf{A}$, $\Pr[\|\mathbf{x}\| > \sqrt{m}\sigma : \mathbf{x} \leftarrow \mathbf{A}_\sigma^{-1}(\mathbf{y})] \leq \mathsf{negl}(m)$.

Function Class $\mathcal{F}_{\ell,d}$. The ensemble function class $\mathcal{F}_{\ell,d} = \{\mathcal{F}_{\ell,d}\}_{\ell,d \in \mathbb{N}^2}$ consists of the set of all functions that take ℓ-bit inputs, one-bit outputs, and can be computed by a depth-d boolean circuit.

Homomorphic Computation. Let (n, m, q) be LWE parameters such that $m \geq 3n \log q$. The lattice homomorphic computation machinery from [GSW13, BGG+14] for function class $\mathcal{F}_{\ell,d}$ provides two *deterministic* polynomial-time algorithms described as follows.

$\mathsf{EvalF}(\mathbf{B}, f) \rightarrow \mathbf{B}_f$**.** The function evaluation algorithm takes as input $\mathbf{B} \in \mathbb{Z}_q^{n \times \ell m}$, function $f \in \mathcal{F}_{\ell,d}$, and outputs matrix $\mathbf{B}_f \in \mathbb{Z}_q^{n \times m}$.

$\mathsf{EvalFX}(\mathbf{B}, f, \mathbf{x}) \rightarrow \mathbf{H}_{f,\mathbf{x}}$**.** The function-input evaluation algorithm takes as input $\mathbf{B} \in \mathbb{Z}_q^{n \times \ell m}$, function $f \in \mathcal{F}_{\ell,d}$, input $\mathbf{x} \in \{0,1\}^\ell$, and outputs matrix $\mathbf{H}_{f,\mathbf{x}} \in \mathbb{Z}_q^{\ell m \times m}$.

These algorithms satisfy these two properties: for any $\mathbf{B}$, $f \in \mathcal{F}_{\ell,d}, \mathbf{x} \in \{0,1\}^\ell$ (1) $(\mathbf{B} + \mathbf{x}^\top \otimes \mathbf{G})\mathbf{H}_{f,\mathbf{x}} = \mathbf{B}_f + f(x)\mathbf{G}$ where $\mathbf{B}_f = \mathsf{EvalF}(\mathbf{B}, f)$, (2) $\|\mathbf{H}_{f,\mathbf{x}}\| \leq m^{O(d)}$.

B Accept LWE Assumption

In this section, we provide the formal description of our Accept LWE (ALWE) assumption and provide useful discussion on its hardness. ALWE is a generalization of the 1-key LWE assumption (Section 9.1). We recommend that the reader peruse Appendix A which contains important definitions relevant to lattice homomorphic computation machinery and trapdoors. We formally the assumption below.

B.1 d-Accept LWE

Assumption B.1. Let (n, m, q, χ) be LWE parameters. Let width parameter $\sigma > m^{O(d)} \log n$. The Accept LWE (ALWE) assumption is defined with respect to the following game between a challenger (Chal) and a PPT adversary ($\mathcal{A}$).

1. $\mathcal{A}(1^\lambda)$ sends input $\mathbf{x}$ to Chal.
2. Chal samples $b \leftarrow \{0,1\}$. Chal then samples $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q), \mathbf{B} \leftarrow \mathbb{Z}_q^{n \times |\mathbf{x}|m}, \mathbf{d} \leftarrow \mathbb{Z}_q^n$. If $b = 0$, Chal samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_1 \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{R}_\mathbf{B} \leftarrow \{\pm 1\}^{m \times |\mathbf{x}|m}$ and sets $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{A} + \mathbf{e}_1$ and $\mathbf{c}_2^\top = \mathbf{s}^\top (\mathbf{B} + \mathbf{x} \otimes \mathbf{G}) + \mathbf{e}_1^\top \mathbf{R}_\mathbf{B}$. Otherwise, if $b = 1$, Chal samples $\mathbf{c}_1 \leftarrow \mathbb{Z}_q^m, \mathbf{c}_2 \leftarrow \mathbb{Z}_q^{|\mathbf{x}|m}$. Chal sends $\mathbf{A}, \mathbf{B}, \mathbf{d}, \mathbf{c}_1^\top, \mathbf{c}_2^\top$ to $\mathcal{A}$.
3. $\mathcal{A}$ sends a function $f \in \mathcal{F}_{|\mathbf{x}|,d}$. Chal computes $\mathbf{B}_f = \mathsf{EvalF}(\mathbf{B}, f)$. Chal also samples $\mathbf{r}_f \leftarrow \mathsf{SamplePre}\left([\mathbf{A} \mid \mathbf{B}_f], \begin{bmatrix} \mathbf{T}_\mathbf{A} \\ \mathbf{0} \end{bmatrix}, \mathbf{d}, \sigma\right)$ and sends $\mathbf{r}_f$ to $\mathcal{A}$.
4. $\mathcal{A}$ sends a bit b'. Output b'.

We say that ALWE assumption holds if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any $\lambda \in \mathbb{N}$,

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

If $d = \mathsf{poly}(\lambda)$, we simply call this ALWE assumption. We remark that the assumption is a falsifiable assumption.

B.2 Discussion on Hardness of Accept LWE

Two Cases. Note that $f(\mathbf{x}) = 0$ (accept in lattice literature) or $f(\mathbf{x}) = 1$ (reject in lattice literature). If f and $\mathbf{x}$ are such that $f(\mathbf{x}) = 1$ always, then we can reduce the assumption to polynomially-hard LWE. The reduction ($\mathcal{B}$) works as follows: the challenger Chal gives $\mathbf{A}', \mathbf{c}^\top$ to $\mathcal{B}$ such that $\mathbf{c}^\top \approx \mathbf{s}^\top \mathbf{A}'$ for some $\mathbf{s} \in \mathbb{Z}_q^n$ or $\mathbf{c}^\top \leftarrow \mathbb{Z}_q^m$. $\mathcal{B}$ sets $\mathbf{A} := \mathbf{A}'$ and $\mathbf{B} := \mathbf{A}'\mathbf{R}_\mathbf{B} - \mathbf{x} \otimes \mathbf{G}$ after receiving $\mathbf{x}$ from $\mathcal{A}$ where $\mathbf{R}_\mathbf{B} \leftarrow \{\pm 1\}^{m \times |\mathbf{x}|m}$. $\mathcal{B}$ will sample $\mathbf{d} \leftarrow \mathbb{Z}_q^n$, set $\mathbf{c}_1^\top := \mathbf{c}^\top, \mathbf{c}_2^\top := \mathbf{c}^\top \mathbf{R}_\mathbf{A}$, and send $\mathbf{A}, \mathbf{B}, \mathbf{d}, \mathbf{c}_1^\top, \mathbf{c}_2^\top$ to $\mathcal{A}$. Note that if $\mathbf{c}^\top \approx \mathbf{s}^\top \mathbf{A}'$, then $\mathbf{c}_2^\top \approx \mathbf{s}^\top \mathbf{A}'\mathbf{R}_\mathbf{A} = \mathbf{s}^\top (\mathbf{B} + \mathbf{x}^\top \otimes \mathbf{G})$ and uniformly random and independent of $\mathbf{B}$ otherwise. Then when $\mathcal{A}$ sends $f \in \mathcal{F}_{|\mathbf{x}|,d}$ such that $f(\mathbf{x}) = 1$, $\mathcal{B}$ will compute $\mathbf{H}_{f,\mathbf{x}} = \mathsf{EvalFX}(\mathbf{B}, f, \mathbf{x})$ and $\mathbf{B}_f = \mathbf{A}'\mathbf{R}_\mathbf{B}\mathbf{H}_{f,\mathbf{x}} - \mathbf{G}$. Now, clearly $[\mathbf{A} \mid \mathbf{B}_f] = [\mathbf{A}' \mid \mathbf{A}'\mathbf{R}_\mathbf{A}\mathbf{H}_{f,\mathbf{x}} - \mathbf{G}]$ has a trapdoor [ABB10, CHKP10, MP12]. So, we can reduce d-ALWE to LWE if $f(\mathbf{x}) = 1$ (reject). The assumption only seems to be non-trivial and doesn't follow from LWE only if $f(\mathbf{x}) = 0$ (accept).

1-key LWE. It is easy to see that the 1-key LWE assumption (Assumption 9.1) is a special case of λ-ALWE. As mem-S can be computed using an NC^1 circuit, for large enough $\lambda \in \mathbb{N}$, we can upper bound the depth of mem-S circuit by λ. Moreover, $\mathbf{x}$ can be the one-hot vector $\mathbb{1}_{N,i}$ to denote $1^N, i$. Thus, this assumption is a special case of ALWE.

Less Interesting Variants. Consider a variation of the assumption in which $\mathcal{A}$ provides f along with $\mathbf{x}$, i.e, selectively. In this case, we can figure out $\mathbf{B}_f$ early. Then we can sample $\mathbf{r}_f \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{2m}$ and set $\mathbf{d} = [\mathbf{A} \mid \mathbf{B}_f]\mathbf{r}_f$ and provide $\mathbf{A}, \mathbf{B}, \mathbf{d}, \mathbf{c}_1^\top, \mathbf{c}_2^\top, \mathbf{r}_f$ together. This distribution is statistically close to the distribution of $[\mathbf{A} \mid \mathbf{B}_f]_\sigma^{-1}(\mathbf{d})$ and $\mathbf{d} \leftarrow \mathbb{Z}_q^n$ (by pre-image distribution property [GPV08, MP12]) for sufficiently large σ. Thus, the assumption is only interesting if f is chosen adaptively after viewing the public matrices.

Consider the variant when the parameter sizes ($\|\mathbf{r}_f\|, |\mathbf{c}_1^\top|$, etc.) are allowed to grow with the description size of f ($\|\mathbf{r}_f\| \propto |f|$). Then, we can simply guess f selectively and reduce the assumption to sub-exponentially secure LWE. However, this would be a problem for our application of programmable pseudorandom set samplers. We require that $\|\mathbf{r}_S\|$ to be independent of N for our compactness property. By guessing set S, we blow up the parameter sizes. Hence, this variant is not relevant and uninteresting for our application.

Relation to Public-Coin Evasive LWE. Evasive LWE assumption [Wee22, Tsa22] is a non-falsifiable LWE-style assumption which posits that

$$\text{If } \{\mathbf{A}, \mathbf{B}, \mathbf{d}, \mathsf{aux}, \mathbf{s}^\top\mathbf{B} + \mathbf{e}_1^\top, \mathbf{s}^\top\mathbf{A} + \mathbf{e}_2^\top, \mathbf{s}^\top\mathbf{D} + \mathbf{e}_3^\top\} \approx_c \{\mathbf{A}, \mathbf{B}, \mathbf{d}, \mathbf{c}_1^\top, \mathbf{c}_2^\top, \mathbf{c}_3^\top\}$$
$$\text{then } \{\mathbf{A}, \mathbf{B}, \mathbf{D}, \mathsf{aux}, \mathbf{s}^\top\mathbf{B} + \mathbf{e}_1^\top, \mathbf{s}^\top\mathbf{A} + \mathbf{e}_2^\top, \mathbf{R}\} \approx_c \{\mathbf{A}, \mathbf{B}, \mathbf{D}, \mathsf{aux}, \mathbf{c}_1^\top, \mathbf{c}_2^\top, \mathbf{R}\}$$

where $(\mathbf{B}, \mathbf{D}, \mathsf{aux}) \leftarrow \mathsf{Samp}(1^\lambda)$, $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{c}_1 \leftarrow \mathbb{Z}_q^{m'}, \mathbf{c}_2, \mathbf{c}_3 \leftarrow \mathbb{Z}_q^m$, $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ are discrete gaussian error vectors from appropriate distributions, and $\mathbf{R} = \mathsf{SamplePre}(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{D}, \sigma)$ for large enough $\sigma = \omega(\sqrt{m \log m})$. 'Public-coin' evasive LWE is a variation of above assumption when aux contains all the random coins used by Samp[10].

ALWE has a similar structure to public-coin evasive LWE. In both assumptions, we give out multiple LWE instances and a pre-image for a related matrix. However, our assumption is falsifiable [Nao03] and roughly corresponds to the post-condition in Evasive LWE. Moreover, our assumption is *interactive*. That is, adversary demands a pre-image for a matrix that can be adaptively chosen instead of statically choosing it. Thus, it is unclear how to reduce our assumption to public-coin evasive LWE.

C Proof of Theorem 4.3

We define the following hybrids.

$\mathbf{Hyb}_0^\mathcal{A}(1^\lambda)$. This is the experiment from Definition 3.9 with $b = 0$.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N]$. Sample $\mathsf{dbe.crs} \leftarrow \mathsf{EqDBE.GSetup}(1^\lambda, 1^N)$. Set $\mathsf{crs} := \mathsf{dbe.crs}$. For each $i \in S'$, sample $(\mathsf{dbe.pk}_i, \mathsf{dbe.sk}_i) \leftarrow \mathsf{EqDBE.KGen}(i)$. Set $\mathsf{pk}_i := \mathsf{dbe.pk}_i$. Send $\mathsf{crs}, \{\mathsf{pk}_i\}_{i \in S'}$ to $\mathcal{A}$.
- **Encryption Phase.** $\mathcal{A}$ sends $S^* \subseteq S'$. Sample $\mathsf{dbe.ct} \leftarrow \mathsf{EqDBE.Enc}(\{\mathsf{dbe.pk}_i\}_{i \in S^*}, 0)$. Send $\mathsf{ct} := \mathsf{dbe.ct}$ to $\mathcal{A}$.

[10] We remark that recent works have shown counter-examples against Evasive LWE [BÜW24, AKY25, AMYY25, DJM+25, HJL25]. In particular [AMYY25] provides an attack against public-coin evasive LWE in the contrived parameter regime. These attacks are not applicable for the parameter regime we consider. Thus, it is still relevant to discuss the relation between public-coin evasive LWE and ALWE.

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

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

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N]$. Sample $(\mathsf{dbe.}\widetilde{\mathsf{crs}}, \{\mathsf{dbe.}\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \mathsf{dbe.}\widetilde{\mathsf{ct}}, \mathsf{dbe.aux}) \leftarrow \mathsf{EqDBE.FakeSetup}(1^\lambda, 1^N, S', 0)$. Set $\mathsf{crs} := \mathsf{dbe.}\widetilde{\mathsf{crs}}$. Set $\mathsf{pk}_i := \mathsf{dbe.}\widetilde{\mathsf{pk}}_i$ for each $i \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_i\}_{i\in S'}$ to $\mathcal{A}$.
- **Encryption Phase.** $\mathcal{A}$ sends $S^* \subseteq S'$. Sample $\mathsf{dbe.ct} \leftarrow \mathsf{EqDBE.Equivocate}(\mathsf{dbe.aux}, S^*)$. Send $\mathsf{ct} := \mathsf{dbe.ct}$ to $\mathcal{A}$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$**.** In this hybrid, we will encrypt 1 using $\mathsf{EqDBE.Equivocate}$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N]$. Sample $(\mathsf{dbe.}\widetilde{\mathsf{crs}}, \{\mathsf{dbe.}\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \mathsf{dbe.}\widetilde{\mathsf{ct}}, \mathsf{dbe.aux}) \leftarrow \mathsf{EqDBE.FakeSetup}(1^\lambda, 1^N, S', 1)$. Set $\mathsf{crs} := \mathsf{dbe.}\widetilde{\mathsf{crs}}$. Set $\mathsf{pk}_i := \mathsf{dbe.}\widetilde{\mathsf{pk}}_i$ for each $i \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_i\}_{i\in S'}$ to $\mathcal{A}$.
- **Encryption Phase.** $\mathcal{A}$ sends $S^* \subseteq S'$. Sample $\mathsf{dbe.ct} \leftarrow \mathsf{EqDBE.Equivocate}(\mathsf{dbe.aux}, S^*)$. Send $\mathsf{ct} := \mathsf{dbe.ct}$ to $\mathcal{A}$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$**.** This is the experiment from Definition 3.9 with $b = 1$.

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

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 $\epsilon(\cdot)$. Then, we will construct an adversary $\mathcal{B}$ that can break the mode-indistinguishability property of EqDBE with probability ϵ.

$\mathcal{B}$ interacts with $\mathcal{A}(1^\lambda)$ and receives $1^N, S'$ from $\mathcal{A}$. $\mathcal{B}$ passes $(1^N, 1^{S'}, 0)$ to the challenger Chal in the EqDBE mode indistinguishability game. Chal sends $\mathsf{dbe.crs}, \{\mathsf{dbe.pk}_i\}_{i\in S'}$ to $\mathcal{B}$. $\mathcal{B}$ sets $\mathsf{crs} := \mathsf{dbe.crs}, \mathsf{pk}_i = \mathsf{dbe.pk}_i$ for each $i \in S'$ and sends $\mathsf{crs}, \{\mathsf{pk}_i\}_{i\in S'}$ to $\mathcal{A}$. When $\mathcal{A}$ sends S^*, $\mathcal{B}$ sends S^* to Chal to receive $\mathsf{dbe.ct}$. $\mathcal{B}$ passes this as ct to $\mathcal{A}$. $\mathcal{B}$ outputs the guess of $\mathcal{A}$ as its guess.

If Chal's coin toss $b = 0$, $\mathcal{B}$ behaves like $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ and if $b = 1$, $\mathcal{B}$ behaves like $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. Thus, advantage of $\mathcal{B}$ is also ϵ. So, it must be the case that if EqDBE satisfies mode indistinguishability, any PPT adversary only has negligible advantage in distinguishing these hybrids. □

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

Proof. 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 $\epsilon(\cdot)$. Then, we will construct an adversary $\mathcal{B}$ that can break the selective security for fake setup property of EqDBE with probability ϵ.

$\mathcal{B}$ interacts with $\mathcal{A}(1^\lambda)$ and receives $1^N, S'$ from $\mathcal{A}$. $\mathcal{B}$ passes this to the challenger Chal in the EqDBE selective security for fake setup game. Chal sends $(\mathsf{dbe.}\widetilde{\mathsf{crs}}, \{\mathsf{dbe.}\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \mathsf{dbe.}\widetilde{\mathsf{ct}}, \mathsf{dbe.aux})$ to $\mathcal{B}$. $\mathcal{B}$ sets $\mathsf{crs} := \mathsf{dbe.}\widetilde{\mathsf{crs}}, \mathsf{pk}_i = \mathsf{dbe.}\widetilde{\mathsf{pk}}_i$ for each $i \in S'$ and sends $\mathsf{crs}, \{\mathsf{pk}_i\}_{i\in S'}$ to $\mathcal{A}$. When $\mathcal{A}$ sends S^*, $\mathcal{B}$ samples $\mathsf{dbe.ct} \leftarrow \mathsf{Equivocate}(\mathsf{dbe.aux}, S^*)$ and sends this to $\mathcal{A}$. $\mathcal{B}$ outputs the guess of $\mathcal{A}$ as its guess.

If Chal's 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 ϵ. So, it must be the case that if EqDBE satisfies selective security for fake setup, any PPT adversary only has negligible advantage in distinguishing these hybrids. □

Lemma C.3. 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. This is similar to proof of Lemma C.1. □

D Proofs from Section 5

D.1 Proof of Theorem 5.2

We define the following hybrids.

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

- **Setup Phase.** $\mathcal{A}$ sends 1^N and $S' \subseteq [N], \mu \in \{0,1\}$. Sample $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$, $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$. Also, sample hash functions $\mathcal{H}_1 : \{0,1\}^\lambda \times [N] \to \{\pm 1\}^{4m^5}, \mathcal{H}_2 : \{0,1\}^\lambda \times [N] \to \mathbb{Z}_q^n$ which are modeled as random oracles. Set $\mathsf{crs} := (N, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, \mathcal{H}_1, \mathcal{H}_2)$. Here, $\mathcal{H}_1, \mathcal{H}_2$ are handles used for random oracle queries.

  Compute $\mathbf{V}_N = \mathsf{MCOM.Verify}(1^N)$. For each $i \in S'$, sample $\mathbf{r}_i \leftarrow \{\pm 1\}^{4m^5}$, $\mathbf{t}_i = \mathbf{B}\mathbf{r}_i + \mathbf{p} - \mathbf{A}\mathbf{v}_i$ where $\mathbf{v}_i$ is the i-th column of $\mathbf{V}_N$. Set $\mathsf{pk}_i = (i, \mathbf{t}_i)$ for each $i \in S'$.

  Send $\mathsf{crs}, \{\mathsf{pk}_i\}_{i\in S'}$ to $\mathcal{A}$.

- **Encryption Phase.** $\mathcal{A}$ sends $S^* \subseteq S'$. Sample $\mathsf{sd}_1, \mathsf{sd}_2 \leftarrow \{0,1\}^\lambda$.

  1. For each $j \in [N]$, if $j \in S^*$, let $\mathbf{x}_j = \mathcal{H}_1(\mathsf{sd}_1, j)$ and if $j \notin S^*$, $\mathbf{d}_j = \mathcal{H}_2(\mathsf{sd}_2, j)$.
  2. Construct an $n \times N$ matrix $\mathbf{M}^*$ where the j-th column $\mathbf{m}_j^* = \mathbf{t}_j + \mathbf{B}\mathbf{x}_j$ if $j \in S^*$. Otherwise, $\mathbf{m}_j^* = \mathbf{d}_j$.
  3. Compute $\mathbf{C}^* = \mathsf{MCOM.Com}(\mathbf{M}^*)$.
  4. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R}_\mathbf{A} \leftarrow \{\pm 1\}^{4m^5\times m}, \mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$.
  5. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top\mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top(\mathbf{A} + \mathbf{C}^*) + \mathbf{e}^\top\mathbf{R}_\mathbf{A}, c_3 = \mathbf{s}^\top\mathbf{p} + \mathbf{e}^\top\mathbf{r}_\mathbf{p} + \lfloor q/2 \rfloor \mu$.

  Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{sd}_1, \mathsf{sd}_2)$ 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 maintain a dictionary H that maps random oracle queries to inputs sampled on-the-fly. The changes are highlighted in red.

