New Techniques for Preimage Sampling: Improved NIZKs and More from LWE

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

Hoeteck Wee
NTT Research
wee@di.ens.fr

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

Abstract

Recent constructions of vector commitments and non-interactive zero-knowledge (NIZK) proofs from LWE implicitly solve the following *shifted multi-preimage sampling problem*: given matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n \times m}$ and targets $\mathbf{t}_1, \ldots, \mathbf{t}_\ell \in \mathbb{Z}_q^n$, sample a shift $\mathbf{c} \in \mathbb{Z}_q^n$ and short preimages $\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell \in \mathbb{Z}_q^m$ such that $\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{t}_i + \mathbf{c}$ for all $i \in [\ell]$. In this work, we introduce a new technique for sampling $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ together with a succinct public trapdoor for solving the multi-preimage sampling problem with respect to $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$. This enables the following applications:

- We provide a dual-mode instantiation of the hidden-bits model (and by correspondence, a dual-mode NIZK proof for NP) with (1) a linear-size common reference string (CRS); (2) a transparent setup in hiding mode (which yields statistical NIZK arguments); and (3) hardness from LWE with a polynomial modulus-to-noise ratio. This improves upon the work of Waters (STOC 2024) which required a quadratic-size structured reference string (in *both* modes) and LWE with a super-polynomial modulus-to-noise ratio.
- We give a statistically-hiding vector commitment with transparent setup and polylogarithmic-size CRS, commitments, and openings from SIS. This simultaneously improves upon the vector commitment schemes of de Castro and Peikert (EUROCRYPT 2023) as well as Wee and Wu (EUROCRYPT 2023).

At a conceptual level, our work provides a unified view of recent lattice-based vector commitments and hidden-bits model NIZKs through the lens of the shifted multi-preimage sampling problem.

1 Introduction

Starting from the seminal works of Ajtai [Ajt96] and of Gentry, Peikert, and Vaikuntanathan [GPV08], lattice trapdoors have played a critical role in building advanced cryptographic primitives from lattices. These include notions like hash-and-sign signatures [GPV08], identity-based and attribute-based encryption [GPV08, ABB10b, ABB10a, CHKP10, GVW13, BGG+14, GVW15a, BTVW17], homomorphic signatures [GVW15b], functional commitments [dCP23, WW23b, BCFL23, WW23a], succinct non-interactive arguments [ACL+22, CLM23], and non-interactive zero-knowledge (NIZK) proofs [Wat24].[1]

Lattice trapdoors. In this work, we focus on gadget trapdoors [MP12]. In this setting, a trapdoor for a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ is a short matrix $\mathbf{T}$ where $\mathbf{AT} = \mathbf{G}$ and $\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g}^{\mathsf{T}} \in \mathbb{Z}_q^{n \times m}$ is the gadget matrix, $\mathbf{I}_n$ denotes the n-by-n identity matrix, and $\mathbf{g}^{\mathsf{T}} = [2^0, 2^1, \ldots, 2^{\lceil \log q \rceil - 1}]$. Given a trapdoor for a matrix $\mathbf{A}$ along with a target vector $\mathbf{c} \in \mathbb{Z}_q^n$, we can efficiently compute a short preimage $\boldsymbol{\pi} \in \mathbb{Z}_q^m$ satisfying $\mathbf{A} \cdot \boldsymbol{\pi} = \mathbf{c}$. In fact, we can even sample *random* short discrete Gaussian preimages, whose distribution we denote by $\mathbf{A}^{-1}(\mathbf{c})$.

Shifted multi-preimage sampling. Recent constructions of lattice-based vector commitments [PPS21, WW23b] and non-interactive zero-knowledge (NIZK) proofs [Wat24] implicitly considered variants of a "shifted multi-preimage sampling problem" which is parameterized by a collection of matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n \times m}$:

[1] Earlier constructions of lattice-based NIZKs [CCH+19, PS19] did not require lattice trapdoors.

Shifted multi-preimage sampling: Given targets $\mathbf{t}_1, \ldots, \mathbf{t}_\ell \in \mathbb{Z}_q^n$, sample a random vector $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ along with short discrete Gaussian preimages $\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell \in \mathbb{Z}_q^m$ satisfying $\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{t}_i + \mathbf{c}$ for all $i \in [\ell]$.

Solving this problem for arbitrary matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ requires knowing some hint (e.g. trapdoors) related to these matrices. The aforementioned applications also require "somewhere hardness:" namely, that the short integer solutions (SIS) or the learning with errors (LWE) problems is hard with respect to any individual $\mathbf{A}_i$ even given the hint. This rules out the trivial solution of taking the hint to be trapdoors for each matrix $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$. As a warm-up, observe that for $\ell = 1$ (and $\mathbf{A}_1$ being uniformly random), this problem is straightforward. One can sample a short Gaussian $\boldsymbol{\pi}_1 \in \mathbb{Z}_q^m$ and set $\mathbf{c} = \mathbf{A}_1 \boldsymbol{\pi}_1 - \mathbf{t}_1$. The work of [GPV08] shows that when $m \geq O(n \log q)$, the distribution of $\mathbf{c}$ is statistically close to uniform over $\mathbb{Z}_q^n$.

For the more general version with $\ell > 1$ targets, prior works [PPS21, WW23b, Wat24] required a hint of size $O(\ell^2)$, even for special cases of the problem where the target vectors $\mathbf{t}_1, \ldots, \mathbf{t}_\ell$ are the all-zeroes vector $\mathbf{0}^n$. Among them, only the work of Wee and Wu [WW23b] solved the problem in full generality for arbitrary target vectors. They showed how to sample a random $\mathbf{c}$ together with *random* short discrete Gaussian preimages $\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell$ satisfying $\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{t}_i + \mathbf{c}$. In their construction, the hint corresponds to a gadget trapdoor for the matrix

$$\mathbf{D}_\ell := \left[\begin{array}{ccc|c} \mathbf{A}_1 & & & \mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & \mathbf{G} \end{array}\right] = [\mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid \mathbf{1}^\ell \otimes \mathbf{G}]. \tag{1.1}$$

The work of [WW23b] uses the gadget trapdoor to sample a random Gaussian preimage of $\mathbf{D}_\ell$ for the target vector $(\mathbf{t}_1, \ldots, \mathbf{t}_\ell) \in \mathbb{Z}_q^{n\ell}$; namely, a vector $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \hat{\mathbf{c}})$ where

$$\mathbf{D}_\ell \begin{bmatrix} \boldsymbol{\pi}_1 \\ \vdots \\ \boldsymbol{\pi}_\ell \\ \hat{\mathbf{c}} \end{bmatrix} = \left[\begin{array}{ccc|c} \mathbf{A}_1 & & & \mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & \mathbf{G} \end{array}\right] \begin{bmatrix} \boldsymbol{\pi}_1 \\ \vdots \\ \boldsymbol{\pi}_\ell \\ \hat{\mathbf{c}} \end{bmatrix} = \begin{bmatrix} \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_\ell \end{bmatrix}.$$

They then set $\mathbf{c} = -\mathbf{G}\hat{\mathbf{c}}$. In this case, for all $i \in [\ell]$, $\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{t}_i - \mathbf{G}\hat{\mathbf{c}} = \mathbf{t}_i + \mathbf{c}$, as required. Moreover, the ensuing distribution of $(\mathbf{c}, \boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell)$ is statistically close to that given by first sampling $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and then $\boldsymbol{\pi}_i \leftarrow \mathbf{A}_i^{-1}(\mathbf{t}_i + \mathbf{c})$.

Drawbacks of prior works. There are two major drawbacks of needing to include hints for $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ as part of the public parameters of the scheme:

- **Trusted setup.** First, sampling the hint typically requires *private* randomness. Existing constructions use private randomness to sample $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ along with their respective trapdoors. In the aforementioned applications (to vector commitments [PPS21, WW23b] and NIZKs [Wat24]), an adversary who knows the private randomness is able to break security of the associated scheme. Thus, the aforementioned constructions all rely on a *trusted* setup to sample the CRS.

- **Hint size.** The aforementioned approaches require a hint whose size is *quadratic* in the dimension ℓ. Here ℓ is the input dimension (in the case of vector commitments) or the length of the hidden-bits string (in the case of using a hidden-bits generator [FLS90, QRW19] to construct a NIZK). Thus, the existing schemes have large public parameters.

This work: *eliminating* the hint. In this work, we show how to construct a *shifted multi-preimage trapdoor sampler* that allows us to solve the shifted multi-preimage sampling problem with respect to a carefully-chosen set of matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ *without* hints. In our construction, the matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ will be correlated, but the marginal distribution of each individual $\mathbf{A}_i$ remains uniformly random, albeit with slightly larger dimensions $n \times (m \cdot \lceil \log \ell \rceil)$. Moreover, both SIS and LWE are hard with respect to any individual $\mathbf{A}_i$. In fact, $\mathbf{A}_i$ is simply $\mathbf{B} - \mathbf{u}_i \otimes \mathbf{G}$, where $\mathbf{B} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m \cdot \lceil \log \ell \rceil}$ and $\mathbf{u}_i \in \{0,1\}^{\lceil \log \ell \rceil}$ is the binary representation of i. Given only the matrix $\mathbf{B}$, we show that we can *publicly* derive a gadget trapdoor for the matrix $\mathbf{D}_\ell$ in Eq. (1.1). Each entry in the public trapdoor of $\mathbf{D}_\ell$ lies in

$\{-1, 0, 1\}$. Here, we rely on the machinery from [GSW13, BGG$^+$14, DHM$^+$24] for homomorphic computation on matrix encodings. Having a publicly-computable gadget trapdoor means that in our applications, the public parameters only needs to specify the single (uniformly) random matrix $\mathbf{B}$; we do not need to include the matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ or its trapdoor for $\mathbf{D}_\ell$ (say, as needed in [WW23b]). This immediately yields improvements to existing constructions of lattice-based vector commitments and dual-mode NIZKs which we discuss in more detail below.

Vector commitments. Recall that a vector commitment allows a user to succinctly commit to a vector $\mathbf{x}$ and also succinctly open to individual components x_i of the committed vector. The security properties are binding and hiding. Binding says that an adversary should not be able to open a commitment σ to two distinct values $x_i \neq x_i'$ at any index i. Hiding says that the openings for any set of indices should not reveal anything about the values at unopened indices.

By integrating our shifted multi-preimage trapdoor sampler with the framework of [WW23b], we obtain a statistically-hiding and computationally-binding vector commitment scheme (Corollary 6.15) from the SIS assumption with a *transparent* (i.e., public-coin) setup. For committing to ℓ-dimensional inputs over $\mathbb{Z}_q^\ell$, the size of the (uniformly-random) common reference string (CRS), the commitment, and the openings are all $\text{poly}(\lambda, \log \ell)$. Our construction simultaneously improves upon and inherits the properties of prior vector commitments from the SIS assumption [PSTY13, LLNW16, PPS21, dCP23, WW23b]. Here, we focus on comparing with the most recent schemes [dCP23, WW23b]:

(1) Like [dCP23, WW23b], our scheme is linearly homomorphic and supports stateless updates (i.e., given a commitment σ to a vector $\mathbf{x}$ it is possible to transform it into a commitment to $\mathbf{x}'$ using only knowledge of σ and the difference $\mathbf{x}' - \mathbf{x}$);

(2) We achieve a *transparent* and polylogarithmic-size CRS, matching [dCP23] and improving upon the quadratic-size CRS in [WW23b];

(3) We achieve statistical hiding and can directly commit to vectors over $\mathbb{Z}_q^\ell$ (while preserving linear homomorphism), as achieved in [WW23b] but not in [dCP23].

We refer to Section 2.3 for further discussion and comparison of our approach with prior work.

Dual-mode NIZKs. Our second application is to (dual-mode) hidden-bits generators [FLS90, QRW19, LPWW20], which imply dual-mode NIZKs. In a dual-mode NIZK [GOS06, GOS12], the CRS can be sampled from one of two computationally-indistinguishable distributions: one distribution yields computational NIZK proofs while the other yields statistical NIZK arguments. Previously, Peikert and Shiehian [PS19] showed how to construct dual-mode NIZKs for NP from LWE by constructing a correlation-intractable hash function [CGH04, KRR17, HL18, CCRR18, CCH$^+$19]. For many years, the correlation-intractability approach was the only way of realizing NIZKs for NP from lattices.

Very recently, Waters [Wat24] showed a new path for constructing NIZKs from lattices by constructing a hidden-bits generator from the LWE assumption. A hidden-bits generator [FLS90, QRW19] is a cryptographic primitive that generates a succinct commitment to a pseudorandom sequence of hidden bits (and relative to a common reference string). Unlike the case of vector commitments, we require the commitment to *statistically* bind to the sequence of hidden bits (relative to the *long* CRS). Since the commitments are succinct and the hidden-bits generator is statistically binding, the number of possible hidden-bit strings that can be associated with a commitment is small.

In this work, we present a new hidden-bits generator by combining our shifted multi-preimage trapdoor sampler with ideas and techniques from [WW23b] and [Wat24]. Our hidden-bits generator improves upon the [Wat24] hidden-bits generator in three key aspects (Corollary 5.18):

(1) We achieve a shorter CRS whose size scales *linearly* with the number of hidden bits ℓ. This improves upon the quadratic dependency in [Wat24].

(2) In hiding mode, our scheme has a transparent setup (i.e., a *uniform* random CRS). This yields statistical NIZK arguments with a transparent setup. The [Wat24] construction required a structured CRS in both modes.

(3) Security relies on LWE with a *polynomial* modulus-to-noise ratio, improving upon the *super-polynomial* modulus-to-noise ratio in [Wat24].

Waters [Wat24] implicitly constructed a hidden-bits generator starting from a shifted multi-preimage trapdoor sampler for the special case where the ℓ target vectors are set to the all-zeroes vector (the vector $\mathbf{c}$ corresponds to the succinct commitment and the preimage $\boldsymbol{\pi}_i$ corresponds to an opening for the i^{th} bit). The sampler in [Wat24] essentially outputs random linear combinations of quantities in the CRS.

Substituting our shifted multi-preimage trapdoor sampler into the [Wat24] approach yields our improved hidden-bits generator. In particular, Properties (1) and (2) follow immediately from the improved parameters for our shifted multi-preimage trapdoor sampler. The binding analysis for our hidden-bits generator is the same as that in [Wat24], whereas the hiding analysis follows the proof of statistical hiding for the vector commitment from [WW23b]. The latter eliminates the use of noise flooding used in [Wat24], which allows us to achieve Property (3). We refer to Section 5 for a more detailed technical comparison of our approach with that of [Wat24].

Taken together, we obtain a dual-mode NIZK for NP from LWE via the hidden-bits model approach that achieves all of the properties of the Peikert-Shiehian construction [PS19] based on correlation-intractable hash functions. Our scheme has the additional appealing feature that it does not need to make non-black-box use of cryptographic primitives or lattice-sampling algorithms. But more broadly, our results show that the hidden-bits model approach is just as versatile for realizing NIZKs for NP from the LWE assumption as the correlation-intractability framework.

A new abstraction. At a conceptual level, our work provides a unified and more modular view of recent lattice-based vector commitments [PPS21, WW23b] and dual-mode NIZKs [Wat24] through the lens of the shifted multi-preimage sampling problem. By focusing on and giving an improved construction of this key cryptographic object (the shifted multi-preimage trapdoor sampler), we immediately obtain improvements to both vector commitments and dual-mode NIZKs. We believe that the notion of shifted multi-preimage sampling, as well as our new techniques for solving this preimage sampling problem, will find additional applications to other lattice-based primitives in the future.

2 Technical Overview

The main technical building block in this work is a shifted multi-preimage trapdoor sampler for solving the shifted multi-preimage sampling problem. Namely, starting from a seed (i.e., a random matrix) $\mathbf{B} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times t}$ where $t = \lceil \log \ell \rceil \cdot m$, we construct a structured matrix of the form in Eq. (1.1):

$$\mathbf{D}_\ell := \left[\begin{array}{ccc|c} \mathbf{A}_1 & & & \mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & \mathbf{G} \end{array}\right] = [\operatorname{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid \mathbf{1}^\ell \otimes \mathbf{G}].$$

The description of $\mathbf{D}_\ell$ is extremely simple: let $\mathbf{u}_i \in \{0,1\}^{\lceil \log \ell \rceil}$ be the binary representation of i, and set

$$\mathbf{A}_i := \mathbf{B} - \mathbf{u}_i^\intercal \otimes \mathbf{G} \in \mathbb{Z}_q^{n \times t} \tag{2.1}$$

Moreover, $\mathbf{D}_\ell$ has the following remarkable property: given only the seed $\mathbf{B}$, we can derive a gadget trapdoor for $\mathbf{D}_\ell$ where the entries of the trapdoor are in the set $\{-1, 0, 1\}$. Before we describe how to construct the trapdoor, we describe three properties of $\mathbf{D}_\ell$ that are useful for our applications:

- **Shifted multi-preimage sampling:** Given $\mathbf{D}_\ell$, its trapdoor, and target vectors $\mathbf{t}_1, \ldots, \mathbf{t}_\ell \in \mathbb{Z}_q^n$, we can sample a uniform randomly $\mathbf{c} \xleftarrow{\text{R}} \mathbb{Z}_q^n$ together with short vectors $\boldsymbol{\pi}_i \in \mathbb{Z}_q^m$ such that $\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{t}_i + \mathbf{c} \in \mathbb{Z}_q^n$ for all $i \in [\ell]$.
- **Hardness:** For all $i \in [\ell]$, both SIS and LWE are hard with respect to any $\mathbf{A}_i$ even given $\mathbf{D}_\ell$ and its trapdoor.
- **Preimage distribution:** We require that the joint distributions of $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$ sampled using $\mathbf{D}_\ell$ and its trapdoor to be statistically close to sampling $\mathbf{c} \xleftarrow{\text{R}} \mathbb{Z}_q^n$ and $\boldsymbol{\pi}_i \leftarrow \mathbf{A}_i^{-1}(\mathbf{t}_i + \mathbf{c})$. In other words, the distribution of each $\boldsymbol{\pi}_i$ should be statistically close to a random discrete Gaussian $\boldsymbol{\pi}_i$ conditioned on $\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{t}_i + \mathbf{c}$.

In the applications to vector commitments (resp., hidden-bits generators), the first property will be used to sample the commitment and the openings, the second will be used to argue binding (resp., mode indistinguishability), and the third will be use to argue hiding.

A gadget trapdoor for $\mathbf{D}_\ell$. To derive a gadget trapdoor $\mathbf{T}$ for $\mathbf{D}_\ell$, we rely on the machinery from [GSW13, BGG⁺14, DHM⁺24] for homomorphic computation on matrix encodings. We consider the specific application to the family of indicator functions $\{\delta_{\mathbf{u}} \colon \{0,1\}^k \to \{0,1\}\}_{\mathbf{u} \in \{0,1\}^k}$ given by

$$\delta_{\mathbf{u}}(\mathbf{x}) := \begin{cases} 1 & \mathbf{x} = \mathbf{u} \\ 0 & \mathbf{x} \neq \mathbf{u}. \end{cases}$$

Next, let $\mathbf{B} \in \mathbb{Z}_q^{n \times km}$ be a matrix. Then, there is an efficient (and explicit) algorithm that takes as input $\mathbf{u}, \mathbf{x} \in \{0,1\}^k$ and outputs a short matrix $\mathbf{H}_{\mathbf{B},\mathbf{u},\mathbf{x}}$ (with entries in $\{-1,0,1\}$) where

$$(\mathbf{B} - \mathbf{x}^{\mathsf{T}} \otimes \mathbf{G}) \cdot \mathbf{H}_{\mathbf{B},\mathbf{u},\mathbf{x}} = \begin{cases} \mathbf{B}_{\mathbf{u}} - \mathbf{G} & \text{if } \mathbf{u} = \mathbf{x} \\ \mathbf{B}_{\mathbf{u}} & \text{if } \mathbf{u} \neq \mathbf{x}, \end{cases} \tag{2.2}$$

and moreover, we can efficiently compute $\mathbf{B}_{\mathbf{u}} \in \mathbb{Z}_q^{n \times m}$ given just $\mathbf{B}$ and $\mathbf{u}$.[2] Setting $k = \lceil \log \ell \rceil$, we have:

$$\underbrace{\left[\begin{array}{ccc|c} \mathbf{B} - \mathbf{u}_1^{\mathsf{T}} \otimes \mathbf{G} & & & \mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{B} - \mathbf{u}_\ell^{\mathsf{T}} \otimes \mathbf{G} & \mathbf{G} \end{array}\right]}_{\mathbf{D}_\ell} \cdot \underbrace{\begin{bmatrix} -\mathbf{H}_{\mathbf{B},\mathbf{u}_1,\mathbf{u}_1} & \cdots & -\mathbf{H}_{\mathbf{B},\mathbf{u}_\ell,\mathbf{u}_1} \\ \vdots & \ddots & \vdots \\ -\mathbf{H}_{\mathbf{B},\mathbf{u}_1,\mathbf{u}_\ell} & \cdots & -\mathbf{H}_{\mathbf{B},\mathbf{u}_\ell,\mathbf{u}_\ell} \\ \mathbf{G}^{-1}(\mathbf{B}_{\mathbf{u}_1}) & \cdots & \mathbf{G}^{-1}(\mathbf{B}_{\mathbf{u}_\ell}) \end{bmatrix}}_{\mathbf{T}} = \begin{bmatrix} \mathbf{G} & & \\ & \ddots & \\ & & \mathbf{G} \end{bmatrix}. \tag{2.3}$$

Shifted multi-preimage sampling using $\mathbf{T}$. Given the trapdoor $\mathbf{T}$ for $\mathbf{D}_\ell = [\operatorname{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid \mathbf{1}^\ell \otimes \mathbf{G}]$, we can use the [WW23b] approach to solve the shifted multi-preimage sampling problem with respect to the matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$. Specifically, given target vectors $\mathbf{t}_1, \ldots, \mathbf{t}_\ell \in \mathbb{Z}_q^n$, we use $\mathbf{T}$ to sample a Gaussian preimage $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \hat{\mathbf{c}})$ to the linear system

$$\underbrace{\left[\begin{array}{ccc|c} \mathbf{A}_1 & & & \mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & \mathbf{G} \end{array}\right]}_{\mathbf{D}_\ell} \cdot \begin{bmatrix} \boldsymbol{\pi}_1 \\ \vdots \\ \boldsymbol{\pi}_\ell \\ \hat{\mathbf{c}} \end{bmatrix} = \begin{bmatrix} \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_\ell \end{bmatrix}. \tag{2.4}$$

By construction, for all $i \in [\ell]$, we have $\mathbf{A}_i \boldsymbol{\pi}_i + \mathbf{G}\hat{\mathbf{c}} = \mathbf{t}_i$, or equivalently, $\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{t}_i - \mathbf{G}\hat{\mathbf{c}}$. Defining $\mathbf{c} = -\mathbf{G}\hat{\mathbf{c}}$, we obtain a solution $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$ to the shifted multi-preimage sampling problem.

Hardness. The second property we require is that the SIS and LWE problems are hard with respect to any $\mathbf{A}_i$ given $\mathbf{D}_\ell = [\operatorname{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid \mathbf{1}^\ell \otimes \mathbf{G}]$ together with the trapdoor for $\mathbf{D}_\ell$. It suffices to show that given any index $i \in [\ell]$ and any matrix $\mathbf{A}^* \in \mathbb{Z}_q^{n \times t}$, we can simulate a seed $\mathbf{B} \in \mathbb{Z}_q^{n \times t}$ the expands into matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ where $\mathbf{A}_i = \mathbf{A}^*$. We refer to this as a "somewhere programmability" property on our shifted multi-preimage trapdoor sampler. Recall from Eq. (2.1) that $\mathbf{A}_i = \mathbf{B} - \mathbf{u}_i^{\mathsf{T}} \otimes \mathbf{G}$. To simulate a seed $\mathbf{B}$ that expands to $\mathbf{A}^*$ in position i, we can set

$$\mathbf{B} := \mathbf{A}^* + \mathbf{u}_i^{\mathsf{T}} \otimes \mathbf{G}.$$

Under this definition, $\mathbf{A}_i = \mathbf{B} - \mathbf{u}_i^{\mathsf{T}} \otimes \mathbf{G} = \mathbf{A}^*$, as required. Moreover, when $\mathbf{A}^*$ is uniformly random (i.e., $\mathbf{A}^*$ is an SIS or LWE challenge), then the simulated seed is distributed identically to the real seed. This shows that given an SIS or LWE challenge matrix $\mathbf{A}^* \in \mathbb{Z}_q^{n \times t}$, we can simulate an identically-distributed seed $\mathbf{B}$ that expands to $\mathbf{A}^*$ in position i. This suffices to demonstrate hardness of the SIS or LWE problems with respect to any of the matrices $\mathbf{A}_i$ associated with a seed.

[2]$\mathbf{B}_{\mathbf{u}}$ is the homomorphic evaluation of $\delta_{\mathbf{u}}$ on $\mathbf{B}$.

Preimage distribution. The preimage distribution property requires that the *joint* distribution of $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$ sampled using $\mathbf{D}_\ell, \mathbf{T}$ to be statistically close to sampling $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and $\boldsymbol{\pi}_i \leftarrow \mathbf{A}_i^{-1}(\mathbf{t}_i + \mathbf{c})$. This basic requirement follows directly by properties of the discrete Gaussian distribution (see Lemma 3.6). For the application to statistically-hiding vector commitments, we require a stronger simulatability property that stipulates that there is an efficient algorithm that samples a seed $\mathbf{B}$ together with trapdoors for the *individual* matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$. These trapdoors are used to efficiently *simulate* the distribution $\boldsymbol{\pi}_i \leftarrow \mathbf{A}_i^{-1}(\mathbf{c})$. To satisfy this stronger hiding requirement, we append to the seed a random $\mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m}$ and derive $\mathbf{D}_\ell$ by setting

$$\mathbf{A}_i \coloneqq [\mathbf{A} \mid \mathbf{B} - \mathbf{u}_i \otimes \mathbf{G}]$$

This way, we can derive gadget trapdoors for $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ starting from a gadget trapdoor for $\mathbf{A}$. It is straightforward to check that we can still derive a gadget trapdoor for the modified $\mathbf{D}_\ell$ given just $\mathbf{A}, \mathbf{B}$, and moreover, that the required hardness properties continue to hold. We provide the formal definition and construction details of our shifted multi-preimage trapdoor sampler in Section 4.