- **Setup Phase.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. In addition, initiate an empty dictionary H.
- **Random Oracle Queries.** Any query to $\mathcal{H}_1$ with input $(\mathsf{sd}, i) \in \{0,1\} \times [N]$ is handled as follows: if H has entry $(\mathsf{sd}, i, 1)$, return $\mathsf{H}[(\mathsf{sd}, i, 1)]$. Otherwise, sample $\mathbf{x} \leftarrow \{\pm 1\}^{4m^5}$, add $\mathsf{H}[(\mathsf{sd}, i, 1)] = \mathbf{x}$ to H, and reply $\mathbf{x}$ to $\mathcal{A}$.

  Similarly, query to $\mathcal{H}_2$ with input $(\mathsf{sd}, i) \in \{0,1\} \times [N]$ is handled as follows: if H has entry $(\mathsf{sd}, i, 2)$, return $\mathsf{H}[(\mathsf{sd}, i, 2)]$. Otherwise, sample $\mathbf{d} \leftarrow \mathbb{Z}_q^n$ add $\mathsf{H}[(\mathsf{sd}, i, 2)] = \mathbf{d}$ to H, and reply $\mathbf{d}$ to $\mathcal{A}$.
- **Encryption Phase.** Same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.
- **Random Oracle Queries.** Handle these similarly to previous random oracle queries phase.
- **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 if $\mathcal{A}$ queries $\mathcal{H}_1$ or $\mathcal{H}_2$ with sd_1 or sd_2 respectively in pre-encryption random oracle queries, we abort. The changes are highlighted in red.

- **Encryption Phase.** $\mathcal{A}$ sends $S^* \subseteq S'$. Sample $\mathsf{sd}_1, \mathsf{sd}_2 \leftarrow \{0,1\}^\lambda$. If H contains entries with $(\mathsf{sd}_1, i, 1)$ or $(\mathsf{sd}_2, j, 1)$ for any $i, j \in [N]$, abort and output 0. The rest of the steps are 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 we also sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$ and $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$ during setup phase. Using these, we compute $\mathbf{M}'$ for set S' using the same procedure as encryption phase. We also compute commitment to $\mathbf{M}'$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends 1^N and $S' \subseteq [N], \mu \in \{0,1\}$. Sample $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$, $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{crs} := (N, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, \mathcal{H}_1, \mathcal{H}_2)$. Here, $\mathcal{H}_1, \mathcal{H}_2$ are handles used for random oracle queries. Sample an empty dictionary H.

  Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$ and $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively. Construct matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column of $\mathbf{M}'$, $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$ and $\mathbf{m}'_j = \mathbf{d}'_j$ otherwise. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$.

  Compute $\mathbf{V}_N = \mathsf{MCOM.Verify}(1^N)$. For each $i \in S'$, sample $\mathbf{r}_i \leftarrow \{\pm 1\}^{4m^5}$, $\mathbf{t}_i = \mathbf{B}\mathbf{r}_i + \mathbf{p} - \mathbf{A}\mathbf{v}_i$ where $\mathbf{v}_i$ is the i-th column of $\mathbf{V}_N$. Set $\mathsf{pk}_i = (i, \mathbf{t}_i)$ for each $i \in S'$.

  Send $\mathsf{crs}, \{\mathsf{pk}_i\}_{i \in S'}$ to $\mathcal{A}$.

$\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ except we also sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$ for each $i \in S'$ during setup phase before sample $\mathbf{A}$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends 1^N and $S' \subseteq [N], \mu \in \{0,1\}$. For each $i \in S'$, sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{pk}_i = (i, \mathbf{t}_i)$ for each $i \in S'$. Sample $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$, $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{crs} := (N, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, \mathcal{H}_1, \mathcal{H}_2)$. Here, $\mathcal{H}_1, \mathcal{H}_2$ are handles used for random oracle queries. Sample an empty dictionary H.

Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$ and $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively. Construct matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column of $\mathbf{M}'$, $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$ and $\mathbf{m}'_j = \mathbf{d}'_j$ otherwise. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Set $\mathsf{aux} := (\mathbf{X}', \mathbf{D}', S')$.

Send $\mathsf{crs}, \{\mathsf{pk}_i\}_{i \in S'}$ to $\mathcal{A}$.

$\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ except we program outputs of $\mathsf{H}[(\mathsf{sd}_1, i, 1)]$ and $\mathsf{H}[(\mathsf{sd}_2, i, 2)]$ for each $i \in [N]$ during encryption phase using aux. The changes are highlighted in red.

- **Encryption Phase.** $\mathcal{A}$ sends $S^* \subseteq S'$. Sample $\mathsf{sd}_1, \mathsf{sd}_2 \leftarrow \{0,1\}^\lambda$. If H contains entries with $(\mathsf{sd}_1, i, 1)$ or $(\mathsf{sd}_2, j, 1)$ for any $i, j \in [N]$, abort and output 0. For each $j \in [N]$, if $j \in S^*$, set $\mathsf{H}[(\mathsf{sd}_1, j, 1)] = \mathbf{x}'_j$. Otherwise, If $j \notin S'$, set $\mathsf{H}[(\mathsf{sd}_2, j, 2)] = \mathbf{d}'_j$. Otherwise, if $j \in S' \setminus S^*$, set $\mathsf{H}[(\mathsf{sd}_1, j, 1)] = \mathbf{x}^*_j$ and $\mathsf{H}[(\mathsf{sd}_2, j, 2)] = \mathbf{d}'_j$ where $\mathbf{x}^*_j \leftarrow \{\pm 1\}^{4m^5}$. The rest of the steps are 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 we program outputs of $\mathsf{H}[(\mathsf{sd}_1, j, 2)] = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ for $j \in S' \setminus S^*$. The changes are highlighted in red.

- **Encryption Phase.** $\mathcal{A}$ sends $S^* \subseteq S'$. Sample $\mathsf{sd}_1, \mathsf{sd}_2 \leftarrow \{0,1\}^\lambda$. If H contains entries with $(\mathsf{sd}_1, i, 1)$ or $(\mathsf{sd}_2, j, 1)$ for any $i, j \in [N]$, abort and output 0. For each $j \in [N]$, if $j \in S^*$, set $\mathsf{H}[(\mathsf{sd}_1, j, 1)] = \mathbf{x}'_j$. Otherwise, If $j \notin S'$, set $\mathsf{H}[(\mathsf{sd}_2, j, 2)] = \mathbf{d}'_j$. Otherwise, if $j \in S' \setminus S^*$, set $\mathsf{H}[(\mathsf{sd}_1, j, 1)] = \mathbf{x}^*_j$ and $\mathsf{H}[(\mathsf{sd}_2, j, 2)] = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ where $\mathbf{x}^*_j \leftarrow \{\pm 1\}^{4m^5}$. The rest of the steps are same as $\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$ except we set $\mathbf{M}^* = \mathbf{M}'$ and $\mathbf{C}^* = \mathbf{C}'$ during encryption phase. We write the whole hybrid for clarity. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends 1^N and $S' \subseteq [N], \mu \in \{0,1\}$. For each $i \in S'$, sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{pk}_i = (i, \mathbf{t}_i)$ for each $i \in S'$. Sample $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$, $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{crs} := (N, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, \mathcal{H}_1, \mathcal{H}_2)$. Here, $\mathcal{H}_1, \mathcal{H}_2$ are handles used for random oracle queries. Sample an empty dictionary H.

  Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$ and $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively. Construct matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column of $\mathbf{M}'$, $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$ and $\mathbf{m}'_j = \mathbf{d}'_j$ otherwise. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$.

  Send $\mathsf{crs}, \{\mathsf{pk}_i\}_{i \in S'}$ to $\mathcal{A}$.

- **Random Oracle Queries.** Any query to $\mathcal{H}_1$ with input $(\mathsf{sd}, i) \in \{0,1\} \times [N]$ is handled as follows: if H has entry $(\mathsf{sd}, i, 1)$, return $\mathsf{H}[(\mathsf{sd}, i, 1)]$. Otherwise, sample $\mathbf{x} \leftarrow \{\pm 1\}^{4m^5}$, add $\mathsf{H}[(\mathsf{sd}, i, 1)] = \mathbf{x}$ to H, and reply $\mathbf{x}$ to $\mathcal{A}$.

  Similarly, query to $\mathcal{H}_2$ with input $(\mathsf{sd}, i) \in \{0,1\} \times [N]$ is handled as follows: if H has entry $(\mathsf{sd}, i, 2)$, return $\mathsf{H}[(\mathsf{sd}, i, 2)]$. Otherwise, sample $\mathbf{d} \leftarrow \mathbb{Z}_q^n$ add $\mathsf{H}[(\mathsf{sd}, i, 2)] = \mathbf{d}$ to H, and reply $\mathbf{d}$ to $\mathcal{A}$.

- **Encryption Phase.** $\mathcal{A}$ sends $S^* \subseteq S'$. Sample $\mathsf{sd}_1, \mathsf{sd}_2 \leftarrow \{0,1\}^\lambda$. If H contains entries with $(\mathsf{sd}_1, i, 1)$ or $(\mathsf{sd}_2, j, 1)$ for any $i, j \in [N]$, abort and output 0. For each $j \in [N]$, if $j \in S^*$, set

$\mathsf{H}[(\mathsf{sd}_1, j, 1)] = \mathbf{x}'_j$. Otherwise, If $j \notin S'$, set $\mathsf{H}[(\mathsf{sd}_2, j, 2)] = \mathbf{d}'_j$. Otherwise, if $j \in S' \setminus S^*$, set $\mathsf{H}[(\mathsf{sd}_1, j, 1)] = \mathbf{x}^*_j$ and $\mathsf{H}[(\mathsf{sd}_2, j, 2)] = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ where $\mathbf{x}^*_j \leftarrow \{\pm 1\}^{4m^5}$.

1. Set $\mathbf{M}^* = \mathbf{M}', \mathbf{C}^* = \mathbf{C}'$.
2. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R}_\mathbf{A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$.
3. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top(\mathbf{A} + \mathbf{C}^*) + \mathbf{e}^\top \mathbf{R}_\mathbf{A}, c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r}_\mathbf{p} + \lfloor q/2 \rfloor \mu$.

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

- **Random Oracle Queries.** Handle these similarly to previous random oracle queries phase.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_8^\mathcal{A}(1^\lambda)$. This is same as $\mathbf{Hyb}_7^\mathcal{A}(1^\lambda)$ except we set $\mathbf{A} = \mathbf{A}' - \mathbf{C}'$ during setup phase. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends 1^N and $S' \subseteq [N], \mu \in \{0,1\}$. For each $i \in S'$, sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{pk}_i = (i, \mathbf{t}_i)$ for each $i \in S'$.

  Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$ and $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively. Construct matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column of $\mathbf{M}'$, $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$ and $\mathbf{m}'_j = \mathbf{d}'_j$ otherwise. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$.

  Sample $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$, $\mathbf{A}' \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$, and set $\mathbf{A} = \mathbf{A}' - \mathbf{C}'$. Set $\mathsf{crs} := (N, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, \mathcal{H}_1, \mathcal{H}_2)$. Here, $\mathcal{H}_1, \mathcal{H}_2$ are handles used for random oracle queries. Sample an empty dictionary H.

  Send $\mathsf{crs}, \{\mathsf{pk}_i\}_{i \in S'}$ to $\mathcal{A}$.

$\mathbf{Hyb}_9^\mathcal{A}(1^\lambda)$. This is the mode indistinguishability experiment with $b = 1$.

Lemma D.1. For any adversary $\mathcal{A}$, $\Pr[1 \leftarrow \mathbf{Hyb}_0^\mathcal{A}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_1^\mathcal{A}(1^\lambda)]$.

Proof. As $\mathcal{H}_1$ and $\mathcal{H}_2$ are modeled as random oracles, this follows from the definition of random oracles. □

Lemma D.2. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any PPT 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. As $\mathcal{A}$ is a polynomial-time adversary, there exists a polynomial $Q = Q(\lambda)$ such that $\mathcal{A}$ makes at most Q random oracle queries before encryption phase. By a standard union bound argument, the probability that sd_1 is one of them is at most $\mathsf{poly}(Q)/2^\lambda$. Similarly, probability that sd_2 is one of them is at most $\mathsf{poly}(Q)/2^\lambda$. So, the abort in $\mathbf{Hyb}_2^\mathcal{A}(1^\lambda)$ only happens with at most $\mathsf{poly}(Q)/2^\lambda$ probability. Thus, the lemma holds. □

Lemma D.3. For any adversary $\mathcal{A}$, $\Pr[1 \leftarrow \mathbf{Hyb}_2^\mathcal{A}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_3^\mathcal{A}(1^\lambda)]$.

Proof. Note that view of the adversary remains same in both hybrids. Thus, the lemma follows. □

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 generalized leftover hash lemma (Lemma 3.2). □

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. As $\mathbf{X}'$ and $\mathbf{D}'$ are sampled independently from all other queries, each column of these matrices are also independent and uniformly sampled. Thus, both hybrid distributions remain identical to any adversary. □

Lemma D.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. This follows from the generalized leftover hash lemma (Lemma 3.2). As $\mathbf{x}'_j$ is not revealed to the adversary via random oracle queries, $(\mathbf{B}, \mathbf{B}\mathbf{x}'_j)$ is statistically close to $\mathbf{B}, \mathbf{d}'_j$. □

Lemma D.7. For any adversary $\mathcal{A}$, $\Pr[1 \leftarrow \mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)]$.

Lemma D.8. Note that $\mathbf{M}^*$ is constructed as $\mathbf{m}_j^* = \mathbf{t}_j + \mathbf{B}\mathsf{H}[(\mathsf{sd}_1, j, 1)]$ if $j \in S^*$ and $\mathbf{m}_j^* = \mathsf{H}[(\mathsf{sd}_2, j, 2)]$ if $j \notin S^*$. If $j \in S^*$ then $\mathsf{H}[(\mathsf{sd}_1, j, 1)] = \mathbf{x}'_j$. If $j \notin S'$ (then $j \notin S^*$ as well), then $\mathbf{m}_j^* = \mathsf{H}[(\mathsf{sd}_2, j, 2)] = \mathbf{d}'_j$. If $j \in S' \setminus S^*$, then $j \notin S^*$, $\mathbf{m}_j^* = \mathsf{H}[(\mathsf{sd}_2, j, 2)] = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$. Thus, if $j \in S', \mathbf{m}_j^* = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ and if $j \notin S', \mathbf{m}_j^* = \mathbf{d}'_j$. So, $\mathbf{M}^* = \mathbf{M}'$. As MCOM algorithms are deterministic, $\mathsf{MCOM.Com}(\mathbf{M}^*) = \mathbf{C}^* = \mathsf{MCOM.Com}(\mathbf{M}') = \mathbf{C}'$.

Lemma D.9. For any adversary $\mathcal{A}$, $\Pr[1 \leftarrow \mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_8^{\mathcal{A}}(1^\lambda)]$.

Proof. As $\mathbf{A}'$ is sampled uniformly randomly and independently, $\mathbf{A}$ in both hybrids is identically distributed. □

Lemma D.10. For any adversary $\mathcal{A}$, $\Pr[1 \leftarrow \mathbf{Hyb}_8^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_9^{\mathcal{A}}(1^\lambda)]$.

Proof. We can sample $\mathbf{s}, \mathbf{e}, \mathbf{R}_\mathbf{A}, \mathbf{r}_\mathbf{p}$ early as well. So, this is just a semantic change. □

D.2 Proof of Theorem 5.3

We define the following hybrids.

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the selective security for fake setup game from Definition 4.1 with $b = 0$.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N]$. Perform these steps.

  1. For each $i \in S' \subseteq [N]$, sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. Set $\widetilde{\mathsf{pk}}_i := (i, \mathbf{t}_i)$ for each $i \in S'$.
  2. Sample $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$.
  3. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}, \mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively.
  4. Compute matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$. Otherwise $\mathbf{m}'_j = \mathbf{d}'_j$. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Sample $\mathbf{A}' \leftarrow \mathbb{Z}_q^{n \times m}$ and set $\mathbf{A} = \mathbf{A}' - \mathbf{C}'$.

5. Sample $\mathbf{p} \leftarrow \mathbb{Z}_q^n$. Set $\mathsf{crs} := (N, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, \mathcal{H}_1, \mathcal{H}_2)$ where $\mathcal{H}_1$ and $\mathcal{H}_2$ are handles to random oracles.
6. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R_A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top \mathbf{R_A}, c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r_p}$. Set $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$.
7. Set $\mathsf{aux} := (\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S}, \widetilde{\mathsf{ct}}, \mathbf{X}', \mathbf{D}', S')$.

Send $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ to $\mathcal{A}$.

- **Random Oracle Queries.** Any query to $\mathcal{H}_1$ with input $(\mathsf{sd}, i) \in \{0,1\} \times [N]$ is handled as follows: if H has entry $(\mathsf{sd}, i, 1)$, return $\mathsf{H}[(\mathsf{sd}, i, 1)]$. Otherwise, sample $\mathbf{x} \leftarrow \{\pm 1\}^{4m^5}$, add $\mathsf{H}[(\mathsf{sd}, i, 1)] = \mathbf{x}$ to H, and reply $\mathbf{x}$ to $\mathcal{A}$.

  Similarly, query to $\mathcal{H}_2$ with input $(\mathsf{sd}, i) \in \{0,1\} \times [N]$ is handled as follows: if H has entry $(\mathsf{sd}, i, 2)$, return $\mathsf{H}[(\mathsf{sd}, i, 2)]$. Otherwise, sample $\mathbf{d} \leftarrow \mathbb{Z}_q^n$ add $\mathsf{H}[(\mathsf{sd}, i, 2)] = \mathbf{d}$ to H, and reply $\mathbf{d}$ 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 $\mathbf{A}' = \mathbf{B}\mathbf{R_A}$ and $\mathbf{p} = \mathbf{B}\mathbf{r_p}$ by sampling $\mathbf{R_A}$ and $\mathbf{r_p}$ early. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N]$. Perform these steps.

1. For each $i \in S' \subseteq [N]$, sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. Set $\widetilde{\mathsf{pk}}_i := (i, \mathbf{t}_i)$ for each $i \in S'$. Sample $\mathbf{R_A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$.
2. Sample $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$.
3. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}, \mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively.
4. Compute matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$. Otherwise $\mathbf{m}'_j = \mathbf{d}'_j$. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Set $\mathbf{A}' = \mathbf{B}\mathbf{R_A}$ and set $\mathbf{A} = \mathbf{A}' - \mathbf{C}'$.
5. Set $\mathbf{p} = \mathbf{B}\mathbf{r_p}$. Set $\mathsf{crs} := (N, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, \mathcal{H}_1, \mathcal{H}_2)$ where $\mathcal{H}_1$ and $\mathcal{H}_2$ are handles to random oracles.
6. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}$. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top \mathbf{R_A}, c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r_p}$. Set $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$.
7. Set $\mathsf{aux} := (\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S}, \widetilde{\mathsf{ct}}, \mathbf{X}', \mathbf{D}', S')$.

Send $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ to $\mathcal{A}$.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ except we set $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1 \times 4m^5}, \mathbf{c}_2^\top := \mathbf{c}_1^\top \mathbf{R_A}, c_3 := \mathbf{c}_1^\top \mathbf{r_p}$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N]$. Perform these steps.

1. For each $i \in S' \subseteq [N]$, sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. Set $\widetilde{\mathsf{pk}}_i := (i, \mathbf{t}_i)$ for each $i \in S'$. Sample $\mathbf{R_A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$.

2. Sample $(\mathbf{W}, \mathbf{D}, \mathbf{B}) \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$.
3. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}, \mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively. Set $\mathsf{aux} := (\mathbf{X}', \mathbf{D}', S')$.
4. Compute matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$. Otherwise $\mathbf{m}'_j = \mathbf{d}'_j$. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Set $\mathbf{A}' = \mathbf{B}\mathbf{R}_{\mathbf{A}}$ and set $\mathbf{A} = \mathbf{A}' - \mathbf{C}'$.
5. Set $\mathbf{p} = \mathbf{B}\mathbf{r}_{\mathbf{p}}$. Set $\mathsf{crs} := (N, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, \mathcal{H}_1, \mathcal{H}_2)$ where $\mathcal{H}_1$ and $\mathcal{H}_2$ are handles to random oracles.
6. Compute $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1 \times 4m^5}, \mathbf{c}_2^\top := \mathbf{c}_1^\top \mathbf{R}_{\mathbf{A}}, c_3 = \mathbf{c}_1^\top \mathbf{r}_{\mathbf{p}}$. Set $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$.
7. Set $\mathsf{aux} := (\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S}, \widetilde{\mathsf{ct}}, \mathbf{X}', \mathbf{D}', S')$

Send $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ 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 $\widetilde{\mathsf{ct}} \leftarrow \mathbb{Z}_q^{1 \times (4m^5 + m + 1)}$. In this hybrid, c'_3 is a random element of $\mathbb{Z}_q$ and as such any dependence on b is eliminated.

Lemma D.11. 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 generalized leftover hash lemma (Lemma 3.2). □

Lemma D.12. Assuming the hardness of $2m^2$-decomposed LWE assumption (Assumption 3.4), for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

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. $\mathcal{B}$ interacts with $\mathcal{A}(1^\lambda)$ to get $1^N, S' \subseteq [N]$, $\mathcal{B}$ samples $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}, \mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. For each $i \in S'$, $\mathcal{B}$ sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. $\mathcal{B}$ also samples $\mathbf{R}_{\mathbf{A}} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}$. $\mathcal{B}$ computes $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ similarly to $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ and constructs $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Then, $\mathcal{B}$ sets $\mathbf{A} = \mathbf{B}\mathbf{R}_{\mathbf{A}} - \mathbf{C}'$. $\mathcal{B}$ sets $\widetilde{\mathsf{crs}} := (N, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, \mathcal{H}_1, \mathcal{H}_2), \{\widetilde{\mathsf{pk}}_i := (i, \mathbf{t}_i)\}_{i \in S'}$. Now, $\mathcal{B}$ sets $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, \mathbf{c}_1^\top \mathbf{R}_{\mathbf{A}}, \mathbf{c}_1^\top \mathbf{r}_{\mathbf{p}})$ and aux similarly to $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ and sends $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ to $\mathcal{A}$.