2.1 Application to Vector Commitments

Our shifted multi-preimage trapdoor sampler can be directly applied to the Wee-Wu vector commitment [WW23b] to obtain a statistically-hiding (and computationally-binding) scheme with a short transparent setup. We first recall their construction (rephrased in the language of shifted multi-preimage sampling). In the following description, let ℓ be the vector dimension.

- The common reference string contains matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n \times t}$ together with a hint for solving the shifted multi-preimage sampling problem with respect to $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$.
- The commitment to an input $\mathbf{x} \in \mathbb{Z}_q^\ell$ is a vector $\mathbf{c} \in \mathbb{Z}_q^n$ and an opening to value $x_i \in \mathbb{Z}_q$ at index $i \in [\ell]$ is a short preimage $\boldsymbol{\pi}_i$ where $\mathbf{A}_i \boldsymbol{\pi}_i = x_i \mathbf{e}_1 + \mathbf{c}$ and $\mathbf{e}_1 = [1, 0, \ldots, 0]^\mathsf{T}$ is the first canonical basis vector.

In other words, a commitment $\mathbf{c}$ and the openings $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell)$ form a solution to the shifted multi-preimage sampling problem with respect to matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ and target vectors $x_1\mathbf{e}_1, \ldots, x_\ell \mathbf{e}_1$. The work of [WW23b] solve the shifted multi-preimage sampling problem by publishing random matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ in the CRS together with a gadget trapdoor for the matrix $\mathbf{D}_\ell = [\mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid \mathbf{1}^\ell \otimes \mathbf{G}]$. Thus, their construction requires a structured CRS whose size scales quadratically with the input dimension ℓ.

Vector commitments with short transparent setup. To obtain a vector commitment scheme with a succinct transparent setup, we replace the structured matrix $\mathbf{D}_\ell$ and its trapdoor $\mathbf{T}$ in the [WW23b] CRS with the seed $[\mathbf{A} \mid \mathbf{B}]$ of our shifted multi-preimage trapdoor sampler. This is sufficient for correctness. Security then follows by the properties of the shifted multi-preimage trapdoor sampler:

- **Binding:** The [WW23b] scheme is computationally binding if SIS is hard with respect to $\mathbf{A}_i$ (even given $\mathbf{D}_\ell$ and $\mathbf{T}$).[3] This is the same hardness property satisfied by our shifted multi-preimage trapdoor sampler.
- **Hiding:** The [WW23b] scheme is statistically hiding if there is an alternative (and statistically indistinguishable) way to sample $\mathbf{D}_\ell, \mathbf{T}$ together with knowledge of a gadget trapdoor for each $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$. This is the simulatability property of our shifted multi-preimage trapdoor sampler.

Taken together, we obtain a statistically-hiding vector commitment scheme with a transparent setup. Moreover, the size of the common random string is polylogarithmic in the vector dimension (in contrast to [WW23b] which required a structured quadratic-size CRS). We compare with other vector commitment schemes in Section 2.3 and give the formal description in Section 6.

[3]Technically, binding holds if the SIS assumption holds with respect to $\mathbf{A}_i$ *without* the first row, but we elide this detail in this overview.

2.2 Application to Dual-Mode NIZKs for NP

The second application of our shifted multi-preimage trapdoor sampler is to dual-mode hidden-bits generators, which in turn, implies a dual-mode NIZK for NP. We start with a more modular view of the dual-mode hidden-bits generator by Waters [Wat24] in the language of the shifted multi-preimage sampling and then show how our shifted multi-preimage trapdoor sampler can simultaneously reduce the CRS size (from quadratic to linear), achieve transparent setup in hiding mode, and only rely on LWE with a polynomial modulus-to-noise ratio. Thus, we simultaneously improve on functionality and security.

Hidden-bits generators. At a high level, a hidden-bits generator [QRW19] is a cryptographic primitive that generates a pseudorandom sequence of hidden bits from a short seed. Moreover, the user can provide local openings to any subset of the bits (with respect to a commitment of the seed). The unopened bits should remain pseudorandom. While both vector commitments and hidden-bits generators have the flavor of committing to a long input with a short commitment, there is a key distinction between the two notions:

- In a vector commitment, the user can choose *any* string of bits $\mathbf{r} \in \{0,1\}^\ell$ and derive a succinct commitment to $\mathbf{r}$. The scheme is *computationally* binding (i.e., it is hard for an adversary to open a commitment to two distinct values at any single index).
- In a hidden-bits generator, the user samples a succinct commitment σ. The commitment together with a *long* CRS determines an associated hidden-bits string $\mathbf{r} \in \{0,1\}^\ell$. In this setting, the binding property is *statistical*; each commitment can only be opened to at most one bit-string $\mathbf{r}$ with respect to the CRS. Since the commitment is succinct, the number of potential bit-strings $\mathbf{r} \in \{0,1\}^\ell$ that has an associated commitment σ is small.

More formally, a hidden-bits generator consists of three main algorithms:

- The $\mathsf{Setup}(1^\lambda, 1^\ell)$ algorithm takes the security parameter λ and the output length ℓ and outputs a common reference string crs.
- The $\mathsf{GenBits}(\mathsf{crs})$ algorithm takes the common reference string and outputs a *short* commitment σ, a bit-string $\mathbf{r} \in \{0,1\}^\ell$, and openings $\pi_1, \ldots, \pi_\ell$.
- The $\mathsf{Verify}(\mathsf{crs}, \sigma, i, r_i, \pi_i)$ algorithm takes the common reference string crs, the commitment σ, an index $i \in [\ell]$, a bit $r_i \in \{0,1\}$, and a proof π_i and decides whether to accept or reject.

The security requirements are (1) binding which says that the adversary should not be able to open a commitment σ to both a 0 and a 1 at any index i; and (2) hiding, which says that given the commitment σ, the bits r_i together with their openings π_i for all $i \in S$, the unopened bits r_i for $i \notin S$ should be pseudorandom. In conjunction with information-theoretic NIZKs for NP in the hidden-bits model [FLS90], a hidden-bits generator gives a NIZK for NP [QRW19].

In a dual-mode hidden-bits generator [LPWW20], we impose an additional property where the crs output by Setup is in one of two computationally indistinguishable modes. In binding mode, the hidden-bits generator should be *statistically* binding while in hiding mode, the hidden-bits generator should be *statistically* hiding. Dual-mode hidden-bits generators imply dual-mode NIZKs for NP (i.e., NIZKs where the CRS can be sampled in one of two computationally-indistinguishable modes, with one mode yielding computational NIZK proofs and the other yielding statistical NIZK arguments). Recently, Waters [Wat24] constructed the first dual-mode hidden-bits generator from LWE.

The [Wat24] hidden-bits generator. We now describe the recent dual-mode hidden-bits generator by Waters [Wat24] in the language of the shifted multi-preimage sampling problem. In the following description, let ℓ be the length of the hidden-bits string.

- The common reference string consists of matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n \times t}$.
- To generate a hidden-bits string, the prover invokes the shifted multi-preimage sampling procedure for $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ to obtain a random $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ together with short preimages $\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell \in \mathbb{Z}_q^t$ satisfying $\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{c}$ for all $i \in [\ell]$. In this setting, all of the target vectors $\mathbf{t}_1, \ldots, \mathbf{t}_\ell$ in the shifted multi-preimage sampling problem are the all-zeroes vector $\mathbf{0}^n$. The commitment is $\mathbf{c}$ and an opening at index i is $\boldsymbol{\pi}_i$.

- The CRS also contains vectors $\mathbf{v}_1, \ldots, \mathbf{v}_\ell \in \mathbb{Z}_q^t$ which in conjunction with the openings, determine the hidden-bits string: the i^{th} bit b_i is given by $\lfloor \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_i \rceil$, where we write $\lfloor \cdot \rceil$ to denote the rounding operation (i.e., $\lfloor x \rceil$ outputs 0 if $|x| < q/4$ and 1 if $|x - q/2| < q/4$).

The distribution of $\mathbf{v}_i$ determines whether the bit b_i is binding or hiding:

- When $\mathbf{v}_i^{\mathsf{T}} = \mathbf{s}_i^{\mathsf{T}} \mathbf{A}_i + \mathbf{e}_i^{\mathsf{T}}$ is an LWE sample, then the bit b_i is completely determined by the CRS component $\mathbf{v}_i$ and the commitment $\mathbf{c}$. To see this, consider *any* valid opening $(\boldsymbol{\pi}_i, b_i)$ for any commitment $\mathbf{c} \in \mathbb{Z}_q^n$ on index i. Then, it must be the case that $\boldsymbol{\pi}_i$ is short and $\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{c}$. Moreover,

  $$b_i = \lfloor \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_i \rceil = \lfloor \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_i - \mathbf{e}_i^{\mathsf{T}} \boldsymbol{\pi}_i \rceil = \lfloor (\mathbf{s}_i^{\mathsf{T}} \mathbf{A}_i + \mathbf{e}_i^{\mathsf{T}}) \boldsymbol{\pi}_i - \mathbf{e}_i^{\mathsf{T}} \boldsymbol{\pi}_i \rceil = \lfloor \mathbf{s}_i^{\mathsf{T}} \mathbf{c} \rceil,$$

  where the second equality holds as long as $\mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_i$ is far from a rounding boundary (enforced by the verification relation) and the fact that if $\boldsymbol{\pi}_i$ is small, then so is $\mathbf{e}_i^{\mathsf{T}} \boldsymbol{\pi}_i$. Thus, in this case, the commitment $\mathbf{c}$ completely determines the associated bit b_i. Importantly, this analysis holds for *any* (even adversarial) choice of the commitment $\mathbf{c}$.

- When $\mathbf{v}_i \xleftarrow{\text{R}} \mathbb{Z}_q^t$ is a uniform random vector, then $\mathbf{v}_i$ and $\mathbf{c}$ leaks no information about b_i; in particular, the bit b_i is statistically close to uniform given the commitment $\mathbf{c}$ and the openings $\boldsymbol{\pi}_j$ for all $j \neq i$. In our work, we argue this by relying on the hiding property of our shifted multi-preimage trapdoor sampler. Namely, when $\boldsymbol{\pi}_i \leftarrow \mathbf{A}_i^{-1}(\mathbf{c})$ and $\mathbf{v}_i$ is uniform, the distribution of $\mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_i \in \mathbb{Z}_q$ is statistically close to uniform over $\mathbb{Z}_q$; this holds by appealing to the leftover hash lemma and the fact that the discrete Gaussian distribution has high min-entropy. Moreover, the preimage distribution property of shifted multi-preimage trapdoor sampler also ensures that each $\boldsymbol{\pi}_i$ is independent of $\boldsymbol{\pi}_j$ for $j \neq i$ (indeed, the preimage distribution property stipulates that each $\boldsymbol{\pi}_i$ is sampled independently from $\mathbf{A}_i^{-1}(\mathbf{c})$, independently of all other $\boldsymbol{\pi}_j$). The work of [Wat24] relied on a noise smudging argument to argue hiding. We refer to Section 5 for a more detailed comparison between the two approaches for analyzing hiding.

Finally, the distribution of $\mathbf{v}_i$ in the two modes is computationally indistinguishable under the LWE assumption. In the context of the shifted multi-preimage trapdoor sampler, we require that LWE hold with respect to the matrix $\mathbf{A}_i$.

Instantiating [Wat24] with the shifted multi-preimage trapdoor sampler. The work of [Wat24] solve the shifted multi-preimage sampling problem by publishing a collection of short preimages in the common reference string. This leads to a structured CRS in both binding and hiding modes, and moreover, the size of the CRS scales quadratically with the output length of the hidden-bits generator. The [Wat24] scheme also requires a super-polynomial modulus to implement the noise smudging argument needed for hiding.

In this work, we replace the CRS with the seed for our shifted multi-preimage trapdoor sampler. This provides an efficient way to solve the shifted multi-preimage sampling problem and thus, suffices to instantiate the general blueprint of [Wat24]. In this case, the CRS for the hidden-bits generator consists of the seed for the sampler together with the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_\ell \in \mathbb{Z}_q^t$. This yields a hidden-bits generator with a CRS that is linear in the output length. Moreover, in statistically-hiding mode, each $\mathbf{v}_i$ is uniformly random, and we obtain a scheme in the common *random* string model (equivalently, a scheme with a *transparent* setup). We provide the full details in Section 5.

Dual-mode NIZKs for NP. Taken together, we obtain a dual-mode hidden-bits generator from plain LWE (Corollary 5.18). In conjunction with existing compilers [FLS90, QRW19, LPWW20, Wat24], this yields a dual-mode NIZK for NP from LWE with a polynomial modulus-to-noise ratio (Corollary 5.19). Our construction achieves the same set of properties as the Peikert-Shiehian construction [PS19] based on correlation-intractable hash functions. Our approach thus yields an alternative route to constructing NIZKs for NP that does not rely on non-black-box use of cryptographic primitives or lattice sampling algorithms.

2.3 Related Work

Vector commitments. Starting from Merkle's construction of vector commitments from any collision-resistant hash function [Mer87], many works have studied constructions of vector commitments from algebraic assumptions over groups with bilinear maps [LY10, KZG10, CF13, LRY16, LM19, TAB+20, GRWZ20], groups of unknown order [CF13, LM19, CFG+20, AR20, TXN20], and lattice-based assumptions [PSTY13, LLNW16, PPS21, dCP23, WW23b].

Compared to schemes like Merkle [Mer87] based on collision-resistant hash functions, the advantage of the algebraic approach we take is the support for properties like linear homomorphism (e.g., given commitments to $\mathbf{x}, \mathbf{x}'$, we can compute a commitment to the sum $\mathbf{x} + \mathbf{x}'$) or the support for stateless updates. Moreover, the basic approach based on collision-resistant hash functions does not satisfy hiding. It is possible to augment Merkle commitments to be (statistically) hiding using (lossy) public-key encryption or (statistical) NIZK arguments.

Prior to this work, the state-of-the-art in lattice-based vector commitments are [dCP23, WW23b]; both works give constructions from the SIS assumption. The work of de Castro and Peikert [dCP23] has a transparent polylogarithmic-size CRS (similar to our scheme). However, their scheme does not natively support hiding. While their work describes a way to achieve statistical hiding, the transformation comes at the price of relaxing binding to the weaker notion of target binding (i.e., where an adversary cannot open an *honestly-generated* commitment to two different values). Alternatively, one could compose the [dCP23] commitment scheme with a statistical NIZK argument [PS19, Wat24] (or a lossy public-key encryption scheme [PW08, HLOV11]) to obtain a statistically-hiding scheme. This approach would additionally bring in the LWE assumption. The advantage of our approach is we achieve statistical hiding directly without needing additional tools. Conversely, the scheme of Wee and Wu [WW23b] is statistically hiding. However, their scheme requires a structured CRS whose size scales *quadratically* with the vector dimension. Our construction has a transparent and polylogarithmic-size CRS.

Non-interactive zero-knowledge. NIZKs have been extensively studied and today, we have constructions from most standard algebraic assumptions; these include factoring [FLS90], pairing-based assumptions [CHK03, GOS06], (sub-exponential) decisional Diffie-Hellman [JJ21], learning with errors [CCH+19, PS19, Wat24], and the combination of learning parity with noise in conjunction with multivariate quadratic equations [DJJ24]. Among the lattice-based constructions, the initial constructions [CCH+19, PS19] leveraged correlation-intractable hash functions to provably instantiate the Fiat-Shamir heuristic, while the more recent work of [Wat24] show how use the LWE assumption to implement the classic hidden-bits model.

3 Preliminaries

Throughout this work, we write λ to denote the security parameter. For a positive integer $n \in \mathbb{N}$, we define the set $[n] \coloneqq \{1, \dots n\}$. For a positive integer $q \in \mathbb{N}$, we write $\mathbb{Z}_q$ to denote the ring of integers modulo q. We write $\mathsf{poly}(\lambda)$ to denote a fixed polynomial in λ. We write $\mathsf{negl}(\lambda)$ to denote a function that is negligible in λ (i.e., a function that is $o(\lambda^{-c})$ for all $c \in \mathbb{N}$). We say an event occurs with overwhelming probability if the probability of its complement occurring is negligible. We say an algorithm is efficient if it runs in probabilistic polynomial time in the length of its input. For two ensembles of distributions $\mathcal{D}_1 = \{\mathcal{D}_{1,\lambda}\}_{\lambda \in \mathbb{N}}$ and $\mathcal{D}_2 = \{\mathcal{D}_{2,\lambda}\}_{\lambda \in \mathbb{N}}$ indexed by a security parameter, we say they are computationally indistinguishable if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\left| \Pr[\mathcal{A}(1^\lambda, x) = 1 : x \leftarrow \mathcal{D}_{1,\lambda}] - \Pr[\mathcal{A}(1^\lambda, x) = 1 : x \leftarrow \mathcal{D}_{2,\lambda}] \right| = \mathsf{negl}(\lambda).$$

We say they are statistically indistinguishable if there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the statistical distance between them is $\mathsf{negl}(\lambda)$. We write $\mathcal{D}_1 \overset{c}{\approx} \mathcal{D}_2$ (resp., $\mathcal{D}_1 \overset{s}{\approx} \mathcal{D}_2$) if $\mathcal{D}_1$ and $\mathcal{D}_2$ are computationally (resp., statistically) indistinguishable. We write $\mathcal{D}_1 \equiv \mathcal{D}_2$ if the distributions are identical.

Vectors and matrices. Throughout, we use bold uppercase letters (e.g., $\mathbf{A}, \mathbf{B}$) to denote matrices, bold lowercase letters (e.g., $\mathbf{u}, \mathbf{v}$) to denote vectors, and non-boldface letters to refer to their components (e.g., $\mathbf{v} = [v_1, \dots, v_n]$). For matrices $\mathbf{A}_1, \dots, \mathbf{A}_\ell$, we write $\mathrm{diag}(\mathbf{A}_1, \dots, \mathbf{A}_\ell)$ to denote the block diagonal matrix where the blocks are the matrices $\mathbf{A}_1, \dots, \mathbf{A}_\ell$. For a vector $\mathbf{v} \in \mathbb{Z}^n$, we write $\|\mathbf{v}\|$ to denote the ℓ_∞-norm of $\mathbf{v}$. When $\mathbf{v} \in \mathbb{Z}_q^n$, we write $\|\mathbf{v}\|$ to denote the ℓ_∞-norm of the vector (over $\mathbb{Z}^n$) obtained by first associating each component $v_i \in \mathbb{Z}_q$ with its unique representative in the set $(-q/2, q/2] \cap \mathbb{Z}$. For a matrix $\mathbf{A}$, we write $\|\mathbf{A}\|$ to denote the ℓ_∞-norm of the vector obtained by concatenating together the columns of $\mathbf{A}$ (i.e., $\|\mathbf{A}\| = \max_{i,j} |A_{i,j}|$).

Lemma 3.1 (Full Rank Matrices [GPV08, Lemma 5.1]). *Let n, m, q be lattice parameters where q is prime and $m \ge 2n \log q$. Then, all but a $\mathsf{negl}(n)$ fraction of matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ are full rank.*

Discrete Gaussians and gadget matrices. We write $D_{\mathbb{Z},s}$ to denote the discrete Gaussian distribution over $\mathbb{Z}$ with width parameter $s > 0$. For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a target vector $\mathbf{y} \in \mathbb{Z}_q^n$ in the column-span of $\mathbf{A}$, we write $\mathbf{A}^{-1}(\mathbf{y})$ to denote the random variable $\mathbf{x} \leftarrow D_{\mathbb{Z},s}^m$ conditioned on $\mathbf{A}\mathbf{x} = \mathbf{y} \bmod q$. Note that if $\mathbf{y}$ is not in the column-span of $\mathbf{A}$, then the distribution $\mathbf{A}^{-1}(\mathbf{y})$ simply outputs $\perp$ with probability 1. We extend $\mathbf{A}^{-1}$ to operate on matrices by applying $\mathbf{A}^{-1}$ column-wise. For positive integers $n, q \in \mathbb{N}$, we write $\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g}^\intercal \in \mathbb{Z}_q^{n \times m'}$ to be the gadget matrix [MP12] where $\mathbf{I}_n$ is the identity matrix of dimension n, $\mathbf{g}^\intercal = [1, 2, \ldots, 2^{\lceil \log q \rceil - 1}]$, and $m' = n \lceil \log q \rceil$. For dimensions $m \geq m'$, we overload the notation and write $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ to denote the "padded gadget matrix" $[\mathbf{I}_n \otimes \mathbf{g}^\intercal \mid \mathbf{0}^{n \times (m - m')}]$. The inverse function $\mathbf{G}^{-1} \colon \mathbb{Z}_q^n \to \mathbb{Z}_q^{m'}$ expands each entry $x \in \mathbb{Z}_q$ into a column of size $\lceil \log q \rceil$ corresponding to the bits in the binary representation of x. Similarly, when $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ is a padded gadget matrix with dimension $m \geq m'$, we extend the output of $\mathbf{G}^{-1} \colon \mathbb{Z}_q^n \to \mathbb{Z}_q^m$ by zero-padding each column. By construction, for all $\mathbf{t} \in \mathbb{Z}_q^n$, it follows that $\mathbf{G} \cdot \mathbf{G}^{-1}(\mathbf{t}) = \mathbf{t} \bmod q$.

Lemma 3.2 (Gaussian Tail Bound [MP12, Lemma 2.6, adapted]). *Let n, m, q be lattice parameters where $m \geq 2n \log q$. Then, for all but a $\mathsf{negl}(n)$-fraction of matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, all width parameters $s > \log m$ and all vectors $\mathbf{y} \in \mathbb{Z}_q^n$ in the column-span of $\mathbf{A}$,*

$$\Pr[\|\mathbf{x}\| > \sqrt{m}s : \mathbf{x} \leftarrow \mathbf{A}_s^{-1}(\mathbf{y})] \leq O(2^{-m}).$$

For the particular case of the discrete Gaussian over the integers and any $\lambda \in \mathbb{N}$,

$$\Pr[|x| > \sqrt{\lambda}s : x \leftarrow D_{\mathbb{Z},s}] \leq 2^{-\lambda}.$$

Lemma 3.3 (Discrete Gaussian Preimages [GPV08, adapted]). *Let n, m, q, s be lattice parameters where $m \geq n \lceil \log q \rceil$ and $s \geq \log m$. Then the statistical distance between the following distributions is at most $\mathsf{negl}(n)$:*

$$\left\{ \mathbf{G}\mathbf{x} : \mathbf{x} \leftarrow D_{\mathbb{Z},s}^m \right\} \quad \textit{and} \quad \left\{ \mathbf{u} : \mathbf{u} \xleftarrow{\text{R}} \mathbb{Z}_q^n \right\}.$$

Discrete Gaussian preimages. We will need to reason about the distribution of $[\operatorname{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid \mathbf{B}]_s^{-1}(\mathbf{t})$ where $\mathbf{A}_i \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{B} \in \mathbb{Z}_q^{n\ell \times k}$.

Definition 3.4 (Minimum Distance of $\Lambda(\mathbf{A}_i)$). For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, we write $\Lambda(\mathbf{A}_i)$ to denote the q-ary lattice $\Lambda(\mathbf{A}) \coloneqq \{\mathbf{y} \in \mathbb{Z}^m : \mathbf{y} = \mathbf{A}^\intercal \mathbf{x} \bmod q \text{ for some } \mathbf{x} \in \mathbb{Z}^m\}$. For a lattice $\Lambda \subset \mathbb{R}^m$, we write $\lambda_1^\infty(\Lambda)$ to denote the minimum distance $\lambda_1^\infty(\Lambda) \coloneqq \min_{\mathbf{0} \neq \mathbf{v} \in \Lambda} \|\mathbf{x}\|$.

Lemma 3.5 (Minimum Distance of Random Matrix [GPV08, Lemma 5.3]). *Let n, m, q be lattice parameters where q is prime and $m \geq 2n \log q$. Then, for all but a $q^{-n} = \mathsf{negl}(n)$ fraction of matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\lambda_1^\infty(\Lambda(\mathbf{A})) \geq q/4$.*

Lemma 3.6 (Discrete Gaussian Preimages [WW23b, Corollary 2.11]). *Let n, m, q, t be parameters where $m \geq n$. Take any $\ell, k = \mathsf{poly}(n, \log q)$, any collection of matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n \times m}$ where $\mathbf{A}_i$ is full rank and $\lambda_1^\infty(\Lambda(\mathbf{A}_i)) \geq t$ for all $i \in [\ell]$, any collection of matrices $\mathbf{B}_1, \ldots, \mathbf{B}_\ell \in \mathbb{Z}_q^{n \times k}$, and any target vector $\mathbf{t} \in \mathbb{Z}_q^{n\ell}$. Define the following matrices*

$$\mathbf{C} = \left[\begin{array}{ccc|c} \mathbf{A}_1 & & & \mathbf{B}_1 \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & \mathbf{B}_\ell \end{array} \right] \quad \textit{and} \quad \mathbf{t} = \begin{bmatrix} \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_\ell \end{bmatrix}.$$

Then, for all width parameters $s \geq q/t \cdot \log(\ell m)$, the statistical distance between the following distributions is $\mathsf{negl}(m)$:

$$\left\{ \mathbf{v} : \mathbf{v} \leftarrow \mathbf{C}_s^{-1}(\mathbf{t}) \right\} \quad \textit{and} \quad \left\{ \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_\ell \\ \mathbf{v}_{\ell+1} \end{bmatrix} : \begin{array}{c} \mathbf{v}_{\ell+1} \leftarrow D_{\mathbb{Z},s}^k, \\ \mathbf{v}_i \leftarrow (\mathbf{A}_i)_s^{-1}(\mathbf{t}_i - \mathbf{B}_i \mathbf{u}). \end{array} \right\}. \tag{3.1}$$

Min-entropy. Let $\mathcal{D}$ be a distribution with finite support $\mathcal{X}$. We define the min-entropy of $\mathcal{D}$ to be $\mathbf{H}_\infty(\mathcal{D}) := -\log \max_{x \in \mathcal{X}} \Pr[X = x : X \leftarrow \mathcal{D}]$. We now state the following corollary of the leftover hash lemma [HILL99]:

Lemma 3.7 (Leftover Hash Lemma). *Let $m, q \in \mathbb{N}$ be positive integers where q is prime. Let $\mathcal{D}$ be a distribution over $\mathbb{Z}_q^m$ where $\mathbf{H}_\infty(\mathcal{D}) \geq 2\lambda + \log q$. Then, for all $\ell = \ell(\lambda)$, the statistical distance between the following two distributions is at most $\ell(\lambda) \cdot 2^{-\lambda}$:*

$$\left\{ (\mathbf{v}, \mathbf{v}^{\mathsf{T}}\mathbf{x}_1, \dots, \mathbf{v}^{\mathsf{T}}\mathbf{x}_\ell) : \begin{array}{c} \mathbf{v} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^m \\ \forall i \in [\ell] : \mathbf{x}_i \leftarrow \mathcal{D} \end{array} \right\} \quad \textit{and} \quad \left\{ (\mathbf{v}, r_1, \dots, r_\ell) : \begin{array}{c} \mathbf{v} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^m \\ \forall i \in [\ell] : r_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q \end{array} \right\}.$$

Lemma 3.8 (Min-Entropy of Discrete Gaussian [PR06, Lemma 2.11, adapted]). *Let n, q be lattice parameters and suppose $m \geq 2n \log q$. Then, for all but a $\mathrm{negl}(n)$-fraction of matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, all width parameters $s \geq \log m$, all target vectors $\mathbf{y} \in \mathbb{Z}_q^n$, the random variable $\mathbf{x} \leftarrow \mathbf{A}_s^{-1}(\mathbf{y})$ has min-entropy $\mathbf{H}_\infty(\mathbf{x}) \geq n/2$.*

Gadget trapdoors. Our constructions will use the gadget trapdoors from [MP12], which builds on a long sequence of works on constructing lattice trapdoors [Ajt96, GPV08, AP09, ABB10a, ABB10b, CHKP10].