If Chal's coin toss $b = 0$, then $\mathcal{B}$ behaves like $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. Otherwise, $\mathcal{B}$ behaves like $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. Thus, $\mathcal{B}$ has an advantage of $\epsilon(\lambda)$. So, if $2m^2$-decomposed LWE assumption is hard, then it must that any PPT adversary's advantage in distinguishing these hybrids is negligible. □

Lemma D.13. 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. This follows from the generalized leftover hash lemma (Lemma 3.2). Specifically, the distributions

$$\left\{ \begin{bmatrix} \mathbf{B} \\ \mathbf{c}_1^\top \end{bmatrix}, \begin{bmatrix} \mathbf{B} \\ \mathbf{c}_1^\top \end{bmatrix} [\mathbf{R}_{\mathbf{A}} \mid \mathbf{r}_{\mathbf{p}}] \right\} \approx_s \left\{ \begin{bmatrix} \mathbf{B} \\ \mathbf{c}_1^\top \end{bmatrix}, \begin{bmatrix} \mathbf{A}' & \mathbf{p} \\ \mathbf{c}_2^\top & c_3 \end{bmatrix} \right\}$$

where $\mathbf{W} \leftarrow \mathbb{Z}_q^{n \times 2m^3}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m \times 2m^3}, \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + (\mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G})$, $\mathbf{R}_{\mathbf{A}} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}$, $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1 \times 4m^5}, \mathbf{c}_2^\top \leftarrow \mathbb{Z}_q^{1 \times m}, c_3 \leftarrow \mathbb{Z}_q$. □

E Proofs from Section 7

E.1 Proof of Theorem 7.2

We define the following hybrids.

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

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$.

  1. For each $i \in [N]$, sample $\mathsf{samp.ct}_i^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, i)$.
  2. For each $i \in [N]$, sample $\mathsf{samp.ct}_i^{(2)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(2)}, i)$.

  Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_i^{(1)})_{i\in[N]}, (\mathsf{samp.ct}_i^{(2)})_{i\in[N]})$.

  Compute $\mathbf{V}_N = \mathsf{MCOM.Verify}(1^N)$. Let $\mathbf{v}_i$ denote the i-th column of $\mathbf{V}_N$. For each $j \in S'$, sample $\mathbf{r}_j \leftarrow \{\pm 1\}^{4m^5}$. Compute $\mathbf{t}_j = \mathbf{B}\mathbf{r}_j + \mathbf{p} - \mathbf{A}\mathbf{v}_j$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ and $\mathsf{sk} := \mathbf{r}_j$.

  Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j\in S'}$ to $\mathcal{A}$.

- **Encryption Phase.** $\mathcal{A}$ sends $S^* \subseteq S'$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)})$, $\mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Then sample $\mathsf{samp.sk}_{S^*} \leftarrow \mathsf{PPS.KGen}(\mathsf{samp.msk}^{(1)}, S^*)$ and $\mathsf{samp.sk}_{\overline{S^*}} \leftarrow \mathsf{KGen}(\mathsf{samp.msk}^{(2)}, [N] \setminus S^*)$.

  1. For each $j \in [N]$, if $j \in S^*$, let $\mathbf{x}_j = \mathsf{PPS.Decode}(j, \mathsf{samp.ct}_j^{(1)}, S^*, \mathsf{samp.sk}_{S^*})$ and if $j \notin S^*$, $\mathbf{d}_j = \mathsf{PPS.Decode}(j, \mathsf{samp.ct}_j^{(2)}, [N] \setminus S^*, \mathsf{samp.sk}_{\overline{S^*}})$.
  2. Construct an $n \times N$ matrix $\mathbf{M}$ where the j-th column $\mathbf{m}_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}_j$ if $j \in S^*$. Otherwise, $\mathbf{m}_j = \mathbf{d}_j$.
  3. Compute $\mathbf{C} = \mathsf{MCOM.Com}(\mathbf{M})$.
  4. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R}_\mathbf{A} \leftarrow \{\pm 1\}^{4m^5\times m}, \mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$.
  5. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top\mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top(\mathbf{A} + \mathbf{C}) + \mathbf{e}^\top\mathbf{R}_\mathbf{A}, c_3 = \mathbf{s}^\top\mathbf{p} + \mathbf{e}^\top\mathbf{r}_\mathbf{p} + \lfloor q/2 \rfloor \mu$.

  Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{samp.sk}_{S^*}, \mathsf{samp.sk}_{\overline{S^*}})$ to $\mathcal{A}$.

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

$\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$, except we sample all public keys $\{\mathbf{t}_j\}$ for $j \in S'$ as uniformly random strings form $\mathbb{Z}_q^n$ early during setup phase. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$.

  1. For each $i \in [N]$, sample $\mathsf{samp.ct}_i^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, i)$.

2. For each $i \in [N]$, sample $\mathsf{samp.ct}_i^{(2)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(2)}, i)$.

Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_i^{(1)})_{i\in[N]}, (\mathsf{samp.ct}_i^{(2)})_{i\in[N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j\in S'}$ to $\mathcal{A}$.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$, except that we sample $\mathsf{samp.msk}^{(1)}, \mathsf{samp.msk}^{(2)}$ early during setup phase. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)})$, $\mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$.

  1. For each $i \in [N]$, sample $\mathsf{samp.ct}_i^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, i)$.
  2. For each $i \in [N]$, sample $\mathsf{samp.ct}_i^{(2)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(2)}, i)$.

  Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_i^{(1)})_{i\in[N]}, (\mathsf{samp.ct}_i^{(2)})_{i\in[N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j\in S'}$ to $\mathcal{A}$.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$, except that we sample $\mathsf{samp.ct}_j^{(2)}$ using $\mathsf{PPS.skEnc}$ that will decode to uniformly random $\mathbf{d}'_j$ for each $j \notin S'$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)})$, $\mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5\times N}$, $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n\times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of these matrices respectively. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$.

  1. For each $i \in [N]$, sample $\mathsf{samp.ct}_i^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, i)$.
  2. For each $i \in [N]$, if $i \in S'$ sample $\mathsf{samp.ct}_i^{(2)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(2)}, i)$. Otherwise, sample $(\mathsf{samp.ct}_j^{(2)})_{j\notin S'} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, \{j \to \mathbf{d}'_j\}_{j\notin S'})$.

  Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_i^{(1)})_{i\in[N]}, (\mathsf{samp.ct}_i^{(2)})_{i\in[N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j\in S'}$ to $\mathcal{A}$.

- **Encryption Phase.** $\mathcal{A}$ sends $S^* \subseteq S'$. Sample $\mathsf{samp.sk}_{S^*} \leftarrow \mathsf{PPS.KGen}(\mathsf{samp.msk}^{(1)}, S^*)$ and $\mathsf{samp.sk}_{\overline{S^*}} \leftarrow \mathsf{KGen}(\mathsf{samp.msk}^{(2)}, [N] \setminus S^*)$.

  1. For each $j \in [N]$, if $j \in S^*$, let $\mathbf{x}_j = \mathsf{PPS.Decode}(j, \mathsf{samp.ct}_j^{(1)}, S^*, \mathsf{samp.sk}_{S^*})$ and if $j \notin S^*$, $\mathbf{d}_j = \mathsf{PPS.Decode}(j, \mathsf{samp.ct}_j^{(2)}, [N] \setminus S^*, \mathsf{samp.sk}_{\overline{S^*}})$.
  2. Construct an $n \times N$ matrix $\mathbf{M}$ where the j-th column $\mathbf{m}_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}_j$ if $j \in S^*$. Otherwise, $\mathbf{m}_j = \mathbf{d}_j$.

3. Compute $\mathbf{C} = \mathsf{MCOM.Com}(\mathbf{M})$.
4. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R_A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$.
5. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top(\mathbf{A} + \mathbf{C}) + \mathbf{e}^\top \mathbf{R_A}, c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu$.

Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{samp.sk}_{S^*}, \mathsf{samp.sk}_{\overline{S^*}})$ to $\mathcal{A}$.

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

$\mathbf{Hyb}_{4,\iota}^{\mathcal{A}}(1^\lambda)$ for $\iota \in [N+1]$. This is same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$, except that we set $\mathsf{samp.ct}_j^{(1)}$ so that it decodes to $\mathbf{x}'_j$ and $\mathsf{samp.ct}_j^{(2)}$ so that it decodes to $\mathbf{t}'_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$ and $j < \iota$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)})$, $\mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$, $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of these matrices respectively. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$.

  1. For each $j \in [N]$, if $j \in S', j < \iota$, sample $(\mathsf{samp.ct}_j^{(1)})_{j \in S', j < \iota} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, \{j \to \mathbf{x}'_j\}_{j \in S', j < \iota})$. Otherwise, sample $\mathsf{samp.ct}_j^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, j)$.
  2. Let $T = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S', j < \iota} \cup \{j \to \mathbf{d}'_j\}_{j \notin S'}$. Sample $(\mathsf{samp.ct}_j^{(2)})_j \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T)$. For each $j \in S', j \geq \iota$, sample $\mathsf{samp.ct}_j^{(2)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(2)}, j)$.

  Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_j^{(1)})_{j \in [N]}, (\mathsf{samp.ct}_j^{(2)})_{j \in [N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j \in S'}$ to $\mathcal{A}$.

$\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_{4,N+1}^{\mathcal{A}}(1^\lambda)$. We explicitly write the full experiment for clarity.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)})$, $\mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$, $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of these matrices respectively. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$.

  1. For each $j \in [N]$, if $j \in S'$, sample $(\mathsf{samp.ct}_j^{(1)})_{j \in S'} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, \{j \to \mathbf{x}'_j\}_{j \in S'})$. Otherwise, sample $\mathsf{samp.ct}_j^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, j)$.
  2. Let $T = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S'} \cup \{j \to \mathbf{d}'_j\}_{j \notin S'}$. Sample $(\mathsf{samp.ct}_j^{(2)})_j \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T)$.

  Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_j^{(1)})_{j \in [N]}, (\mathsf{samp.ct}_j^{(2)})_{j \in [N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j \in S'}$ to $\mathcal{A}$.

- **Encryption Phase.** $\mathcal{A}$ sends $S^* \subseteq S'$. Sample $\mathsf{samp.sk}_{S^*} \leftarrow \mathsf{PPS.KGen}(\mathsf{samp.msk}^{(1)}, S^*)$ and $\mathsf{samp.sk}_{\overline{S^*}} \leftarrow \mathsf{KGen}(\mathsf{samp.msk}^{(2)}, [N] \setminus S^*)$.

  1. For each $j \in [N]$, if $j \in S^*$, let $\mathbf{x}_j = \mathsf{PPS.Decode}(j, \mathsf{samp.ct}_j^{(1)}, S^*, \mathsf{samp.sk}_{S^*})$ and if $j \notin S^*$, $\mathbf{d}_j = \mathsf{PPS.Decode}(j, \mathsf{samp.ct}_j^{(2)}, [N] \setminus S^*, \mathsf{samp.sk}_{\overline{S^*}})$.
  2. Construct an $n \times N$ matrix $\mathbf{M}$ where the j-th column $\mathbf{m}_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}_j$ if $j \in S^*$. Otherwise, $\mathbf{m}_j = \mathbf{d}_j$.
  3. Compute $\mathbf{C} = \mathsf{MCOM.Com}(\mathbf{M})$.
  4. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R_A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r_p} \leftarrow \{\pm 1\}^{4m^5}$.
  5. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top(\mathbf{A} + \mathbf{C}) + \mathbf{e}^\top \mathbf{R_A}, c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r_p} + \lfloor q/2 \rfloor \mu$.

  Send $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{samp.sk}_{S^*}, \mathsf{samp.sk}_{\overline{S^*}})$ to $\mathcal{A}$.

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

$\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$ except that we sample matrix $\mathbf{M}'$ during setup and commit it using MCOM. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)})$, $\mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$, $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of these matrices respectively. Compute matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where j-th column $\mathbf{m}'_j = \mathbf{t}'_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$ and $\mathbf{m}'_j = \mathbf{d}'_j$ otherwise. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$.

  1. For each $j \in [N]$, if $j \in S'$, sample $(\mathsf{samp.ct}_j^{(1)})_{j \in S'} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, \{j \to \mathbf{x}'_j\}_{j \in S'})$. Otherwise, sample $\mathsf{samp.ct}_j^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, j)$.
  2. Let $T = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S'} \cup \{j \to \mathbf{d}'_j\}_{j \notin S'}$. Sample $(\mathsf{samp.ct}_j^{(2)})_j \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T)$.

  Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_j^{(1)})_{j \in [N]}, (\mathsf{samp.ct}_j^{(2)})_{j \in [N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j \in S'}$ to $\mathcal{A}$.

$\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$ except that we sample matrix $\mathbf{A}' \leftarrow \mathbb{Z}_q^{n \times m}$ and set $\mathbf{A} = \mathbf{A}' - \mathbf{C}'$ during setup. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)})$, $\mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$, $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of these matrices respectively. Compute matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where j-th column $\mathbf{m}'_j = \mathbf{t}'_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$ and $\mathbf{m}'_j = \mathbf{d}'_j$ otherwise. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Sample $\mathbf{A}' \leftarrow \mathbb{Z}_q^{n \times m}$ and set $\mathbf{A} = \mathbf{A}' - \mathbf{C}'$. Sample $\mathbf{p} \leftarrow \mathbb{Z}_q^n$.

1. For each $j \in [N]$, if $j \in S'$, sample $(\mathsf{samp.ct}_j^{(1)})_{j\in S'} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, \{j \to \mathbf{x}'_j\}_{j\in S'})$. Otherwise, sample $\mathsf{samp.ct}_j^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, j)$.
2. Let $T = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j\in S'} \cup \{j \to \mathbf{d}'_j\}_{j\notin S'}$. Sample $(\mathsf{samp.ct}_j^{(2)})_j \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T)$.

Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_j^{(1)})_{j\in[N]}, (\mathsf{samp.ct}_j^{(2)})_{j\in[N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j\in S'}$ to $\mathcal{A}$.

$\mathbf{Hyb}_8^{\mathcal{A}}(1^\lambda)$. This is mode indistinguishability experiment with $b = 1$.

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

Proof. This follows by generalized leftover hash lemma (Lemma 3.2). □

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

Proof. This is just a semantic change. As $\mathsf{samp.msk}^{(1)}$ and $\mathsf{samp.msk}^{(2)}$ are not in the view of $\mathcal{A}$ anywhere, sampling them early will keep the distribution identical. □

Lemma E.3. If PPS satisfies mode indistinguishability property (Definition 6.1), then for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)]| \le \mathsf{negl}(\lambda)$.

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

$\mathcal{B}$ interacts with $\mathcal{A}$ and receives $1^N, S' \subseteq [N], \mu \in \{0,1\}$. $\mathcal{B}$ samples $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ as public keys for all $j \in S'$. $\mathcal{B}$ samples $\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}$ as described in $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. Then, $\mathcal{B}$ sends $1^N, \{\}, [N] \setminus S'$ (empty programming dictionary) to challenger Chal and receives $\mathsf{samp.pp}^{(2)}$ and $(\mathsf{samp.ct}_i^{(2)})_{i\in[N]}$. $\mathcal{B}$ additionally samples $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)})$, and samples $\mathsf{samp.ct}_i^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, i)$ for each $i \in [N]$. $\mathcal{B}$ sets $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_i^{(1)})_{i\in[N]}, (\mathsf{samp.ct}_i^{(2)})_{i\in[N]})$ and sends $\mathsf{crs}, \{(j, \mathbf{t}_j)\}_{j\in S'}$ to $\mathcal{A}$. Note that $\mathcal{B}$ implicitly sets $\mathbf{d}'_j$ for each $j' \notin S'$ using Chal. These can later be revealed once we get a secret key from Chal.

$\mathcal{A}$ then queries encryption for $S^* \subseteq S'$. $\mathcal{B}$ samples $\mathsf{samp.sk}_{S^*} \leftarrow \mathsf{PPS.KGen}(\mathsf{samp.msk}^{(1)}, S^*)$ and queries Chal for set $[N] \setminus S^*$ to receive $\mathsf{samp.sk}_{\overline{S^*}}$. Then $\mathcal{B}$ computes set $\mathbf{M}$ as described in $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. $\mathcal{B}$ then commits $\mathbf{M}$, and computes $\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3$ as described in $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. $\mathcal{B}$ sends $(\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{samp.sk}_{S^*}, \mathsf{samp.sk}_{\overline{S^*}})$ to $\mathcal{A}$.

Firstly, note that as $S^* \subseteq S'$, $([N] \setminus S') \subseteq ([N] \setminus S^*)$. Thus, $\mathcal{B}$ is an admissible adversary in the mode indistinguishability game against Chal If Chal's coin toss is $b = 0$, then Chal uses pkEnc for generating $\mathsf{samp.ct}_i^{(2)}$. Otherwise, Chal uses skEnc with $\{j \to \mathbf{d}'_j\}_{j\in[N]\setminus S'}$ to generate $(\mathsf{samp.ct}_j^{(2)})_{j\in[N]\setminus S'}$ for some uniformly random $\mathbf{d}'_j \leftarrow \mathbb{Z}_q^n$ and rest of them using pkEnc. So, if $\mathcal{A}$ can distinguish between $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ and $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ with $\epsilon(\lambda)$ advantage then $\mathcal{B}$ can break mode indistinguishability with $\epsilon(\lambda)$ advantage. □

Lemma E.4. $\Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_{4,1}^{\mathcal{A}}(1^\lambda)]$

Proof. As $j < 1$ is not true for all $j \in [N]$, these two hybrids are identically distributed. □

Lemma E.5. If PPS satisfies mode indistinguishability and payload hiding, then for any PPT adversary $\mathcal{A}$, $\forall\ \iota \in [N]$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_{4,\iota}^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_{4,\iota+1}^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. Note that these hybrids are not meaningful if $\iota \notin S'$. So, we additionally assume $\iota \in S'$. Define $\mathbf{Hyb}_{4,0}^{\mathcal{A}}(1^\lambda) \equiv \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. To prove this lemma, we define *two sets* of sub hybrids. The distinction is marked by the prefix $(\iota \in S^*)$ and $(\iota \notin S^*)$.

$(\iota \in S^*)$-SubHyb$_1^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_{4,\iota-1}^{\mathcal{A}}(1^\lambda)$ except $\mathsf{samp.ct}_\iota^{(1)}$ will be set such that it decodes to $\mathbf{x}'_\iota$ using secret key $\mathsf{samp.sk}_{S^*}$. The changes are highlighted in red.

• **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)}), \mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$, $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of these matrices respectively. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$.

1. For each $j \in [N]$, if $j \in S', j \leq \iota$, sample $(\mathsf{samp.ct}_j^{(1)})_{j \in S', j < \iota} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, \{j \to \mathbf{x}'_j\}_{j \in S', j \leq \iota})$. Otherwise, sample $\mathsf{samp.ct}_j^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, j)$.
2. Let $T = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S', j < \iota} \cup \{j \to \mathbf{d}'_j\}_{j \notin S'}$. Sample $(\mathsf{samp.ct}_j^{(2)})_j \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T)$. For each $j \in S', j \geq \iota$, sample $\mathsf{samp.ct}_j^{(2)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(2)}, j)$.

Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_j^{(1)})_{j \in [N]}, (\mathsf{samp.ct}_j^{(2)})_{j \in [N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j \in S'}$ to $\mathcal{A}$.

$(\iota \in S^*)$-SubHyb$_2^{\mathcal{A}}(1^\lambda)$. This is $\mathbf{Hyb}_{4,\iota+1}^{\mathcal{A}}(1^\lambda)$. Specifically, this is the same as $(\iota \in S^*)$-**SubHyb**$_1^{\mathcal{A}}(1^\lambda)$ except $\mathsf{samp.ct}_\iota^{(2)}$ will be set such that it decodes to $\mathbf{t}_\iota + \mathbf{B}\mathbf{x}'_\iota$ using secret key $\mathsf{samp.sk}_{\overline{S^*}}$. The changes are highlighted in red.

• **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)}), \mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$, $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of these matrices respectively. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$.

1. For each $j \in [N]$, if $j \in S', j \leq \iota$, sample $(\mathsf{samp.ct}_j^{(1)})_{j \in S', j < \iota} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, \{j \to \mathbf{x}'_j\}_{j \in S', j \leq \iota})$. Otherwise, sample $\mathsf{samp.ct}_j^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, j)$.
2. Let $T = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S', j \leq \iota} \cup \{j \to \mathbf{d}'_j\}_{j \notin S'}$. Sample $(\mathsf{samp.ct}_j^{(2)})_j \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T)$. For each $j \in S', j > \iota$, sample $\mathsf{samp.ct}_j^{(2)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(2)}, j)$.

Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_j^{(1)})_{j\in[N]}, (\mathsf{samp.ct}_j^{(2)})_{j\in[N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j\in S'}$ to $\mathcal{A}$.

Claim E.6. If PPS satisfies mode indistinguishability, then $\mathbf{Hyb}_{4,\iota}^{\mathcal{A}}(1^\lambda)$ and $(\iota \in S^*)$-$\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)$ are computationally indistinguishable.

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_{4,\iota}^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow (\iota \in S^*)\text{-}\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)]| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. We will construct an adversary $\mathcal{B}$ that can break the mode indistinguishability property of PPS with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ interacts with $\mathcal{A}$ and receives $1^N, S' \subseteq [N], \mu$. $\mathcal{B}$ sample $\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}$ similarly to $\mathbf{Hyb}_{4,\iota}^{\mathcal{A}}(1^\lambda)$. Then, $\mathcal{B}$ samples $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$ and $\mathbf{X'} \leftarrow \{\pm 1\}^{4m^5\times N}$ and $\mathbf{D'} \leftarrow \mathbb{Z}_q^{n\times N}$. $\mathcal{B}$ sends $(1^N, \{j \to \mathbf{x}'_j\}_{j\in S', j<\iota}, \iota)$ to challenger Chal. $\mathcal{B}$ receives $\mathsf{samp.pp}^{(1)}, (\mathsf{samp.ct}_i^{(1)})_{i\in[N]}$ from Chal. $\mathcal{B}$ also samples $\mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$ and $\mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. $\mathcal{B}$ samples $\mathsf{samp.ct}_i^{(2)}$ for each $i \in [N]$ similarly to $(\iota \in S^*)$-$\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)$. $\mathcal{B}$ sets $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_i^{(1)})_{i\in[N]}, (\mathsf{samp.ct}^{(2)})_{i\in[N]})$ and sends $\mathsf{crs}, \{(j, \mathbf{t}_j)\}_{j\in S'}$ to $\mathcal{A}$. Note that $\mathcal{B}$ implicitly sets $\mathbf{x}'_\iota$ as the one sampled by Chal and *not* from $\mathbf{X'}$. It can later be revealed using $\mathsf{samp.sk}_{S^*}$.

$\mathcal{B}$ then receives $S^* \subseteq S'$ from $\mathcal{A}$. $\mathcal{B}$ sends S^* to Chal and receives $\mathsf{samp.sk}_{S^*}$. $\mathcal{B}$ samples $\mathsf{samp.sk}_{\overline{S^*}} \leftarrow \mathsf{PPS.KGen}(\mathsf{samp.msk}^{(2)}, [N] \setminus S^*)$. Then, $\mathcal{B}$ samples matrix $\mathbf{M}$, commits it as $\mathbf{C'}$, samples $\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3$ like in $\mathbf{Hyb}_{4,\iota}^{\mathcal{A}}(1^\lambda)$. It sends $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{samp.sk}_{S^*}, \mathsf{samp.sk}_{\overline{S^*}})$ to $\mathcal{A}$.

Note that as $\iota \in S^*$ and $\iota \notin$ programming dictionary, $\mathcal{B}$ is an admissible adversary against the mode indistinguishability game against PPS challenger. If Chal's bit $b = 0$, then $\mathsf{samp.ct}_\iota^{(1)}$ is sampled using PPS.pkEnc and if $b = 1$, then it is programmed to some $\mathbf{x}'_\iota \leftarrow \{\pm 1\}^{4m^5}$ which can later be revealed. Thus, if $\mathcal{A}$ has $\epsilon(\lambda)$ advantage in distinguishing then so does $\mathcal{B}$ in breaking mode indistinguishability property. □

Claim E.7. If PPS satisfies payload hiding, then $(\iota \in S^*)$-$\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)$ and $(\iota \in S^*)$-$\mathbf{SubHyb}_2^{\mathcal{A}}(1^\lambda)$ are computationally indistinguishable.

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $|\Pr[1 \leftarrow (\iota \in S^*)\text{-}\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow (\iota \in S^*)\text{-}\mathbf{SubHyb}_2^{\mathcal{A}}(1^\lambda)]| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. We will construct an adversary $\mathcal{B}$ that can break the payload hiding property of PPS with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ interacts with $\mathcal{A}$ and receives $1^N, S' \subseteq [N], \mu$. $\mathcal{B}$ sample $\mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}$ similarly to $(\iota \in S^*)$-$\mathbf{SubHyb}_2^{\mathcal{A}}(1^\lambda)$. Then, $\mathcal{B}$ samples $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$ and $\mathbf{X'} \leftarrow \{\pm 1\}^{4m^5\times N}$ and $\mathbf{D'} \leftarrow \mathbb{Z}_q^{n\times N}$. $\mathcal{B}$ sends $(1^N, \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j\in S', j<\iota} \cup \{j \to \mathbf{d}'_j\}_{j\notin S'}, \iota, \mathbf{t}_\iota + \mathbf{B}\mathbf{x}'_\iota)$ to challenger Chal. $\mathcal{B}$ receives $\mathsf{samp.pp}^{(2)}, (\mathsf{samp.ct}_i^{(2)})_{i\in[N]}$ from Chal. $\mathcal{B}$ also samples $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$ and $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)})$. $\mathcal{B}$ samples $\mathsf{samp.ct}_i^{(1)}$ for each $i \in [N]$ similarly to $(\iota \in S^*)$-$\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)$. $\mathcal{B}$ sets $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_i^{(1)})_{i\in[N]}, (\mathsf{samp.ct}^{(2)})_{i\in[N]})$ and sends $\mathsf{crs}, \{(j, \mathbf{t}_j)\}_{j\in S'}$ to $\mathcal{A}$.

$\mathcal{B}$ then receives $S^* \subseteq S'$ from $\mathcal{A}$. $\mathcal{B}$ sends $[N] \setminus S^*$ to Chal and receives $\mathsf{samp.sk}_{\overline{S^*}}$. $\mathcal{B}$ samples $\mathsf{samp.sk}_{S^*} \leftarrow \mathsf{PPS.KGen}(\mathsf{samp.msk}^{(1)}, S^*)$. Then, $\mathcal{B}$ samples matrix $\mathbf{M}$, commits it as $\mathbf{C'}$, samples $\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3$ like in $\mathbf{Hyb}_{4,\iota}^{\mathcal{A}}(1^\lambda)$. It sends $\mathsf{ct} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3, \mathsf{samp.sk}_{S^*}, \mathsf{samp.sk}_{\overline{S^*}})$ to $\mathcal{A}$.

Note that as $\iota \in S^* \implies \iota \notin [N] \setminus S^*$ and $\iota \notin$ programming dictionary, $\mathcal{B}$ is an admissible adversary against the payload hiding game against PPS challenger. If Chal's bit $b = 0$, then $\mathsf{samp.ct}_\iota^{(2)}$ is sampled using PPS.pkEnc and if $b = 1$, then it is programmed to $\mathbf{t}_\iota + \mathbf{B}\mathbf{x}'_\iota$ given by

$\mathcal{B}$. Thus, if $\mathcal{A}$ has $\epsilon(\lambda)$ advantage in distinguishing then so does $\mathcal{B}$ in breaking payload hiding property. $\square$

Above claims prove on path where $\iota \in S^*$, we can safely move from $\mathbf{Hyb}^{\mathcal{A}}_{4,\iota}(1^\lambda)$ to $\mathbf{Hyb}^{\mathcal{A}}_{4,\iota+1}(1^\lambda)$. Now, we will prove that even if $\iota \notin S^*$, we can do the same using the following set of sub-hybrids and claims.

($\iota \notin S^*$)-SubHyb$^{\mathcal{A}}_1(1^\lambda)$. This is the same as $\mathbf{Hyb}^{\mathcal{A}}_{4,\iota}(1^\lambda)$ except we set $\mathsf{samp.ct}^{(2)}_\iota$ such that it can be decoded to $\mathbf{d}_\jmath \iota'$ using $\mathsf{samp.sk}_{\overline{S^*}}$. The changes are highlighted in red.
• **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)}), \mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$, $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of these matrices respectively. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$.

1. For each $j \in [N]$, if $j \in S', j < \iota$, sample $(\mathsf{samp.ct}^{(1)}_j)_{j \in S', j < \iota} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, \{j \to \mathbf{x}'_j\}_{j \in S', j < \iota})$. Otherwise, sample $\mathsf{samp.ct}^{(1)}_j \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, j)$.
2. Let $T = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S', j < \iota} \cup \{j \to \mathbf{d}'_j\}_{j \in ([N] \setminus S') \cup \{\iota\}}$. Sample $(\mathsf{samp.ct}^{(2)}_j)_j \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T)$. For each $j \in S', j > \iota$, sample $\mathsf{samp.ct}^{(2)}_j \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(2)}, j)$.

Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}^{(1)}_j)_{j \in [N]}, (\mathsf{samp.ct}^{(2)}_j)_{j \in [N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j \in S'}$ to $\mathcal{A}$.

($\iota \notin S^*$)-SubHyb$^{\mathcal{A}}_2(1^\lambda)$. This is the same as ($\iota \notin S^*$)-**SubHyb**$^{\mathcal{A}}_1(1^\lambda)$ except we set $\mathsf{samp.ct}^{(2)}_\iota$ such that it can be decoded to $\mathbf{t}_\iota + \mathbf{B}\mathbf{x}'_\iota$ using $\mathsf{samp.sk}_{\overline{S^*}}$. The changes are highlighted in red.
• **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)}), \mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$, $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of these matrices respectively. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$.

1. For each $j \in [N]$, if $j \in S', j < \iota$, sample $(\mathsf{samp.ct}^{(1)}_j)_{j \in S', j < \iota} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, \{j \to \mathbf{x}'_j\}_{j \in S', j < \iota})$. Otherwise, sample $\mathsf{samp.ct}^{(1)}_j \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, j)$.
2. Let $T = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S', j \le \iota} \cup \{j \to \mathbf{d}'_j\}_{j \notin S'}$. Sample $(\mathsf{samp.ct}^{(2)}_j)_j \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T)$. For each $j \in S', j > \iota$, sample $\mathsf{samp.ct}^{(2)}_j \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(2)}, j)$.

Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}^{(1)}_j)_{j \in [N]}, (\mathsf{samp.ct}^{(2)}_j)_{j \in [N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j \in S'}$ to $\mathcal{A}$.

$(\iota \notin S^*)$-$\mathbf{SubHyb}_3^{\mathcal{A}}(1^\lambda)$. This is $\mathbf{Hyb}_{4,\iota+1}^{\mathcal{A}}(1^\lambda)$. Specifically, this is the same as $(\iota \notin S^*)$-$\mathbf{SubHyb}_2^{\mathcal{A}}(1^\lambda)$ except we set $\mathsf{samp.ct}_\iota^{(1)}$ such that it can be decoded to $\mathbf{x}'_\iota$ using $\mathsf{samp.sk}_{S^*}$. The changes are highlighted in red.

• **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$. Sample $\mathbf{t}_j \leftarrow \mathbb{Z}_q^n$ for each $j \in S'$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N), \mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)}), \mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}$, $\mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of these matrices respectively. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{p} \leftarrow \mathbb{Z}_q^n$.

1. For each $j \in [N]$, if $j \in S', j \leq \iota$, sample $(\mathsf{samp.ct}_j^{(1)})_{j \in S', j \leq \iota} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, \{j \to \mathbf{x}'_j\}_{j \in S', j \leq \iota})$. Otherwise, sample $\mathsf{samp.ct}_j^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, j)$.
2. Let $T = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S', j \leq \iota} \cup \{j \to \mathbf{d}'_j\}_{j \notin S'}$. Sample $(\mathsf{samp.ct}_j^{(2)})_j \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T)$. For each $j \in S', j > \iota$, sample $\mathsf{samp.ct}_j^{(2)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(2)}, j)$.

Set $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.ct}_j^{(1)})_{j \in [N]}, (\mathsf{samp.ct}_j^{(2)})_{j \in [N]})$. Set $\mathsf{pk}_j := (j, \mathbf{t}_j)$ for $j \in S'$. Send $\mathsf{crs}, \{\mathsf{pk}_j\}_{j \in S'}$ to $\mathcal{A}$.

Claim E.8. If PPS satisfies mode indistinguishability property, then $\mathbf{Hyb}_{4,\iota}^{\mathcal{A}}(1^\lambda)$ and $(\iota \notin S^*)$-$\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)$ are computationally indistinguishable.

Proof. This is similar to proof of Claim E.6 □

Claim E.9. $(\iota \notin S^*)$-$\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)$ and $(\iota \notin S^*)$-$\mathbf{SubHyb}_2^{\mathcal{A}}(1^\lambda)$ are statistically indistinguishable.

Proof. This follows by leftover hash lemma. Note that $\mathbf{x}'_\iota$ is not present in the view of $\mathcal{A}$ at all. So, this claim holds. □

Claim E.10. If PPS satisfies payload hiding, then $(\iota \notin S^*)$-$\mathbf{SubHyb}_2^{\mathcal{A}}(1^\lambda)$ and $(\iota \notin S^*)$-$\mathbf{SubHyb}_3^{\mathcal{A}}(1^\lambda)$ are computationally indistinguishable.

Proof. This is similar to proof of Claim E.7. □

Using these claims and the piecewise guessing framework of [JKK+17, KW20], Lemma E.5 holds. □

Lemma E.11. $\Pr[1 \leftarrow \mathbf{Hyb}_{4,N+1}^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)]$

Proof. This is self-evident as we wrote the hybrid experiment for clarity. □

Lemma E.12. $\Pr[1 \leftarrow \mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)]$

Proof. Note that we are not using $\mathbf{M}'$ and $\mathbf{C}'$ anywhere in the view of $\mathcal{A}$. So, the lemma holds. □

Lemma E.13. $\Pr[1 \leftarrow \mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)]$

Proof. As $\mathbf{A}'$ is sampled uniformly randomly in $\mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)$, in both hybrids, distribution of $\mathbf{A}$ is identical. Thus, the lemma holds. □

Lemma E.14. $\Pr[1 \leftarrow \mathbf{Hyb}_7^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_8^{\mathcal{A}}(1^\lambda)]$

Proof. This is just a semantic change as we can sample $\mathbf{s}, \mathbf{e}, \mathbf{R}_\mathbf{A}, \mathbf{r}_\mathbf{p}$ early as well. So, the lemma holds. □

E.2 Proof of Theorem 7.3

We define the following hybrids.

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the selective security with fake setup game with $b = 0$.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N]$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$, $\mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$, and perform the following steps.

  1. For each $i \in S' \subseteq [N]$, sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. Set $\widetilde{\mathsf{pk}}_i := (i, \mathbf{t}_i)$ for each $i \in S'$.
  2. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$.
  3. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}, \mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively.
  4. Compute matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$. Otherwise $\mathbf{m}'_j = \mathbf{d}'_j$. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Sample $\mathbf{A}' \leftarrow \mathbb{Z}_q^{n \times m}$ and set $\mathbf{A} = \mathbf{A}' - \mathbf{C}'$.
  5. Sample $\mathbf{p} \leftarrow \mathbb{Z}_q^n$.
  6. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)}), \mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$.
  7. Set $T^{(1)} = \{j \to \mathbf{x}'_j\}_{j \in S'}$. Also, set $T^{(2)} = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S'} \cup \{j \to \mathbf{d}'_j\}_{j \notin S'}$. For each $j \notin S'$, sample $\widetilde{\mathsf{samp.ct}}_j^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, i)$.
  8. Sample $(\widetilde{\mathsf{samp.ct}}^{(1)})_{j \in S'} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, T^{(1)})$ and $(\widetilde{\mathsf{samp.ct}}^{(2)})_{j \in [N]} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T^{(2)})$.
  9. Set $\widetilde{\mathsf{crs}} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\widetilde{\mathsf{samp.ct}}_i^{(1)})_{i \in [N]}, (\widetilde{\mathsf{samp.ct}}_i^{(2)})_{i \in [N]})$.
  10. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R}_\mathbf{A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top \mathbf{R}_\mathbf{A}, c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r}_\mathbf{p}$. Set $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$. Set $\mathsf{aux} := (S', \mathsf{samp.msk}^{(1)}, \mathsf{samp.msk}^{(2)}, \widetilde{\mathsf{ct}})$.

  Send $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ 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 $\mathbf{A}' = \mathbf{B}\mathbf{R}_\mathbf{A}$ and $\mathbf{p} = \mathbf{B}\mathbf{r}_\mathbf{p}$ by sampling $\mathbf{R}_\mathbf{A}$ and $\mathbf{r}_\mathbf{p}$ early. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$, $\mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$, and perform the following steps.

  1. For each $i \in S' \subseteq [N]$, sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. Set $\widetilde{\mathsf{pk}}_i := (i, \mathbf{t}_i)$ for each $i \in S'$. Sample $\mathbf{R}_\mathbf{A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$.

2. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$.
3. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}, \mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively.
4. Compute matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$. Otherwise $\mathbf{m}'_j = \mathbf{d}'_j$. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$.Set $\mathbf{A} = \mathbf{B}\mathbf{R}_\mathbf{A} - \mathbf{C}'$.
5. Set $\mathbf{p} = \mathbf{B}\mathbf{r}_\mathbf{p}$.
6. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)}), \mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$.
7. Set $T^{(1)} = \{j \to \mathbf{x}'_j\}_{j \in S'}$. Also, set $T^{(2)} = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S'} \cup \{j \to \mathbf{d}'_j\}_{j \notin S'}$. For each $j \notin S'$, sample $\mathsf{samp.}\widetilde{\mathsf{ct}}_j^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, i)$.
8. Sample $(\mathsf{samp.}\widetilde{\mathsf{ct}}^{(1)})_{j \in S'} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, T^{(1)})$ and $(\mathsf{samp.}\widetilde{\mathsf{ct}}^{(2)})_{j \in [N]} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T^{(2)})$.
9. Set $\widetilde{\mathsf{crs}} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.}\widetilde{\mathsf{ct}}_i^{(1)})_{i \in [N]}, (\mathsf{samp.}\widetilde{\mathsf{ct}}_i^{(2)})_{i \in [N]})$.
10. Sample $\mathbf{s} \leftarrow \mathbb{Z}_q^n, \mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{4m^5}, \mathbf{R}_\mathbf{A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$. Compute $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{B} + \mathbf{e}^\top, \mathbf{c}_2^\top = \mathbf{s}^\top \mathbf{A}' + \mathbf{e}^\top \mathbf{R}_\mathbf{A}, c_3 = \mathbf{s}^\top \mathbf{p} + \mathbf{e}^\top \mathbf{r}_\mathbf{p}$. Set $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$. Set $\mathsf{aux} := (S', \mathsf{samp.msk}^{(1)}, \mathsf{samp.msk}^{(2)}, \widetilde{\mathsf{ct}})$.

Send $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i \in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ 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 we set $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1 \times 4m^5}, \mathbf{c}_2^\top := \mathbf{c}_1^\top \mathbf{R}_\mathbf{A}, c_3 := \mathbf{c}_1^\top \mathbf{r}_\mathbf{p}$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, S' \subseteq [N], \mu \in \{0,1\}$. Sample $\mathsf{samp.pp}^{(1)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$, $\mathsf{samp.pp}^{(2)} \leftarrow \mathsf{PPS.PPGen}(1^\lambda, 1^N)$, and perform the following steps.

1. For each $i \in S' \subseteq [N]$, sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. Set $\widetilde{\mathsf{pk}}_i := (i, \mathbf{t}_i)$ for each $i \in S'$. Sample $\mathbf{R}_\mathbf{A} \leftarrow \{\pm 1\}^{4m^5 \times m}, \mathbf{r}_\mathbf{p} \leftarrow \{\pm 1\}^{4m^5}$.
2. Sample $\mathbf{W}, \mathbf{D}, \mathbf{B} \leftarrow \mathsf{MCOM.Setup}(1^n, 1^m, q, \sigma)$.
3. Sample $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5 \times N}, \mathbf{D}' \leftarrow \mathbb{Z}_q^{n \times N}$. Let $\mathbf{x}'_j$ and $\mathbf{d}'_j$ denote the j-th column of $\mathbf{X}'$ and $\mathbf{D}'$ respectively.
4. Compute matrix $\mathbf{M}' \in \mathbb{Z}_q^{n \times N}$ where the j-th column $\mathbf{m}'_j = \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j$ if $j \in S'$. Otherwise $\mathbf{m}'_j = \mathbf{d}'_j$. Compute $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Set $\mathbf{A} = \mathbf{B}\mathbf{R}_\mathbf{A} - \mathbf{C}'$.
5. Set $\mathbf{p} = \mathbf{B}\mathbf{r}_\mathbf{p}$.
6. Sample $\mathsf{samp.msk}^{(1)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(1)}), \mathsf{samp.msk}^{(2)} \leftarrow \mathsf{PPS.MSKGen}(\mathsf{samp.pp}^{(2)})$.
7. Set $T^{(1)} = \{j \to \mathbf{x}'_j\}_{j \in S'}$. Also, set $T^{(2)} = \{j \to \mathbf{t}_j + \mathbf{B}\mathbf{x}'_j\}_{j \in S'} \cup \{j \to \mathbf{d}'_j\}_{j \notin S'}$. For each $j \notin S'$, sample $\mathsf{samp.}\widetilde{\mathsf{ct}}_j^{(1)} \leftarrow \mathsf{PPS.pkEnc}(\mathsf{samp.pp}^{(1)}, i)$.
8. Sample $(\mathsf{samp.}\widetilde{\mathsf{ct}}^{(1)})_{j \in S'} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(1)}, T^{(1)})$ and $(\mathsf{samp.}\widetilde{\mathsf{ct}}^{(2)})_{j \in [N]} \leftarrow \mathsf{PPS.skEnc}(\mathsf{samp.msk}^{(2)}, T^{(2)})$.

9. Set $\widetilde{\mathsf{crs}} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.}\widetilde{\mathsf{ct}}_i^{(1)})_{i\in[N]}, (\mathsf{samp.}\widetilde{\mathsf{ct}}_i^{(2)})_{i\in[N]})$.
10. Compute $\mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1\times 4m^5}, \mathbf{c}_2^\top := \mathbf{c}_1^\top \mathbf{R}_{\mathbf{A}}, c_3 = \mathbf{c}_1^\top \mathbf{r}_{\mathbf{p}}$. Set $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$. Set $\mathsf{aux} := (S', \mathsf{samp.msk}^{(1)}, \mathsf{samp.msk}^{(2)}, \widetilde{\mathsf{ct}})$.

Send $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ to $\mathcal{A}$.

- **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 set $\widetilde{\mathsf{ct}} \leftarrow \mathbb{Z}_q^{1\times(4m^5+m+1)}$. In this hybrid, c_3 is a random element of $\mathbb{Z}_q$ and as such any dependence on b is eliminated.