Theorem 3.9 (Gadget Trapdoor [MP12, adapted]). *Let n, m, q be lattice parameters with $m \geq 3n \lceil \log q \rceil$. Then there exist efficient algorithms* $(\mathsf{TrapGen}, \mathsf{SamplePre})$ *with the following syntax:*

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

Moreover, the above algorithms satisfy the following properties:

- **Trapdoor distribution:** *If $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$, then the distribution of $\mathbf{A}$ is 2^{-n}-close to the uniform distribution over $\mathbb{Z}_q^{n \times m}$. Moreover, $\mathbf{A}\mathbf{T} = \mathbf{G}$ and $\|\mathbf{T}\| = 1$.*
- **Preimage sampling:** *For all matrices $\mathbf{T}$, width parameters $s > 0$, and all target vectors $\mathbf{y} \in \mathbb{Z}_q^n$ in the column span of $\mathbf{A}$, the output $\mathbf{x} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{T}, \mathbf{y}, s)$ of $\mathsf{SamplePre}$ satisfies $\mathbf{A}\mathbf{x} = \mathbf{y}$.*
- **Preimage distribution:** *Suppose $\mathbf{T}$ is a gadget trapdoor for $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ (i.e., $\mathbf{A}\mathbf{T} = \mathbf{G}$). Then, for all $s \geq m \|\mathbf{T}\| \log n$, and all target vectors $\mathbf{y} \in \mathbb{Z}_q^n$, the statistical distance between the following distributions is at most 2^{-n}:*

$$\{\mathbf{x} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{T}, \mathbf{y}, s)\} \quad \textit{and} \quad \left\{\mathbf{x} \leftarrow \mathbf{A}_s^{-1}(\mathbf{y})\right\}.$$

Homomorphic evaluation. Our construction of succinct functional commitments will rely on the lattice homomorphic evaluation procedure developed in [GSW13, BGG+14, DHM+24]. In this work, we consider a specialization to indicator functions $\delta_{\mathbf{u}} \colon \{0,1\}^\ell \to \{0,1\}$ where

$$\delta_{\mathbf{u}}(\mathbf{x}) := \begin{cases} 1 & \mathbf{x} = \mathbf{u} \\ 0 & \mathbf{x} \neq \mathbf{u}. \end{cases}$$

Specifically, we focus on the version for "database reads" from [DHM+24, §4.1].

Theorem 3.10 (Homomorphic Encodings [DHM+24, Theorem 4.5, adapted]). *Let λ be a security parameter and $n = n(\lambda)$, $q = q(\lambda)$ be lattice parameters. Take any $m \geq n \lceil \log q \rceil$ and let $\ell = \ell(\lambda)$ be an input length. Then, there exist a pair of efficient algorithms* $(\mathsf{EvalF}, \mathsf{EvalFX})$ *with the following properties:*

- $\mathsf{EvalF}(\mathbf{A}, \delta_{\mathbf{u}}) \to \mathbf{A}_{\mathbf{u}}$*: On input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times \ell m}$ and the indicator function $\delta_{\mathbf{u}}$ (where $\mathbf{u} \in \{0,1\}^\ell$), the input-independent evaluation algorithm outputs a matrix $\mathbf{A}_{\mathbf{u}} \in \mathbb{Z}_q^{n \times m}$.*

- $\mathsf{EvalFX}(\mathbf{A}, \delta_{\mathbf{u}}, \mathbf{x}) \to \mathbf{H}_{\mathbf{A},\delta_{\mathbf{u}},\mathbf{x}}$: *On input a matrix* $\mathbf{A} \in \mathbb{Z}_q^{n \times \ell m}$, *an indicator function* $\delta_{\mathbf{u}}$ *(where* $\mathbf{u} \in \{0,1\}^\ell$*), and an input* $\mathbf{x} \in \{0,1\}^\ell$*, the input-dependent evaluation algorithm outputs a matrix* $\mathbf{H}_{\mathbf{A},\mathbf{u},\mathbf{x}} \in \mathbb{Z}_q^{\ell m \times m}$.

Moreover for all security parameters $\lambda \in \mathbb{N}$*, matrices* $\mathbf{A} \in \mathbb{Z}_q^{n \times \ell m}$*, all indicator functions* $\delta_{\mathbf{u}} \in \mathcal{F}$*, and all inputs* $\mathbf{x} \in \{0,1\}^\ell$*, the matrices* $\mathbf{A}_{\mathbf{u}} \leftarrow \mathsf{EvalF}(\mathbf{A}, \delta_{\mathbf{u}})$ *and* $\mathbf{H}_{\mathbf{A},\mathbf{u},\mathbf{x}} \leftarrow \mathsf{EvalFX}(\mathbf{A}, \delta_{\mathbf{u}}, \mathbf{x})$ *satisfy the following properties:*

- $\mathbf{H}_{\mathbf{A},\mathbf{u},\mathbf{x}} \in \{-1,0,1\}^{\ell m \times m}$.
- $(\mathbf{A} - \mathbf{x}^{\mathsf{T}} \otimes \mathbf{G}) \cdot \mathbf{H}_{\mathbf{A},\mathbf{u},\mathbf{x}} = \mathbf{A}_{\mathbf{u}} - \delta_{\mathbf{u}}(\mathbf{x}) \cdot \mathbf{G}$.

Moreover, the running time of $\mathsf{EvalF}(\mathbf{A}, \delta_{\mathbf{u}})$ *and* $\mathsf{EvalFX}(\mathbf{A}, \delta_{\mathbf{u}}, \mathbf{x})$ *is bounded by* $2^\ell \cdot \mathrm{poly}(n, m, \ell, \log q)$.

Lattice assumptions. We recall the short integer solutions (SIS) [Ajt96] and learning with errors (LWE) [Reg05] problems.

Assumption 3.11 (Short Integer Solutions [Ajt96])**.** Let λ be a security parameter and $n = n(\lambda)$, $m = m(\lambda)$, $q = q(\lambda)$, and $\beta = \beta(\lambda)$ be lattice parameters. We say the short integer solution problem $\mathsf{SIS}_{n,m,q,\beta}$ holds if for all efficient adversaries $\mathcal{A}$,

$$\Pr\left[\mathbf{A}\mathbf{x} = \mathbf{0} \text{ and } 0 < \|\mathbf{x}\| \le \beta : \begin{array}{c} \mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m}; \\ \mathbf{x} \leftarrow \mathcal{A}(1^\lambda, \mathbf{A}) \end{array}\right] = \mathsf{negl}(\lambda).$$

Assumption 3.12 (Learning with Errors [Reg05])**.** Let λ be a security parameter and $n = n(\lambda)$, $m = m(\lambda)$, $q = q(\lambda)$, and $s = s(\lambda)$ be lattice parameters. We say the learning with errors problem $\mathsf{LWE}_{n,m,q,s}$ holds if for all efficient adversaries $\mathcal{A}$,

$$\left|\Pr\left[\mathcal{A}(\mathbf{A}, \mathbf{s}^{\mathsf{T}}\mathbf{A} + \mathbf{e}^{\mathsf{T}}) = 1 : \begin{array}{c} \mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m} \\ \mathbf{s} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^m, \mathbf{e} \leftarrow D_{\mathbb{Z},s}^m \end{array}\right] - \Pr\left[\mathcal{A}(\mathbf{A}, \mathbf{u}^{\mathsf{T}}) = 1 : \begin{array}{c} \mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m} \\ \mathbf{u} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^m \end{array}\right]\right| = \mathsf{negl}(\lambda).$$

4 Shifted Multi-Preimage Trapdoor Sampler

In this section, we describe our technique for deriving a collection of matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ together with a trapdoor for solving the shifted multi-preimage sampling problem from a short common *random* string. Our construction relies on the following key lemma asserting that a certain structured matrix has a *public* trapdoor:

Lemma 4.1 (Structured Lattice with a Public Trapdoor)**.** *Let* λ *be a security parameter and* $n = n(\lambda)$, $m = m(\lambda)$ *and* $q = q(\lambda)$ *be lattice parameters. Suppose* $m \ge n \lceil \log q \rceil$*. Then, for all* $\ell \in \mathbb{N}$*, and* $t = m \lceil \log \ell \rceil$*, there exists an explicit polynomial-time algorithm* $\mathsf{StructTrapGen}$ *that takes as input* $(\mathbf{B}, \mathbf{u}_1, \ldots, \mathbf{u}_\ell)$ *where* $\mathbf{B} \in \mathbb{Z}_q^{n \times t}$*, and* $\mathbf{u}_1, \ldots, \mathbf{u}_\ell \in \{0,1\}^{\lceil \log \ell \rceil}$ *are distinct vectors, and outputs a gadget trapdoor* $\mathbf{T} \in \mathbb{Z}_q^{(\ell t + m) \times \ell m}$ *where* $\|\mathbf{T}\| = 1$ *for the matrix*

$$\mathbf{D}'_\ell = \left[\begin{array}{ccc|c} \mathbf{B} - \mathbf{u}_1^{\mathsf{T}} \otimes \mathbf{G} & & & \mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{B} - \mathbf{u}_\ell^{\mathsf{T}} \otimes \mathbf{G} & \mathbf{G} \end{array}\right] \in \mathbb{Z}_q^{\ell n \times (\ell t + m)}.$$

Proof. Algorithm $\mathsf{StructTrapGen}(\mathbf{B}, \mathbf{u}_1, \ldots, \mathbf{u}_\ell)$ works as follows:

1. For $i \in [\ell]$, let $\delta_{\mathbf{u}_i} \colon \{0,1\}^{\lceil \log \ell \rceil} \to \{0,1\}$ be the indicator function where $\delta_{\mathbf{u}_i}(\mathbf{v})$ outputs 1 if $\mathbf{v} = \mathbf{u}_i$ and 0 otherwise. For each $i, j \in [\ell]$, compute $\mathbf{H}_{\mathbf{B},\mathbf{u}_i,\mathbf{u}_j} \leftarrow \mathsf{EvalFX}(\mathbf{B}, \delta_{\mathbf{u}_i}, \mathbf{u}_j)$ and $\mathbf{B}_{\mathbf{u}_i} \leftarrow \mathsf{EvalF}(\mathbf{B}, \delta_{\mathbf{u}_i})$.
2. Output the trapdoor

$$\mathbf{T} = \begin{bmatrix} -\mathbf{H}_{\mathbf{B},\mathbf{u}_1,\mathbf{u}_1} & \cdots & -\mathbf{H}_{\mathbf{B},\mathbf{u}_\ell,\mathbf{u}_1} \\ \vdots & \ddots & \vdots \\ -\mathbf{H}_{\mathbf{B},\mathbf{u}_1,\mathbf{u}_\ell} & \cdots & -\mathbf{H}_{\mathbf{B},\mathbf{u}_\ell,\mathbf{u}_\ell} \\ \mathbf{G}^{-1}(\mathbf{B}_{\mathbf{u}_1}) & \cdots & \mathbf{G}^{-1}(\mathbf{B}_{\mathbf{u}_\ell}) \end{bmatrix} \in \mathbb{Z}_q^{(\ell t + m) \times \ell m}. \tag{4.1}$$

To complete the proof, we show that the trapdoor $\mathbf{T}$ output by StructTrapGen satisfies the required properties. Namely, we need to show that

$$\mathbf{D}'_\ell \mathbf{T} = \left[\begin{array}{ccc|c} \mathbf{B} - \mathbf{u}_1^{\mathsf{T}} \otimes \mathbf{G} & & & \mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{B} - \mathbf{u}_\ell^{\mathsf{T}} \otimes \mathbf{G} & \mathbf{G} \end{array}\right] \cdot \begin{bmatrix} -\mathbf{H}_{\mathbf{B},\mathbf{u}_1,\mathbf{u}_1} & \cdots & -\mathbf{H}_{\mathbf{B},\mathbf{u}_\ell,\mathbf{u}_1} \\ \vdots & \ddots & \vdots \\ -\mathbf{H}_{\mathbf{B},\mathbf{u}_1,\mathbf{u}_\ell} & \cdots & -\mathbf{H}_{\mathbf{B},\mathbf{u}_\ell,\mathbf{u}_\ell} \\ \mathbf{G}^{-1}(\mathbf{B}_{\mathbf{u}_1}) & \cdots & \mathbf{G}^{-1}(\mathbf{B}_{\mathbf{u}_\ell}) \end{bmatrix} = \begin{bmatrix} \mathbf{G} & & \\ & \ddots & \\ & & \mathbf{G} \end{bmatrix}. \tag{4.2}$$

Since $t = m \cdot \lceil \log \ell \rceil \geq n \lceil \log q \rceil \lceil \log \ell \rceil$, by Theorem 3.10, we have

$$(\mathbf{B} - \mathbf{u}_i^{\mathsf{T}} \otimes \mathbf{G}) \cdot \mathbf{H}_{\mathbf{B},\mathbf{u}_j,\mathbf{u}_i} = \mathbf{B}_{\mathbf{u}_j} - \delta_{\mathbf{u}_j}(\mathbf{u}_i) \cdot \mathbf{G} = \begin{cases} \mathbf{B}_{\mathbf{u}_j} - \mathbf{G} & i = j \\ \mathbf{B}_{\mathbf{u}_j} & i \neq j. \end{cases}$$

Correspondingly, Eq. (4.2) holds. Again by Theorem 3.10, for all $i, j \in [\ell]$, we have that $\mathbf{H}_{\mathbf{B},\mathbf{u}_i,\mathbf{u}_j} \in \{-1, 0, 1\}^{t \times m}$. Moreover $\mathbf{G}^{-1}(\mathbf{B}_{\mathbf{u}_i}) \in \{0, 1\}^{m \times m}$, so it follows that $\|\mathbf{T}\| = 1$. □

Shifted multi-preimage trapdoor syntax. At a high-level, a shifted multi-preimage trapdoor sampler provides a way to sample a set of matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ together with a trapdoor td that allows us to efficiently solve the shifted multi-preimage sampling problem with respect to $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$. Formally, the sampler consists of a Gen algorithm that samples a common reference string crs, an Expand algorithm that expands crs into the matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ and the trapdoor td, and a shifted multi-preimage sampler algorithm SampleMultPre. The main properties we require are as follows:

- **Correctness:** Given the trapdoor td and any set of target vectors $\mathbf{t}_1, \ldots, \mathbf{t}_\ell$, the shifted multi-preimage sampler $\mathsf{SampleMultPre}(\mathsf{td}, \mathbf{t}_1, \ldots, \mathbf{t}_\ell)$ outputs a solution $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$ where for all $i \in [\ell]$, $\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{t}_i + \mathbf{c}$.

- **Preimage distribution:** We require that the solutions output by SampleMultPre to have a "nice" distribution. Formally, we require that the joint distribution of the solution $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$ output by the sampler $\mathsf{SampleMultPre}(\mathsf{td}, \mathbf{t}_1, \ldots, \mathbf{t}_\ell)$ to be statistically close to sampling $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and $\boldsymbol{\pi}_i \leftarrow \mathbf{A}_i^{-1}(\mathbf{t}_i + \mathbf{c})$. In other words, the distribution of each $\boldsymbol{\pi}_i$ should be statistically close to an independent discrete Gaussian $\boldsymbol{\pi}_i$ conditioned on $\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{t}_i + \mathbf{c}$. We use this property to argue hiding for our dual-mode hidden-bits generator (Theorem 5.10) and for our vector commitment scheme (Theorem 6.8).

- **Somewhere programmable:** We also require that SIS and LWE are hard with respect to any $\mathbf{A}_i$ given $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ and the trapdoor td. We model this by defining a "somewhere programmable" property which stipulates that there is an auxiliary sampling algorithm GenProg that takes as input an index i and a (random) matrix $\mathbf{A}_i$ and outputs a common reference string $\widetilde{\mathsf{crs}}$ that is indistinguishable from the common reference string output by Gen. Moreover, the ith matrix associated with $\widetilde{\mathsf{crs}}$ is precisely the programmed matrix $\mathbf{A}_i$. This property ensures that the marginal distribution of any individual $\mathbf{A}_i$ associated with an honestly-generated crs is statistically close to uniform, and in addition, that problems like SIS or LWE remain hard with respect to any individual $\mathbf{A}_i$ even given the common reference string. We use this property to argue mode indistinguishability for our dual-mode hidden-bits generator (Theorem 5.5) and binding for our vector commitment scheme (Theorem 6.4).

We now give the formal definition and construction.

Definition 4.2 (Shifted Multi-Preimage Trapdoor Sampler)**.** Let λ be a security parameter and ℓ be a dimension. Let n, t, q, s be parameters that are functions of λ and ℓ. An (n, t, q, s)-shifted multi-preimage trapdoor sampler is a tuple of efficient algorithms $\Pi_{\mathsf{samp}} = (\mathsf{Gen}, \mathsf{Expand}, \mathsf{SampleMultPre})$ with the following syntax:

- $\mathsf{Gen}(1^\lambda, 1^\ell) \to \mathsf{crs}$: On input the security parameter λ and the dimension ℓ, the generator algorithm outputs a common reference string crs.

- $\mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}) \to (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td})$: On input the security parameter λ, the dimension ℓ, and the common reference string crs, the expand algorithm outputs matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n \times t}$ and a trapdoor td. This algorithm is *deterministic.*

- $\mathsf{SampleMultPre}(\mathsf{td}, \mathbf{t}_1, \ldots, \mathbf{t}_\ell) \to (\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$: On input a trapdoor td and a collection of preimages $\mathbf{t}_1, \ldots, \mathbf{t}_\ell$, the shifted multi-preimage sampling algorithm outputs a shift $\mathbf{c} \in \mathbb{Z}_q^n$ together with preimages $\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell \in \mathbb{Z}_q^t$.

The shifted multi-preimage trapdoor sampler should satisfy the following properties:

- **Correctness**: For all $\lambda, \ell \in \mathbb{N}$, all crs in the support of $\mathsf{Gen}(1^\lambda, 1^\ell)$, all target vectors $\mathbf{t}_1, \ldots, \mathbf{t}_\ell \in \mathbb{Z}_q^n$, and setting $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs})$, it holds that
$$\Pr[\mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{t}_i + \mathbf{c} \text{ for all } i \in [\ell] : (\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c}) \leftarrow \mathsf{SampleMultPre}(\mathsf{td}, \mathbf{t}_1, \ldots, \mathbf{t}_\ell)] = 1$$

- **Preimage distribution:** For all polynomials $\ell = \ell(\lambda)$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that with overwhelming probability over the choice of $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$, letting $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, \mathsf{crs})$, and for all targets $\mathbf{t}_1, \ldots, \mathbf{t}_\ell \in \mathbb{Z}_q^n$ and all $\lambda \in \mathbb{N}$, the statistical distance between the following distributions is $\mathsf{negl}(\lambda)$:
  - $\mathcal{D}_0$: Output $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c}) \leftarrow \mathsf{SampleMultPre}(\mathsf{td}, \mathbf{t}_1, \ldots, \mathbf{t}_\ell)$.
  - $\mathcal{D}_1$: Sample $\mathbf{c} \xleftarrow{\text{R}} \mathbb{Z}_q^n$ and $\boldsymbol{\pi}_i \leftarrow (\mathbf{A}_i)_s^{-1}(\mathbf{t}_i + \mathbf{c})$ for each $i \in [\ell]$. Output $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$.

- **Somewhere programmable:** There exists an efficient algorithm GenProg such that for all polynomials $\ell = \ell(\lambda)$, the following hold:
  - For all $\lambda \in \mathbb{N}$, all indices $i \in [\ell]$, and all matrices $\mathbf{A}_i \in \mathbb{Z}_q^{n \times t}$, it holds that
$$\Pr\left[\mathbf{A}_i = \widetilde{\mathbf{A}}_i : \begin{array}{c} \widetilde{\mathsf{crs}} \leftarrow \mathsf{GenProg}(1^\lambda, 1^\ell, i, \mathbf{A}_i) \\ (\widetilde{\mathbf{A}}_1, \ldots, \widetilde{\mathbf{A}}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \widetilde{\mathsf{crs}}) \end{array}\right] = 1.$$
  - There exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ and all $i \in [\ell]$, the statistical distance between the following distributions is $\mathsf{negl}(\lambda)$:
$$\left\{\mathsf{crs} : \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)\right\} \quad \text{and} \quad \left\{\widetilde{\mathsf{crs}} : \begin{array}{c} \mathbf{A}_i \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times t} \\ \widetilde{\mathsf{crs}} \leftarrow \mathsf{GenProg}(1^\lambda, 1^\ell, i, \mathbf{A}_i) \end{array}\right\}.$$

    When these distributions are identical, we say the shifted multi-preimage trapdoor sampler satisfies *perfect* somewhere programmability.

Definition 4.3 (Transparent Setup)**.** A shifted multi-preimage trapdoor sampler (Gen, Expand, SampleMultPre) supports transparent setup if the common reference string crs output by Gen just consists of the random coins used to sample crs. Otherwise, we say the common reference string is a *structured* reference string (i.e., sampled using *private* randomness).

Local expansion. The Expand algorithm in Definition 4.2 outputs a collection of ℓ matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ along with a trapdoor $\mathbf{T}$. Consequently, this necessarily takes $\mathsf{poly}(\lambda, \ell)$ time. In some applications (e.g., to vector commitments), we require the shifted multi-preimage trapdoor sampler to support a more fine-grained algorithm ExpandLocal that takes as input a single index $i \in [\ell]$ and outputs $\mathbf{A}_i$. In particular, the local expansion algorithm only needs to read $\mathsf{poly}(\lambda, \log \ell)$ bits of the CRS and runs in time $\mathsf{poly}(\lambda, \log \ell)$. We give the formal definition below:

Definition 4.4 (Local Expansion)**.** A shifted multi-preimage trapdoor sampler (Gen, Expand, SampleMultPre) supports local expansion if there exists an efficient algorithm ExpandLocal that takes as input crs together with an index $i \in [\ell]$ and outputs $\mathbf{A}_i$:

- $\mathsf{ExpandLocal}(1^\lambda, \mathsf{crs}, i)$: On input the security parameter λ, the common reference string crs, and an index i, the local expand algorithm outputs a matrix $\mathbf{A}_i \in \mathbb{Z}_q^{n \times t}$. This algorithm is deterministic.

The requirement for ExpandLocal is that for all $\lambda, \ell \in \mathbb{N}$ and all crs, if $\mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}) = (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{T})$, then for all $i \in [\ell]$, $\mathsf{ExpandLocal}(1^\lambda, \mathsf{crs}, i) = \mathbf{A}_i$. In addition, $\mathsf{ExpandLocal}(1^\lambda, \mathsf{crs}, i)$ only needs to read $\mathsf{poly}(\lambda, \log \ell)$ bits of crs, and moreover, runs in time $\mathsf{poly}(\lambda, \log \ell)$ given these bits.

Simulatable openings. In some applications (e.g., to statistically-hiding vector commitments; see Section 6), we require a stronger requirement where we can sample a common reference string crs (that is indistinguishable from a real CRS) together with trapdoors $\mathbf{T}_1, \ldots, \mathbf{T}_\ell$ for each of the matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ associated with crs. Notably, the trapdoor $\mathbf{T}_i$ can be used to *efficiently* sample from $\mathbf{A}_i^{-1}(\mathbf{c})$ for any vector $\mathbf{c}$ in the column-space of $\mathbf{A}_i$. We give the formal definition below:

Definition 4.5 (Simulatable Openings)**.** An (n, t, q, s)-shifted multi-preimage trapdoor sampler $\Pi_{\mathsf{samp}} = (\mathsf{Gen}, \mathsf{Expand}, \mathsf{SampleMultPre})$ has simulatable openings if there exists an efficient algorithm GenTD with the following syntax:

- $\mathsf{GenTD}(1^\lambda, 1^\ell) \to (\mathsf{crs}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell)$: On input the security parameter λ and the dimension ℓ, the trapdoor generator algorithm outputs a common reference string crs together with trapdoors $\mathbf{T}_1, \ldots, \mathbf{T}_\ell$.