Lemma E.15. 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 generalized leftover hash lemma (Lemma 3.2). □

Lemma E.16. Assuming the hardness of $2m^2$-decomposed LWE assumption (Assumption 3.4), 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. $\mathcal{B}$ interacts with $\mathcal{A}(1^\lambda)$ to get $1^N, S' \subseteq [N]$, $\mathcal{B}$ samples $\mathbf{X}' \leftarrow \{\pm 1\}^{4m^5\times N}, \mathbf{D}' \leftarrow \mathbb{Z}_q^{n\times N}$. For each $i \in S'$, $\mathcal{B}$ sample $\mathbf{t}_i \leftarrow \mathbb{Z}_q^n$. $\mathcal{B}$ also samples $\mathbf{R}_{\mathbf{A}} \leftarrow \{\pm 1\}^{4m^5\times m}, \mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}$. $\mathcal{B}$ computes $\mathbf{M}' \in \mathbb{Z}_q^{n\times N}$ similarly to $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ and constructs $\mathbf{C}' = \mathsf{MCOM.Com}(\mathbf{M}')$. Then, $\mathcal{B}$ sets $\mathbf{A} = \mathbf{B}\mathbf{R}_{\mathbf{A}} - \mathbf{C}'$. $\mathcal{B}$ sends $\mathsf{crs} := (N, \mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathbf{W}, \mathbf{D}, \mathbf{B}, \mathbf{A}, \mathbf{p}, (\mathsf{samp.}\widetilde{\mathsf{ct}}_i^{(1)})_{i\in[N]}, (\mathsf{samp.}\widetilde{\mathsf{ct}}_i^{(1)})_{i\in[N]}), \{\mathsf{pk}_i := \mathbf{t}_i\}_{i\in S'}$ to $\mathcal{A}$. Here, $\mathsf{samp.pp}^{(1)}, \mathsf{samp.pp}^{(2)}, \mathsf{samp.}\widetilde{\mathsf{ct}}_i^{(1)}$, and $\mathsf{samp.}\widetilde{\mathsf{ct}}_j^{(2)}$ are constructed similarly to $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. Now, $\mathcal{B}$ sets $\widetilde{\mathsf{ct}} := (\mathbf{c}_1^\top, \mathbf{c}_1^\top \mathbf{R}_{\mathbf{A}}, \mathbf{c}_1^\top \mathbf{r}_{\mathbf{p}})$. $\mathcal{B}$ sets $\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \mathsf{aux}$ similarly to $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. It sends $(\widetilde{\mathsf{crs}}, \{\widetilde{\mathsf{pk}}_i\}_{i\in S'}, \widetilde{\mathsf{ct}}, \mathsf{aux})$ to $\mathcal{A}$.

If Chal's coin toss $b = 0$, then $\mathcal{B}$ behaves like $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. Otherwise, $\mathcal{B}$ behaves like $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. Thus, $\mathcal{B}$ has an advantage of $\epsilon(\lambda)$. So, if $2m^2$-decomposed LWE assumption is hard, then it must that any PPT adversary's advantage in distinguishing these hybrids is negligible. □

Lemma E.17. 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 generalized leftover hash lemma (Lemma 3.2). Specifically, the distributions

$$\left\{ \begin{bmatrix} \mathbf{B} \\ \mathbf{c}_1^\top \end{bmatrix}, \begin{bmatrix} \mathbf{B} \\ \mathbf{c}_1^\top \end{bmatrix} [\mathbf{R}_{\mathbf{A}} \mid \mathbf{r}_{\mathbf{p}}] \right\} \approx_s \left\{ \begin{bmatrix} \mathbf{B} \\ \mathbf{c}_1^\top \end{bmatrix}, \begin{bmatrix} \mathbf{A}' & \mathbf{p} \\ \mathbf{c}_2^\top & c_3 \end{bmatrix} \right\}$$

where $\mathbf{W} \leftarrow \mathbb{Z}_q^{n\times 2m^3}, \mathbf{D} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{m\times 2m^3}, \mathbf{B} = \mathbf{W}(\mathbf{I}_{2m^2} \otimes \mathbf{D}) + (\mathsf{vec}(\mathbf{I}_{2m^2})^\top \otimes \mathbf{G}), \mathbf{R}_{\mathbf{A}} \leftarrow \{\pm 1\}^{4m^5\times m}, \mathbf{r}_{\mathbf{p}} \leftarrow \{\pm 1\}^{4m^5}, \mathbf{c}_1^\top \leftarrow \mathbb{Z}_q^{1\times 4m^5}, \mathbf{c}_2^\top \leftarrow \mathbb{Z}_q^{1\times m}, c_3 \leftarrow \mathbb{Z}_q$. □

F Proofs from Section 8

F.1 Proof of Theorem 8.3

We define the following hybrids.

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the mode indistinguishability experiment from Definition 6.1 with $b = 1$.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, \{j \to \mu_j\}_{j\in P}, i^*$ such that $P \subseteq [N], i^* \notin P, \mu_j \in \mathbb{Z}_2 \forall\ j \in P$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, set $\mathsf{pp} := (n, m, q, \chi, B, \mathbf{A}, N)$. Also sample $\mathbf{s}_{k,b} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{k,b} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m$ for each $k \in [N], b \in \{0,1\}$, set $\mathbf{b}_{k,b}^\top = \mathbf{s}_{k,b}^\top \mathbf{A} + \mathbf{e}_{k,b}^\top$. Set $\mathsf{msk} := (\mathbf{s}_{k,b}, \mathbf{b}_{k,b})_{k,b}$.

  For each $j \in P$,

  1. Sample $\mathbf{x}_j \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{j,0} = \mathbf{A}\mathbf{x}_j$.
  2. Sample for each $k \in [N]$, $z_{j,k} \leftarrow \mathbb{Z}_q$ such that $\sum_{k\in[N]} z_{j,k} = 0$. Sample $t_j \leftarrow \mathbb{Z}_q$ subject to the constraint that $t_j \notin [-\lfloor q/4\rfloor - B, -\lfloor q/4\rfloor + B] \cup [\lfloor q/4\rfloor - B, \lfloor q/4\rfloor + B]$.
  3. Set $c_{j,j,1} = \mathbf{b}_{j,1}^\top \mathbf{x}_j + z_{j,j} + t_j$ and sample $c_{j,j,0} \leftarrow \mathbb{Z}_q$.
  4. For $k \in [N], k \neq j$ for each $b \in \{0,1\}$, set $c_{j,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_j + z_{j,k}$.
  5. Set $c_{j,T} = \lceil t_j \rfloor_2 + \mu_j$
  6. Set $\mathsf{ct}_j = (\mathbf{c}_{j,0}, (c_{j,k,b})_{k,b}, c_{j,T})$.

  For each $j = i^*$, sample $\mu_{i^*} \leftarrow \mathbb{Z}_2$. Then,

  1. Sample $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{i^*,0} = \mathbf{A}\mathbf{x}_{i^*}$.
  2. Sample for each $k \in [N]$, $z_{i^*,k} \leftarrow \mathbb{Z}_q$ such that $\sum_{k\in[N]} z_{i^*,k} = 0$. Sample $t_{i^*} \leftarrow \mathbb{Z}_q$ subject to the constraint that $t_{i^*} \notin [-\lfloor q/4\rfloor - B, -\lfloor q/4\rfloor + B] \cup [\lfloor q/4\rfloor - B, \lfloor q/4\rfloor + B]$.
  3. Set $c_{i^*,i^*,1} = \mathbf{b}_{i^*,1}^\top \mathbf{x}_{i^*} + z_{i^*,i^*} + t_{i^*}$ and sample $c_{i^*,i^*,0} \leftarrow \mathbb{Z}_q$.
  4. For $k \in [N], k \neq i^*$ for each $b \in \{0,1\}$, set $c_{i^*,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_{i^*} + z_{i^*,k}$.
  5. Set $c_{i^*,T} = \lceil t_{i^*} \rfloor_2 + \mu_{i^*}$
  6. Set $\mathsf{ct}_{i^*} = (\mathbf{c}_{i^*,0}, (c_{i^*,k,b})_{k,b}, c_{i^*,T})$.

  For $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$.

  Send $(\mathsf{pp}, (\mathsf{ct}_k)_{k\in[N]})$ to $\mathcal{A}$.

- **Key Generation Query.** $\mathcal{A}$ sends a set S^* such that $i^* \in S^* \subseteq [N]$. Set $\mathsf{sk}_{S^*} = \sum_{j\in[N]} \mathbf{s}_{j,r_j}$ where $\mathbf{r} = (r_1, \ldots, r_N)$ is the characteristic vector of S^*. Send sk_{S^*} 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 that we set $c_{i^*,T} \leftarrow \mathbb{Z}_2$ and stop sampling $\mu_{i^*} \leftarrow \mathbb{Z}_2$.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ except that we choose $t_{i^*} \leftarrow \mathbb{Z}_q$ without the additional constraint.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ except that we sample $c_{i^*,i^*,1} \leftarrow \mathbb{Z}_q$.

$\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ except that we sample $(z_{i^*,k})_{k\in[N]\setminus\{i^*\}}$ as random elements from $\mathbb{Z}_q$.

$\mathbf{Hyb}_{5,\kappa}^{\mathcal{A}}(1^\lambda)$ for $\kappa \in [N+1]$. In this hybrid, we will set $c_{i^*,k,0}, c_{i^*,k,1}$ as random elements from $\mathbb{Z}_q$ for each $1 \le k < \kappa$. We write the full hybrid for clarity.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, \{j \to \mu_j\}_{j\in P}, i^*$ such that $P \subseteq [N], i^* \notin P, \mu_j \in \mathbb{Z}_2 \forall\ j \in P$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, set $\mathsf{pp} := (n, m, q, \chi, B, \mathbf{A}, N)$. Also sample $\mathbf{s}_{k,b} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{k,b} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m$ for each $k \in [N], b \in \{0,1\}$, set $\mathbf{b}_{k,b}^\top = \mathbf{s}_{k,b}^\top \mathbf{A} + \mathbf{e}_{k,b}^\top$.

  For each $j \in P$,

  1. Sample $\mathbf{x}_j \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{j,0} = \mathbf{A}\mathbf{x}_j$.
  2. Sample for each $k \in [N]$, $z_{j,k} \leftarrow \mathbb{Z}_q$ such that $\sum_{k\in[N]} z_{j,k} = 0$. Sample $t_j \leftarrow \mathbb{Z}_q$ subject to the constraint that $t_j \notin [-\lfloor q/4\rfloor - B, -\lfloor q/4\rfloor + B] \cup [\lfloor q/4\rfloor - B, \lfloor q/4\rfloor + B]$.
  3. Set $c_{j,j,1} = \mathbf{b}_{j,1}^\top \mathbf{x}_j + z_{j,j} + t_j$ and sample $c_{j,j,0} \leftarrow \mathbb{Z}_q$.
  4. For $k \in [N], k \ne j$ for each $b \in \{0,1\}$, set $c_{j,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_j + z_{j,k}$.
  5. Set $c_{j,T} = \lceil t_j \rfloor_2 + \mu_j$
  6. Set $\mathsf{ct}_j = (\mathbf{c}_{j,0}, (c_{j,k,b})_{k,b}, c_{j,T})$.

  For each $j = i^*$,

  1. Sample $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{i^*,0} = \mathbf{A}\mathbf{x}_{i^*}$.
  2. Sample for each $k \in [N] \setminus \{i^*\}$, $z_{i^*,k} \leftarrow \mathbb{Z}_q$.
  3. Sample $c_{i^*,i^*,1}, c_{i^*,i^*,0} \leftarrow \mathbb{Z}_q$.
  4. For $k \in [N], k \ne i^*$, for each $b \in \{0,1\}$, if $k < \kappa$, sample $c_{i^*,k,b} \leftarrow \mathbb{Z}_q$. Otherwise, if $k \ge \kappa$, set $c_{i^*,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_{i^*} + z_{i^*,k}$.
  5. Sample $c_{i^*,T} \leftarrow \mathbb{Z}_2$
  6. Set $\mathsf{ct}_{i^*} = (\mathbf{c}_{i^*,0}, (c_{i^*,k,b})_{k,b}, c_{i^*,T})$.

  For $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$.

  Send $(\mathsf{pp}, (\mathsf{ct}_k)_{k\in[N]})$ to $\mathcal{A}$.

- **Key Generation Query.** $\mathcal{A}$ sends a set S^* such that $i^* \in S^* \subseteq [N]$. Set $\mathsf{sk}_{S^*} = \sum_{j\in[N]} \mathbf{s}_{j,r_j}$ where $\mathbf{r} = (r_1, \ldots, r_N)$ is the characteristic vector of S^*. Send sk_{S^*} to $\mathcal{A}$.

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

$\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$. This is mode indistinguishability experiment from Definition 6.1 with $b = 0$.

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

Proof. As μ_{i^*} is sampled independently from all other components, $c_{i^*,T}$ in both hybrids are identically distributed. Thus, the lemma holds. □

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 only difference between these hybrids is the sampling constraint on t. The statistical distance between these two distributions is $(4B + 2)/q \leq \mathsf{negl}(\lambda)$ by our parameters. So, these hybrids are statistically close. □

Lemma F.3. $\Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)]$

Proof. As t_{i^*} is sampled independently from all other components, $c_{i^*,i^*,1}$ in both hybrids are identically distributed. Thus, the lemma holds. □

Lemma F.4. $\Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)]$

Proof. As z_{i^*,i^*} is nowhere in the view of the adversary, the rest of the shares of $0 \in \mathbb{Z}_q$, $z_{i^*,k} \leftarrow \mathbb{Z}_q$ for $k \neq i^*$, are distributed uniformly randomly. Thus, in both hybrids, $(z_{i^*,k})_{k \neq i^*}$ are uniformly random and hence, the lemma holds. □

Lemma F.5. $\Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_{5,0}^{\mathcal{A}}(1^\lambda)]$

Proof. In $\mathbf{Hyb}_{5,0}^{\mathcal{A}}(1^\lambda)$, we are setting any $c_{i^*,k,b}$ for $k \in [N] \setminus \{i^*\}, b \in \{0,1\}$ to random. This is the same distribution as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$. □

Lemma F.6. Assuming the hardness of LWE (Assumption 3.3), for any PPT adversary $\mathcal{A}$, $\forall\ \kappa \in [N]$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_{5,\kappa}^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_{5,\kappa+1}^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. Note that these hybrids are not useful when $\kappa = i^*$ that $\mathcal{A}$ provides. So, we will assume that $\kappa \neq i^*$ in these hybrids. To prove this lemma, we define *two sets* of sub hybrids. The distinction is marked by the prefix $(\kappa \in S^*)$ and $(\kappa \notin S^*)$.

$(\kappa \in S^*)$-$\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_{5,\kappa}^{\mathcal{A}}(1^\lambda)$ except that as $\kappa \in S^*$, $\mathbf{s}_{\kappa,0}$ is nowhere used. So, we will set $\mathbf{b}_{\kappa,0} \leftarrow \mathbb{Z}_q^m$. The changes are marked in red.

• **Setup Phase.** $\mathcal{A}$ sends $1^N, \{j \to \mu_j\}_{j \in P}, i^*$ such that $P \subseteq [N], i^* \notin P, \mu_j \in \mathbb{Z}_2 \forall\ j \in P$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, set $\mathsf{pp} := (n, m, q, \chi, B, \mathbf{A}, N)$. Also sample $\mathbf{s}_{k,b} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{k,b} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m$ for each $k \in [N], b \in \{0,1\}, (k,b) \neq (\kappa, 0)$, set $\mathbf{b}_{k,b}^\top = \mathbf{s}_{k,b}^\top \mathbf{A} + \mathbf{e}_{k,b}^\top$ for $(k,b) \neq (\kappa, 0)$, $\mathbf{b}_{\kappa,0} \leftarrow \mathbb{Z}_q^m$. Perform the rest of the steps same as $\mathbf{Hyb}_{5,\kappa}^{\mathcal{A}}(1^\lambda)$.

$(\kappa \in S^*)$-$\mathbf{SubHyb}_2^{\mathcal{A}}(1^\lambda)$. This is the same as $(\kappa \in S^*)$-$\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)$ except that we set $\mathbf{c}_0 \leftarrow \mathbb{Z}_q^n, c_{i^*,\kappa,0} \leftarrow \mathbb{Z}_q$. The changes are marked in red.

• **Setup Phase.** $\mathcal{A}$ sends $1^N, \{j \to \mu_j\}_{j\in P}, i^*$ such that $P \subseteq [N], i^* \notin P, \mu_j \in \mathbb{Z}_2 \forall\ j \in P$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, set $\mathsf{pp} := (n, m, q, \chi, B, \mathbf{A}, N)$. Also sample $\mathbf{s}_{k,b} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{k,b} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m$ for each $k \in [N], b \in \{0,1\}, (k,b) \neq (\kappa, 0)$, set $\mathbf{b}_{k,b}^\top = \mathbf{s}_{k,b}^\top \mathbf{A} + \mathbf{e}_{k,b}^\top$ for $(k,b) \neq (\kappa, 0)$, $\mathbf{b}_{\kappa,0} \leftarrow \mathbb{Z}_q^m$.
For each $j \in P$,

1. Sample $\mathbf{x}_j \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{j,0} = \mathbf{A}\mathbf{x}_j$.
2. Sample for each $k \in [N]$, $z_{j,k} \leftarrow \mathbb{Z}_q$ such that $\sum_{k\in[N]} z_{j,k} = 0$. Sample $t_j \leftarrow \mathbb{Z}_q$ subject to the constraint that $t_j \notin [-\lfloor q/4 \rfloor - B, -\lfloor q/4 \rfloor + B] \cup [\lfloor q/4 \rfloor - B, \lfloor q/4 \rfloor + B]$.
3. Set $c_{j,j,1} = \mathbf{b}_{j,1}^\top \mathbf{x}_j + z_{j,j} + t_j$ and sample $c_{j,j,0} \leftarrow \mathbb{Z}_q$.
4. For $k \in [N], k \neq j$ for each $b \in \{0,1\}$, set $c_{j,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_j + z_{j,k}$.
5. Set $c_{j,T} = \lceil t_j \rfloor_2 + \mu_j$
6. Set $\mathsf{ct}_j = (\mathbf{c}_{j,0}, (c_{j,k,b})_{k,b}, c_{j,T})$.

For each $j = i^*$,

1. Sample $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{i^*,0} \leftarrow \mathbb{Z}_q^n$.
2. Sample for each $k \in [N] \setminus \{i^*\}$, $z_{i^*,k} \leftarrow \mathbb{Z}_q$.
3. Sample $c_{i^*,i^*,1}, c_{i^*,i^*,0} \leftarrow \mathbb{Z}_q$.
4. For $k \in [N], k \neq i^*$, for each $b \in \{0,1\}$, if $k < \kappa$, sample $c_{i^*,k,b} \leftarrow \mathbb{Z}_q$. Otherwise, if $k > \kappa$, set $c_{i^*,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_{i^*} + z_{i^*,k}$. Otherwise, if $k = \kappa$, sample $c_{i^*,\kappa,0} \leftarrow \mathbb{Z}_q$, $c_{i^*,\kappa,1} = \mathbf{b}_{\kappa,1}^\top \mathbf{x}_{i^*} + z_{i^*,\kappa}$.
5. Sample $c_{i^*,T} \leftarrow \mathbb{Z}_2$
6. Set $\mathsf{ct}_{i^*} = (\mathbf{c}_{i^*,0}, (c_{i^*,k,b})_{k,b}, c_{i^*,T})$.

For $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$. Send $(\mathsf{pp}, (\mathsf{ct}_k)_{k\in[N]})$ to $\mathcal{A}$.

• **Key Generation Query.** $\mathcal{A}$ sends a set S^* such that $i^* \in S^* \subseteq [N]$. Set $\mathsf{sk}_{S^*} = \sum_{j\in[N]} \mathbf{s}_{j,r_j}$ where $\mathbf{r} = (r_1, \ldots, r_N)$ is the characteristic vector of S^*. Send sk_{S^*} to $\mathcal{A}$.

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

$(\kappa \in S^*)$-$\mathbf{SubHyb}_3^{\mathcal{A}}(1^\lambda)$. This is the same as $(\kappa \in S^*)$-$\mathbf{SubHyb}_2^{\mathcal{A}}(1^\lambda)$ except $c_{i^*,\kappa,1} \leftarrow \mathbb{Z}_q$.

Claim F.7. Assuming the hardness of LWE (Assumption 3.3), for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_{5,\kappa}^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow (\kappa \in S^*)\text{-}\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}^{\mathcal{A}}_{5,\kappa}(1^\lambda)] - \Pr[1 \leftarrow (\kappa \in S^*)\text{-}\mathbf{SubHyb}^{\mathcal{A}}_1(1^\lambda)]| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. We will construct an adversary $\mathcal{B}$ that can break LWE assumption as follows.

$\mathcal{B}$ interacts with $\mathcal{A}$ and receives $(1^N, \{j \to \mu_j\}_{j\in P}, i^*)$. $\mathcal{B}$ also receives $\mathbf{A}, \mathbf{b}^\top \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^{1\times m}$ from LWE challenger. $\mathcal{B}$ sets $(n, m, q, \chi, B, \mathbf{A}, N)$ as pp. For each $(k, b) \in [N]\times\{0, 1\}, (k, b) \neq (\kappa, 0)$, $\mathcal{B}$ samples $\mathbf{s}_{k,b} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{k,b} \leftarrow \mathcal{D}^m_{\mathbb{Z},\chi}$, and sets $\mathbf{b}^\top_{k,b} = \mathbf{s}^\top_{k,b}\mathbf{A} + \mathbf{e}^\top_{k,b}$. $\mathcal{B}$ sets $\mathbf{b}_{\kappa,0} = \mathbf{b}$ (from LWE challenger).