Moreover, the GenTD algorithm should satisfy the following properties:

- **Mode indistinguishability:** There exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the statistical distance between the following distributions is $\mathsf{negl}(\lambda)$:

$$\left\{\mathsf{crs} : \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)\right\} \quad \text{and} \quad \left\{\mathsf{crs} : (\mathsf{crs}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell) \leftarrow \mathsf{GenTD}(1^\lambda, 1^\ell)\right\}.$$

- **Trapdoor generation:** For all $\lambda, \ell \in \mathbb{N}$, and all $(\mathsf{crs}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell)$ in the support of $\mathsf{GenTD}(1^\lambda, 1^\ell)$, and setting $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs})$, we have that

$$\forall i \in [\ell] : \mathbf{A}_i \mathbf{T}_i = \mathbf{G} \quad \text{and} \quad \|\mathbf{T}_i\| \le s/(t \log n).$$

Shifted multi-preimage trapdoor sampler construction. We now give our shifted multi-preimage trapdoor sampler and analysis. The construction critically leverages Lemma 4.1. We refer to Section 2 for an overview of our construction.

Construction 4.6 (Shifted Multi-Preimage Trapdoor Sampler)**.** Let λ be a security parameter and ℓ be a dimension. Let $n = n(\lambda, \ell)$, $q = q(\lambda, \ell)$, and $s = s(\lambda, \ell)$ be lattice parameters. Let $m = 3n \lceil \log q \rceil$ and $t = m(\lceil \log \ell \rceil + 1)$. In the following construction, we associate each index $i \in [\ell]$ with a distinct canonical bit-vector $\mathbf{u}_i \in \{0,1\}^{\lceil \log \ell \rceil}$ (e.g., the bit-vector associated with the binary representation of $i - 1$). We construct an (n, t, q, s)-shifted multi-preimage trapdoor sampler as follows:

- $\mathsf{Gen}(1^\lambda, 1^\ell)$: On input the security parameter λ and the dimension ℓ, the generator algorithm samples $[\mathbf{A} \mid \mathbf{B}] \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times t}$ and outputs $\mathsf{crs} = [\mathbf{A} \mid \mathbf{B}]$.
- $\mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs})$: On input the security parameter λ, the dimension ℓ, and the common reference string $\mathsf{crs} = [\mathbf{A} \mid \mathbf{B}]$, where $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{B} \in \mathbb{Z}_q^{n \times m \lceil \log \ell \rceil}$. Then, the expand algorithm proceeds as follows:
  - For each $i \in [\ell]$, let $\mathbf{B}_i = \mathbf{B} - \mathbf{u}_i^{\mathsf{T}} \otimes \mathbf{G}$. Then, define

$$\mathbf{A}_i = [\mathbf{A} \mid \mathbf{B} - \mathbf{u}_i^{\mathsf{T}} \otimes \mathbf{G}] = [\mathbf{A} \mid \mathbf{B}_i] \in \mathbb{Z}_q^{n \times t}.$$

    Let $\mathbf{D}_\ell = [\mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid \mathbf{1}^\ell \otimes \mathbf{G}] \in \mathbb{Z}_q^{\ell n \times (\ell t + m)}$.
  - Let $\mathbf{\Pi} \in \{0,1\}^{(\ell t + m) \times (\ell t + m)}$ be the permutation matrix where

$$\mathbf{D}_\ell = \left[\begin{array}{ccccccc|c} \mathbf{A} & \mathbf{B}_1 & & & & & & \mathbf{G} \\ & & \mathbf{A} & \mathbf{B}_2 & & & & \mathbf{G} \\ & & & & \ddots & & & \vdots \\ & & & & & \mathbf{A} & \mathbf{B}_\ell & \mathbf{G} \end{array}\right] = \left[\begin{array}{ccc|ccc|c} \mathbf{A} & & & \mathbf{B}_1 & & & \mathbf{G} \\ & \ddots & & & \ddots & & \vdots \\ & & \mathbf{A} & & & \mathbf{B}_\ell & \mathbf{G} \end{array}\right] \mathbf{\Pi}. \tag{4.3}$$

– Compute $\mathbf{T}' \leftarrow \mathsf{StructTrapGen}(\mathbf{B}, \mathbf{u}_1, \dots, \mathbf{u}_\ell) \in \mathbb{Z}_q^{(\ell m \lceil \log \ell \rceil + m) \times \ell m}$ where StructTrapGen is the algorithm from Lemma 4.1. Define the trapdoor $\mathbf{T} \in \mathbb{Z}_q^{(\ell t + m) \times \ell m}$ where

$$\mathbf{T} = \mathbf{\Pi}^{-1} \begin{bmatrix} \mathbf{0}^{\ell m \times \ell m} \\ \mathbf{T}' \end{bmatrix} \in \mathbb{Z}_q^{(\ell t + m) \times \ell m}. \tag{4.4}$$

The algorithm sets $\mathsf{td} = (\mathbf{D}_\ell, \mathbf{T})$ and outputs $(\mathbf{A}_1, \dots, \mathbf{A}_\ell, \mathsf{td})$.

- $\mathsf{SampleMultPre}(\mathsf{td}, \mathbf{t}_1, \dots, \mathbf{t}_\ell)$: On input the trapdoor $\mathsf{td} = (\mathbf{D}_\ell, \mathbf{T})$ and target vectors $\mathbf{t}_1, \dots, \mathbf{t}_\ell \in \mathbb{Z}_q^n$, the shifted multi-preimage sampler defines the vector $\mathbf{t} \in \mathbb{Z}_q^{\ell n}$ to be the vertical concatenation of $\mathbf{t}_1, \dots, \mathbf{t}_\ell$ and outputs $(\boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_\ell, -\mathbf{G}\hat{\mathbf{c}})$ where
$$\begin{bmatrix} \boldsymbol{\pi}_1 \\ \vdots \\ \boldsymbol{\pi}_\ell \\ \hat{\mathbf{c}} \end{bmatrix} \leftarrow \mathsf{SamplePre}(\mathbf{D}_\ell, \mathbf{T}, \mathbf{t}, s).$$

Theorem 4.7 (Shifted Multi-Preimage Trapdoor Sampler)**.** *Let $n = n(\lambda, \ell)$, $q = q(\lambda, \ell)$ be arbitrary non-negative functions where $n \geq \lambda$. Let $m = 3n \lceil \log q \rceil$ and $t = m(\lceil \log \ell \rceil + 1)$. Then for all $s \geq (\ell t + m) \log(\ell n)$, Construction 4.6 is an (n, t, q, s)-shifted multi-preimage trapdoor sampler with perfect somewhere programmability. Moreover, Construction 4.6 is transparent (Definition 4.3), supports local expansion (Definition 4.4) and has simulatable openings (Definition 4.5). The size of the CRS output by $\mathsf{Gen}(1^\lambda, 1^\ell)$ is $nt \log q$.*

Proof. Take any polynomial $\ell = \ell(\lambda)$ and any $\lambda \in \mathbb{N}$. Take any $\mathsf{crs} = [\mathbf{A} \mid \mathbf{B}] \in \mathbb{Z}_q^{n \times t}$ where $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{B} \in \mathbb{Z}_q^{n \times m \lceil \log q \rceil}$. Let $(\mathbf{A}_1, \dots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(\mathsf{crs})$. Then, $\mathsf{td} = (\mathbf{D}_\ell, \mathbf{T})$ where $\mathbf{D}_\ell = [\mathrm{diag}(\mathbf{A}_1, \dots, \mathbf{A}_\ell) \mid \mathbf{1}^\ell \otimes \mathbf{G}]$ and $\mathbf{T}$ is defined as in Eq. (4.4). By construction, $\mathbf{A}_i = [\mathbf{A} \mid \mathbf{B} - \mathbf{u}_i^{\mathsf{T}} \otimes \mathbf{G}] = [\mathbf{A} \mid \mathbf{B}_i]$. By Eqs. (4.3) and (4.4) and Lemma 4.1, we have

$$\begin{aligned} \mathbf{D}_\ell \mathbf{T} &= \left[\begin{array}{ccc|ccc|c} \mathbf{A} & & & \mathbf{B}_1 & & & \mathbf{G} \\ & \ddots & & & \ddots & & \vdots \\ & & \mathbf{A} & & & \mathbf{B}_\ell & \mathbf{G} \end{array}\right] \cdot \mathbf{\Pi} \cdot \mathbf{\Pi}^{-1} \cdot \begin{bmatrix} \mathbf{0}^{\ell m \times \ell m} \\ \mathbf{T}' \end{bmatrix} \\ &= \left[\begin{array}{ccc|c} \mathbf{B} - \mathbf{u}_1^{\mathsf{T}} \otimes \mathbf{G} & & & \mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{B} - \mathbf{u}_\ell^{\mathsf{T}} \otimes \mathbf{G} & \mathbf{G} \end{array}\right] \mathbf{T}' = \begin{bmatrix} \mathbf{G} & & \\ & \ddots & \\ & & \mathbf{G} \end{bmatrix}. \end{aligned}$$

Thus, $\mathbf{T}$ is a gadget trapdoor for $\mathbf{D}_\ell$. Again by Lemma 4.1, $\|\mathbf{T}'\| = 1$, so $\|\mathbf{T}\| = 1$. We now show Construction 4.6 satisfies each of the required properties.

Correctness. Take any set of target vectors $\mathbf{t}_1, \dots, \mathbf{t}_\ell \in \mathbb{Z}_q^n$ and any $(\boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_\ell, \mathbf{c})$ in the support of the sampler $\mathsf{SampleMultPre}(\mathsf{td}, \mathbf{t}_1, \dots, \mathbf{t}_\ell)$. By construction, SampleMultPre first constructs $\mathbf{t} \in \mathbb{Z}_q^{\ell n}$ to be the vertical concatenation of $\mathbf{t}_1, \dots, \mathbf{t}_\ell$. Then it samples
$$\begin{bmatrix} \boldsymbol{\pi}_1 \\ \vdots \\ \boldsymbol{\pi}_\ell \\ \hat{\mathbf{c}} \end{bmatrix} \leftarrow \mathsf{SamplePre}(\mathbf{D}_\ell, \mathbf{T}, \mathbf{t}, s)$$
and sets $\mathbf{c} = -\mathbf{G}\hat{\mathbf{c}}$. Since $\mathbf{D}_\ell \mathbf{T} = \mathbf{I}_\ell \otimes \mathbf{G}$, the columns of $\mathbf{D}_\ell$ span $\mathbb{Z}_q^{\ell n}$. By Theorem 3.9, this means that for all $i \in [\ell]$,

$$\mathbf{A}_i \boldsymbol{\pi}_i + \mathbf{G}\hat{\mathbf{c}} = \mathbf{t}_i \implies \mathbf{A}_i \boldsymbol{\pi}_i = \mathbf{t}_i - \mathbf{G}\hat{\mathbf{c}} = \mathbf{t}_i + \mathbf{c}.$$

Preimage distribution. Take any set of target vectors $\mathbf{t}_1, \ldots, \mathbf{t}_\ell \in \mathbb{Z}_q^n$ and let $\mathbf{t} \in \mathbb{Z}_q^{\ell n}$ be the vertical concatenation of $\mathbf{t}_1, \ldots, \mathbf{t}_\ell$. We now define a sequence of intermediate distributions:

- $\mathcal{D}_0$: Output $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c}) \leftarrow \mathsf{SampleMultPre}(\mathsf{td}, \mathbf{t}_1, \ldots, \mathbf{t}_\ell)$. Specifically, sample $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \hat{\mathbf{c}})$ according to $\mathsf{SamplePre}(\mathbf{D}_\ell, \mathbf{T}, \mathbf{t}, s)$ and set $\mathbf{c} = -\mathbf{G}\hat{\mathbf{c}}$.
- $\mathcal{D}_1$: Sample $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \hat{\mathbf{c}})$ from $(\mathbf{D}_\ell)_s^{-1}(\mathbf{t})$. Set $\mathbf{c} = -\mathbf{G}\hat{\mathbf{c}}$ and output $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$.
- $\mathcal{D}_2$: Sample $\hat{\mathbf{c}} \leftarrow D_{\mathbb{Z},s}^m$, set $\mathbf{c} = -\mathbf{G}\hat{\mathbf{c}}$, and sample $\boldsymbol{\pi}_i \leftarrow (\mathbf{A}_i)_s^{-1}(\mathbf{t}_i + \mathbf{c})$ for each $i \in [\ell]$. Output $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$.
- $\mathcal{D}_3$: Sample $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and $\boldsymbol{\pi}_i \leftarrow (\mathbf{A}_i)_s^{-1}(\mathbf{t}_i + \mathbf{c})$ for each $i \in [\ell]$. Output $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$.

By definition, $\mathcal{D}_0$ and $\mathcal{D}_3$ corresponds to the two distributions in the preimage distribution property. We analyze each adjacent pair of distributions:

- First, $\mathcal{D}_0$ and $\mathcal{D}_1$ are statistically indistinguishable by Theorem 3.9. As argued above, $\mathbf{T}$ is a gadget trapdoor for $\mathbf{D}_\ell$ and $\|\mathbf{T}\| = 1$. Since $t = 3n \lceil \log q \rceil$, $s \geq (\ell t + m) \log(\ell n) = (\ell t + m) \|\mathbf{T}\| \cdot \log(\ell n)$, we appeal to Theorem 3.9 to conclude that the statistical distance between the distribution $\mathsf{SamplePre}(\mathbf{D}_\ell, \mathbf{T}, \mathbf{t}, s)$ and $(\mathbf{D}_\ell)_s^{-1}(\mathbf{t})$ is at most $2^{-\ell n} = \mathsf{negl}(\lambda)$. Thus, $\mathcal{D}_0$ and $\mathcal{D}_0'$ are statistically indistinguishable.
- Next, $\mathcal{D}_1$ and $\mathcal{D}_2$ are statistically indistinguishable by Lemma 3.6. To argue this, we show that with overwhelming probability over the choice of crs, all of the matrices $\mathbf{A}_i$ output by $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs})$ are full rank and satisfy $\lambda_1^\infty(\mathbf{A}_i) \geq q/4$. Consider the marginal distribution of each $\mathbf{A}_i$. By construction, $\mathbf{A}_i = [\mathbf{A} \mid \mathbf{B} - \mathbf{u}_i^\mathsf{T} \otimes \mathbf{G}]$ where $[\mathbf{A} \mid \mathbf{B}] \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times t}$. Thus, the marginal distribution of $\mathbf{A}_i$ is uniform over $\mathbb{Z}_q^{n \times t}$. By Lemmas 3.1 and 3.5, all but a negligible fraction of matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times t}$ are full rank and satisfy $\lambda_1^\infty(\Lambda(\mathbf{A})) \geq q/4$. Thus, for each $i \in [\ell]$, with overwhelming probability over the choice of crs, the associated matrix $\mathbf{A}_i$ is full rank. Since $\ell = \mathsf{poly}(\lambda)$, we use a union bound to argue that with overwhelming probability over the choice of crs, for all $i \in [\ell]$, it holds that $\mathbf{A}_i$ is full rank and $\lambda_1^\infty(\mathbf{A}_i) \geq q/4$. As long as $s \geq 4 \log(\ell t)$, the claim holds by Lemma 3.6.
- Finally, $\mathcal{D}_2$ and $\mathcal{D}_3$ are statistically indistinguishable by Lemma 3.3. In particular, when $s \geq \log m$, the distribution of $\mathbf{c} = -\mathbf{G}\hat{\mathbf{c}}$ when $\hat{\mathbf{c}} \leftarrow D_{\mathbb{Z},s}^m$ is statistically close to uniform.

By a hybrid argument, we conclude that with overwhelming probability over the choice of $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$, distributions $\mathcal{D}_0$ and $\mathcal{D}_3$ are statistically indistinguishable, as required.

Somewhere programmable. We define the GenProg algorithm as follows:

- $\mathsf{GenProg}(1^\lambda, 1^\ell, i, \mathbf{A}_i)$: On input the security parameter λ, the dimension ℓ, the index $i \in [\ell]$ and a matrix $\mathbf{A}_i = [\mathbf{A} \mid \mathbf{B}]$ where $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{B} \in \mathbb{Z}_q^{n \times m \lceil \log \ell \rceil}$, output $\widetilde{\mathsf{crs}} = [\mathbf{A} \mid \mathbf{B} + \mathbf{u}_i^\mathsf{T} \otimes \mathbf{G}]$.

We now show that GenProg satisfies the required properties. Take any polynomial $\ell = \ell(\lambda)$ and index $i \in [\ell]$. We consider each property individually:

- Take any $\lambda \in \mathbb{N}$ and any matrix $\mathbf{A}_i = [\mathbf{A} \mid \mathbf{B}] \in \mathbb{Z}_q^{n \times t}$. Let $\widetilde{\mathsf{crs}} \leftarrow \mathsf{GenProg}(1^\lambda, 1^\ell, i, \mathbf{A}_i)$ and consider $(\widetilde{\mathbf{A}}_1, \ldots, \widetilde{\mathbf{A}}_\ell, \mathsf{td}) = \mathsf{Expand}(\widetilde{\mathsf{crs}})$. Then $\widetilde{\mathsf{crs}} = [\mathbf{A} \mid \mathbf{B} + \mathbf{u}_i^\mathsf{T} \otimes \mathbf{G}]$. By definition of Expand, we now have
$$\widetilde{\mathbf{A}}_i = [\mathbf{A} \mid (\mathbf{B} + \mathbf{u}_i^\mathsf{T} \otimes \mathbf{G}) - \mathbf{u}_i^\mathsf{T} \otimes \mathbf{G}] = [\mathbf{A} \mid \mathbf{B}] = \mathbf{A}_i.$$
- Suppose $[\mathbf{A} \mid \mathbf{B}] \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$. By construction, the Gen algorithm samples $[\mathbf{A} \mid \mathbf{B}] \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times t}$. We claim that this matches the distribution output by $\mathsf{GenProg}(1^\lambda, 1^\ell, i, \mathbf{A}_i)$ when $\mathbf{A}_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times t}$. If $\mathbf{A}_i = [\mathbf{A}' \mid \mathbf{B}']$, then $\mathsf{GenProg}(1^\lambda, 1^\ell, i, \mathbf{A}_i)$ outputs $[\mathbf{A}' \mid \mathbf{B}' + \mathbf{u}_i^\mathsf{T} \otimes \mathbf{G}]$, which is still distributed uniformly over $\mathbb{Z}_q^{n \times t}$. Thus, these two distributions are identical.

We conclude that Construction 4.6 satisfies perfect somewhere programmability.

Transparent setup. The common reference string output by Gen consists of a uniform random matrix $[\mathbf{A} \mid \mathbf{B}] \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times t}$, so the scheme has a transparent setup by construction.

Local expansion. The local expansion property follows by construction of Expand. Namely, we can define $\mathsf{ExpandLocal}(1^\lambda, \mathsf{crs}, i)$ to output $\mathbf{A}_i = [\mathbf{A} \mid \mathbf{B} - \mathbf{u}_i^{\mathsf{T}} \otimes \mathbf{G}] \in \mathbb{Z}_q^{n \times t}$, where $\mathsf{crs} = [\mathbf{A} \mid \mathbf{B}]$.

Simulatable openings. We define the GenTD algorithm as follows:

- $\mathsf{GenTD}(1^\lambda, 1^\ell)$: On input the security parameter λ and the dimension ℓ, the trapdoor generator algorithm samples $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, m)$ and $\mathbf{B} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m\lceil \log \ell \rceil}$. It sets $\mathsf{crs} = [\mathbf{A} \mid \mathbf{B}]$ and for each $i \in [\ell]$, it sets $\mathbf{T}_i = \left[\begin{smallmatrix} \mathbf{T} \\ \mathbf{0} \end{smallmatrix}\right] \in \mathbb{Z}_q^{t \times m}$. Finally, it outputs $(\mathsf{crs}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell)$.

We now show that GenTD satisfies mode indistinguishability and the trapdoor generation properties:

- **Mode indistinguishability:** Consider the distribution of $(\mathsf{crs}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell)$ output by $\mathsf{GenTD}(1^\lambda, 1^\ell)$. By construction, $\mathsf{crs} = [\mathbf{A} \mid \mathbf{B}]$ where $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, m)$ and $\mathbf{B} \xleftarrow{\text{R}} \mathbb{Z}_q^{m\lceil \log \ell \rceil \times m}$. Since $n \geq \lambda$ and $m = 3n \lceil \log q \rceil$, we appeal to Theorem 3.9 to conclude that the distribution of $\mathbf{A}$ is statistically close to uniform. Correspondingly, this means $[\mathbf{A} \mid \mathbf{B}]$ is statistically close to uniform over $\mathbb{Z}_q^{n \times t}$. This is the distribution output by $\mathsf{Gen}(1^\lambda, 1^\ell)$.

- **Trapdoor generation:** Take any $\lambda, \ell \in \mathbb{N}$ and consider $(\mathsf{crs}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell)$ output by $\mathsf{GenTD}(1^\lambda, 1^\ell)$. Let $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs})$. By construction, $\mathsf{crs} = [\mathbf{A} \mid \mathbf{B}]$ where $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, m)$ and $\mathbf{T}_i = \left[\begin{smallmatrix} \mathbf{T} \\ \mathbf{0} \end{smallmatrix}\right]$. Since $m \geq 3n \lceil \log q \rceil$, by Theorem 3.9, this means $\mathbf{A}\mathbf{T} = \mathbf{G}$. Moreover, by definition of Expand, $\mathbf{A}_i = [\mathbf{A} \mid \mathbf{B} - \mathbf{u}_i \otimes \mathbf{G}]$. Thus,

$$\mathbf{A}_i \mathbf{T}_i = [\mathbf{A} \mid \mathbf{B} - \mathbf{u}_i \otimes \mathbf{G}] \begin{bmatrix} \mathbf{T} \\ \mathbf{0}^{m\lceil \log \ell \rceil \times m} \end{bmatrix} = \mathbf{A}\mathbf{T} = \mathbf{G}.$$

  Again by Theorem 3.9, $\|\mathbf{T}\| = 1$ so $\|\mathbf{T}_i\| = 1$. Since $s \geq (\ell t + m) \log(\ell n)$, we have $\|\mathbf{T}_i\| = 1 \leq s/(t \log n)$. □

Marginal distribution of matrices output by the shifted multi-preimage trapdoor sampler. The somewhere programmability requirement of a shifted multi-preimage trapdoor sampler ensures that the marginal distribution of each $\mathbf{A}_i$ obtained by running $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$ and $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs})$ is statistically close to uniform. This will be useful in our applications, so we give the formal statement below:

Lemma 4.8 (Marginal Distribution of $\mathbf{A}_i$)**.** *Let λ be a security parameter and ℓ be a dimension. Suppose* (Gen, Expand, SampleMultPre) *is an (n, t, q, s)-shifted multi-preimage trapdoor sampler. Then, for all polynomials $\ell = \ell(\lambda)$, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all indices $i \in [\ell]$ and all $\lambda \in \mathbb{N}$, the statistical distance between the following distributions is* $\mathsf{negl}(\lambda)$*:*

$$\left\{ \mathbf{A}_i : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell) \\ (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}) \end{array} \right\} \quad \text{and} \quad \left\{ \mathbf{A}_i : \mathbf{A}_i \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times t} \right\}.$$

Proof. Follows immediately by somewhere programmability. □

5 Dual-Mode Hidden-Bits Model NIZK from LWE

In this section, we show how to use the shifted multi-preimage trapdoor sampler from Section 4 to construct a dual-mode hidden-bits generator. Our construction improves upon the previous dual-mode hidden-bits model NIZK of Waters [Wat24] in several key dimensions: (1) the size of the CRS in our scheme is linear in the length of the hidden-bits string as opposed to quadratic; (2) in statistically-hiding mode, the CRS in our scheme is a common *random* string as opposed to a structured reference string; (3) security relies on the LWE assumption with a *polynomial* modulus-to-noise ratio as opposed to a sub-exponential one.

Dual-mode hidden-bits generator. We start by recalling the notion of a dual-mode hidden-bits generator [QRW19, LPWW20]. In the definition, we use the statistical single-bit hiding property from [Wat24], which is sufficient for the application to NIZKs for NP. The single-bit security notion is simpler to analyze, so we focus on it for ease of exposition.

Definition 5.1 (Dual-Mode Hidden-Bits Generator [QRW19, LPWW20, Wat24, adapted]). A dual-mode hidden-bits generator is a tuple of efficient algorithms $\Pi_{\mathsf{HBG}} = (\mathsf{Setup}, \mathsf{GenBits}, \mathsf{Verify})$ with the following syntax:

- $\mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{mode}) \to \mathsf{crs}$: On input the security parameter λ, the output length ℓ, and $\mathsf{mode} \in \{\mathsf{binding}, \mathsf{hiding}\}$, the setup algorithm outputs a common reference string crs.
- $\mathsf{GenBits}(\mathsf{crs}) \to (\sigma, \mathbf{r}, (\pi_1, \ldots, \pi_\ell))$: On input the common reference string crs, the generator algorithm outputs a commitment σ, a bit-string $\mathbf{r} \in \{0,1\}^\ell$, and a tuple of proofs $\pi_1, \ldots, \pi_\ell$.
- $\mathsf{Verify}(\mathsf{crs}, \sigma, i, \beta, \pi) \to b$: On input the common reference string crs, a commitment σ, an index i, a bit $\beta \in \{0,1\}$, and a proof π, the verification algorithm outputs a bit $b \in \{0,1\}$.

Moreover, we require that Π_{HBG} satisfy the following properties:

- **Correctness:** For all $\lambda, \ell \in \mathbb{N}$, all modes $\mathsf{mode} \in \{\mathsf{binding}, \mathsf{hiding}\}$, and all indices $i \in [\ell]$, we have that

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, \sigma, i, r_i, \pi_i) = 1 : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{mode}) \\ (\sigma, \mathbf{r}, (\pi_1, \ldots, \pi_\ell)) \leftarrow \mathsf{GenBits}(\mathsf{crs}) \end{array}\right] = 1.$$

- **Mode indistinguishability:** For an adversary $\mathcal{A}$, an output length ℓ, and a bit $b \in \{0,1\}$, we define the mode-indistinguishability game as follows:

  1. If $b = 0$, the challenger sets $\mathsf{mode} = \mathsf{binding}$. If $b = 1$, the challenger sets $\mathsf{mode} = \mathsf{hiding}$. The challenger samples $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{mode})$ and gives $(1^\lambda, 1^\ell, \mathsf{crs})$ to $\mathcal{A}$.
  2. Algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.