For each $j \in P$, $\mathcal{B}$ uses $(\mathbf{b}_{k,b})_{k\in[N],b\in\{0,1\}}$ to compute $\mathsf{skEnc}(\mathsf{msk}, j, \mu_j)$ similar to $\mathbf{Hyb}^{\mathcal{A}}_{5,\kappa}(1^\lambda)$. For each $j \notin P \cup \{i^*\}$, $\mathcal{B}$ sets $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$. $\mathcal{B}$ now samples $z_{i^*,k} \leftarrow \mathbb{Z}_q$ for each $k \neq i^*$. For $j = i^*$, $\mathcal{B}$ samples $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$, sets $\mathbf{c}_0 := \mathbf{A}\mathbf{x}_{i^*}$. For each $k \in [N], k \neq i^*$, if $k < \kappa$, $\mathcal{B}$ samples $c_{i^*,k,b} \leftarrow \mathbb{Z}_q$ for $b \in \{0, 1\}$. Otherwise, if $k > \kappa$, $\mathcal{B}$ sets $c_{i^*,k,b} = \mathbf{b}^\top_{k,b}\mathbf{x}_{i^*} + z_{i^*,k}$. Otherwise, if $k = \kappa$, sample $c_{i^*,\kappa,0} = \mathbf{b}^\top\mathbf{x}_{i^*} + z_{i^*,\kappa}$ and $c_{i^*,\kappa,1} = \mathbf{b}^\top_{\kappa,1}\mathbf{x}_{i^*} + z_{i^*,\kappa}$. $\mathcal{B}$ also samples $c_{i^*,i^*,1} \leftarrow \mathbb{Z}_q, c_{i^*,i^*,0} \leftarrow \mathbb{Z}_q, c_{i^*,T} \leftarrow \mathbb{Z}_2$. $\mathcal{B}$ sets $\mathsf{ct}_{i^*} = (\mathbf{c}_{i^*,0}, (c_{i^*,k,b})_{k,b}, c_{i^*,T})$. $\mathcal{B}$ sends $(\mathsf{pp}, (\mathsf{ct}_j)_{j\in[N]})$ to $\mathcal{A}$.

$\mathcal{A}$ queries for secret key for set $S^* \subseteq [N], i^* \in S^*$. $\mathcal{B}$ sets $\mathsf{sk}_{S^*} = \mathbf{s}_{\kappa,1} + \sum_{k\in[N]\setminus\{\kappa\}} \mathbf{s}_{k,r_k}$ where $\mathbf{r} = (r_1, \ldots, r_N)$ such that $r_{i^*} = r_\kappa = 1$ is the characteristic vector of S^*. $\mathcal{B}$ sends sk_{S^*} to $\mathcal{A}$.

If LWE challenger's coin toss $b = 0$, then $\mathcal{B}$ acts as $\mathbf{Hyb}^{\mathcal{A}}_{5,\kappa}(1^\lambda)$. If coin toss was $b = 1$, then $\mathcal{B}$ acts as $(\kappa \in S^*)\text{-}\mathbf{SubHyb}^{\mathcal{A}}_1(1^\lambda)$. Thus, $\mathcal{B}$ can break the LWE assumption with advantage $\epsilon(\lambda)$. □

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

Proof. The only difference between these hybrids is that we set $(\mathbf{c}_{i^*,0}, c_{i^*,\kappa,0})$ in as $(\mathbf{A}\mathbf{x}_{i^*}, \mathbf{b}^\top\mathbf{x}_{i^*} + \mathbf{z}_{i^*,\kappa})$ for $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$ in one of them and $(\mathbf{c}_{i^*,0}, c_{i^*,\kappa,0}) \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q$ in the other. In addition, there is leakage of $\mathbf{x}_{i^*}$ in the form of $(\mathbf{b}^\top_{\kappa,1}\mathbf{x}_{i^*}, \mathbf{b}^\top_{\kappa+1,0}\mathbf{x}_{i^*}, \mathbf{b}^\top_{\kappa+1,1}\mathbf{x}_{i^*}, \ldots, \mathbf{b}^\top_{N,0}\mathbf{x}_{i^*}, \mathbf{b}^\top_{N,1}\mathbf{x}_{i^*})$ in both of the hybrids. As $m > (n + 1 + 2N)\log q + \lambda$, by generalized leftover hash lemma (Lemma 3.2, Item 2), these two hybrids are statistically close. □

Claim F.9. $\Pr[1 \leftarrow (\kappa \in S^*)\text{-}\mathbf{SubHyb}^{\mathcal{A}}_2(1^\lambda)] = \Pr[1 \leftarrow (\kappa \in S^*)\text{-}\mathbf{SubHyb}^{\mathcal{A}}_3(1^\lambda)]$

Proof. In both hybrids, $z_{i^*,\kappa}$ is sampled uniformly randomly from $\mathbb{Z}_q$ and independent from all other components. Thus, both in both hybrids $c_{i^*,\kappa,1}$ is identically distributed to uniformly random element from $\mathbb{Z}_q$. □

Claim F.10. Assuming the hardness of LWE (Assumption 3.3), there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any PPT adversary $\mathcal{A}$, $|\Pr[1 \leftarrow (\kappa \in S^*)\text{-}\mathbf{SubHyb}^{\mathcal{A}}_3(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}^{\mathcal{A}}_{5,\kappa+1}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. Firstly, in one of these hybrids, we set $\mathbf{c}_{i^*,0}$ as $\mathbf{A}\mathbf{x}_{i^*}$ for $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$, and $\mathbf{c}_{i^*,0} \leftarrow \mathbb{Z}_q^n$ in the other. In addition, there is leakage of $\mathbf{x}_{i^*}$ in the form of $(\mathbf{b}^\top_{\kappa+1,0}\mathbf{x}_{i^*}, \mathbf{b}^\top_{\kappa+1,1}\mathbf{x}_{i^*}, \ldots, \mathbf{b}^\top_{N,0}\mathbf{x}_{i^*}, \mathbf{b}^\top_{N,1}\mathbf{x}_{i^*})$ in both of the hybrids. As $m > (n + 1 + 2N)\log q + \lambda$, by generalized leftover hash lemma (Lemma 3.2, Item 2), these two distributions are statistically close.

Secondly, in one of these hybrids, we set $\mathbf{b}_{\kappa,0} \leftarrow \mathbb{Z}_q^m$ and in the other, $\mathbf{b}^\top_{\kappa,0} = \mathbf{s}^\top_{\kappa,0}\mathbf{A} + \mathbf{e}^\top_{\kappa,0}$ where $\mathbf{s}_{\kappa,0} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{\kappa,0} \leftarrow \mathcal{D}^m_{\mathbb{Z},\chi}$. However, as $\mathbf{s}_{\kappa,0}$ is not used anywhere, by relying on LWE security, we carn argue that this change is also not noticeable by any PPT adversary $\mathcal{A}$ similar to Claim F.7. □

Above claims prove on path where $\kappa \in S^*$, we can safely move from $\mathbf{Hyb}^{\mathcal{A}}_{5,\kappa}(1^\lambda)$ to $\mathbf{Hyb}^{\mathcal{A}}_{5,\kappa+1}(1^\lambda)$. Now, we will prove that even if $\kappa \notin S^*$, we can do the same using the following set of sub-hybrids and claims.

$(\kappa \notin S^*)$-**SubHyb**$_1^{\mathcal{A}}(1^\lambda)$**.** This is the same as **Hyb**$_{5,\kappa}^{\mathcal{A}}(1^\lambda)$ except that as $\kappa \notin S^*$, $\mathbf{s}_{\kappa,1}$ is nowhere used. So, we will set $\mathbf{b}_{\kappa,1} \leftarrow \mathbb{Z}_q^m$. The changes are marked in red.

• **Setup Phase.** $\mathcal{A}$ sends $1^N, \{j \to \mu_j\}_{j\in P}, i^*$ such that $P \subseteq [N], i^* \notin P, \mu_j \in \mathbb{Z}_2 \forall\ j \in P$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, set $\mathsf{pp} := (n, m, q, \chi, B, \mathbf{A}, N)$. Also sample $\mathbf{s}_{k,b} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{k,b} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m$ for each $k \in [N], b \in \{0,1\}, (k,b) \neq (\kappa, 1)$, set $\mathbf{b}_{k,b}^\top = \mathbf{s}_{k,b}^\top \mathbf{A} + \mathbf{e}_{k,b}^\top$ for $(k,b) \neq (\kappa,1)$, $\mathbf{b}_{\kappa,1} \leftarrow \mathbb{Z}_q^m$. Perform the rest of the steps same as **Hyb**$_{5,\kappa}^{\mathcal{A}}(1^\lambda)$.

$(\kappa \notin S^*)$-**SubHyb**$_2^{\mathcal{A}}(1^\lambda)$**.** This is the same as $(\kappa \notin S^*)$-**SubHyb**$_1^{\mathcal{A}}(1^\lambda)$ except that we set $\mathbf{c}_0 \leftarrow \mathbb{Z}_q^n, c_{i^*,\kappa,1} \leftarrow \mathbb{Z}_q$. The changes are marked in red.

• **Setup Phase.** $\mathcal{A}$ sends $1^N, \{j \to \mu_j\}_{j\in P}, i^*$ such that $P \subseteq [N], i^* \notin P, \mu_j \in \mathbb{Z}_2 \forall\ j \in P$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, set $\mathsf{pp} := (n, m, q, \chi, B, \mathbf{A}, N)$. Also sample $\mathbf{s}_{k,b} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{k,b} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m$ for each $k \in [N], b \in \{0,1\}, (k,b) \neq (\kappa, 1)$, set $\mathbf{b}_{k,b}^\top = \mathbf{s}_{k,b}^\top \mathbf{A} + \mathbf{e}_{k,b}^\top$ for $(k,b) \neq (\kappa,1)$, $\mathbf{b}_{\kappa,1} \leftarrow \mathbb{Z}_q^m$.
For each $j \in P$,

1. Sample $\mathbf{x}_j \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{j,0} = \mathbf{A}\mathbf{x}_j$.
2. Sample for each $k \in [N]$, $z_{j,k} \leftarrow \mathbb{Z}_q$ such that $\sum_{k\in[N]} z_{j,k} = 0$. Sample $t_j \leftarrow \mathbb{Z}_q$ subject to the constraint that $t_j \notin [-\lfloor q/4 \rfloor - B, -\lfloor q/4 \rfloor + B] \cup [\lfloor q/4 \rfloor - B, \lfloor q/4 \rfloor + B]$.
3. Set $c_{j,j,1} = \mathbf{b}_{j,1}^\top \mathbf{x}_j + z_{j,j} + t_j$ and sample $c_{j,j,0} \leftarrow \mathbb{Z}_q$.
4. For $k \in [N], k \neq j$ for each $b \in \{0,1\}$, set $c_{j,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_j + z_{j,k}$.
5. Set $c_{j,T} = \lceil t_j \rfloor_2 + \mu_j$
6. Set $\mathsf{ct}_j = (\mathbf{c}_{j,0}, (c_{j,k,b})_{k,b}, c_{j,T})$.

For each $j = i^*$,

1. Sample $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{i^*,0} \leftarrow \mathbb{Z}_q^n$.
2. Sample for each $k \in [N] \setminus \{i^*\}$, $z_{i^*,k} \leftarrow \mathbb{Z}_q$.
3. Sample $c_{i^*,i^*,1}, c_{i^*,i^*,0} \leftarrow \mathbb{Z}_q$.
4. For $k \in [N], k \neq i^*$, for each $b \in \{0,1\}$, if $k < \kappa$, sample $c_{i^*,k,b} \leftarrow \mathbb{Z}_q$. Otherwise, if $k > \kappa$, set $c_{i^*,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_{i^*} + z_{i^*,k}$. Otherwise, if $k = \kappa$, sample $c_{i^*,\kappa,1} \leftarrow \mathbb{Z}_q$, $c_{i^*,\kappa,0} = \mathbf{b}_{\kappa,1}^\top \mathbf{x}_{i^*} + z_{i^*,\kappa}$.
5. Sample $c_{i^*,T} \leftarrow \mathbb{Z}_2$
6. Set $\mathsf{ct}_{i^*} = (\mathbf{c}_{i^*,0}, (c_{i^*,k,b})_{k,b}, c_{i^*,T})$.

For $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$. Send $(\mathsf{pp}, (\mathsf{ct}_k)_{k\in[N]})$ to $\mathcal{A}$.

• **Key Generation Query.** $\mathcal{A}$ sends a set S^* such that $i^* \in S^* \subseteq [N]$. Set $\mathsf{sk}_{S^*} = \sum_{j\in[N]} \mathbf{s}_{j,r_j}$ where $\mathbf{r} = (r_1, \ldots, r_N)$ is the characteristic vector of S^*. Send sk_{S^*} to $\mathcal{A}$.

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

$(\kappa \notin S^*)$-$\mathbf{SubHyb}_3^{\mathcal{A}}(1^\lambda)$. This is the same as $(\kappa \notin S^*)$-$\mathbf{SubHyb}_2^{\mathcal{A}}(1^\lambda)$ except $c_{i^*,\kappa,0} \leftarrow \mathbb{Z}_q$.

Claim F.11. Assuming the hardness of LWE (Assumption 3.3), for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_{5,\kappa}^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow (\kappa \notin S^*)\text{-}\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$

Proof. This is similar to proof of Claim F.7. □

Claim F.12. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow (\kappa \notin S^*)\text{-}\mathbf{SubHyb}_1^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow (\kappa \notin S^*)\text{-}\mathbf{SubHyb}_2^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. This is similar to proof of Claim F.8. □

Claim F.13. $\Pr[1 \leftarrow (\kappa \notin S^*)\text{-}\mathbf{SubHyb}_2^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow (\kappa \notin S^*)\text{-}\mathbf{SubHyb}_3^{\mathcal{A}}(1^\lambda)]$

Proof. This is similar to proof of Claim F.9 □

Claim F.14. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow (\kappa \notin S^*)\text{-}\mathbf{SubHyb}_3^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_{5,\kappa+1}^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. This is similar to proof of Claim F.10. □

Using these claims and the piecewise guessing framework of [JKK+17, KW20], Lemma F.6 holds. □

Lemma F.15. There exists a negligible function $\mathsf{negl}(\cdot)$ such that for any adversary $\mathcal{A}$, $|\Pr[1 \leftarrow \mathbf{Hyb}_{5,N+1}^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. In $\mathbf{Hyb}_{5,N+1}^{\mathcal{A}}(1^\lambda)$, $\mathbf{c}_{i^*,0} = \mathbf{A}\mathbf{x}_{i^*}$ for $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$ and $c_{i^*,j,b} \leftarrow \mathbb{Z}_q$ for each $j \in [N], b \in \{0,1\}$, $c_{i^*,T} \leftarrow \mathbb{Z}_2$. In $\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$, $(\mathbf{c}_{i^*,0}, (c_{i^*,j,b})_{j,b}, c_T) \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$. These two hybrids are statistically close by leftover hash lemma (Lemma 3.2). □

F.2 Proof of Theorem 8.4

We define the following hybrids.

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the payload hiding experiment from Definition 6.1 with $b = 1$.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, \{j \to \mu_j\}_{j \in P}, i^*, \mu_{i^*}$ such that $P \subseteq [N], i^* \notin P, \mu_j \in \mathbb{Z}_2 \forall\, j \in P \cup \{i^*\}$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, set $\mathsf{pp} := (n, m, q, \chi, B, \mathbf{A}, N)$. Also sample $\mathbf{s}_{k,b} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{k,b} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m$ for each $k \in [N], b \in \{0,1\}$, set $\mathbf{b}_{k,b}^\top = \mathbf{s}_{k,b}^\top \mathbf{A} + \mathbf{e}_{k,b}^\top$. Set $\mathsf{msk} := (\mathbf{s}_{k,b}, \mathbf{b}_{k,b})_{k,b}$.

  For each $j \in P$,

  1. Sample $\mathbf{x}_j \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{j,0} = \mathbf{A}\mathbf{x}_j$.
  2. Sample for each $k \in [N]$, $z_{j,k} \leftarrow \mathbb{Z}_q$ such that $\sum_{k \in [N]} z_{j,k} = 0$. Sample $t_j \leftarrow \mathbb{Z}_q$ subject to the constraint that $t_j \notin [-\lfloor q/4 \rfloor - B, -\lfloor q/4 \rfloor + B] \cup [\lfloor q/4 \rfloor - B, \lfloor q/4 \rfloor + B]$.
  3. Set $c_{j,j,1} = \mathbf{b}_{j,1}^\top \mathbf{x}_j + z_{j,j} + t_j$ and sample $c_{j,j,0} \leftarrow \mathbb{Z}_q$.
  4. For $k \in [N], k \neq j$ for each $b \in \{0,1\}$, set $c_{j,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_j + z_{j,k}$.

5. Set $c_{j,T} = \lceil t_j \rfloor_2 + \mu_j$
6. Set $\mathsf{ct}_j = (\mathbf{c}_{j,0}, (c_{j,k,b})_{k,b}, c_{j,T})$.

For each $j = i^*$,

1. Sample $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{i^*,0} = \mathbf{A}\mathbf{x}_{i^*}$.
2. Sample for each $k \in [N]$, $z_{i^*,k} \leftarrow \mathbb{Z}_q$ such that $\sum_{k\in[N]} z_{i^*,k} = 0$. Sample $t_{i^*} \leftarrow \mathbb{Z}_q$ subject to the constraint that $t_{i^*} \notin [-\lfloor q/4 \rfloor - B, -\lfloor q/4 \rfloor + B] \cup [\lfloor q/4 \rfloor - B, \lfloor q/4 \rfloor + B]$.
3. Set $c_{i^*,i^*,1} = \mathbf{b}_{i^*,1}^\top \mathbf{x}_{i^*} + z_{i^*,i^*} + t_{i^*}$ and sample $c_{i^*,i^*,0} \leftarrow \mathbb{Z}_q$.
4. For $k \in [N], k \neq i^*$ for each $b \in \{0,1\}$, set $c_{i^*,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_{i^*} + z_{i^*,k}$.
5. Set $c_{i^*,T} = \lceil t_{i^*} \rfloor_2 + \mu_{i^*}$
6. Set $\mathsf{ct}_{i^*} = (\mathbf{c}_{i^*,0}, (c_{i^*,k,b})_{k,b}, c_{i^*,T})$.

For $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$.

Send $(\mathsf{pp}, (\mathsf{ct}_k)_{k\in[N]})$ to $\mathcal{A}$.

- **Key Generation Query.** $\mathcal{A}$ sends a set S^* such that $i^* \notin S^* \subseteq [N]$. Set $\mathsf{sk}_{S^*} = \sum_{j\in[N]} \mathbf{s}_{j,r_j}$ where $\mathbf{r} = (r_1, \ldots, r_N)$ is the characteristic vector of S^*. Send sk_{S^*} to $\mathcal{A}$.

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

$\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ except that we sample $\mathbf{b}_{i^*,1} \leftarrow \mathbb{Z}_q^m$.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ except that we sample $c_{i^*,i^*,1} \leftarrow \mathbb{Z}_q$ and $\mathbf{c}_{i^*,0} \leftarrow \mathbb{Z}_q^n$.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ except that we set $\mathbf{c}_{i^*,0} = \mathbf{A}\mathbf{x}_{i^*}$.

$\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ except that we sample $t_{i^*} \leftarrow \mathbb{Z}_q$ without the additional constraint.

$\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ except that we sample $c_{i^*,T} \leftarrow \mathbb{Z}_2$.

$\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$. This is the same as $\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$ except that we sample $z_{i^*,k} \leftarrow \mathbb{Z}_q$ for each $k \neq i^*$ and not sample z_{i^*,i^*} at all.

$\mathbf{Hyb}_{7,\kappa}^{\mathcal{A}}(1^\lambda)$ for $\kappa \in [N+1]$. In this hybrid, we will set $c_{i^*,k,0}$, $c_{i^*,k,1}$ as random elements from $\mathbb{Z}_q$ for each $1 \leq k < \kappa$. We write the full hybrid for clarity.

- **Setup Phase.** $\mathcal{A}$ sends $1^N, \{j \to \mu_j\}_{j\in P}, i^*, \mu_{i^*}$ such that $P \subseteq [N], i^* \notin P, \mu_j \in \mathbb{Z}_2 \forall\, j \in P \cup \{i^*\}$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, set $\mathsf{pp} := (n, m, q, \chi, B, \mathbf{A}, N)$. Also sample $\mathbf{s}_{k,b} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{k,b} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m$ for each $k \in [N], b \in \{0,1\}$, set $\mathbf{b}_{k,b}^\top = \mathbf{s}_{k,b}^\top \mathbf{A} + \mathbf{e}_{k,b}^\top$. Set $\mathsf{msk} := (\mathbf{s}_{k,b}, \mathbf{b}_{k,b})_{k,b}$.

  For each $j \in P$,

  1. Sample $\mathbf{x}_j \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{j,0} = \mathbf{A}\mathbf{x}_j$.

2. Sample for each $k \in [N]$, $z_{j,k} \leftarrow \mathbb{Z}_q$ such that $\sum_{k\in[N]} z_{j,k} = 0$. Sample $t_j \leftarrow \mathbb{Z}_q$ subject to the constraint that $t_j \notin [-\lfloor q/4 \rfloor - B, -\lfloor q/4 \rfloor + B] \cup [\lfloor q/4 \rfloor - B, \lfloor q/4 \rfloor + B]$.
3. Set $c_{j,j,1} = \mathbf{b}_{j,1}^\top \mathbf{x}_j + z_{j,j} + t_j$ and sample $c_{j,j,0} \leftarrow \mathbb{Z}_q$.
4. For $k \in [N], k \neq j$ for each $b \in \{0,1\}$, set $c_{j,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_j + z_{j,k}$.
5. Set $c_{j,T} = \lceil t_j \rfloor_2 + \mu_j$
6. Set $\mathsf{ct}_j = (\mathbf{c}_{j,0}, (c_{j,k,b})_{k,b}, c_{j,T})$.