  The hidden-bits generator satisfies mode indistinguishability if for all efficient adversaries $\mathcal{A}$ and all polynomials $\ell = \ell(\lambda)$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$|\Pr[b' = 1 : b = 0] - \Pr[b' = 1 : b = 1]| = \mathsf{negl}(\lambda)$$

  in the mode-indistinguishability game.

- **Succinctness:** There exists a fixed polynomial $p(\cdot,\cdot)$ such that for all $\lambda, \ell \in \mathbb{N}$, all $\mathsf{mode} \in \{\mathsf{binding}, \mathsf{hiding}\}$, all crs in the support of $\mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{mode})$, and all commitments σ in the support of $\mathsf{GenBits}(\mathsf{crs})$, we have that $|\sigma| \le p(\lambda, \log \ell)$.

- **Statistically binding in binding mode:** For all polynomials $\ell = \ell(\lambda)$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{c} \exists(\sigma, i, \pi_0, \pi_1) : \\ \mathsf{Verify}(\mathsf{crs}, \sigma, i, 0, \pi_0) = 1 = \mathsf{Verify}(\mathsf{crs}, \sigma, i, 1, \pi_1) \end{array} : \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{binding})\right] = \mathsf{negl}(\lambda).$$

- **Single-bit statistical hiding in hiding mode:** For an adversary $\mathcal{A}$, an output length ℓ, and a bit $b \in \{0,1\}$, we define the hiding game as follows:

  1. On input the security parameter 1^λ and the length parameter 1^ℓ, algorithm $\mathcal{A}$ outputs an index $i^* \in [\ell]$.
  2. The challenger samples $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{hiding})$ and $(\sigma, \mathbf{r}, (\pi_1, \ldots, \pi_\ell)) \leftarrow \mathsf{GenBits}(\mathsf{crs})$. If $b = 0$, the challenger sets $\beta = r_{i^*}$. If $b = 1$, it samples $\beta \xleftarrow{\mathrm{R}} \{0,1\}$.
  3. The challenger gives $\left(\mathsf{crs}, \sigma, \{(i, r_i, \pi_i)\}_{i \neq i^*}, \beta\right)$ to $\mathcal{A}$.
  4. Algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.

The hidden-bits generator satisfies statistical hiding in hiding mode if for all (computationally-unbounded) adversaries $\mathcal{A}$ and all polynomials $\ell = \ell(\lambda)$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$|\Pr[b' = 1 : b = 0] - \Pr[b' = 1 : b = 1]| = \mathsf{negl}(\lambda)$$

in the hiding game.

Theorem 5.2 (NIZKs from Hidden-Bits Generators [FLS90, QRW19, LPWW20, Wat24])**.** *If there exists a dual-mode hidden-bits generator, then there exists a dual-mode NIZK for* NP.

Constructing a dual-mode hidden-bits generator. We now show how to use a shifted multi-preimage trapdoor sampler to construct a dual-mode hidden-bits model generator (Definition 4.2). We refer to Section 2.2 for an overview of the construction. As discussed in Section 2.2, our construction shares a similar structure with the scheme of Waters [Wat24], but differs in a few key respects:

- **CRS structure:** The [Wat24] construction publish structured preimages in the CRS and the commitment and the openings are derived by computing short linear combinations of the components in the CRS. Because the CRS contains structured preimages (in both modes), the [Wat24] construction relied on a (quadratic-size) structured CRS in both modes. In our scheme, we replace the structured preimages in the CRS with our shifted multi-preimage trapdoor sampler. This allows us to achieve a linear-size CRS, and moreover in hiding mode, the CRS is a uniform random string (i.e., supports transparent setup).
- **Binding analysis:** In the [Wat24] scheme, the verification algorithm checks that the opening does not land near a "rounding boundary" (and rejects if the opening is too close). This ensures that at every index, a given commitment can only be opened one way. This is useful for arguing binding. In [Wat24], the modulus q is super-polynomial so the probability that an opening lands near a rounding boundary is negligible. In our construction, we rely on a polynomial modulus q, so there is an inverse polynomial probability that GenBits samples a commitment and a set of openings where one of the opening lands inside the rounding boundary (and causes the verification algorithm to fail). However, since the verification algorithm is public, the GenBits algorithm can simply resample a commitment and set of openings whenever this happens. By choosing q to be a sufficiently-large polynomial, we ensure that each sampling attempt succeeds with at least constant probability. After λ attempts, the algorithm will sample a commitment and a set of valid openings with overwhelming probability. Thus, we can avoid the super-polynomial modulus q in the binding analysis.
- **Hiding analysis:** The [Wat24] also critically relies on a super-polynomial modulus q for the hiding analysis. Specifically, when the CRS is sampled in hiding mode, [Wat24] first establishes that there exists small perturbations that can be added to a commitment and flip the i^{th} output bit of the hidden-bits string, while leaving all remaining bits unchanged. Then, using a noise smudging argument, [Wat24] argues that the adversary cannot tell whether a commitment is "normal" or "perturbed." This suffices to show that the i^{th} bit is statistically hidden from the view of the adversary. In our work, we take a different approach. To argue hiding, we rely on the fact that Gaussian preimages have sufficient min-entropy and use this to extract a string of uniform random bits. This avoids the need for noise smudging and allows us to use a polynomial modulus for the overall construction. This in turn allows us to prove security from LWE with a polynomial modulus-to-noise ratio.

We now describe our construction.

Construction 5.3 (Dual-Mode Hidden-Bits Generator)**.** Let λ be a security parameter and ℓ be a length parameter. Let $\Pi_{\mathsf{samp}} = (\mathsf{Gen}, \mathsf{Expand}, \mathsf{SampleMultPre})$ be a $(n, t, q, s_{\mathsf{samp}})$-shifted multi-preimage trapdoor sampler. Let $s_{\mathsf{LWE}} = s_{\mathsf{LWE}}(\lambda, \ell)$ be a Gaussian width parameter and $B_{\max} = B_{\max}(\lambda, \ell)$, $B_{\mathsf{round}} = B_{\mathsf{round}}(\lambda, \ell)$ be bounds. Throughout, we assume that $B_{\mathsf{round}} < q/4$. We construct our dual-mode hidden bits generator $\Pi_{\mathsf{HBG}} = (\mathsf{Setup}, \mathsf{GenBits}, \mathsf{Verify})$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{mode})$: On input the security parameter λ, the output length ℓ, and $\mathsf{mode} \in \{\mathsf{binding}, \mathsf{hiding}\}$, the setup algorithm samples $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$. Next, it computes $(\mathbf{A}_1, \dots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$, where $\mathbf{A}_i \in \mathbb{Z}_q^{n \times t}$. Next, for each $i \in [\ell]$, it samples a vector $\mathbf{v}_i \in \mathbb{Z}_q^t$ as follows:

  - If mode = binding, it samples $\mathbf{s}_i \xleftarrow{R} \mathbb{Z}_q^n$, $\mathbf{e}_i \leftarrow D_{\mathbb{Z},s_{\mathsf{LWE}}}^t$, and sets $\mathbf{v}_i^{\mathsf{T}} = \mathbf{s}_i^{\mathsf{T}}\mathbf{A}_i + \mathbf{e}_i^{\mathsf{T}}$.
  - If mode = hiding, it samples $\mathbf{v}_i \xleftarrow{R} \mathbb{Z}_q^t$.

  It outputs $\mathsf{crs} = (1^\lambda, \mathsf{crs}_{\mathsf{samp}}, \mathbf{v}_1, \ldots, \mathbf{v}_\ell)$.

- $\mathsf{GenBits}(\mathsf{crs})$: On input the common reference string $\mathsf{crs} = (1^\lambda, \mathsf{crs}_{\mathsf{samp}}, \mathbf{v}_1, \ldots, \mathbf{v}_\ell)$, the generator algorithm first computes $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$. Then, it repeats the following procedure up to λ times:

  - Sample $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c}) \leftarrow \mathsf{SampleMultPre}(\mathsf{td}, \mathbf{0}^n, \ldots, \mathbf{0}^n)$.
  - For each $i \in [\ell]$, if $\|\boldsymbol{\pi}_i\| > B_{\max}$, it sets $r_i = \bot$. Otherwise, compute $u_i = \mathbf{v}_i^{\mathsf{T}}\boldsymbol{\pi}_i$ and set r_i as follows:

  $$r_i = \begin{cases} 0 & u_i \in [-B_{\mathsf{round}}, B_{\mathsf{round}}] \\ 1 & u_i \in [\lfloor q/2 \rfloor - B_{\mathsf{round}}, \lfloor q/2 \rfloor + B_{\mathsf{round}}] \\ \bot & \text{otherwise.} \end{cases}$$

    Note that the intervals $[-B_{\mathsf{round}}, B_{\mathsf{round}}]$ and $[\lfloor q/2 \rfloor - B_{\mathsf{round}}, \lfloor q/2 \rfloor + B_{\mathsf{round}}]$ are guaranteed to be disjoint when $B_{\mathsf{round}} < q/4$.
  - If $r_i \in \{0, 1\}$ for all $i \in [\ell]$, then it outputs $(\mathbf{c}, \mathbf{r}, (\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell))$. Otherwise, if there exists an index $i \in [\ell]$ where $r_i = \bot$, the generator algorithm restarts the sampling procedure.

  If the sampling procedure does not succeed after λ attempts, then the generator algorithm sets $\mathbf{c} = \bot$, $\mathbf{r} = \mathbf{0}^\ell$, and $\boldsymbol{\pi}_i = \bot$ for all $i \in [\ell]$. It outputs $(\mathbf{c}, \mathbf{0}^\ell, (\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell))$.

- $\mathsf{Verify}(\mathsf{crs}, \mathbf{c}, i, \beta, \boldsymbol{\pi})$: On input the common reference string $\mathsf{crs} = (1^\lambda, \mathsf{crs}_{\mathsf{samp}}, \mathbf{v}_1, \ldots, \mathbf{v}_\ell)$, a commitment $\mathbf{c}$, an index $i \in [\ell]$, a bit $\beta \in \{0, 1\}$, and a proof $\boldsymbol{\pi}$, the verification algorithm proceeds as follows:

  - If $\mathbf{c} = \bot$ output 1 if $\beta = 0$ and 0 otherwise.
  - If $\mathbf{c} \in \mathbb{Z}_q^n$ and $\boldsymbol{\pi} \in \mathbb{Z}_q^t$, then compute $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$. Output 1 if

  $$\|\boldsymbol{\pi}\| \le B_{\max} \quad \text{and} \quad \mathbf{A}_i\boldsymbol{\pi} = \mathbf{c} \quad \text{and} \quad \mathbf{v}_i^{\mathsf{T}}\boldsymbol{\pi}_i \in [\lfloor q/2 \rfloor \beta - B_{\mathsf{round}}, \lfloor q/2 \rfloor \beta + B_{\mathsf{round}}]$$

    and 0 otherwise.

  In all other cases, output 0.

Theorem 5.4 (Correctness). *If Π_{samp} is correct, then Construction 5.3 is correct.*

Proof. Take any $\lambda, \ell \in \mathbb{N}$ and mode $\in$ {binding, hiding}. Suppose $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{mode})$ and $(\mathbf{c}, \mathbf{r}, (\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell)) \leftarrow \mathsf{GenBits}(\mathsf{crs})$. Then, we can write $\mathsf{crs} = (1^\lambda, \mathsf{crs}_{\mathsf{samp}}, \mathbf{v}_1, \ldots, \mathbf{v}_\ell)$. Let $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$. We consider two possibilities:

- Suppose $\mathbf{c} = \bot$. By construction of GenBits, this means $\mathbf{r} = \mathbf{0}^\ell$. Then $\mathsf{Verify}(\mathsf{crs}, \mathbf{c}, i, 0, \boldsymbol{\pi}_i)$ outputs 1 by construction.

- Suppose $\mathbf{c} \in \mathbb{Z}_q^n$. This means the GenBits algorithm sampled $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c}) \leftarrow \mathsf{SampleMultPre}(\mathsf{td}, \mathbf{0}^n, \ldots, \mathbf{0}^n)$. By correctness of Π_{samp}, this means $\mathbf{A}_i\boldsymbol{\pi}_i = \mathbf{c}$ for all $i \in [\ell]$. Finally, GenBits outputs $\mathbf{c}$ only if

$$\|\boldsymbol{\pi}_i\| \le B_{\max} \quad \text{and} \quad \mathbf{v}_i^{\mathsf{T}}\boldsymbol{\pi}_i \in [\lfloor q/2 \rfloor r_i - B_{\mathsf{round}}, \lfloor q/2 \rfloor r_i + B_{\mathsf{round}}].$$

  In this case, all of the verification checks pass and $\mathsf{Verify}(\mathsf{crs}, \mathbf{c}, i, r_i, \boldsymbol{\pi}_i) = 1$. □

Theorem 5.5 (Mode Indistinguishability). *Suppose Π_{samp} satisfies somewhere programmability. Then, under the $\mathsf{LWE}_{n,t,q,s_{\mathsf{LWE}}}$ assumption, Construction 5.3 satisfies mode indistinguishability.*

Proof. Let $\mathcal{A}$ be an efficient adversary for the mode indistinguishability game. We being by defining a sequence of hybrid experiments parameterized by an index $i \in \{0, \ldots, \ell\}$:

- Hyb_i: In this experiment, the challenger samples $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$ and computes $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$. Then, for each $j \in [\ell]$, it constructs the vector $\mathbf{v}_j \in \mathbb{Z}_q^t$ as follows:
  - If $j > i$, sample $\mathbf{s}_j \xleftarrow{\text{R}} \mathbb{Z}_q^n$, $\mathbf{e}_i \leftarrow D_{\mathbb{Z}, s_{\mathsf{LWE}}}^t$, and compute $\mathbf{v}_j^{\mathsf{T}} = \mathbf{s}_j^{\mathsf{T}} \mathbf{A}_j + \mathbf{e}_j^{\mathsf{T}}$.
  - If $j \le i$, sample $\mathbf{v}_j \xleftarrow{\text{R}} \mathbb{Z}_q^t$.

  The challenger gives $\mathsf{crs} = (1^\lambda, \mathsf{crs}_{\mathsf{samp}}, \mathbf{v}_1, \ldots, \mathbf{v}_\ell)$ to $\mathcal{A}$. The output of the experiment is the output of $\mathcal{A}$.

We write $\mathsf{Hyb}_i(\mathcal{A})$ to denote the output of an execution of Hyb_i with adversary $\mathcal{A}$. By construction, $\mathsf{Hyb}_0(\mathcal{A})$ corresponds to the experiment where the challenger samples $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{binding})$ while $\mathsf{Hyb}_\ell(\mathcal{A})$ corresponds to the experiment where the challenger samples $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{hiding})$. To complete the proof, we show that for all $i \in [\ell]$, the output distributions $\mathsf{Hyb}_{i-1}(\mathcal{A})$ and $\mathsf{Hyb}_i(\mathcal{A})$ are computationally indistinguishable. To do so, we introduce two intermediate hybrids:

- $\mathsf{Hyb}_{i,1}$: Same as Hyb_i, except the challenger samples $\mathbf{A}_i \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times t}$ and $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{GenProg}(1^\lambda, 1^\ell, i, \mathbf{A}_i)$.
- $\mathsf{Hyb}_{i,2}$: Same as $\mathsf{Hyb}_{i,1}$, except the challenger samples $\mathbf{v}_j \xleftarrow{\text{R}} \mathbb{Z}_q^t$.

We now show that each adjacent pair of hybrid distributions are indistinguishable.

Lemma 5.6. *If Π_{samp} is somewhere programmable, then* $\mathsf{Hyb}_i(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_{i,1}(\mathcal{A})$.

Proof. The only difference between Hyb_i and $\mathsf{Hyb}_{i,1}$ is the distribution of $\mathsf{crs}_{\mathsf{samp}}$. In Hyb_i, the challenger samples $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$ while in $\mathsf{Hyb}_{i,1}$, the challenger samples $\mathbf{A}_i \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times t}$ and $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{GenProg}(1^\lambda, 1^\ell, i, \mathbf{A}_i)$. These two distributions are statistically indistinguishable by somewhere programmability. □

Lemma 5.7. *Suppose Π_{samp} is somewhere programmable. Then, under the $\mathsf{LWE}_{n,t,q,s_{\mathsf{LWE}}}$ assumption, for all $i \in [\ell]$,* $\mathsf{Hyb}_{i,1}(\mathcal{A}) \overset{c}{\approx} \mathsf{Hyb}_{i,2}(\mathcal{A})$.

Proof. Suppose $|\Pr[\mathsf{Hyb}_{i,1}(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_{i,2}(\mathcal{A}) = 1]| = \varepsilon(\lambda)$ for some non-negligible ε. We use $\mathcal{A}$ to construct an efficient adversary $\mathcal{B}$ for the $\mathsf{LWE}_{n,t,q,s_{\mathsf{LWE}}}$ assumption:

- On input the LWE challenge $(\mathbf{A}, \mathbf{u}^{\mathsf{T}})$ where $\mathbf{A} \in \mathbb{Z}_q^{n \times t}$ and $\mathbf{u} \in \mathbb{Z}_q^t$, algorithm $\mathcal{B}$ computes $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{GenProg}(1^\lambda, 1^\ell, i, \mathbf{A})$. Then, it computes $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$. For each $j \in [\ell]$, the challenger computes the vector $\mathbf{v}_j$ as follows:
  - If $j < i$, sample $\mathbf{v}_j \xleftarrow{\text{R}} \mathbb{Z}_q^t$.
  - If $j = i$, set $\mathbf{v}_j = \mathbf{u}$.
  - If $j > i$, sample $\mathbf{s}_j \xleftarrow{\text{R}} \mathbb{Z}_q^n$, $\mathbf{e}_j \leftarrow D_{\mathbb{Z}, s_{\mathsf{LWE}}}^t$, and compute $\mathbf{v}_j^{\mathsf{T}} = \mathbf{s}_j^{\mathsf{T}} \mathbf{A}_j + \mathbf{e}_j^{\mathsf{T}}$.
- Algorithm $\mathcal{B}$ gives $\mathsf{crs} = (1^\lambda, \mathsf{crs}_{\mathsf{samp}}, \mathbf{v}_1, \ldots, \mathbf{v}_\ell)$ to $\mathcal{A}$ and outputs whatever $\mathcal{A}$ outputs.

We now analyze the advantage of algorithm $\mathcal{B}$. First, the LWE challenger samples $\mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times t}$, so the distribution of $\mathsf{crs}_{\mathsf{samp}}$ is identical to the distribution in $\mathsf{Hyb}_{i,1}$ and $\mathsf{Hyb}_{i,2}$. By somewhere programmability, we also have that $\mathbf{A}_i = \mathbf{A}$. Consider now the distribution of $\mathbf{u}$:

- Suppose $\mathbf{u}^{\mathsf{T}} = \mathbf{s}^{\mathsf{T}} \mathbf{A} + \mathbf{e}^{\mathsf{T}}$ where $\mathbf{s} \xleftarrow{\text{R}} \mathbb{Z}_q^n$ and $\mathbf{e} \leftarrow D_{\mathbb{Z}, s_{\mathsf{LWE}}}^t$. Since $\mathbf{A}_i = \mathbf{A}$, we have $\mathbf{v}_i^{\mathsf{T}} = \mathbf{u}^{\mathsf{T}} = \mathbf{s}^{\mathsf{T}} \mathbf{A}_i + \mathbf{e}^{\mathsf{T}}$, which matches the distribution in $\mathsf{Hyb}_{i,1}$.
- Suppose $\mathbf{u} \xleftarrow{\text{R}} \mathbb{Z}_q^t$. Then, $\mathbf{v}_i$ is uniform over $\mathbb{Z}_q^t$, which matches the distribution in $\mathsf{Hyb}_{i,2}$.

Algorithm $\mathcal{B}$ breaks $\mathsf{LWE}_{n,t,q,s_{\mathsf{LWE}}}$ with the same non-negligible advantage ε and the lemma follows. □

Lemma 5.8. *If Π_{samp} is somewhere programmable, then* $\mathsf{Hyb}_{i,2}(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_{i+1}(\mathcal{A})$.

Proof. The only difference between $\mathsf{Hyb}_{i,2}$ and Hyb_{i+1} is the distribution of $\mathsf{crs}_{\mathsf{samp}}$. In $\mathsf{Hyb}_{i,2}$, the challenger samples $\mathbf{A}_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n\times t}$ and $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{GenProg}(1^\lambda, 1^\ell, i, \mathbf{A}_i)$ whereas in Hyb_i, the challenger samples $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$. These two distributions are statistically indistinguishable by somewhere programmability. □

Mode indistinguishability now follows from Lemmas 5.6 to 5.8. □

Theorem 5.9 (Statistical Binding in Binding Mode)**.** *Suppose $q > 4t\sqrt{\lambda}s_{\mathsf{LWE}}B_{\max} + 4B_{\mathsf{round}}$. Then, Construction 5.3 is statistically binding in binding mode.*

Proof. Take a security parameter $\lambda \in \mathbb{N}$ and any polynomial $\ell = \ell(\lambda)$. Let $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{binding})$. We parse $\mathsf{crs} = (1^\lambda, \mathsf{crs}_{\mathsf{samp}}, \mathbf{v}_1, \ldots, \mathbf{v}_\ell)$ and let $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$. In binding mode, for all $i \in [\ell]$, $\mathbf{v}_i^{\mathsf{T}} = \mathbf{s}_i^{\mathsf{T}}\mathbf{A}_i + \mathbf{e}_i^{\mathsf{T}}$, where $\mathbf{e}_i \leftarrow D^t_{\mathbb{Z},s_{\mathsf{LWE}}}$. By Lemma 3.2, $\|\mathbf{e}_i\| \le \sqrt{\lambda}s_{\mathsf{LWE}}$ with probability $1 - t\cdot 2^{-\lambda}$. By a union bound, with probability $1 - t\ell \cdot 2^{-\lambda} = 1 - \mathsf{negl}(\lambda)$, the following holds:

$$\forall i \in [\ell] : \|\mathbf{e}_i\| \le \sqrt{\lambda}s_{\mathsf{LWE}}. \tag{5.1}$$

Suppose now that there exists a tuple $(\mathbf{c}, i, \boldsymbol{\pi}_0, \boldsymbol{\pi}_1)$ where

$$\mathsf{Verify}(\mathsf{crs}, \mathbf{c}, i, 0, \boldsymbol{\pi}_0) = 1 = \mathsf{Verify}(\mathsf{crs}, \mathbf{c}, i, 1, \boldsymbol{\pi}_1). \tag{5.2}$$

We now consider two possibilities:

- Suppose that $\mathbf{c} = \perp$. Then, by construction, $\mathsf{Verify}(\mathsf{crs}, \mathbf{c}, i, 1, \boldsymbol{\pi}_1)$ outputs 0 which contradicts Eq. (5.2).
- Suppose $\mathbf{c} \in \mathbb{Z}_q^n$. By Eq. (5.2) and construction of Verify, the following conditions also hold:

  $$\|\boldsymbol{\pi}_0\|, \|\boldsymbol{\pi}_1\| \le B_{\max} \quad \text{and} \quad \mathbf{A}_i\boldsymbol{\pi}_0 = \mathbf{c} = \mathbf{A}_i\boldsymbol{\pi}_1,$$

  and in addition,

  $$\begin{aligned} &\mathbf{v}_i^{\mathsf{T}}\boldsymbol{\pi}_0 \in [-B_{\mathsf{round}}, B_{\mathsf{round}}] \\ &\mathbf{v}_i^{\mathsf{T}}\boldsymbol{\pi}_1 \in [\lfloor q/2 \rfloor - B_{\mathsf{round}}, \lfloor q/2 \rfloor + B_{\mathsf{round}}]. \end{aligned}$$

  In particular, this means that

  $$|\mathbf{v}_i^{\mathsf{T}}(\boldsymbol{\pi}_0 - \boldsymbol{\pi}_1)| \ge \lfloor q/2 \rfloor - 2B_{\mathsf{round}}. \tag{5.3}$$

  Since $\mathbf{A}_i\boldsymbol{\pi}_0 = \mathbf{A}_i\boldsymbol{\pi}_1$, we have

  $$\mathbf{v}_i^{\mathsf{T}}(\boldsymbol{\pi}_0 - \boldsymbol{\pi}_1) = \mathbf{s}_i^{\mathsf{T}}(\mathbf{A}_i\boldsymbol{\pi}_0 - \mathbf{A}_i\boldsymbol{\pi}_1) + \mathbf{e}_i^{\mathsf{T}}(\boldsymbol{\pi}_0 - \boldsymbol{\pi}_1) = \mathbf{e}_i^{\mathsf{T}}(\boldsymbol{\pi}_0 - \boldsymbol{\pi}_1).$$

  Since $\|\boldsymbol{\pi}_0\|, \|\boldsymbol{\pi}_1\| \le B_{\max}$, and $\|\mathbf{e}_i\| \le \sqrt{\lambda}s_{\mathsf{LWE}}$ from Eq. (5.1), we conclude that

  $$|\mathbf{v}_i^{\mathsf{T}}(\boldsymbol{\pi}_0 - \boldsymbol{\pi}_1)| = |\mathbf{e}_i^{\mathsf{T}}(\boldsymbol{\pi}_0 - \boldsymbol{\pi}_1)| \le 2t\sqrt{\lambda}s_{\mathsf{LWE}}B_{\max}.$$

  However, this contradicts Eq. (5.3) whenever $q > 4t\sqrt{\lambda}s_{\mathsf{LWE}}B_{\max} + 4B_{\mathsf{round}}$.

We conclude that no such tuple $(\mathbf{c}, i, \boldsymbol{\pi}_0, \boldsymbol{\pi}_1)$ can exist when Eq. (5.1) holds. Since Eq. (5.1) holds with $1 - \mathsf{negl}(\lambda)$ over the randomness of Setup, the theorem follows. □

Theorem 5.10 (Single-Bit Statistical Hiding)**.** *Suppose Π_{samp} satisfies the preimage distribution property and $n \ge 4\lambda + 2\log q$, $t \ge 3n\lceil \log q \rceil$, q is prime, $q \ge 4B_{\mathsf{round}} + 2$, $s_{\mathsf{samp}} \ge \log t$, $B_{\max} \ge \sqrt{t}s_{\mathsf{samp}}$, and $B_{\mathsf{round}} \ge q/4 - q/(8\ell) + 1/2$. Then, Construction 5.3 satisfies single-bit statistical hiding in hiding mode.*

Proof. Let $\mathcal{A}$ be a distinguisher for the hiding game. We start by defining a sequence of hybrid experiments:

- $\mathsf{Hyb}_0^{(b)}$: This is the hiding experiment with the bit b. Specifically, the adversary (on input 1^λ and 1^ℓ) starts by outputting an index $i^* \in [\ell]$. The challenger then samples $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, \mathsf{hiding})$. Namely, the challenger samples $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$ and for each $i \in [\ell]$, $\mathbf{v}_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q^t$. It sets $\mathsf{crs} = (1^\lambda, \mathsf{crs}_{\mathsf{samp}}, \mathbf{v}_1, \ldots, \mathbf{v}_\ell)$. Next, for each $d \in [\lambda]$, the challenger proceeds as follows:[4]
  - Sample $(\boldsymbol{\pi}_{d,1}, \ldots, \boldsymbol{\pi}_{d,\ell}, \mathbf{c}_d) \leftarrow \mathsf{SampleMultPre}(\mathsf{td}, \mathbf{0}^n, \ldots, \mathbf{0}^n)$.
  - For each $i \in [\ell]$, the challenger computes $u_{d,i} = \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_{d,i}$. If $\|\boldsymbol{\pi}_{d,i}\| > B_{\max}$, set $r_{d,i} = \bot$. Otherwise, set

$$r_{d,i} = \begin{cases} 0 & u_{d,i} \in [-B_{\mathsf{round}}, B_{\mathsf{round}}] \\ 1 & u_{d,i} \in [\lfloor q/2 \rfloor - B_{\mathsf{round}}, \lfloor q/2 \rfloor + B_{\mathsf{round}}] \\ \bot & \text{otherwise.} \end{cases} \tag{5.4}$$

  The challenger then constructs the challenge as follows:
  - Suppose for all $d \in [\lambda]$, there exists an index $i \in [\ell]$ where $r_{d,i} = \bot$. Then, the challenger sets $\mathbf{c} = \bot$ and for all $i \in [\ell]$, $r_i = 0$ and $\boldsymbol{\pi}_i = \bot$.
  - Otherwise, let $d^* \in [\lambda]$ be the first index where $r_{d^*,i} \in \{0,1\}$ for all $i \in [\ell]$. Then the challenger sets $\mathbf{c} = \mathbf{c}_{d^*}$ and for all $i \in [\ell]$, $r_i = r_{d^*,i}$ and $\boldsymbol{\pi}_i = \boldsymbol{\pi}_{d^*,i}$.

  Finally, if $b = 0$, then the challenger sets $\beta = r_{i^*}$. If $b = 1$, the challenger samples $\beta \xleftarrow{\mathrm{R}} \{0,1\}$. The challenger gives $\big(\mathsf{crs}, \mathbf{c}, \{(i, r_i, \boldsymbol{\pi}_i)\}_{i \neq i^*}, \beta\big)$ to $\mathcal{A}$. At the end of the game, algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.

- $\mathsf{Hyb}_1^{(b)}$: Same as $\mathsf{Hyb}_0^{(b)}$, except for all $d \in [\lambda]$, the challenger changes how it samples $\mathbf{c}_d$ and $\boldsymbol{\pi}_{d,i}$. Specifically, after sampling crs as in $\mathsf{Hyb}_0^{(b)}$, the challenger proceeds as follows for each $d \in [\lambda]$:
  - Sample $\mathbf{c}_d \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and for each $i \in [\ell]$, sample $\boldsymbol{\pi}_{d,i} \leftarrow (\mathbf{A}_i)^{-1}_{s_{\mathsf{samp}}}(\mathbf{c}_d)$.
  - For each $i \in [\ell]$, the challenger computes $u_{d,i} = \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_{d,i}$. If $\|\boldsymbol{\pi}_{d,i}\| > B_{\max}$, set $r_{d,i} = \bot$. Otherwise, it sets $r_{d,i}$ according to Eq. (5.4).

  The rest of the experiment proceeds as in $\mathsf{Hyb}_0^{(b)}$.

- $\mathsf{Hyb}_2^{(b)}$: Same as $\mathsf{Hyb}_1^{(b)}$, except the challenger no longer checks the norm constraint on $\boldsymbol{\pi}_{d,i}$ when computing $r_{d,i}$. Specifically, after sampling crs as in $\mathsf{Hyb}_0^{(b)}$, the challenger proceeds as follows for each $d \in [\lambda]$:
  - Sample $\mathbf{c}_d \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and for each $i \in [\ell]$, sample $\boldsymbol{\pi}_{d,i} \leftarrow (\mathbf{A}_i)^{-1}_{s_{\mathsf{samp}}}(\mathbf{c}_d)$.
  - For each $i \in [\ell]$, the challenger computes $u_{d,i} = \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_{d,i}$ and sets $r_{d,i}$ according to Eq. (5.4). In particular, the challenger no longer checks if $\|\boldsymbol{\pi}_{d,i}\| \le B_{\max}$.

  The rest of the experiment proceeds as in $\mathsf{Hyb}_0^{(b)}$.

- $\mathsf{Hyb}_3^{(b)}$: Same as $\mathsf{Hyb}_2^{(b)}$, except for all $d \in [\lambda]$, the challenger samples $u_{d,i^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q$. Specifically, after sampling crs as in $\mathsf{Hyb}_0^{(b)}$, the challenger proceeds as follows for each $d \in [\lambda]$:
  - Sample $\mathbf{c}_d \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and for each $i \in [\ell]$, sample $\boldsymbol{\pi}_{d,i} \leftarrow (\mathbf{A}_i)^{-1}_{s_{\mathsf{samp}}}(\mathbf{c}_d)$.
  - For each $i \in [\ell] \setminus \{i^*\}$, the challenger computes $u_{d,i} = \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_{d,i}$. It then samples $u_{d,i^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q$. Finally, the challenger sets $r_{d,i}$ according to Eq. (5.4).

  The rest of the experiment proceeds as in $\mathsf{Hyb}_0^{(b)}$.

[4]In this description, we explicitly unroll the (up to) λ iterations of rejection sampling that the GenBits algorithm performs. Specifically, the challenger samples λ commitments and openings, and the output is defined to be the first instance that is successful (i.e., the first instance that GenBits would have accepted). As such, the description here is identical to the procedure in GenBits, but is more convenient to analyze.

- $\mathsf{Hyb}_4^{(b)}$: Same as $\mathsf{Hyb}_3^{(b)}$, except for all $d \in [\lambda]$, the challenger changes how it samples r_{d,i^*}. Specifically, after sampling crs as in $\mathsf{Hyb}_0^{(b)}$, the challenger proceeds as follows for each $d \in [\lambda]$:
  - Sample $\mathbf{c}_d \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and for each $i \in [\ell]$, sample $\boldsymbol{\pi}_{d,i} \leftarrow (\mathbf{A}_i)^{-1}_{s_{\mathsf{samp}}}(\mathbf{c}_d)$.
  - For each $i \in [\ell] \setminus \{i^*\}$, the challenger computes $u_{d,i} = \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_{d,i}$ and sets $r_{d,i}$ according to Eq. (5.4).
  - With probability $(4B_{\mathsf{round}} + 2)/q$, the challenger samples $r_{d,i^*} \xleftarrow{\mathrm{R}} \{0,1\}$, and with probability $1 - (4B_{\mathsf{round}} + 2)/q$, the challenger sets $r_{d,i^*} = \perp$.

  The rest of the experiment proceeds as in $\mathsf{Hyb}_0^{(b)}$.

- $\mathsf{Hyb}_5^{(b)}$: Same as $\mathsf{Hyb}_4^{(b)}$, except the challenger samples $r_{i^*} \xleftarrow{\mathrm{R}} \{0,1\}$ in the case for all $d \in [\lambda]$, there exists an index $i \in [\ell]$ where $r_{d,i} = \perp$. Specifically, after sampling crs as in $\mathsf{Hyb}_0^{(b)}$, the challenger proceeds as follows for each $d \in [\lambda]$:
  - Sample $\mathbf{c}_d \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and for each $i \in [\ell]$, sample $\boldsymbol{\pi}_{d,i} \leftarrow (\mathbf{A}_i)^{-1}_{s_{\mathsf{samp}}}(\mathbf{c}_d)$.
  - For each $i \in [\ell] \setminus \{i^*\}$, the challenger computes $u_{d,i} = \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_{d,i}$ and sets $r_{d,i}$ according to Eq. (5.4).
  - With probability $(4B_{\mathsf{round}} + 2)/q$, the challenger samples $r_{d,i^*} \xleftarrow{\mathrm{R}} \{0,1\}$, and with probability $1 - (4B_{\mathsf{round}} + 2)/q$, the challenger sets $r_{d,i^*} = \perp$.

  The challenger then constructs the challenge as follows:
  - Suppose for all $d \in [\lambda]$, there exists an index $i \in [\ell]$ where $r_{d,i} = \perp$. Then, the challenger sets $\mathbf{c} = \perp$ and for all $i \in [\ell] \setminus \{i^*\}$, it sets $r_i = 0$ and $\boldsymbol{\pi}_i = \perp$. The challenger samples $r_{i^*} \xleftarrow{\mathrm{R}} \{0,1\}$.
  - Otherwise, let $d^* \in [\lambda]$ be the first index where $r_{d^*,i} \in \{0,1\}$ for all $i \in [\ell]$. Then the challenger sets $\mathbf{c} = \mathbf{c}_{d^*}$ and for all $i \in [\ell]$, $r_i = r_{d^*,i}$ and $\boldsymbol{\pi}_i = \boldsymbol{\pi}_{d^*,i}$.

  Finally, if $b = 0$, then the challenger sets $\beta = r_{i^*}$. If $b = 1$, the challenger samples $\beta \xleftarrow{\mathrm{R}} \{0,1\}$. The challenger gives $\left(\mathsf{crs}, \mathbf{c}, \{(i, r_i, \boldsymbol{\pi}_i)\}_{i \neq i^*}, \beta\right)$ to $\mathcal{A}$. At the end of the game, algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.

We write $\mathsf{Hyb}_i^{(b)}(\mathcal{A})$ to denote the random variable corresponding to the output of an execution of hybrid $\mathsf{Hyb}_i^{(b)}$ with adversary $\mathcal{A}$. We now analyze each adjacent pair of distributions.

Lemma 5.11. *Suppose Π_{samp} satisfies the preimage distribution property. Then, for all $b \in \{0,1\}$, $\mathsf{Hyb}_0^{(b)}(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_1^{(b)}(\mathcal{A})$.*

Proof. For each $j \in \{0, \ldots, \lambda\}$, we define an intermediate hybrid as follows:

- $\mathsf{Hyb}_{0,j}^{(b)}$: Same as $\mathsf{Hyb}_0^{(b)}$, except for all $d \leq j$, the challenger samples $(\boldsymbol{\pi}_{d,1}, \ldots, \boldsymbol{\pi}_{d,\ell}, \mathbf{c}_d)$ according to the procedure in $\mathsf{Hyb}_1^{(b)}$. For all $d > j$, the challenger samples $(\boldsymbol{\pi}_{d,1}, \ldots, \boldsymbol{\pi}_{d,\ell}, \mathbf{c}_d)$ according to the procedure in $\mathsf{Hyb}_0^{(b)}$.

By construction, $\mathsf{Hyb}_{0,0}^{(b)}(\mathcal{A}) \equiv \mathsf{Hyb}_0^{(b)}(\mathcal{A})$ and $\mathsf{Hyb}_{0,\lambda}^{(b)}(\mathcal{A}) \equiv \mathsf{Hyb}_1^{(b)}(\mathcal{A})$. We now argue that for every $j \in [\lambda]$, the statistical distance between $\mathsf{Hyb}_{0,j-1}^{(b)}(\mathcal{A})$ and $\mathsf{Hyb}_{0,j}^{(b)}(\mathcal{A})$ is $\mathsf{negl}(\lambda)$. The only difference between these two distributions is the distribution of $(\boldsymbol{\pi}_{j,1}, \ldots, \boldsymbol{\pi}_{j,\ell}, \mathbf{c}_j)$. With overwhelming probability over the choice of $\mathsf{crs}_{\mathsf{samp}}$, these two distributions are statistically indistinguishable by the preimage distribution property of Π_{samp}. □

Lemma 5.12. *Suppose $n \geq \lambda$, $t \geq 2n \log q$, $s_{\mathsf{samp}} \geq \log t$, and $B_{\max} \geq \sqrt{t} s_{\mathsf{samp}}$. Then, for all $b \in \{0,1\}$, $\mathsf{Hyb}_1^{(b)}(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_2^{(b)}(\mathcal{A})$.*

Proof. These experiments are identical unless in an execution of $\mathsf{Hyb}_1^{(b)}$, there exists an index $d \in [\lambda]$ and $i \in [\ell]$ where $\|\boldsymbol{\pi}_{d,i}\| > B_{\mathsf{max}}$. In $\mathsf{Hyb}_1^{(b)}$, the challenger samples $\boldsymbol{\pi}_{d,i} \leftarrow (\mathbf{A}_i)^{-1}_{s_{\mathsf{samp}}}(\mathbf{c}_d)$. By Lemma 4.8, the marginal distribution of $\mathbf{A}_i$ in $\mathsf{Hyb}_1^{(b)}$ is statistically close to uniform. Since $t \geq 2n \log q$, $s_{\mathsf{samp}} \geq \log t$, and $B_{\mathsf{max}} \geq \sqrt{t} s_{\mathsf{samp}}$, by Lemma 3.2, with overwhelming probability over the choice of $\boldsymbol{\pi}_{d,i}$, it holds that $\|\boldsymbol{\pi}_{d,i}\| \leq B_{\mathsf{max}}$. By a union bound over all $d \in [\lambda]$ and $i \in [\ell]$, we conclude that with overwhelming probability, in an execution of $\mathsf{Hyb}_1^{(b)}$, it holds that $\|\boldsymbol{\pi}_{d,i}\| \leq B_{\mathsf{max}}$ for all $d \in [\lambda]$ and $i \in [\ell]$. In this case, the output of $\mathsf{Hyb}_1^{(b)}$ and $\mathsf{Hyb}_2^{(b)}$ is the same. □

Lemma 5.13. *Suppose $n \geq 4\lambda + 2\log q$, $t \geq 2n \log q$, q is prime, and $s_{\mathsf{samp}} \geq \log t$. Then, for all $b \in \{0, 1\}$, $\mathsf{Hyb}_2^{(b)}(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_3^{(b)}(\mathcal{A})$.*

Proof. For each $j \in \{0, \ldots, \lambda\}$, we define an intermediate hybrid as follows:

- $\mathsf{Hyb}_{2,j}^{(b)}$: Same as $\mathsf{Hyb}_2^{(b)}$, except for all $d \leq j$, the challenger samples $u_{d,i^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q$. For all $d > j$, the challenger sets $u_{d,i^*} = \mathbf{v}_{i^*}^{\mathsf{T}} \boldsymbol{\pi}_{d,i^*}$ as in $\mathsf{Hyb}_1^{(b)}$.

By construction, $\mathsf{Hyb}_{2,0}^{(b)}(\mathcal{A}) \equiv \mathsf{Hyb}_2^{(b)}(\mathcal{A})$ and $\mathsf{Hyb}_{2,\lambda}^{(b)}(\mathcal{A}) \equiv \mathsf{Hyb}_3^{(b)}(\mathcal{A})$. We argue that for all $j \in [\lambda]$, the statistical distance between $\mathsf{Hyb}_{2,j-1}^{(b)}$ and $\mathsf{Hyb}_{2,j}^{(b)}$ is $\mathsf{negl}(\lambda)$. The only difference between these two distributions is the distribution of u_{j,i^*}. We show that these two distributions are statistically indistinguishable.

- By Lemma 4.8, the marginal distribution of $\mathbf{A}_{i^*}$ in $\mathsf{Hyb}_{2,j-1}^{(b)}$ and $\mathsf{Hyb}_{2,j}^{(b)}$ is statistically close to uniform over $\mathbb{Z}_q^{n \times t}$.
- In $\mathsf{Hyb}_{2,j-1}^{(b)}$ and $\mathsf{Hyb}_{2,j}^{(b)}$, the challenger samples $\boldsymbol{\pi}_{j,i^*} \leftarrow (\mathbf{A}_{i^*})^{-1}_{s_{\mathsf{samp}}}(\mathbf{c}_d)$. Since $t \geq 2n \log q$ and $s_{\mathsf{samp}} \geq \log t$, we appeal to Lemma 3.8 to conclude that with overwhelming probability over the choice of $\mathbf{A}_{i^*}$,
$$\mathbf{H}_\infty(\boldsymbol{\pi}_{j,i^*}) \geq n/2 \geq 2\lambda + \log q.$$
- In $\mathsf{Hyb}_{2,j-1}^{(b)}$ and $\mathsf{Hyb}_{2,j}^{(b)}$, the challenger samples $\mathbf{v}_{i^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^t$. By the leftover hash lemma (Lemma 3.7), the statistical distance between the distributions
$$\left\{(\mathbf{v}_{i^*}, \mathbf{v}_{i^*}^{\mathsf{T}} \boldsymbol{\pi}_{j,i^*}) : \mathbf{v}_{i^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^t\right\} \quad \text{and} \quad \left\{(\mathbf{v}_{i^*}, u_{j,i^*}) : \mathbf{v}_{i^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^t, u_{j,i^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q\right\}$$
is at most $2^{-\lambda} = \mathsf{negl}(\lambda)$. The distribution in the left-hand side corresponds to $\mathsf{Hyb}_{2,j-1}^{(b)}$ while the one on the right-hand side corresponds to $\mathsf{Hyb}_{2,j}^{(b)}$.

We conclude that for all $j \in [\lambda]$, the distributions $\mathsf{Hyb}_{2,j-1}^{(b)}(\mathcal{A})$ and $\mathsf{Hyb}_{2,j}^{(b)}$ are statistically indistinguishable. The lemma now follows by a hybrid argument. □

Lemma 5.14. *If $q > 4B_{\mathsf{round}} + 2$, then for all $b \in \{0, 1\}$, $\mathsf{Hyb}_3^{(b)}(\mathcal{A}) \equiv \mathsf{Hyb}_4^{(b)}(\mathcal{A})$.*

Proof. The distributions $\mathsf{Hyb}_3^{(b)}(\mathcal{A})$ and $\mathsf{Hyb}_4^{(b)}(\mathcal{A})$ are identically distributed as long as $q > 4B_{\mathsf{round}} + 2$. In this case, the intervals $[-B_{\mathsf{round}}, B_{\mathsf{round}}]$ and $[\lfloor q/2 \rfloor - B_{\mathsf{round}}, \lfloor q/2 \rfloor + B_{\mathsf{round}}]$ are disjoint and each has size $2B_{\mathsf{round}} + 1$. The two experiments only differ in how they compute r_{d,i^*} for $d \in [\lambda]$. We show that these two procedures are distributed identically for each $d \in [\lambda]$.

- In $\mathsf{Hyb}_3^{(b)}$, the challenger samples $u_{d,i^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q$ and then sets $r_{d,i^*} = 0$ if $u_{d,i^*} \in [-B_{\mathsf{round}}, B_{\mathsf{round}}]$. Over the randomness of u_{d,i^*}, this happens with probability $(2B_{\mathsf{round}} + 1)/q$. Alternatively, it sets $r_{d,i^*} = 1$ if $u_{d,i^*} \in [\lfloor q/2 \rfloor - B_{\mathsf{round}}, \lfloor q/2 \rfloor + B_{\mathsf{round}}]$. This also happens with probability $(2B_{\mathsf{round}} + 1)/q$ over the randomness of u_{d,i^*}. Finally, if neither event holds, which occurs with probability $1 - (4B_{\mathsf{round}} + 2)/q$, then the challenger sets $r_{d,i^*} = \perp$.
- In $\mathsf{Hyb}_4^{(b)}$, $r_{d,i^*} = 0$ with probability $(1/2) \cdot (4B_{\mathsf{round}} + 2)/q = (2B_{\mathsf{round}} + 1)/q$, which matches the probability in $\mathsf{Hyb}_3^{(b)}$. The challenger sets $r_{d,i^*} = 1$ with the same probability. Finally, the challenger sets $r_{d,i^*} = \perp$ with probability $1 - (4B_{\mathsf{round}} + 2)/q$, which is identical to the behavior in $\mathsf{Hyb}_3^{(b)}$.

We conclude that the distribution of r_{d,i^*} is identical in the two experiments for all $d \in [\lambda]$. Correspondingly, the outputs of these two experiments are identically distributed. $\square$

Lemma 5.15. *Suppose $n \geq 4\lambda + 2\log q$, $t \geq 2n \log q$, q is prime, $s_{\mathsf{samp}} > \log t$, and $B_{\mathsf{round}} \geq q/4 - q/(8\ell) + 1/2$. Then, for all $b \in \{0,1\}$, $\mathsf{Hyb}_4^{(b)}(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_5^{(b)}(\mathcal{A})$.*

Proof. $\mathsf{Hyb}_4^{(b)}$ and $\mathsf{Hyb}_5^{(b)}$ are identical experiments unless for all $d \in [\lambda]$, there exists an index $i \in [\ell]$ where $r_{d,i} = \perp$. We show that this event happens with negligible probability. To analyze the probability of this event, we first define the following sequence of distributions and argue that each adjacent pair is statistically indistinguishable:

- $\mathcal{D}_0$: This is the distribution of $r_{d,i}$ in $\mathsf{Hyb}_4^{(b)}$ and $\mathsf{Hyb}_5^{(b)}$. Namely, the distribution samples $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$, $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$, and $\mathbf{v}_1, \ldots, \mathbf{v}_\ell \xleftarrow{R} \mathbb{Z}_q^t$. Then for each $d \in [\lambda]$, it samples $\mathbf{c}_d \leftarrow \mathbb{Z}_q^n$. For each $d \in [\lambda]$ and $i \in [\ell] \setminus \{i^*\}$, it samples $\boldsymbol{\pi}_{d,i} \leftarrow (\mathbf{A}_i)^{-1}_{s_{\mathsf{samp}}}(\mathbf{c}_d)$, sets $u_{d,i} = \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_{d,i}$, and sets $r_{d,i}$ according to Eq. (5.4). Finally, with probability $(4B_{\mathsf{round}} + 2)/q$, sample $r_{d,i^*} \xleftarrow{R} \{0,1\}$ and with probability $1 - (4B_{\mathsf{round}} + 2)/q$, set $r_{d,i^*} = \perp$. The output is $(r_{1,1}, \ldots, r_{1,\ell}, \ldots, r_{\lambda,1}, \ldots, r_{\lambda,\ell})$.
- $\mathcal{D}_1$: In this distribution, for all $d \in [\lambda]$ and $i \in [\ell] \setminus \{i^*\}$, sample $u_{d,i} \xleftarrow{R} \mathbb{Z}_q$ and set $r_{d,i}$ according to Eq. (5.4). Then, with probability $(4B_{\mathsf{round}} + 2)/q$, sample $r_{d,i^*} \xleftarrow{R} \{0,1\}$ and with probability $1 - (4B_{\mathsf{round}} + 2)/q$, set $r_{d,i^*} = \perp$. The output is $(r_{1,1}, \ldots, r_{1,\ell}, \ldots, r_{\lambda,1}, \ldots, r_{\lambda,\ell})$.
- $\mathcal{D}_1$: In this distribution, for all $d \in [\lambda]$ and $i \in [\ell]$, sample $r_{d,i} \xleftarrow{R} \{0,1\}$ and with probability $1 - (4B_{\mathsf{round}} + 2)/q$, set $r_{d,i} = \perp$. The output is $(r_{1,1}, \ldots, r_{1,\ell}, \ldots, r_{\lambda,1}, \ldots, r_{\lambda,\ell})$.

We start by showing that distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ are statistically indistinguishable. Formally, we analyze the distribution $\mathcal{D}_0$:

- Since $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$ and $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(\mathsf{crs}_{\mathsf{samp}})$, we appeal to Lemma 4.8 to conclude that the marginal distribution of each $\mathbf{A}_i$ is statistically close to uniform over $\mathbb{Z}_q^{n \times t}$. We declare $\mathbf{A}_i \in \mathbb{Z}_q^{n \times t}$ to be "good" if the property in Lemma 3.8 holds: namely, $\mathbf{A}_i$ is good if for all $\mathbf{y} \in \mathbb{Z}_q^n$, the distribution $(\mathbf{A}_i)^{-1}_{s_{\mathsf{samp}}}(\mathbf{y})$ has min-entropy at least $n/2$. Since $t \geq 2n \log q$ and $s_{\mathsf{samp}} \geq \log t$, all but a $\mathsf{negl}(n)$ fraction of $\mathbf{A}_i$'s are "good". Since the marginal distribution of each $\mathbf{A}_i$ is statistically close to uniform over $\mathbb{Z}_q^{n \times t}$, it follows that each $\mathbf{A}_i$ is good with probability $1 - \mathsf{negl}(n)$. By a union bound (and since $\ell = \mathsf{poly}(\lambda)$), we conclude that with overwhelming probability, all of the $\mathbf{A}_i$ are good.
- If $\mathbf{A}_i$ is good for all $i \in [\ell]$, this means that $\mathbf{H}_\infty(\boldsymbol{\pi}_{d,i}) \geq n/2 \geq 2\lambda + \log q$ for all $d \in [\lambda]$ and $i \in [\ell]$, where $\boldsymbol{\pi}_{d,i} \leftarrow (\mathbf{A}_i)^{-1}_{s_{\mathsf{samp}}}(\mathbf{c}_d)$.
- By Lemma 3.7, the following pair of distributions are statistically indistinguishable for all $i \in [\ell]$:
  - Sample $\mathbf{v}_i \xleftarrow{R} \mathbb{Z}_q^t$. For each $d \in [\lambda]$, sample $\mathbf{c}_d \xleftarrow{R} \mathbb{Z}_q^n$ and $\boldsymbol{\pi}_{d,i} \leftarrow (\mathbf{A}_i)^{-1}_{s_{\mathsf{samp}}}(\mathbf{c}_d)$. Output $(\mathbf{v}_i, \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_{1,i}, \ldots, \mathbf{v}_i^{\mathsf{T}} \boldsymbol{\pi}_{\lambda,i})$.
  - Sample $\mathbf{v}_i \xleftarrow{R} \mathbb{Z}_q^t$. For each $d \in [\lambda]$, sample $u_{d,i} \xleftarrow{R} \mathbb{Z}_q$. Output $(\mathbf{v}_i, u_{1,i}, \ldots, u_{\lambda,i})$.