For each $j = i^*$,

1. Sample $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$, set $\mathbf{c}_{i^*,0} = \mathbf{A}\mathbf{x}_{i^*}$.
2. Sample for each $k \in [N] \setminus \{i^*\}$, $z_{i^*,k} \leftarrow \mathbb{Z}_q$.
3. Sample $c_{i^*,i^*,1}, c_{i^*,i^*,0} \leftarrow \mathbb{Z}_q$.
4. For $k \in [N], k \neq i^*$, for each $b \in \{0,1\}$, if $k < \kappa$, sample $c_{i^*,k,b} \leftarrow \mathbb{Z}_q$. Otherwise, if $k \geq \kappa$, set $c_{i^*,k,b} := \mathbf{b}_{k,b}^\top \mathbf{x}_{i^*} + z_{i^*,k}$.
5. Sample $c_{i^*,T} \leftarrow \mathbb{Z}_2$
6. Set $\mathsf{ct}_{i^*} = (\mathbf{c}_{i^*,0}, (c_{i^*,k,b})_{k,b}, c_{i^*,T})$.

For $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q^{2N} \times \mathbb{Z}_2$.

Send $(\mathsf{pp}, (\mathsf{ct}_k)_{k\in[N]})$ to $\mathcal{A}$.

- **Key Generation Query.** $\mathcal{A}$ sends a set S^* such that $i^* \notin S^* \subseteq [N]$. Set $\mathsf{sk}_{S^*} = \sum_{j\in[N]} \mathbf{s}_{j,r_j}$ where $\mathbf{r} = (r_1, \ldots, r_N)$ is the characteristic vector of S^*. Send sk_{S^*} to $\mathcal{A}$.
- **Guess Phase.** Output whatever $\mathcal{A}$ outputs.

$\mathbf{Hyb}_8^{\mathcal{A}}(1^\lambda)$. This is the payload hiding experiment from Definition 6.1 with $b = 0$.

Lemma F.16. Assuming the hardness of LWE assumption (Assumption 3.3), there exists a negligible function $\mathsf{negl}(\cdot)$ such that for any PPT 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. As skEnc does not use $\mathbf{s}_{j,b}$ for any $j \in [N], b \in \{0,1\}$, and solely relies on the corresponding $\mathbf{b}_{j,b}$, and as $i^* \notin S^*$, $\mathbf{s}_{i^*,1}$ is nowhere used, we can argue that the lemma holds similarly to Claim F.7. □

Lemma F.17. 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 only difference between these hybrids is that in $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$, we sample $(\mathbf{c}_{i^*,0}, c_{i^*,i^*,1})$ as $(\mathbf{A}\mathbf{x}_{i^*}, \mathbf{b}_{i^*,1}^\top \mathbf{x}_{i^*} + z_{i^*,i^*})$ for $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$, and in $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$, we sample $(\mathbf{c}_{i^*,0}, c_{i^*,i^*,1}) \leftarrow \mathbb{Z}_q^n \times \mathbb{Z}_q$. Moreover, there is leakage on $\mathbf{x}_{i^*}$ in the form of

$$(\mathbf{b}_{1,0}^\top \mathbf{x}_{i^*}, \mathbf{b}_{1,1}^\top \mathbf{x}_{i^*}, \ldots, \mathbf{b}_{i^*-1,0}^\top \mathbf{x}_{i^*}, \mathbf{b}_{i^*-1,1}^\top \mathbf{x}_{i^*}, \mathbf{b}_{i^*+1,0}^\top \mathbf{x}_{i^*}, \mathbf{b}_{i^*+1,1}^\top \mathbf{x}_{i^*}, \ldots, \mathbf{b}_{N,0}^\top \mathbf{x}_{i^*}, \mathbf{b}_{N,1}^\top \mathbf{x}_{i^*})$$

in both of these hybrids. As $m > (n+1+2N)\log q + \lambda$, both of these hybrids are statistically close by generalized leftover hash lemma (Lemma 3.2, Item 2). □

Lemma F.18. 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 these hybrids is that in $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$, we sample $\mathbf{c}_{i^*,0}$ as $\mathbf{A}\mathbf{x}_{i^*}$ for $\mathbf{x}_{i^*} \leftarrow \{\pm 1\}^m$, and in $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$, we sample $\mathbf{c}_{i^*,0} \leftarrow \mathbb{Z}_q^n$. Moreover, there is leakage on $\mathbf{x}_{i^*}$ in the form of

$$(\mathbf{b}_{1,0}^\top \mathbf{x}_{i^*}, \mathbf{b}_{1,1}^\top \mathbf{x}_{i^*}, \ldots, \mathbf{b}_{i^*-1,0}^\top \mathbf{x}_{i^*}, \mathbf{b}_{i^*-1,1}^\top \mathbf{x}_{i^*}, \mathbf{b}_{i^*+1,0}^\top \mathbf{x}_{i^*}, \mathbf{b}_{i^*+1,1}^\top \mathbf{x}_{i^*}, \ldots, \mathbf{b}_{N,0}^\top \mathbf{x}_{i^*}, \mathbf{b}_{N,1}^\top \mathbf{x}_{i^*})$$

in both of these hybrids. As $m > (n+1+2N)\log q + \lambda$, both of these hybrids are statistically close by generalized leftover hash lemma (Lemma 3.2, Item 2). □

Lemma F.19. 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 is similar to proof of Lemma F.2. □

Lemma F.20. 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. As t_{i^*} is not used anywhere in ct_{i^*}, and $t_{i^*} \leftarrow \mathbb{Z}_q$ and $\lceil t_{i^*} \rfloor_2$ is statistically close to a uniform element from $\mathbb{Z}_2$ (as q is an odd prime and super-polynomial, statistical distance is at most $1/2q$), we have that, $\lceil t_{i^*} \rfloor_2 + \mu_{i^*}$ is statistically close to a uniformly random element form $\mathbb{Z}_2$ which is what is done in $\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$. □

Lemma F.21. $\Pr[1 \leftarrow \mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)]$

Proof. As z_{i^*,i^*} is not used anywhere ($c_{i^*,i^*,1}$ is a random $\mathbb{Z}_q$ element), $\{z_{i^*,k}\}_{k \neq i^*}$ is distributed identically to uniformly random $N-1$ elements from $\mathbb{Z}_q$. Thus, the lemma holds. □

Lemma F.22. $\Pr[1 \leftarrow \mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_{7,0}^{\mathcal{A}}(1^\lambda)]$

Proof. As we are not setting $c_{i^*,k,0}$ or $c_{i^*,k,1}$ to be random elements from $\mathbb{Z}_q$ for any $k \in [N]$ in both these hybrids, they are identically distributed. □

Lemma F.23. Assuming the hardness of LWE (Assumption 3.3), for any PPT adversary $\mathcal{A}$, $\forall\ \kappa \in [N]$ there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_{7,\kappa}^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_{7,\kappa+1}^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. This is similar to proof of Lemma F.6. We omit full proof for brevity. □

Lemma F.24. $\Pr[1 \leftarrow \mathbf{Hyb}_{7,N+1}^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_8^{\mathcal{A}}(1^\lambda)]$

Proof. This is similar to proof of Lemma F.15. □

G Proofs from Section 9

G.1 Proof of Theorem 9.4

We define the following hybrids.

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the mode indistinguishability experiment from Definition 6.1 with $b = 0$.

- **Setup Phase.** $\mathcal{A}$ sends $(1^N, \{j \to \mu_j\}_{j\in P}, i^*)$ for $P \subseteq [N]$, each $\mu_j \in \mathbb{Z}_2$, and $i^* \notin P$. Chal sets $\mathsf{pp} := (n, m, q, \chi, \sigma, N)$.

  Chal also samples $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, $\mathbf{B} \leftarrow \mathbb{Z}_q^{n\times \lceil \log N\rceil m}, \mathbf{d} \leftarrow \mathbb{Z}_q^n$. For each $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^{1\times(\lceil \log N\rceil+1)m} \times \mathbb{Z}_2$. For each $j \in P$,

  - Sample $\mathbf{s}_j \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{1,j} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{X}_j \leftarrow \{\pm 1\}^{m\times\lceil \log N\rceil m}$.
  - Compute $\mathbf{c}_{1,j}^\top = \mathbf{s}_j^\top \mathbf{A} + \mathbf{e}_{1,j}^\top$, $\mathbf{c}_{2,j}^\top = \mathbf{s}_j^\top(\mathbf{B} + \mathbf{j}^\top \otimes \mathbf{G}) + \mathbf{e}_{1,j}^\top \mathbf{X}_j$, and $c_{3,j} = \lceil \mathbf{s}_j^\top \mathbf{d}\rfloor_2 + \mu_j$.
  - Set $\mathsf{ct}_j = (\mathbf{c}_{1,j}^\top, \mathbf{c}_{2,j}^\top, c_{3,j})$.

  For $j = i^*$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^{1\times(\lceil \log N\rceil+1)m} \times \mathbb{Z}_2$. Send $\mathsf{pp}, (\mathsf{ct}_j)_{j\in[N]}$ to $\mathcal{A}$.

- **Key Generation Query.** $\mathcal{A}$ sends $S^* \subseteq [N]$ such that $i^* \in S^*$. Chal computes $\mathbf{B}_{S^*} = \mathsf{EvalF}(\mathbf{B}, \mathsf{mem}\text{-}S^*)$. Chal samples $\mathbf{r}_{S^*} \leftarrow \mathsf{SamplePre}\left([\mathbf{A} \mid \mathbf{B}_{S^*}], \begin{bmatrix}\mathbf{T_A}\\ \mathbf{0}\end{bmatrix}, \mathbf{d}, \sigma\right)$. Send $(\mathbf{r}_{S^*}, \mathbf{B})$ to $\mathcal{A}$.

$\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ except that we set $\mathbf{c}_{1,i^*}^\top, \mathbf{c}_{2,i^*}^\top$ to be $\mathbf{s}_{i^*}^\top \mathbf{A} + \mathbf{e}_{1,i^*}^\top, \mathbf{s}_{i^*}^\top(\mathbf{B} + \mathbf{i} \otimes \mathbf{G}) + \mathbf{e}_{2,i^*}^\top$ respectively. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $(1^N, \{j \to \mu_j\}_{j\in P}, i^*)$ for $P \subseteq [N]$, each $\mu_j \in \mathbb{Z}_2$, and $i^* \notin P$. Chal sets $\mathsf{pp} := (n, m, q, \chi, \sigma, N)$.

  Chal also samples $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, $\mathbf{B} \leftarrow \mathbb{Z}_q^{n\times \lceil \log N\rceil m}, \mathbf{d} \leftarrow \mathbb{Z}_q^n$. For each $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^{1\times(\lceil \log N\rceil+1)m} \times \mathbb{Z}_2$. For each $j \in P$,

  - Sample $\mathbf{s}_j \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{1,j} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{X}_j \leftarrow \{\pm 1\}^{m\times\lceil \log N\rceil m}$.
  - Compute $\mathbf{c}_{1,j}^\top = \mathbf{s}_j^\top \mathbf{A} + \mathbf{e}_{1,j}^\top$, $\mathbf{c}_{2,j}^\top = \mathbf{s}_j^\top(\mathbf{B} + \mathbf{j}^\top \otimes \mathbf{G}) + \mathbf{e}_{1,j}^\top \mathbf{X}_j$, and $c_{3,j} = \lceil \mathbf{s}_j^\top \mathbf{d}\rfloor_2 + \mu_j$.
  - Set $\mathsf{ct}_j = (\mathbf{c}_{1,j}^\top, \mathbf{c}_{2,j}^\top, c_{3,j})$.

  For $j = i^*$, sample $\mathbf{s}_{i^*}^\top \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{1,i^*}^\top \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{X}_{i^*} \leftarrow \{\pm 1\}^{m\times\lceil \log N\rceil m}$ and compute $\mathbf{c}_{1,i^*}^\top = \mathbf{s}_{i^*}^\top \mathbf{A} + \mathbf{e}_{1,i^*}^\top, \mathbf{c}_{2,i^*}^\top = \mathbf{s}_{i^*}^\top(\mathbf{B} + (\mathbf{i}^*)^\top \otimes \mathbf{G}) + \mathbf{e}_{1,i^*}^\top \mathbf{X}_{i^*}$, $c_3 \leftarrow \mathbb{Z}_2$. Set $\mathsf{ct}_{i^*} = (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$. Send $\mathsf{pp}, (\mathsf{ct}_j)_{j\in[N]}$ to $\mathcal{A}$.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is the mode indistinguishability game from Definition 6.1 with $b = 1$. Specifically, we set $c_3 = \lceil \mathbf{s}_{i^*}^\top \mathbf{d}\rfloor_2 + \mu_{i^*}^*$ after sampling $\mu_{i^*}^* \leftarrow \mathbb{Z}_2$.

Lemma G.1. Assuming the hardness of 1-key LWE (Assumption 9.1), for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)]| \le \mathsf{negl}(\lambda)$.

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 1-key LWE assumption with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ interacts with $\mathcal{A}$ and receives $1^N, \{j \to \mu_j\}_{j\in P}, i^*$. $\mathcal{B}$ sends $\mathbf{i}^*$ to Chal and receives $\mathbf{A}, \mathbf{B}, \mathbf{d}, \mathbf{c}_1^\top, \mathbf{c}_2^\top$. $\mathcal{B}$ samples pp like in $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. For each $j \notin P \cup \{i^*\}$, $\mathcal{B}$ samples $\mathsf{ct} \leftarrow \mathbb{Z}_q^{1\times(\lceil \log N\rceil+1)m} \times \mathbb{Z}_2$. For each $j \in P$, sample $\mathbf{s}_j \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{1,j} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{X}_j \leftarrow \{\pm 1\}^{m\times\lceil \log N\rceil m}$. Set $\mathsf{ct}_j = (\mathbf{s}_j^\top \mathbf{A} + \mathbf{e}_{1,j}^\top, \mathbf{s}_j^\top(\mathbf{B} + \mathbf{j}^\top \otimes \mathbf{G}) + \mathbf{e}_{1,j}^\top \mathbf{X}_j, \lceil \mathbf{s}_j^\top \mathbf{d}\rfloor_2 + \mu_j)$. $\mathcal{B}$ sets $\mathsf{ct}_{i^*} = (\mathbf{c}_1^\top, \mathbf{c}_2^\top, c_3)$ where $c_3 \leftarrow \mathbb{Z}_2$. $\mathcal{B}$ sends $(\mathsf{pp}, (\mathsf{ct}_j)_{j\in[N]})$ to $\mathcal{A}$. When $\mathcal{A}$ makes key generation query for $S^* \subseteq [N]$ such that $i^* \in S^*$. $\mathcal{B}$ sends S^* to Chal, receives $\mathbf{r}_S$, and sends $(\mathbf{r}_S, \mathbf{B})$ to $\mathcal{A}$.

If Chal's coin toss $b = 0$, then $\mathbf{c}_1^\top = \mathbf{s}^\top \mathbf{A} + \mathbf{e}_1^\top$ and $\mathbf{c}_2^\top = \mathbf{s}^\top(\mathbf{B} + (\mathbf{i}^*)^\top \otimes \mathbf{G}) + \mathbf{e}_1^\top \mathbf{X}$ for some $\mathbf{s} \in \mathbb{Z}_q^n, \mathbf{e}_1 \in \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{X} \in \{\pm 1\}^{m\times\lceil \log N\rceil m}$. In this case, $\mathcal{B}$ behaves like $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. If Chal's coin toss $b = 1$, then $\mathcal{B}$ behaves like $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. Thus, $\mathcal{B}$ has advantage ϵ in breaking the 1-key LWE assumption. □

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

Proof. Because $\mu_{i^*}^*$ is independently and uniformly randomly chosen, in both of these hybrids, the third component of ct_{i^*} is uniformly randomly distributed. Thus, lemma holds. □

G.2 Proof of Theorem 9.5

We define the following hybrids.

$\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$. This is the payload hiding experiment from Definition 6.1 with $b = 0$.

- **Setup Phase.** $\mathcal{A}$ sends $(1^N, \{j \to \mu_j\}_{j\in P}, i^*, \mu^*)$ for $P \subseteq [N]$, each $\mu_j, \mu^* \in \mathbb{Z}_2$, and $i^* \notin P$. Chal sets $\mathsf{pp} := (n, m, q, \chi, \sigma, N)$.

  Chal also samples $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, $\mathbf{B} \leftarrow \mathbb{Z}_q^{n\times\lceil \log N\rceil m}, \mathbf{d} \leftarrow \mathbb{Z}_q^n$. For each $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^{1\times(\lceil \log N\rceil+1)m} \times \mathbb{Z}_2$. For each $j \in P$,

  - Sample $\mathbf{s}_j \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{1,j} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{X}_j \leftarrow \{\pm 1\}^{m\times\lceil \log N\rceil m}$.
  - Compute $\mathbf{c}_{1,j}^\top = \mathbf{s}_j^\top \mathbf{A} + \mathbf{e}_{1,j}^\top$, $\mathbf{c}_{2,j}^\top = \mathbf{s}_j^\top(\mathbf{B} + \mathbf{j}^\top \otimes \mathbf{G}) + \mathbf{e}_{1,j}^\top \mathbf{X}_j$, and $c_{3,j} = \lceil \mathbf{s}_j^\top \mathbf{d}\rfloor_2 + \mu_j$.
  - Set $\mathsf{ct}_j = (\mathbf{c}_{1,j}^\top, \mathbf{c}_{2,j}^\top, c_{3,j})$.

  For $j = i^*$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^{1\times(\lceil \log N\rceil+1)m} \times \mathbb{Z}_2$. Send $\mathsf{pp}, (\mathsf{ct}_j)_{j\in[N]}$ to $\mathcal{A}$.
- **Key Generation Query.** $\mathcal{A}$ sends $S^* \subseteq [N]$ such that $i^* \notin S^*$. Chal computes $\mathbf{B}_{S^*} = \mathsf{EvalF}(\mathbf{B}, \mathsf{mem}\text{-}S^*)$. Chal samples $\mathbf{r}_{S^*} \leftarrow \mathsf{SamplePre}\left([\mathbf{A} \mid \mathbf{B}_{S^*}], \begin{bmatrix}\mathbf{T}_\mathbf{A}\\ \mathbf{0}\end{bmatrix}, \mathbf{d}, \sigma\right)$. Send $(\mathbf{r}_{S^*}, \mathbf{B})$ to $\mathcal{A}$.

$\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$ except that we set $\mathbf{B} = \mathbf{AX} - (\mathbf{i}^*)^\top \otimes \mathbf{G}$ where $\mathbf{X} \leftarrow \{\pm 1\}^{m\times\lceil \log N\rceil m}$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $(1^N, \{j \to \mu_j\}_{j\in P}, i^*, \mu^*)$ for $P \subseteq [N]$, each $\mu_j, \mu^* \in \mathbb{Z}_2$, and $i^* \notin P$. Chal sets $\mathsf{pp} := (n, m, q, \chi, \sigma, N)$.

  Chal also samples $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, sample $\mathbf{X} \leftarrow \{\pm 1\}^{m\times\lceil \log N\rceil m}$, set $\mathbf{B} = \mathbf{AX} - (\mathbf{i}^*)^\top \otimes \mathbf{G}$, $\mathbf{d} \leftarrow \mathbb{Z}_q^n$. Perform the rest of the steps same as $\mathbf{Hyb}_0^{\mathcal{A}}(1^\lambda)$.

$\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_1^{\mathcal{A}}(1^\lambda)$ except that we sample $\mathbf{A}$ uniformly randomly and sample pre-image using $\mathbf{X}$ and $\mathbf{H}_{S^*,\mathbf{i}^*}$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $(1^N, \{j \to \mu_j\}_{j\in P}, i^*, \mu^*)$ for $P \subseteq [N]$, each $\mu_j, \mu^* \in \mathbb{Z}_2$, and $i^* \notin P$. Chal sets $\mathsf{pp} := (n, m, p, q, \chi, \sigma, N)$.

  Chal also samples $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, sample $\mathbf{X} \leftarrow \{\pm 1\}^{m\times \lceil \log N\rceil m}$, set $\mathbf{B} = \mathbf{AX} - (\mathbf{i}^*)^\top \otimes \mathbf{G}$, $\mathbf{d} \leftarrow \mathbb{Z}_q^n$. For each $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^{1\times(\lceil \log N\rceil+1)m} \times \mathbb{Z}_2$. For each $j \in P$,

  - Sample $\mathbf{s}_j \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{1,j} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{X}_j \leftarrow \{\pm 1\}^{m\times\lceil \log N\rceil m}$.
  - Compute $\mathbf{c}_{1,j}^\top = \mathbf{s}_j^\top \mathbf{A} + \mathbf{e}_{1,j}^\top$, $\mathbf{c}_{2,j}^\top = \mathbf{s}_j^\top(\mathbf{B} + \mathbf{j}^\top \otimes \mathbf{G}) + \mathbf{e}_{1,j}^\top \mathbf{X}_j$, and $c_{3,j} = \lceil \mathbf{s}_j^\top \mathbf{d}\rfloor_2 + \mu_j$.
  - Set $\mathsf{ct}_j = (\mathbf{c}_{1,j}^\top, \mathbf{c}_{2,j}^\top, c_{3,j})$.

  For $j = i^*$, sample $\mathbf{c}_{1,i^*}^\top \leftarrow \mathbb{Z}_q^m, c_{3,i^*} \leftarrow \mathbb{Z}_2$, set $\mathsf{ct}_{i^*} := (\mathbf{c}_{1,i^*}^\top, \mathbf{c}_{1,i^*}^\top \mathbf{X}, c_{3,i^*})$. Send $\mathsf{pp}, (\mathsf{ct}_j)_{j\in[N]}$ to $\mathcal{A}$.

- **Key Generation Query.** $\mathcal{A}$ sends $S^* \subseteq [N]$ such that $i^* \notin S^*$. Chal computes $\mathbf{H}_{S^*,\mathbf{i}^*} = \mathsf{EvalFX}(\mathbf{B}, \mathsf{mem}\text{-}S^*, \mathbf{i})$. Chal samples $\mathbf{r}_{S^*} \leftarrow \mathsf{SamplePre}\left([\mathbf{A} \mid \mathbf{B}_{S^*}], \begin{bmatrix}\mathbf{X}\mathbf{H}_{S^*,\mathbf{i}^*} \\ -\mathbf{I}\end{bmatrix}, \mathbf{d}, \sigma\right)$. Send $(\mathbf{r}_{S^*}, \mathbf{B})$ to $\mathcal{A}$.