  Since this holds for all $i \in [\ell]$ and $\ell = \mathsf{poly}(\lambda)$, we conclude that the joint distribution of $(u_{d,i})_{(d \in [\lambda], i \in [\ell])}$ in $\mathcal{D}_0$ and $\mathcal{D}_1$ is statistically indistinguishable. Since both distributions derive $r_{d,i}$ from $u_{d,i}$ using the same procedure, we conclude that $\mathcal{D}_0$ and $\mathcal{D}_1$ are statistically indistinguishable.

Next, $\mathcal{D}_1$ and $\mathcal{D}_2$ are identical distributions. Namely, if $u_{d,i} \xleftarrow{R} \mathbb{Z}_q$, then $r_{d,i} = 0$ with probability $(2B_{\mathsf{round}} + 1)/q$, $r_{d,i} = 1$ with probability $(2B_{\mathsf{round}} + 1)/q$, and $r_{d,i} = \perp$ with probability $1 - (4B_{\mathsf{round}} + 2)/q$. By a hybrid argument, we conclude that $\mathcal{D}_0$ and $\mathcal{D}_2$ are statistically indistinguishable.

Consider now the probability that in an execution of $\mathsf{Hyb}_4^{(b)}$ and $\mathsf{Hyb}_5^{(b)}$, it happens that for all $d \in [\lambda]$, there exists an index $i \in [\ell]$ where $r_{d,i} = \perp$. For a tuple $(r_{1,1}, \ldots, r_{1,\ell}, \ldots, r_{d,1}, \ldots, r_{d,\lambda})$ and an index $d \in [\lambda]$, we define the event Bad_d to be the event that there exists $i \in [\ell]$ where $r_{d,i} = \perp$. By a union bound, we have for all $d \in [\lambda]$,

$$\Pr[\mathsf{Bad}_d : (r_{d,i})_{d \in [\lambda], i \in [\ell]} \leftarrow \mathcal{D}_2] \leq \sum_{i \in [\ell]} \frac{q - (4B_{\mathsf{round}} + 2)}{q} \leq \sum_{i \in [\ell]} \frac{1}{2\ell} = \frac{1}{2},$$

when $B_{\text{round}} \geq q/4 - q/(8\ell) + 1/2$. By definition of $\mathcal{D}_2$, we moreover have that

$$\Pr\left[\bigwedge_{d\in[\lambda]} \mathsf{Bad}_d : (u_{d,i})_{d\in[\lambda],i\in[\ell]} \leftarrow \mathcal{D}_1\right] = \prod_{d\in[\lambda]} \Pr[\mathsf{Bad}_d : (u_{d,i})_{d\in[\lambda],i\in[\ell]} \leftarrow \mathcal{D}_2] = \frac{1}{2^\lambda}.$$

Finally, since $\mathcal{D}_0$ and $\mathcal{D}_1$ are statistically indistinguishable, there exists a negligible function $\mathsf{negl}(\cdot)$ where

$$\Pr\left[\bigwedge_{d\in[\lambda]} \mathsf{Bad}_d : (u_{d,i})_{d\in[\lambda],i\in[\ell]} \leftarrow \mathcal{D}_0\right] \leq \Pr\left[\bigwedge_{d\in[\lambda]} \mathsf{Bad}_d : (u_{d,i})_{d\in[\lambda],i\in[\ell]} \leftarrow \mathcal{D}_2\right] + \mathsf{negl}(\lambda) = \frac{1}{2^\lambda} + \mathsf{negl}(\lambda).$$

Thus, in an execution of $\mathsf{Hyb}_4^{(b)}$ and $\mathsf{Hyb}_5^{(b)}$, the probability that for all $d \in [\lambda]$, there exists an index $i \in [\ell]$ where $r_{d,i} = \perp$ is at most $2^{-\lambda} + \mathsf{negl}(\lambda)$. Thus, with overwhelming probability, the adversary's view in $\mathsf{Hyb}_4^{(b)}$ and $\mathsf{Hyb}_5^{(b)}$ is identical. The lemma follows. □

Lemma 5.16. *It holds that* $\mathsf{Hyb}_5^{(0)}(\mathcal{A}) \equiv \mathsf{Hyb}_5^{(1)}(\mathcal{A})$.

Proof. In $\mathsf{Hyb}_5^{(0)}$, the challenger sets $\beta = r_{i^*}$ whereas in $\mathsf{Hyb}_5^{(1)}$, the challenger samples $\beta \xleftarrow{\text{R}} \{0,1\}$. We argue that the distribution of r_{i^*} in $\mathsf{Hyb}_5^{(0)}$ is uniformly random (and independent of all other quantities in the adversary's view). Consider an execution of $\mathsf{Hyb}_5^{(0)}$. We consider two cases:

- Suppose for all $d \in [\lambda]$, there exists an index $i \in [\ell]$ where $r_{d,i} = \perp$. In this case, the challenger samples $r_{i^*} \xleftarrow{\text{R}} \{0,1\}$.
- Otherwise, let $d^* \in [\lambda]$ be the smallest index where $r_{d^*,i} \in \{0,1\}$ for all $i \in [\ell]$. Then, the challenger sets $r_{i^*} = r_{d^*,i^*}$. In $\mathsf{Hyb}_5^{(0)}$, the challenger either sets $r_{d^*,i^*} = \perp$ or samples $r_{d^*,i^*} \xleftarrow{\text{R}} \{0,1\}$. Since $r_{d^*,i^*} \neq \perp$, this means the challenger must have sampled $r_{d^*,i^*} \xleftarrow{\text{R}} \{0,1\}$.

Finally, none of the other components in the adversary's view depend on the value of r_{d^*,i^*}. As such, we conclude that the distribution of r_{i^*} in $\mathsf{Hyb}_5^{(0)}$ is uniform and independent of all other quantities in the adversary's view. Thus, the distribution of β in the two experiments is identical. □

Combining Lemmas 5.11 to 5.16, we conclude that $\mathsf{Hyb}_0^{(0)} \overset{s}{\approx} \mathsf{Hyb}_0^{(1)}$, which completes the proof. □

Theorem 5.17 (Succinctness)**.** *If* $n \log q = \mathrm{poly}(\lambda, \log \ell)$*, then Construction 5.3 is succinct.*

Proof. The size of the commitment $\mathbf{c}$ output by GenBits in Construction 5.3 is either an element of $\mathbb{Z}_q^n$ or $\perp$. Thus, we can describe $\mathbf{c}$ by a string of length $n \log q + 1$. If $n \log q = \mathrm{poly}(\lambda, \log \ell)$, then succinctness holds. □

Parameter instantiation. Let λ be a security parameter and let ℓ be the length of the hidden-bits string. We now provide one possible instantiation of the parameters in Construction 5.3 to satisfy Theorems 5.4, 5.5, 5.9 and 5.10. In the following, we assume that $\ell \leq 2^\lambda$, so $\log \ell \leq \lambda$.

- When setting parameters, we work under the assumption that $q \leq 2^{O(\lambda)}$. Our final parameter instantiations will satisfy this property. In this case, $\log q = O(\lambda)$.
- We require that $n \geq 4\lambda + 2 \log q$, so we can take $n = 4\lambda + O(\lambda) = O(\lambda)$.
- We set $t = 3n \lceil \log q \rceil \cdot (\lceil \log \ell \rceil + 1) = O(\lambda^3)$.
- We set $s_{\mathsf{LWE}} = \lambda^\delta$ for some constant $\delta > 0$, which we will set later.
- We set $s_{\mathsf{samp}} = (\ell t + 3n \lceil \log q \rceil) \log(\ell n) = O(\lambda^4 \ell)$. We take $B_{\max} > \sqrt{t} s_{\mathsf{samp}} = O(\lambda^{11/2} \ell)$.

- We choose $q = \mathsf{poly}(\lambda, \ell)$ and δ such that $q > 8\ell t\sqrt{\lambda} s_{\mathsf{LWE}} B_{\max} + 4\ell = O(\lambda^{9+\delta}\ell^2)$ and the $\mathsf{LWE}_{n,t,q,s_{\mathsf{LWE}}}$ assumption holds. In particular, since $q = \mathsf{poly}(\lambda, \ell)$ and $\ell < 2^\lambda$, this means $q \leq 2^{O(\lambda)}$, which satisfies our initial assumption.
- We set $B_{\mathsf{round}} = q/4 - q/(8\ell) + 1/2$.
- Finally, we instantiate $(\mathsf{Gen}, \mathsf{Expand}, \mathsf{SampleMultPre})$ with the $(n, t, q, s_{\mathsf{samp}})$-shifted multi-preimage trapdoor sampler from Theorem 4.7 (Construction 4.6).

We briefly verify that these parameters satisfy the necessary requirements:

- Theorem 5.9 requires that $q > 4t\sqrt{\lambda} s_{\mathsf{LWE}} B_{\max} + 4B_{\mathsf{round}}$. Since $B_{\mathsf{round}} = q/4 - q/(8\ell) + 1/2$, this is equivalent to requiring that $q/(2\ell) > 4t\sqrt{\lambda} s_{\mathsf{LWE}} B_{\max} + 2$, or equivalently, that $q > 8\ell t\sqrt{\lambda} s_{\mathsf{LWE}} B_{\max} + 4\ell$.
- All of the conditions of Theorem 5.10 are satisfied by construction. In particular, the requirement $q > 4B_{\mathsf{round}} + 2$ is satisfied whenever $q > 8\ell$.

With this setting of parameters, we obtain a dual-mode hidden-bits generator with the following properties:

- **CRS size:** By Theorem 4.7, the size of the CRS is $\lambda + nt \log q + \ell t \log q = \ell \cdot \mathsf{poly}(\lambda, \log \ell)$. Moreover, in hiding mode, the CRS sampling algorithm is *transparent*.
- **Commitment and opening size:** The size of a commitment $\mathbf{c} \in \mathbb{Z}_q^n$ is $n \log q = \mathsf{poly}(\lambda, \log \ell)$ bits. The size of an opening $\boldsymbol{\pi} \in \mathbb{Z}_q^t$ is $t \log q = \mathsf{poly}(\lambda, \log \ell)$ bits.

Finally, for all $\ell = \mathsf{poly}(\lambda)$, security relies on the $\mathsf{LWE}_{n,t,q,s_{\mathsf{LWE}}}$ assumption with a *polynomial* modulus-to-noise ratio (in this case, $q = \mathsf{poly}(\lambda, \ell) = \mathsf{poly}(\lambda)$ and $s_{\mathsf{LWE}} = \lambda^\delta$ for constant $\delta > 0$). We summarize our instantiation in the following corollaries:

Corollary 5.18 (Dual-Mode Hidden-Bits Generator from LWE)**.** *Let λ be a security parameter. Then, for all polynomials $\ell = \ell(\lambda)$, under the LWE assumption with a polynomial modulus-to-noise ratio, there exists a dual-mode hidden-bits generator with a CRS of size $\ell \cdot \mathsf{poly}(\lambda, \log \ell)$. Moreover, in (statistically) hiding mode, the common reference string can be sampled using a* transparent *setup algorithm.*

Corollary 5.19 (Dual-Mode NIZK for NP from LWE)**.** *Under the LWE assumption with a polynomial modulus-to-noise ratio, there exists a dual-mode NIZK for* NP*. Specifically, we obtain a computational NIZK proof in the structured reference string model and a statistical NIZK argument in the common* random *string model.*

6 Statistically-Hiding Vector Commitments from SIS

In this section, we show how to use our shifted multi-preimage trapdoor sampler to construct a statistically-hiding vector commitment from the SIS assumption. Our vector commitment scheme supports transparent setup. Moreover, the size of the CRS, the commitment, and the opening all scale polylogarithmically with the input dimension. This improves upon the earlier constructions of de Castro and Peikert [dCP23], which does not support statistically-hiding openings as well as the construction of Wee and Wu [WW23b], which required a structured common reference string (with size *quadratic* in the input dimension). We start by recalling the definition of a vector commitment and then provide our construction and analysis. We refer to Section 2.1 for a high-level overview of the construction. Our definitions are adapted from [WW23b]:

Definition 6.1 (Vector Commitment)**.** Let λ be a security parameter and ℓ be a dimension. A vector commitment scheme with succinct local openings over a message space $\mathcal{M} = \{\mathcal{M}_{\lambda,\ell}\}_{\lambda,\ell\in\mathbb{N}}$ consists of a tuple of efficient algorithms $\Pi_{\mathsf{VC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Verify})$ with the following properties:

- $\mathsf{Setup}(1^\lambda, 1^\ell) \to \mathsf{crs}$: On input the security parameter λ and the vector length ℓ, the setup algorithm outputs a common reference string crs.
- $\mathsf{Commit}(\mathsf{crs}, \mathbf{x}) \to (\sigma, \mathsf{st})$: On input the common reference string crs and a vector $\mathbf{x}$, the commit algorithm outputs a commitment σ and a state st.

- $\mathsf{Open}(\mathsf{st}, i) \to \pi$: On input a commitment state st and an index i, the open algorithm outputs an opening π. Note that the opening algorithm could be randomized.
- $\mathsf{Verify}(\mathsf{crs}, \sigma, i, x, \pi) \to b$: On input the common reference string crs, a commitment σ, an index i, a message x, and an opening π, the verification algorithm outputs a bit $b \in \{0, 1\}$.

We now define several standard properties on vector commitment schemes:

- **Correctness:** For all polynomials $\ell = \ell(\lambda)$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$ and all inputs $\mathbf{x} = (x_1, \ldots, x_\ell) \in \mathcal{M}_{\lambda,\ell}^{\ell}$,

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, \sigma, i, x_i, \pi) = 1 : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell) \\ (\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x}) \\ \pi \leftarrow \mathsf{Open}(\mathsf{st}, i) \end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

- **Succinctness:** The vector commitment scheme is succinct if there exist fixed polynomials p_1, p_2 such that for all $\lambda, \ell \in \mathbb{N}$, all crs in the support of $\mathsf{Setup}(1^\lambda, 1^\ell)$, all vectors $\mathbf{x} \in \mathcal{M}_{\lambda,\ell}^{\ell}$, all (σ, st) in the support of $\mathsf{Commit}(\mathsf{crs}, \mathbf{x})$, and all π in the support of $\mathsf{Open}(\mathsf{st}, i)$, we have that $|\sigma| = p_1(\lambda, \log \ell)$ and $|\pi| = p_2(\lambda, \log \ell)$.

- **Computational binding:** We say the commitment scheme is computationally binding if for all polynomials $\ell = \ell(\lambda)$ and all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$\Pr\left[\begin{array}{c} \mathsf{Verify}(\mathsf{crs}, \sigma, i, x, \pi) = 1 \\ \text{and } x \neq x' \text{ and} \\ \mathsf{Verify}(\mathsf{crs}, \sigma, i, x', \pi') = 1 \end{array} : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell); \\ \big(\sigma, i, (x, \pi), (x', \pi')\big) \leftarrow \mathcal{A}(1^\lambda, 1^\ell, \mathsf{crs}) \end{array}\right] = \mathsf{negl}(\lambda).$$

- **Statistical hiding:** For a vector dimension ℓ, an adversary $\mathcal{A}$, and a simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$, we define two distributions $\mathsf{Real}_{\mathcal{A}}(\lambda, \ell)$ and $\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(\lambda, \ell)$ as follows:

$\mathsf{Real}_{\mathcal{A}}(\lambda, \ell)$:

1. Sample $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ and give crs to $\mathcal{A}$.
2. Algorithm $\mathcal{A}$ outputs an input $\mathbf{x} \in \mathcal{M}_{\lambda,\ell}^{\ell}$.
3. Compute $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$ and give σ to $\mathcal{A}$.
4. Algorithm $\mathcal{A}$ can adaptively query for openings. On each query, it provides an index $i \in [\ell]$, and the challenger replies with $\pi_i \leftarrow \mathsf{Open}(\mathsf{st}, i)$.
5. Algorithm $\mathcal{A}$ outputs a bit $b \in \{0, 1\}$ which is the output of the experiment.

$\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(\lambda, \ell)$:

1. Sample $(\mathsf{crs}, \sigma, \mathsf{st}) \leftarrow \mathcal{S}_0(1^\lambda, 1^\ell)$ and give crs to $\mathcal{A}$.
2. Algorithm $\mathcal{A}$ outputs an input $\mathbf{x} \in \mathcal{M}_{\lambda,\ell}^{\ell}$.
3. Give σ to $\mathcal{A}$.
4. Algorithm $\mathcal{A}$ can adaptively query for openings. On each query, it provides an index $i \in [\ell]$, and the challenger computes $(\pi_i, \mathsf{st}) \leftarrow \mathcal{S}_1(\mathsf{st}, i, x_i)$. It replies to $\mathcal{A}$ with π_i.
5. Algorithm $\mathcal{A}$ outputs a bit $b \in \{0, 1\}$ which is the output of the experiment.

We say that the vector commitment scheme is statistically hiding if there exists an efficient simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$ and such that for all polynomials $\ell = \ell(\lambda)$ and all (possibly unbounded) adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$|\Pr[\mathsf{Real}_{\mathcal{A}}(\lambda, \ell) = 1] - \Pr[\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(\lambda, \ell) = 1]| = \mathsf{negl}(\lambda).$$

Vector commitment scheme. We now describe our vector commitment scheme. As described in Section 2.1, our construction can be viewed as replacing the CRS in the Wee-Wu vector commitment scheme [WW23b, Construction 3.9] with the CRS for our shifted multi-preimage trapdoor sampler. Our analysis follows via a similar structure as the analysis in [WW23b], except we now appeal to the properties of the shifted multi-preimage trapdoor sampler. We give the full description and analysis below:

Construction 6.2 (Vector Commitment)**.** Let λ be a security parameter and ℓ be an input length parameter. Let $\Pi_{\mathsf{samp}} = (\mathsf{Gen}, \mathsf{GenTD}, \mathsf{Expand}, \mathsf{ExpandLocal})$ be a (n, t, q, s)-shifted multi-preimage trapdoor sampler that supports local expansion (Definition 4.4). Let $B = B(\lambda, \ell)$ be a bound. We construct a vector commitment $\Pi_{\mathsf{VC}} =$ (Setup, Commit, Open, Verify) scheme over the message space $\mathcal{M} = \mathbb{Z}_q = \left\{\mathbb{Z}_{q(\lambda,\ell)}\right\}_{\lambda,\ell\in\mathbb{N}}$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^\ell)$: On input the security parameter λ and the vector dimension ℓ, the setup algorithm samples $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$. It outputs the common reference string $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$.
- $\mathsf{Commit}(\mathsf{crs}, \mathbf{x})$: On input the common reference string $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$ and a vector $\mathbf{x} \in \mathbb{Z}_q^\ell$, the commit algorithm computes $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$. Then it samples $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c}) \leftarrow \mathsf{SampleMultPre}(\mathsf{td}, x_1\mathbf{e}_1, \ldots, x_\ell\mathbf{e}_1)$ where $\mathbf{e}_1 = [1, 0, \ldots, 0]^\mathsf{T} \in \mathbb{Z}_q^n$ is the first standard basis vector. It outputs the commitment $\mathbf{c} \in \mathbb{Z}_q^n$ and the state $\mathsf{st} = (\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell)$.
- $\mathsf{Open}(\mathsf{st}, i)$: On input the state $\mathsf{st} = (\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell)$ and the index $i \in [\ell]$, the opening algorithm outputs $\boldsymbol{\pi}_i \in \mathbb{Z}_q^t$.
- $\mathsf{Verify}(\mathsf{crs}, \mathbf{c}, i, x, \boldsymbol{\pi})$: On input the common reference string $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$, a commitment $\mathbf{c} \in \mathbb{Z}_q^n$, an index $i \in [\ell]$, a message $x \in \mathbb{Z}_q$, and an opening $\boldsymbol{\pi} \in \mathbb{Z}_q^t$, the verification algorithm computes $\mathbf{A}_i = \mathsf{ExpandLocal}(1^\lambda, \mathsf{crs}_{\mathsf{samp}}, i)$ and outputs 1 if $\|\boldsymbol{\pi}\| \le B$ and $\mathbf{A}_i\boldsymbol{\pi} = \mathbf{c} + x\mathbf{e}_1$.

Theorem 6.3 (Correctness). *Suppose Π_{samp} satisfies correctness and the preimage distribution property. If $s \ge \log t$ and $B \ge \sqrt{t}s$, then Construction 6.2 is correct.*

Proof. Take any polynomial $\ell = \ell(\lambda)$ and any $\mathbf{x} \in \mathbb{Z}_q^\ell$. Let $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$, $(\mathbf{c}, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$, and $\boldsymbol{\pi}_i \leftarrow \mathsf{Open}(\mathsf{st}, i)$. Parse $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$. Let $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$. By construction, the commit algorithm samples $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c}) \leftarrow \mathsf{SampleMultPre}(\mathsf{td}, x_1\mathbf{e}_1, \ldots, x_\ell\mathbf{e}_1)$. Consider the value of $\mathsf{Verify}(\mathsf{crs}, \mathbf{c}, i, x_i, \boldsymbol{\pi}_i)$.

- By correctness of Π_{samp}, $\mathbf{A}_i\boldsymbol{\pi}_i = \mathbf{c} + x_i\mathbf{e}_1$. It suffices to argue that $\|\boldsymbol{\pi}_i\| \le B$.
- Since Π_{samp} satisfies the preimage distribution property, the distribution of $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$ is statistically close to the distribution obtained by sampling $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ and $\boldsymbol{\pi}_i \leftarrow (\mathbf{A}_i)_s^{-1}(x_i\mathbf{e}_1 + \mathbf{c})$ for all $i \in [\ell]$.
- By Lemma 4.8, the marginal distribution of $\mathbf{A}_i$ is statistically close to uniform. Since $s \ge \log t$, by Lemma 3.2, it holds that $\|\boldsymbol{\pi}_i\| \le \sqrt{t}s \le B$ with overwhelming probability.

Finally, the local expansion property (Definition 4.4) ensures that $\mathsf{ExpandLocal}(1^\lambda, \mathsf{crs}_{\mathsf{samp}}, i) = \mathbf{A}_i$. The above analysis shows that $\mathbf{A}_i\boldsymbol{\pi}_i = \mathbf{c} + x_i\mathbf{e}_1$ and $\|\boldsymbol{\pi}_i\| \le B$, so Verify outputs 1 with overwhelming probability. $\square$

Theorem 6.4 (Computational Binding). *Suppose Π_{samp} satisfies somewhere programmability. Then, under the $\mathsf{SIS}_{n-1,t,q,2B}$ assumption, Construction 6.2 is computationally binding.*

Proof. Take any polynomial $\ell = \ell(\lambda)$ and any efficient adversary $\mathcal{A}$ for the computational binding game. We begin by defining a sequence of hybrid experiments:

- Hyb_0: This is the real binding experiment:
  - The challenger begins by sampling $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$ and gives $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$ to $\mathcal{A}$.
  - Algorithm $\mathcal{A}$ outputs a commitment $\mathbf{c} \in \mathbb{Z}_q^n$, an index $i \in [\ell]$ and two pairs $(x, \boldsymbol{\pi})$ and $(x', \boldsymbol{\pi}')$, where $x, x' \in \mathcal{M}_{\lambda,\ell}$ and $\boldsymbol{\pi}, \boldsymbol{\pi}' \in \mathbb{Z}_q^t$.
  - The challenger then computes $\mathbf{A}_i = \mathsf{ExpandLocal}(1^\lambda, \mathsf{crs}_{\mathsf{samp}}, i)$ and outputs 1 if $x \ne x'$, $\|\boldsymbol{\pi}\|, \|\boldsymbol{\pi}'\| \le B$, and $\mathbf{c} = \mathbf{A}_i\boldsymbol{\pi} - x\mathbf{e}_1 = \mathbf{A}_i\boldsymbol{\pi}' - x'\mathbf{e}_1$. Otherwise, the challenger outputs 0.
- Hyb_1: Same as Hyb_0, except at the beginning of the game, the challenger samples an index $i^* \xleftarrow{\mathrm{R}} [\ell]$. The output of the experiment is 1 if the conditions in Hyb_0 hold and $i = i^*$.
- Hyb_2: Same as Hyb_1, except the challenger samples $\mathbf{A}_{i^*} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times t}$ and $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{GenProg}(1^\lambda, 1^\ell, i^*, \mathbf{A}_{i^*})$.

We write $\mathsf{Hyb}_i(\mathcal{A})$ to denote the random variable corresponding to the output of an execution of Hyb_i with adversary $\mathcal{A}$. We now analyze each pair of adjacent experiments:

Lemma 6.5. *It holds that* $\Pr[\mathsf{Hyb}_0(\mathcal{A}) = 1] = \ell \cdot \Pr[\mathsf{Hyb}_1(\mathcal{A})]$.

Proof. The only difference between these two experiments is the additional condition that $i^* = i$ in Hyb_1. Since the adversary's view in Hyb_1 is independent of i^* (and in fact, the challenger can sample i^* *after* the adversary outputs i), the probability that $\mathsf{Hyb}_1(\mathcal{A}) = 1$ is exactly $1/\ell \cdot \Pr[\mathsf{Hyb}_0(\mathcal{A}) = 1]$. The claim follows. □

Lemma 6.6. *If Π_{samp} is somewhere programmable, then $\mathsf{Hyb}_1(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_2(\mathcal{A})$.*

Proof. The only difference between these two experiments is the distribution of crs. These two distributions are statistically indistinguishable by somewhere programmability of Π_{samp} □

Lemma 6.7. *Suppose Π_{samp} satisfies somewhere programmability and local expansion. Then, under the $\mathsf{SIS}_{n-1,t,q,2B}$ assumption, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1] = \mathsf{negl}(\lambda)$.*

Proof. Suppose $\Pr[\mathsf{Hyb}_2 = 1] \geq \varepsilon(\lambda)$ for some non-negligible ε. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ for the SIS assumption:

- On input an SIS challenge matrix $\mathbf{A} \in \mathbb{Z}_q^{(n-1)\times t}$, algorithm $\mathcal{B}$ samples $i^* \xleftarrow{R} [\ell]$ and $\mathbf{a} \xleftarrow{R} \mathbb{Z}_q^t$. It sets $\mathbf{A}_{i^*} = \begin{bmatrix} \mathbf{a}^{\mathsf{T}} \\ \mathbf{A} \end{bmatrix} \in \mathbb{Z}_q^{n\times t}$. Next, algorithm $\mathcal{B}$ samples $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{GenProg}(1^\lambda, 1^\ell, i^*, \mathbf{A}_{i^*})$ and gives $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$ to $\mathcal{A}$.
- Algorithm $\mathcal{A}$ outputs a commitment $\mathbf{c}$, an index i, and two pairs $(x, \boldsymbol{\pi})$ and $(x', \boldsymbol{\pi}')$. Algorithm $\mathcal{B}$ outputs $\boldsymbol{\pi} - \boldsymbol{\pi}'$.

Since the SIS challenger samples $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{(n-1)\times t}$, the distribution of $\mathbf{A}_{i^*}$ is uniform over $\mathbb{Z}_q^{n\times t}$. Thus, algorithm $\mathcal{B}$ perfectly simulates an execution of Hyb_2 for $\mathcal{A}$. Thus, with probability ε, algorithm $\mathcal{A}$ outputs $\mathbf{c} \in \mathbb{Z}_q^n$, $i = i^*$, $x, x' \in \mathcal{M}_{\lambda,\ell}$, and $\boldsymbol{\pi}, \boldsymbol{\pi}' \in \mathbb{Z}_q^t$ where

$$x \neq x' \quad \text{and} \quad \|\boldsymbol{\pi}\|, \|\boldsymbol{\pi}'\| \leq B \quad \text{and} \quad \mathbf{c} = \mathbf{A}_{i^*}\boldsymbol{\pi} - x\mathbf{e}_1 = \mathbf{A}_{i^*}\boldsymbol{\pi}' - x'\mathbf{e}_1.$$

Here, we have implicitly used the fact that the $(i^*)^{\text{th}}$ matrix output by $\mathsf{ExpandLocal}(1^\lambda, i, \mathsf{crs})$ is $\mathbf{A}_{i^*}$, which is guaranteed by somewhere programmability of Π_{samp} and the local expansion property (Definition 4.4). This means

$$\mathbf{A}_{i^*}(\boldsymbol{\pi} - \boldsymbol{\pi}') = (x - x')\mathbf{e}_1 = \begin{bmatrix} x - x' \\ \mathbf{0}^{n-1} \end{bmatrix}. \tag{6.1}$$

Since $x \neq x'$, we conclude that $\boldsymbol{\pi} - \boldsymbol{\pi}' \neq \mathbf{0}$. Since $\mathbf{A}_{i^*} = \begin{bmatrix} \mathbf{a}^{\mathsf{T}} \\ \mathbf{A} \end{bmatrix}$, Eq. (6.1) now implies that $\mathbf{A}(\boldsymbol{\pi} - \boldsymbol{\pi}') = \mathbf{0}^{n-1}$. Thus $\boldsymbol{\pi} - \boldsymbol{\pi}'$ is a non-trivial SIS solution. Finally, $\|\boldsymbol{\pi}\|, \|\boldsymbol{\pi}'\| \leq B$, so $\|\boldsymbol{\pi} - \boldsymbol{\pi}'\| \leq 2B$ and algorithm $\mathcal{B}$ succeeds in breaking SIS with the same advantage ε. □

Combining Lemmas 6.5 to 6.7, we conclude that $\Pr[\mathsf{Hyb}_0(\mathcal{A}) = 1] \leq \ell \cdot \mathsf{negl}(\lambda)$. Since $\ell = \mathsf{poly}(\lambda)$, computational binding holds. □

Theorem 6.8 (Statistical Hiding)**.** *Suppose Π_{samp} supports simulatable openings (Definition 4.5) and moreover, that $n \geq \lambda$ and q is prime. Then Construction 6.2 satisfies statistical hiding.*

Proof. Since Π_{samp} supports simulatable openings, let GenTD be the trapdoor generator algorithm. We construct an efficient simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$ as follows:

- $\mathcal{S}_0(1^\lambda, 1^\ell)$: On input the security parameter 1^λ and the vector dimension 1^ℓ, the simulator first samples $(\mathsf{crs}_{\mathsf{samp}}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell) \leftarrow \mathsf{GenTD}(1^\lambda, 1^\ell)$. Next, it samples $\mathbf{c} \xleftarrow{R} \mathbb{Z}_q^n$. The simulator also initializes an empty dictionary D to keep track of indices and openings. Finally, it outputs the common reference string $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$, the commitment $\mathbf{c}$, and the simulation state $\mathsf{st} = (1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}}, \mathbf{c}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell, \mathrm{D})$.
- $\mathcal{S}(\mathsf{st}, i, x_i)$: On input the simulation state $\mathsf{st} = (1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}}, \mathbf{c}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell, \mathrm{D})$, an index $i \in [\ell]$, and an input $x_i \in \mathcal{M}_{\lambda,\ell}$, the simulator first checks if there is a mapping $(i \mapsto \boldsymbol{\pi}_i)$ in D. If so, it replies with $\boldsymbol{\pi}_i$. Otherwise, the simulator computes $\mathbf{A}_i \leftarrow \mathsf{ExpandLocal}(1^\lambda, \mathsf{crs}_{\mathsf{samp}}, i)$ and samples $\boldsymbol{\pi}_i \leftarrow \mathsf{SamplePre}(\mathbf{A}_i, \mathbf{T}_i, x_i\mathbf{e}_1 + \mathbf{c})$. It adds the mapping $(i \mapsto \boldsymbol{\pi}_i)$ to D and outputs the opening $\boldsymbol{\pi}_i$ together with the updated state $\mathsf{st} = (1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}}, \mathbf{c}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell, \mathrm{D})$.

We now show that this simulator $\mathcal{S}$ satisfies the statistical hiding definition. Take any (possibly unbounded) adversary $\mathcal{A}$. We proceed via a hybrid argument:

- Hyb_0: This is the distribution $\mathsf{Real}_{\mathcal{A}}(\lambda, \ell)$. Specifically, in this experiment, the challenger proceeds as follows:
  - The challenger samples $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$ and gives $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$ to $\mathcal{A}$.
  - Algorithm $\mathcal{A}$ outputs a vector $\mathbf{x}$ and the challenger computes $(\mathbf{c}, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$. Specifically, the challenger first computes $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$. Then it samples $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c}) \leftarrow \mathsf{SampleMultPre}(\mathsf{td}, x_1\mathbf{e}_1, \ldots, x_\ell\mathbf{e}_1)$. The challenger responds to $\mathcal{A}$ with the commitment $\mathbf{c}$.
  - Whenever the adversary requests an opening on an index $i \in [\ell]$, the challenger replies with $\boldsymbol{\pi}_i$.
  - At the end of the game, algorithm $\mathcal{A}$ outputs a bit $b \in \{0, 1\}$ which is the output of the experiment.
- Hyb_1: Same as Hyb_0, except the challenger changes how it samples the commitment and the openings:
  - The challenger samples $\mathsf{crs}_{\mathsf{samp}} \leftarrow \mathsf{Gen}(1^\lambda, 1^\ell)$ and gives $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$ to $\mathcal{A}$.
  - Algorithm $\mathcal{A}$ outputs a vector $\mathbf{x}$. The challenger responds to $\mathcal{A}$ with the commitment $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$. Then, the challenger computes $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$ and for each $i \in [\ell]$, $\boldsymbol{\pi}_i \leftarrow (\mathbf{A}_i)_s^{-1}(x_i\mathbf{e}_1 + \mathbf{c})$.
  - The rest of the experiment proceeds as in Hyb_0.
- Hyb_2: Same as Hyb_1, except the challenger changes how it samples the common reference string:
  - The challenger samples $(\mathsf{crs}_{\mathsf{samp}}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell) \leftarrow \mathsf{GenTD}(1^\lambda, 1^\ell)$ and gives $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$ to $\mathcal{A}$.
  - Algorithm $\mathcal{A}$ outputs a vector $\mathbf{x}$. The challenger responds to $\mathcal{A}$ with the commitment $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$. Then, the challenger computes $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$ and for each $i \in [\ell]$, $\boldsymbol{\pi}_i \leftarrow (\mathbf{A}_i)_s^{-1}(x_i\mathbf{e}_1 + \mathbf{c})$.
  - The rest of the experiment proceeds as in Hyb_0.
- Hyb_3: Same as Hyb_2, except the challenger changes how it constructs the openings:
  - The challenger samples $(\mathsf{crs}_{\mathsf{samp}}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell) \leftarrow \mathsf{GenTD}(1^\lambda, 1^\ell)$ and gives $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$ to $\mathcal{A}$.
  - Algorithm $\mathcal{A}$ outputs a vector $\mathbf{x}$. The challenger responds to $\mathcal{A}$ with the commitment $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$. Then, the challenger computes $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathsf{td}) = \mathsf{Expand}(1^\lambda, 1^\ell, \mathsf{crs}_{\mathsf{samp}})$ and for each $i \in [\ell]$, it samples $\boldsymbol{\pi}_i \leftarrow \mathsf{SamplePre}(\mathbf{A}_i, \mathbf{T}_i, x_i\mathbf{e}_1 + \mathbf{c}, s)$.
  - The rest of the experiment proceeds as in Hyb_0.
- Hyb_4: Same as Hyb_3, except the challenger changes how it computes $\mathbf{A}_i$:
  - The challenger samples $(\mathsf{crs}_{\mathsf{samp}}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell) \leftarrow \mathsf{GenTD}(1^\lambda, 1^\ell)$ and gives $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$ to $\mathcal{A}$.
  - Algorithm $\mathcal{A}$ outputs a vector $\mathbf{x}$. The challenger responds to $\mathcal{A}$ with the commitment $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$. For each $i \in [\ell]$, the challenger computes $\mathbf{A}_i = \mathsf{ExpandLocal}(1^\lambda, \mathsf{crs}_{\mathsf{samp}}, i)$ and $\boldsymbol{\pi}_i \leftarrow \mathsf{SamplePre}(\mathbf{A}_i, \mathbf{T}_i, x_i\mathbf{e}_1 + \mathbf{c}, s)$.
  - The rest of the experiment proceeds as in Hyb_0.
- Hyb_5: Same as Hyb_4, except the challenger samples $\boldsymbol{\pi}_i$ only when the adversary requests an opening on index $i \in [\ell]$:
  - The challenger samples $(\mathsf{crs}_{\mathsf{samp}}, \mathbf{T}_1, \ldots, \mathbf{T}_\ell) \leftarrow \mathsf{GenTD}(1^\lambda, 1^\ell)$ and gives $\mathsf{crs} = (1^\lambda, \ell, \mathsf{crs}_{\mathsf{samp}})$ to $\mathcal{A}$. The challenger also initializes an (empty) dictionary D to keep track of indices and openings.
  - Algorithm $\mathcal{A}$ outputs a vector $\mathbf{x}$. The challenger responds to $\mathcal{A}$ with the commitment $\mathbf{c} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$.
  - Whenever the adversary requests an opening on an index $i \in [\ell]$, the challenger first checks if there is a mapping $(i \mapsto \boldsymbol{\pi}_i)$ in D. If so, it replies with $\boldsymbol{\pi}_i$. Otherwise, the challenger computes $\mathbf{A}_i = \mathsf{ExpandLocal}(1^\lambda, \mathsf{crs}_{\mathsf{samp}}, i)$ and samples $\boldsymbol{\pi}_i \leftarrow \mathsf{SamplePre}(\mathbf{A}_i, \mathbf{T}_i, x_i\mathbf{e}_1 + \mathbf{c}, s)$. It adds the mapping $(i \mapsto \boldsymbol{\pi}_i)$ to D and gives $\boldsymbol{\pi}_i$ to $\mathcal{A}$.

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

This is the experiment $\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(\lambda, \ell)$.

We write $\mathsf{Hyb}_i(\mathcal{A},\mathcal{S})$ to denote the random variable (indexed implicitly by the security parameter λ) corresponding to the output of Hyb_i with adversary $\mathcal{A}$ and simulator $\mathcal{S}$. We now analyze each pair of hybrid experiments:

Lemma 6.9. *Suppose Π_{samp} satisfies the preimage distribution property. Then* $\mathsf{Hyb}_0(\mathcal{A},\mathcal{S}) \overset{s}{\approx} \mathsf{Hyb}_1(\mathcal{A},\mathcal{S})$.

Proof. By the preimage sampling property, with overwhelming probability over the choice of $\mathsf{crs}_{\mathsf{samp}}$, the distribution of $(\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_\ell, \mathbf{c})$ output by $\mathsf{SampleMultPre}(\mathsf{td}, x_1\mathbf{e}_1, \ldots, x_\ell\mathbf{e}_1)$ is statistically close to the distribution obtained by sampling $\mathbf{c} \leftarrow \mathbb{Z}_q^n$ and $\boldsymbol{\pi}_i \leftarrow (\mathbf{A}_i)_s^{-1}(x_i\mathbf{e}_1 + \mathbf{c})$ for all $i \in [\ell]$. Thus, $\mathsf{Hyb}_0(\mathcal{A},\mathcal{S})$ and $\mathsf{Hyb}_1(\mathcal{A},\mathcal{S})$ are statistically indistinguishable. □

Lemma 6.10. *Suppose Π_{samp} has simulatable openings. Then,* $\mathsf{Hyb}_1(\mathcal{A},\mathcal{S}) \overset{s}{\approx} \mathsf{Hyb}_2(\mathcal{A},\mathcal{S})$.

Proof. By definition of mode indistinguishability (Definition 4.5), the distributions of $\mathsf{crs}_{\mathsf{samp}}$ output by $\mathsf{Gen}(1^\lambda, 1^\ell)$ and $\mathsf{GenTD}(1^\lambda, 1^\ell)$ are statistically indistinguishable. □

Lemma 6.11. *Suppose Π_{samp} has simulatable openings. Then,* $\mathsf{Hyb}_2(\mathcal{A},\mathcal{S}) \overset{s}{\approx} \mathsf{Hyb}_3(\mathcal{A},\mathcal{S})$.

Proof. By the trapdoor generation property (Definition 4.5), we have that $\mathbf{A}_i\mathbf{T}_i = \mathbf{G}$ and moreover, that $\|\mathbf{T}_i\| \le s/(t \log n)$. By Theorem 3.9, this means the distribution of $\boldsymbol{\pi}_i \leftarrow \mathsf{SamplePre}(\mathbf{A}_i, \mathbf{T}_i, x_i\mathbf{e}_1 + \mathbf{c}, s)$ is statistically close to sampling $\boldsymbol{\pi}_i \leftarrow (\mathbf{A}_i)_s^{-1}(x_i\mathbf{e}_1 + \mathbf{c})$. This holds for all $i \in [\ell]$. Since $\ell = \mathsf{poly}(\lambda)$, we conclude that $\mathsf{Hyb}_2(\mathcal{A},\mathcal{S})$ and $\mathsf{Hyb}_3(\mathcal{A},\mathcal{S})$ are statistically indistinguishable by a hybrid argument. □

Lemma 6.12. *Suppose* $\mathsf{ExpandLocal}$ *is a correct local-expansion procedure. Then,* $\mathsf{Hyb}_3(\mathcal{A},\mathcal{S}) \equiv \mathsf{Hyb}_4(\mathcal{A},\mathcal{S})$.

Proof. Immediate by correctness of the local expansion procedure (Definition 4.4). □

Lemma 6.13. *It holds that* $\mathsf{Hyb}_4(\mathcal{A},\mathcal{S}) \equiv \mathsf{Hyb}_5(\mathcal{A},\mathcal{S})$.

Proof. The only difference between these experiments is the order in which the challenger samples different components. As such, the output of these two experiments is identically distributed. □

Statistical hiding now follows by combining Lemmas 6.9 to 6.13. □

Theorem 6.14 (Succinctness)**.** *Suppose $n \log q \le \mathsf{poly}(\lambda, \log \ell)$ and $t \log q \le \mathsf{poly}(\lambda, \log \ell)$. Then Construction 6.2 is succinct.*

Proof. This is immediate from the assumptions. Namely, each commitment in Construction 6.2 is an element $\mathbf{c} \in \mathbb{Z}_q^n$, which has size $|\mathbf{c}| = n \log q \le \mathsf{poly}(\lambda, \log \ell)$. Similarly, each opening $\boldsymbol{\pi}_i \in \mathbb{Z}_q^t$ has length at most $t \log q \le \mathsf{poly}(\lambda, \log \ell)$. □

Parameter instantiations. Let λ be a security parameter and ℓ be the vector dimension. We provide one possible instantiation of the parameters in Construction 6.2 to satisfy Theorems 6.3, 6.4, 6.8 and 6.14. In the following, we will assume that ℓ is polynomially-bounded in λ (i.e., $\ell \le \lambda^c$ for some constant $c \in \mathbb{N}$).

- We set the lattice dimension to be $n = \lambda$.
- We set $t = 3n \lceil \log q \rceil \cdot (\lceil \log \ell \rceil + 1)$. When setting parameters, we work under the assumption that $\log q \le k \log \lambda$ for some constant $k > 0$. It is easy to check that this is satisfied by our final instantiation. In this case, $t = O(\lambda \log \ell \log \lambda) = O(\lambda \log^2 \lambda)$ since $\log \ell \le c \log \lambda$.
- We set $s = (\ell t + 3n \lceil \log q \rceil) \log(\ell n) = O(\ell \lambda \log^3 \lambda)$.
- We set $B = \sqrt{t} s = O(\ell \lambda^{3/2} \log^4 \lambda)$.

- We choose a prime $q = 2B \cdot \mathsf{poly}(n)$ where the $\mathsf{SIS}_{n-1,t,q,2B}$ assumption holds. In this case, $\log q = O(\log \lambda + \log \ell)$. Since $\log \ell \leq c \log \lambda$, we can bound $(\log q)$ by $(k \log \lambda)$ for some sufficiently-large constant $k > 0$, as required.
- We instantiate $(\mathsf{Gen}, \mathsf{GenTD}, \mathsf{Expand}, \mathsf{ExpandLocal})$ with the (n, t, q, s)-shifted multi-preimage trapdoor sampler from Theorem 4.7 (Construction 4.6).

With this setting of parameters, we obtain a vector commitment scheme over $\mathbb{Z}_q^\ell$ with the following properties:

- **CRS size:** From Theorem 4.7, Construction 6.2 supports a transparent setup and the size of the CRS is $\lambda + \log \ell + nt \log q = O(\lambda^2 \log^3 \lambda)$.
- **Commitment size:** A commitment in Construction 6.2 consists of a vector $\mathbf{c} \in \mathbb{Z}_q^n$ which has size $n \log q = O(\lambda \log \lambda)$.
- **Opening size:** An opening in Construction 6.2 consists of a vector $\boldsymbol{\pi}_i \in \mathbb{Z}_q^t$, which has size $t \log q = O(\lambda \log^3 \lambda)$.

We summarize the instantiation in the following corollary:

Corollary 6.15 (Vector Commitments from SIS)**.** *Let λ be a security parameter. Then, for all polynomials $\ell = \ell(\lambda)$, under the SIS assumption with a polynomial noise bound $\beta = \mathsf{poly}(\lambda, \ell)$ and a polynomial modulus $q = \mathsf{poly}(\lambda, \ell)$, there exists a vector commitment scheme over $\mathbb{Z}_q^\ell$ with a transparent CRS of size $O(\lambda^2 \log^3 \lambda) = \mathsf{poly}(\lambda)$, commitments of size $O(\lambda \log \lambda)$ and openings of size $O(\lambda \log^3 \lambda)$. The vector commitment is computationally binding and statistically hiding.*

Acknowledgments

Brent Waters is supported by NSF CNS-1908611, CNS-2318701, and a Simons Investigator award. David J. Wu is supported by NSF CNS-2140975, CNS-2318701, a Microsoft Research Faculty Fellowship, and a Google Research Scholar award.

References

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

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

[ACL+22] Martin R. Albrecht, Valerio Cini, Russell W. F. Lai, Giulio Malavolta, and Sri AravindaKrishnan Thyagarajan. Lattice-based SNARKs: Publicly verifiable, preprocessing, and recursively composable. In *CRYPTO*, 2022.

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

[AP09] Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. In *STACS*, 2009.

[AR20] Shashank Agrawal and Srinivasan Raghuraman. KVaC: Key-value commitments for blockchains and beyond. In *ASIACRYPT*, 2020.

[BCFL23] David Balbás, Dario Catalano, Dario Fiore, and Russell W. F. Lai. Chainable functional commitments for unbounded-depth circuits. In *TCC*, 2023.

[BGG+14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *EUROCRYPT*, 2014.

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

[CCH$^+$19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-shamir: from practice to theory. In *STOC*, 2019.

[CCRR18] Ran Canetti, Yilei Chen, Leonid Reyzin, and Ron D. Rothblum. Fiat-shamir and correlation intractability from strong kdm-secure encryption. In *EUROCRYPT*, 2018.

[CF13] Dario Catalano and Dario Fiore. Vector commitments and their applications. In *PKC*, 2013.

[CFG$^+$20] Matteo Campanelli, Dario Fiore, Nicola Greco, Dimitris Kolonelos, and Luca Nizzardo. Incrementally aggregatable vector commitments and applications to verifiable decentralized storage. In *ASIACRYPT*, 2020.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4), 2004.

[CHK03] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In *EUROCRYPT*, 2003.

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

[CLM23] Valerio Cini, Russell W. F. Lai, and Giulio Malavolta. Lattice-based succinct arguments from vanishing polynomials - (extended abstract). In *CRYPTO*, 2023.

[dCP23] Leo de Castro and Chris Peikert. Functional commitments for all functions, with transparent setup and from SIS. In *EUROCRYPT*, 2023.

[DHM$^+$24] Fangqi Dong, Zihan Hao, Ethan Mook, Hoeteck Wee, and Daniel Wichs. Laconic function evaluation and ABE for RAMs from (ring-)LWE. In *CRYPTO*, 2024.

[DJJ24] Quang Dao, Aayush Jain, and Zhengzhong Jin. Non-interactive zero-knowledge from LPN and MQ. In *CRYPTO*, pages 321–360, 2024.

[FLS90] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *FOCS*, 1990.

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero knowledge for NP. In *EUROCRYPT*, 2006.

[GOS12] Jens Groth, Rafail Ostrovsky, and Amit Sahai. New techniques for noninteractive zero-knowledge. *J. ACM*, 59(3), 2012.

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

[GRWZ20] Sergey Gorbunov, Leonid Reyzin, Hoeteck Wee, and Zhenfei Zhang. Pointproofs: Aggregating proofs for multiple vector commitments. In *ACM CCS*, 2020.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO*, 2013.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In *STOC*, 2013.

[GVW15a] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from LWE. In *CRYPTO*, 2015.

[GVW15b] Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. In *STOC*, 2015.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4), 1999.

[HL18] Justin Holmgren and Alex Lombardi. Cryptographic hashing from strong one-way functions (or: One-way product functions and their applications). In *FOCS*, 2018.

[HLOV11] Brett Hemenway, Benoît Libert, Rafail Ostrovsky, and Damien Vergnaud. Lossy encryption: Constructions from general assumptions and efficient selective opening chosen ciphertext security. In *ASIACRYPT*, 2011.

[JJ21] Abhishek Jain and Zhengzhong Jin. Non-interactive zero knowledge from sub-exponential DDH. In *EUROCRYPT*, pages 3–32, 2021.

[KRR17] Yael Tauman Kalai, Guy N. Rothblum, and Ron D. Rothblum. From obfuscation to the security of fiat-shamir for proofs. In *CRYPTO*, 2017.

[KZG10] Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *ASIACRYPT*, 2010.

[LLNW16] Benoît Libert, San Ling, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In *EUROCRYPT*, 2016.

[LM19] Russell W. F. Lai and Giulio Malavolta. Subvector commitments with application to succinct arguments. In *CRYPTO*, 2019.

[LPWW20] Benoît Libert, Alain Passelègue, Hoeteck Wee, and David J. Wu. New constructions of statistical NIZKs: Dual-mode DV-NIZKs and more. In *EUROCRYPT*, 2020.

[LRY16] Benoît Libert, Somindu C. Ramanna, and Moti Yung. Functional commitment schemes: From polynomial commitments to pairing-based accumulators from simple assumptions. In *ICALP*, 2016.

[LY10] Benoît Libert and Moti Yung. Concise mercurial vector commitments and independent zero-knowledge sets with short proofs. In *TCC*, 2010.

[Mer87] Ralph C. Merkle. A digital signature based on a conventional encryption function. In *CRYPTO*, 1987.

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

[PPS21] Chris Peikert, Zachary Pepin, and Chad Sharp. Vector and functional commitments from lattices. In *TCC*, 2021.

[PR06] Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC*, 2006.

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In *CRYPTO*, 2019.

[PSTY13] Charalampos Papamanthou, Elaine Shi, Roberto Tamassia, and Ke Yi. Streaming authenticated data structures. In *EUROCRYPT*, 2013.

[PW08] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In *STOC*, 2008.

[QRW19] Willy Quach, Ron D. Rothblum, and Daniel Wichs. Reusable designated-verifier NIZKs for all NP from CDH. In *EUROCRYPT*, 2019.

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

[TAB$^+$20] Alin Tomescu, Ittai Abraham, Vitalik Buterin, Justin Drake, Dankrad Feist, and Dmitry Khovratovich. Aggregatable subvector commitments for stateless cryptocurrencies. In *SCN*, 2020.

[TXN20] Alin Tomescu, Yu Xia, and Zachary Newman. Authenticated dictionaries with cross-incremental proof (dis)aggregation. *IACR Cryptol. ePrint Arch.*, 2020.

[Wat24] Brent Waters. A new approach for non-interactive zero-knowledge from learning with errors. In *STOC*, 2024.

[WW23a] Hoeteck Wee and David J. Wu. Lattice-based functional commitments: Fast verification and cryptanalysis. In *ASIACRYPT*, 2023.

[WW23b] Hoeteck Wee and David J. Wu. Succinct vector, polynomial, and functional commitments from lattices. In *EUROCRYPT*, 2023.