$\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)$ except that we set ct_{i^*} as $(\mathbf{c}_{1,i^*}^\top, \mathbf{c}_{1,i^*}^\top \mathbf{X}, c_{3,i^*} + \mu^*)$.

$\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$ except that we set $\mathbf{c}_{1,i^*}^\top = \mathbf{s}_{i^*}^\top \mathbf{A} + \mathbf{e}_{1,i^*}^\top, c_{3,i^*} = \lceil \mathbf{s}_{i^*}^\top \mathbf{d}\rfloor_2$. The changes are highlighted in red.

- **Setup Phase.** $\mathcal{A}$ sends $(1^N, \{j \to \mu_j\}_{j\in P}, i^*, \mu^*)$ for $P \subseteq [N]$, each $\mu_j, \mu^* \in \mathbb{Z}_2$, and $i^* \notin P$. Chal sets $\mathsf{pp} := (n, m, p, q, \chi, \sigma, N)$.

  Chal also samples $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, sample $\mathbf{X} \leftarrow \{\pm 1\}^{m\times \lceil \log N\rceil m}$, set $\mathbf{B} = \mathbf{AX} - (\mathbf{i}^*)^\top \otimes \mathbf{G}$, $\mathbf{d} \leftarrow \mathbb{Z}_q^n$. For each $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^{1\times(\lceil \log N\rceil+1)m} \times \mathbb{Z}_2$. For each $j \in P$,

  - Sample $\mathbf{s}_j \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{1,j} \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{X}_j \leftarrow \{\pm 1\}^{m\times\lceil \log N\rceil m}$.
  - Compute $\mathbf{c}_{1,j}^\top = \mathbf{s}_j^\top \mathbf{A} + \mathbf{e}_{1,j}^\top$, $\mathbf{c}_{2,j}^\top = \mathbf{s}_j^\top(\mathbf{B} + \mathbf{j}^\top \otimes \mathbf{G}) + \mathbf{e}_{1,j}^\top \mathbf{X}_j$, and $c_{3,j} = \lceil \mathbf{s}_j^\top \mathbf{d}\rfloor_2 + \mu_j$.
  - Set $\mathsf{ct}_j = (\mathbf{c}_{1,j}^\top, \mathbf{c}_{2,j}^\top, c_{3,j})$.

  For $j = i^*$, sample $\mathbf{s}_{i^*} \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{1,i^*}^\top \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{1\times m}, \mathbf{e}_{3,i^*}^\top \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^{1\times k}, c_{3,i^*} \leftarrow \mathbb{Z}_2$, set $\mathsf{ct}_j := (\mathbf{s}_{i^*}^\top \mathbf{A} + \mathbf{e}_{1,i^*}^\top, (\mathbf{s}_{i^*}^\top \mathbf{A} + \mathbf{e}_{1,i^*}^\top)\mathbf{X}, \lceil \mathbf{s}_{i^*}^\top \mathbf{d}\rfloor_2 + \mu^*)$. Send $\mathsf{pp}, (\mathsf{ct}_j)_{j\in[N]}$ to $\mathcal{A}$.

$\mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)$. This is same as $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ except that we set $\mathsf{ct}_{i^*} = (\mathbf{s}_{i^*}^\top \mathbf{A} + \mathbf{e}_{1,i^*}^\top, \mathbf{s}_{i^*}^\top(\mathbf{B} + (\mathbf{i}^*)^\top \otimes \mathbf{G}) + \mathbf{e}_{1,i^*}^\top \mathbf{X}, \lceil \mathbf{s}_{i^*}^\top \mathbf{d}\rfloor_2 + \mu^*)$.

$\mathbf{Hyb}_6^{\mathcal{A}}(1^\lambda)$. This is payload hiding experiment with $b = 1$.

Lemma G.3. There exists a negligible function 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 generalized leftover hash lemma (Lemma 3.2). Although $\mathbf{A}$ is not uniformly random, $\mathbf{A}$ is statistically close to a uniformly random matrix by matrix distribution property of lattice trapdoors (Appendix A). Thus, the distribution of $(\mathbf{A}, \mathbf{AX})$ is statistically close to $(\mathbf{A}, \mathbf{B})$ where $\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times \lceil \log N \rceil m}$. Thus, the lemma holds. □

Lemma G.4. There exists a negligible function such that or 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. Firstly, by matrix distribution property of lattice trapdoors (Appendix A), $\mathbf{A}$ is statistically close to a uniformly random matrix. So, sampling $\mathbf{A}$ as a uniformly random matrix is not noticeable by any $\mathcal{A}$. Moreover, $\mathbf{A}, \mathbf{c}_{1,i^*}^\top, \mathbf{AX}, \mathbf{c}_{1,i^*}^\top \mathbf{X}$ is statistically close to $\mathbf{A}, \mathbf{c}_{1,i^*}^\top, \mathbf{AX}, \mathbf{c}_{2,i^*}^\top$ for random $(\mathbf{c}_{1,i^*}^\top, \mathbf{c}_{2,i^*}^\top)$ by generalized leftover hash lemma (Lemma 3.2).

Secondly, by homomorphic computation machinery, $(\mathbf{B} + (\mathbf{i}^*)^\top \otimes \mathbf{G})\mathbf{H}_{S^*,\mathbf{i}^*} = \mathbf{B}_{S^*} + \mathbf{G}$ as $i^* \notin S^*$. So, $[\mathbf{A} \mid \mathbf{B}_{S^*}] = [\mathbf{A} \mid (\mathbf{B} + (\mathbf{i}^*)^\top \otimes \mathbf{G})\mathbf{H}_{S^*,\mathbf{i}^*} - \mathbf{G}] = [\mathbf{A} \mid \mathbf{AXH}_{S^*,\mathbf{i}^*} - \mathbf{G}]$. Thus, $[\mathbf{A} \mid \mathbf{B}_{S^*}] \begin{bmatrix} \mathbf{XH}_{S^*,\mathbf{i}^*} \\ -\mathbf{I} \end{bmatrix} = \mathbf{G}$. Thus, $\begin{bmatrix} \mathbf{XH}_{S^*,\mathbf{i}^*} \\ -\mathbf{I} \end{bmatrix}$ acts as a gadget trapdoor for $[\mathbf{A} \mid \mathbf{B}_{S^*}]$. So, as long as $\sigma > m^{O(\lambda)} \log n$, the pre-image $\mathbf{r}_{S^*}$'s distribution in both of these hybrids are statistically close to a discrete Gaussian from the coset lattice $(\Lambda_q^{\mathbf{d}}([\mathbf{A} \mid \mathbf{B}_{S^*}]))$ with σ width. Thus, the lemma holds. □

Lemma G.5. $\Pr[1 \leftarrow \mathbf{Hyb}_2^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)]$

Proof. In both these hybrids, the third component of ct_{i^*} is uniformly randomly distributed in $\mathbb{Z}_2$. Thus, the lemma holds. □

Lemma G.6. Assuming the hardness of LWE assumption (Assumption 3.3), for any PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)]| \leq \mathsf{negl}(\lambda)$.

Proof. Assume that there exists an adversary $\mathcal{A}$ such that $|\Pr[1 \leftarrow \mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)] - \Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)]| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. We will construct an adversary $\mathcal{B}$ that can break the LWE assumption with advantage $\epsilon(\lambda)$.

$\mathcal{B}$ receives $[\mathbf{A} \mid \mathbf{d}], [\mathbf{c}_1^\top \mid c_3]$ from LWE challenger Chal. $\mathcal{B}$ then interacts with $\mathcal{A}$ and receives $(1^N, \{j \to \mu_j\}_{j \in P}, i^*, \mu^*)$ where $i^* \notin P \subseteq [N], \mu_j, \mu^* \in \mathbb{Z}_2$. $\mathcal{B}$ sets $\mathsf{pp} := (n, m, q, \chi, \sigma, N)$ similar to $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$. Also, $\mathcal{B}$ samples $\mathbf{X} \leftarrow \{\pm 1\}^{m \times \lceil \log N \rceil m}$ and sets $\mathbf{B} = \mathbf{AX} - (\mathbf{i}^*)^\top \mathbf{G}$ and samples $\mathbf{d} \leftarrow \mathbb{Z}_2$. For each $j \notin P \cup \{i^*\}$, sample $\mathsf{ct}_j \leftarrow \mathbb{Z}_q^{1 \times (\lceil \log N \rceil + 1)m} \times \mathbb{Z}_2$. For each $j \in P$, $\mathcal{B}$ samples $\mathbf{s}_j \leftarrow \mathbb{Z}_q^n, \mathbf{e}_{1,j}^\top \leftarrow \mathcal{D}_{\mathbb{Z},\chi}^m, \mathbf{X}_j \leftarrow \{\pm 1\}^{m \times \lceil \log N \rceil m}$ and sets $\mathsf{ct}_j = (\mathbf{s}^\top \mathbf{A} + \mathbf{e}_{1,j}^\top, \mathbf{s}^\top (\mathbf{B} + \mathbf{j}^\top \otimes \mathbf{G}) + \mathbf{e}_{1,j}^\top \mathbf{X}_j, \lceil \mathbf{s}^\top \mathbf{d} \rfloor_2 + \mu_j)$. For $j = i^*$, $\mathcal{B}$ sets $\mathsf{ct}_{i^*} = (\mathbf{c}_1^\top, \mathbf{c}_1^\top \mathbf{X}, \lceil c_3 \rfloor_2 + \mu^*)$. $\mathcal{B}$ sends $(\mathsf{pp}, (\mathsf{ct}_j)_{j \in [N]})$ to $\mathcal{A}$. When $\mathcal{A}$ makes key generation query for set S^* such that $i^* \notin S^*$, $\mathcal{B}$ creates a pre-image similar to $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$ using $\mathbf{X}, \mathbf{H}_{S^*,\mathbf{i}^*}$.

If Chal's coin toss $b = 0$, then $\mathbf{c}_1^\top, c_3$ is $\mathbf{s}^\top \mathbf{A} + \mathbf{e}_1^\top, \mathbf{s}^\top \mathbf{d} + e_3$ for some $\mathbf{s}^\top \in \mathbb{Z}_q^n, \mathbf{e}_1^\top \in \mathcal{D}_{\mathbb{Z},\chi}^{1 \times m}, e_3 \in \mathcal{D}_{\mathbb{Z},\chi}$. Then $\mathcal{B}$ behaves like $\mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)$. Otherwise, if $\mathbf{c}_1^\top, c_3$ are independently and uniformly randomly sampled (coin toss $b = 1$), then $\mathcal{B}$ behaves like $\mathbf{Hyb}_3^{\mathcal{A}}(1^\lambda)$. Thus, $\mathcal{B}$ can break the LWE assumption with advantage ϵ. □

Lemma G.7. $\Pr[1 \leftarrow \mathbf{Hyb}_4^{\mathcal{A}}(1^\lambda)] = \Pr[1 \leftarrow \mathbf{Hyb}_5^{\mathcal{A}}(1^\lambda)]$

Proof. This follows as $\mathbf{B} = \mathbf{AX} - (\mathbf{i}^*)^\top \otimes \mathbf{G}$. □

Lemma G.8. There exists a negligible function 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 only changes are $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$, $\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times \lceil \log N \rceil m}$, and the way we generate pre-image $\mathbf{r}_S$. Thus, these hybrids can be argued to be statistically close similar to Lemma G.3 and Lemma G.4. □

H Overview of the Piecewise Guessing Strategy

In this work, we utilize the nested hybrids / partial selectivization / piecewise guessing framework utilized in prior works [HW09, FKPR14, FJP15, HJO+16, JW16, JKK+17, KW20] in the context of achieving adaptive security. A reader unfamiliar with this framework might not completely follow our proofs. In this section, we provide a self-contained yet vastly simplified view of the piecewise guessing framework adapted from [KW20] tailored specifically for our work. At times, we take some text almost verbatim from [KW20]. For a full detailed exposition of this framework, we refer the reader to Kowalczyk and Wee [KW20].

Let $\mathsf{G}_0, \mathsf{G}_1$ be two games that we want to show are computationally indistinguishable. Let $\mathsf{H}_{0,1}, \ldots, \mathsf{H}_{0,L}$ and $\mathsf{H}_{1,1} \ldots, \mathsf{H}_{1,L-1}$ be two sets of hybrids that we use to prove this indistinguishability based on guessing a bit b. That is, the reduction adversary guesses a bit b and uses $\mathsf{H}_{b,\ell}$ to simulate the interaction with $\mathcal{A}$ and hopes that certain adaptive choice of adversary $\mathcal{A}$ is in line with this bit b. Moreover, based on this guess we have two paths[11]

$$\begin{aligned} &\mathsf{G}_0 \equiv \mathsf{H}_{0,0} \approx_c \mathsf{H}_{0,1} \approx_c \ldots \approx_c \mathsf{H}_{0,L-1} \approx_c \mathsf{H}_{0,L} \equiv \mathsf{G}_1 \text{ and} \\ &\mathsf{G}_0 \equiv \mathsf{H}_{1,0} \approx_c \mathsf{H}_{1,1} \approx_c \ldots \approx_c \mathsf{H}_{1,L-1} \approx_c \mathsf{H}_{1,L} \equiv \mathsf{G}_1 \end{aligned} \tag{3}$$

Let $\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_0 \rangle]$ and $\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_1 \rangle]$ denote the probability that an admissible PPT adversary $\mathcal{A}$ playing the games $\mathsf{G}_0, \mathsf{G}_1$ with a challenger wins the game. Here, the probability is taken over the choice of random coins of $\mathcal{A}$ and the random coins of G_0 (and G_1 respectively). Assume that there is an adversary $\mathcal{A}$ such that $|\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_0 \rangle] - \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_1 \rangle]| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\cdot)$. Then, we will show that at least two of these hybrids are distinguishable.

Before proceeding further, let us take a step back and explain what these games and hybrids correspond to in the context of our work. It is easy to see that $\mathsf{G}_0, \mathsf{G}_1$ correspond to $\mathbf{Hyb}_{4,\iota}^{\mathcal{A}}(1^\lambda), \mathbf{Hyb}_{4,\iota+1}^{\mathcal{A}}(1^\lambda)$ for some $\iota \in [N]$ from the mode indistinguishability proof of our EqDBE scheme in standard model (Section 7, Appendix E). And, $\mathsf{H}_{b,\ell}$ for $b \in \{0,1\}, \ell \in [L]$ correspond to the sub-hybrids $(\iota \in S^*)$-$\mathbf{SubHyb}_\ell^{\mathcal{A}}(1^\lambda), (\iota \notin S^*)$-$\mathbf{SubHyb}_\ell^{\mathcal{A}}(1^\lambda)$ from Lemma E.5. An analogous correspondence can be established between proof of mode indistinguishability and payload hiding of our PPS scheme from LWE (Section 8, Appendix F, Lemma F.6). Although the number of these sub-hybrids are different, we can pad them by considering some dummy equivalent sub-hybrids. We will now proceed with our explanation of piecewise guessing framework.

We define hypothetical hybrids $\mathsf{H}^*(S^*)$ that guess the full set S^* purely as an analytical tool to decompose the probability space. Crucially, our actual security reduction does not guess the full

[11] Hybrids $\mathsf{H}_{0,0}$, $\mathsf{H}_{1,0}$ are same (so are $\mathsf{H}_{0,L}$, $\mathsf{H}_{1,L}$) and independent of guess b.

set S^*. It only guesses a single bit (corresponding to the transition between hybrids), resulting in a polynomial security loss rather than 2^N.

Selective Hybrids. In order to prove the above let us define the following selective games and hybrids.

Selector function h. Consider the deterministic function[12] $h : 2^{[N]} \to \{0,1\}$. We will use this to decide which strand we are choosing, i.e, $\mathsf{H}_{h(S^*),\ell}$-th route. For the sub-hybrids defined above, $h(S^*) := 1$ if and only if $(\iota \in S^*)$.

$\mathsf{G}_0^*(S^*), \mathsf{G}_1^*(S^*)$. These games are same as $\mathsf{G}_0, \mathsf{G}_1$ (respectively) except that we will additionally guess $S^* \subseteq [N]$ and abort if the $\mathcal{A}$'s challenge set in the challenge phase is different than S^*. Note that we can similarly define $\mathsf{H}_{0,0}^*(S^*), \mathsf{H}_{1,0}^*(S^*), \mathsf{H}_{0,L}^*(S^*)$, and $\mathsf{H}_{1,L}^*(S^*)$.

$\mathsf{H}_\ell^*(S^*)$ for $\ell \in \{1, \ldots, L-1\}$. These hybrids are same as $\mathsf{H}_{b,\ell}$ except that we will additionally guess $S^* \subseteq [N]$, and based on S^*, we will simulate $\mathsf{H}_{h(S^*),\ell}$ and abort if the $\mathcal{A}$'s challenge set in the challenge phase is different than S^*.

Relating Regular and Selective Hybrids. We will now relate $\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}' \rangle]$ for $\mathsf{H}' \in \{\mathsf{G}_0, \mathsf{G}_1, \mathsf{H}_{0,1}, \ldots, \mathsf{H}_{0,L-1}, \mathsf{H}_{1,1}, \ldots, \mathsf{H}_{1,L-1}\}$ (regular hybrids) with $\mathsf{H}' \in \{\mathsf{G}_0^*(S^*), \mathsf{G}_1^*(S^*), \mathsf{H}_{0,1}^*(S^*), \ldots, \mathsf{H}_{L-1}^*(S^*), \mathsf{H}_1^*(S^*)\}$ (selective hybrids).

As we guess S^* independently of all random coins of $\mathcal{A}, \mathsf{G}_0, \mathsf{G}_1$,

$$|\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_0^*(S^*) \rangle] - \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_1^*(S^*) \rangle]| = \frac{1}{2^N} |\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_0 \rangle] - \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_1 \rangle]|$$

Next, define $B_0 = \{S^* : S^* \subseteq [N], h(S^*) = 0\}$ $B_1 = \{S^* : S^* \subseteq [N], h(S^*) = 0\}$. Note that, $B_0 \cup B_1 = 2^{[N]}, B_0 \cap B_1 = \emptyset$. Now,

$$\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_{b,\ell} \rangle] = \sum_{S \in B_b} \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_\ell^*(S) \rangle] \tag{4}$$

as $\mathsf{H}_\ell^*(S^*)$ invokes $\mathsf{H}_{h(S^*),\ell}$ simply based on $h(S^*)$. So, for every choice of S such that $h(S) = b$, $\mathsf{H}_{b,\ell}$ is invoked. Moreover, by Equation (3), we have that $\mathsf{G}_0^*(S^*) \equiv \mathsf{H}_0^*(S^*) \approx_c \mathsf{H}_1^*(S^*) \approx_c \ldots \approx_c \mathsf{H}_{L-1}^*(S^*) \approx_c \mathsf{H}_L^*(S^*) \equiv \mathsf{G}_1^*(S^*)$ as behavior of all of these hybrids whether S^* is a valid guess or not is computationally close.

Completing the Proof. As we have $|\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_0 \rangle] - \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_1 \rangle]| = \epsilon(\lambda)$, we have that

$$|\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_0^*(S^*) \rangle] - \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_1^*(S^*) \rangle]| = \frac{\epsilon(\lambda)}{2^N}$$

where probability on the left hand side of the above equation is taken on the random coins of $\mathcal{A}$, $\mathsf{G}_0^*(S^*), \mathsf{G}_1^*(S^*)$, and the independent choice of S^*. Then, by a standard hybrid argument, there exists an $\ell \in [L]$ such that

$$|\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_{\ell-1}^*(S^*) \rangle] - \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_\ell^*(S^*) \rangle]| \geq \frac{\epsilon(\lambda)}{2^N \cdot L}$$

[12] 2^A denotes the power set of set A.

Then, by law of total probability, we can remove this dependence on S^* and can write

$$\left|\sum_{S\subseteq[N]} \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}^*_{\ell-1}(S)\rangle] - \sum_{S\subseteq[N]} \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}^*_{\ell}(S)\rangle]\right| \geq \frac{\epsilon(\lambda)}{L}$$

$$\implies \left|\sum_{b\in\{0,1\}} \left(\sum_{S\subseteq B_b} \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}^*_{\ell-1}(S)\rangle] - \sum_{S\in B_b} \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}^*_{\ell}(S)\rangle]\right)\right| \geq \frac{\epsilon(\lambda)}{L}$$

$$\implies \left|\sum_{b\in\{0,1\}} (\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_{b,\ell-1}\rangle] - \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_{b,\ell}\rangle])\right| \geq \frac{\epsilon(\lambda)}{L} \quad \text{(By Equation (4))}$$

By triangle inequality, we have

$$\sum_{b\in\{0,1\}} |\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_{b,\ell-1}\rangle] - \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_{b,\ell}\rangle]| \geq \frac{\epsilon(\lambda)}{L}$$

Now, we will use an averaging argument, which basically means that since sum of two quantities in the above inequality is at least $\epsilon(\lambda)/L$, there must be one item such that it is at least $\epsilon(\lambda)/2L$. If this was not the case, the sum will always be less than $\epsilon(\lambda)/L$. Thus, there exists $\beta \in \{0,1\}, \ell \in [L]$ such that

$$|\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_{\beta,\ell-1}\rangle] - \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_{\beta,\ell}\rangle]| \geq \frac{\epsilon(\lambda)}{2L}$$

In conclusion, if we have an admissible PPT adversary $\mathcal{A}$ such that $|\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_0\rangle] - \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{G}_1\rangle]| = \epsilon(\lambda)$ for some non-negligible function $\epsilon(\lambda)$, then there exists a $\beta \in \{0,1\}, \ell \in [L]$ such that $|\Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_{\beta,\ell-1}\rangle] - \Pr[1 \leftarrow \langle \mathcal{A}, \mathsf{H}_{\beta,\ell}\rangle]| \geq \epsilon(\lambda)/2L$. Thus, by a hybrid argument we can conclude that $\mathsf{G}_0 \approx_c \mathsf{G}_1$ using either of the paths.