Adaptively Sound Zero-Knowledge SNARKs for UP

Surya Mathialagan
MIT*

Spencer Peters
Cornell University†

Vinod Vaikuntanathan
MIT‡

February 16, 2024

Abstract

We study succinct non-interactive arguments (SNARGs) and succinct non-interactive arguments of knowledge (SNARKs) for the class UP in the reusable designated verifier model. UP is an expressive subclass of NP consisting of all NP languages where each instance has at most one witness; a designated verifier SNARG (dvSNARG) is one where verification of the SNARG proof requires a private verification key; and such a dvSNARG is reusable if soundness holds even against a malicious prover with oracle access to the (private) verification algorithm.

Our main results are as follows.

1. A reusably and adaptively sound zero-knowledge (zk) dvSNARG for UP, from subexponential LWE and evasive LWE (a relatively new but popular variant of LWE). Our SNARGs achieve very short proofs of length $(1 + o(1)) \cdot \lambda$ bits for $2^{-\lambda}$ soundness error.
2. A generic transformation that lifts any "Sahai-Waters-like" (zk) SNARG to an adaptively sound (zk) SNARG, in the *designated-verifier* setting. In particular, this shows that the Sahai-Waters SNARG for NP is adaptively sound in the designated verifier setting, assuming subexponential hardness of the underlying assumptions. The resulting SNARG proofs have length $(1 + o(1)) \cdot \lambda$ bits for $2^{-\lambda}$ soundness error. Our result sidesteps the Gentry-Wichs barrier for adaptive soundness by employing an exponential-time security reduction.
3. A generic transformation, building on the work of Campanelli, Ganesh, Khoshakhlagh and Siim, that lifts any adaptively sound (zk) SNARG for UP to an adaptively sound (zk) SNARK for UP, while preserving zero-knowledge. The resulting SNARK achieves the strong notion of black-box extraction. There are barriers to achieving such SNARKs for all of NP from falsifiable assumptions, so our restriction to UP is, in a sense, necessary.

Applying (3) to our SNARG for UP from evasive LWE (1), we obtain a reusably and adaptively sound designated-verifier zero-knowledge SNARK for UP from subexponential LWE and evasive LWE. Moreover, applying both (2) and (3) to the Sahai-Waters SNARG, we obtain the same result from LWE, subexponentially secure one-way functions, and subexponentially secure indistinguishability obfuscation. Both constructions have succinct proofs of size $\mathsf{poly}(\lambda)$. These are the first SNARK constructions (even in the designated-verifier setting) for a non-trivial subset of NP from (sub-exponentially) falsifiable assumptions.

*smathi@mit.edu
†sp2473@cornell.edu
‡vinodv@mit.edu

Contents

1 Introduction **1**
- 1.1 Concurrent Work . . . 5
- 1.2 Organization of the Paper . . . 5

2 Technical Overview **6**
- 2.1 Our SNARG for UP from Evasive LWE . . . 6
- 2.2 Achieving Adaptive Security for "Sahai-Waters-like" SNARGs . . . 12
- 2.3 From Adaptive SNARGs for UP to SNARKs for UP . . . 14

3 Preliminaries **16**
- 3.1 Notation . . . 16
- 3.2 LWE Assumption . . . 17
- 3.3 Security Parameters . . . 17
- 3.4 Trapdoor and Pre-image Sampling . . . 17
- 3.5 Primitives . . . 18
  - 3.5.1 Pseudorandom Functions . . . 18
  - 3.5.2 SNARGs, SNARKs and NIZKs . . . 21
  - 3.5.3 Fully Homomorphic Encryption . . . 23
  - 3.5.4 Public-key Encryption . . . 24
  - 3.5.5 Indistinguishability Obfuscation . . . 25

4 Obfuscating Matrix PRFs with Noise (σ-Matrix PRFs) **25**
- 4.1 Matrix Branching Programs . . . 26
- 4.2 σ-Matrix PRFs . . . 26
- 4.3 Transforming Read-c PRFs into Read-Once PRFs . . . 27
- 4.4 GGH Encodings . . . 28
- 4.5 Evasive LWE . . . 29
- 4.6 Obfuscating Sufficiently Secure σ-matrix PRFs . . . 30

5 Witness PRFs for UP **35**
- 5.1 Definitions . . . 35
- 5.2 Construction . . . 36

6 Designated Verifier zk-SNARGs from Witness PRFs **40**
- 6.1 Construction . . . 40

7 Adaptively Secure Designated Verifier SNARG for NP **43**

8 SNARK for UP **47**
- 8.1 Our Transformation from SNARG for UP to SNARK for UP . . . 48
- 8.2 Proof of Theorem 8.1 . . . 51

A Deferred proofs **59**
A.1 Proof of Lemma 3.8 59
A.2 Proof of Lemma 3.14 60
A.3 Proof of Lemma 4.7 60

1 Introduction

Can we generate short proofs of correctness, consisting of a few bytes, of long and possibly non-deterministic computations, consisting of billions of steps? The notion of succinct non-interactive arguments (SNARGs) [Mic00, GW11] gives us a positive answer to this question by introducing two relaxations to the problem statement: *computational soundness* and the existence of a trusted *common random, or reference, string*. The construction of SNARGs has been a tremendously active area of research in cryptography, leading us to three broadly defined clusters of constructions.

The first cluster of constructions, starting from [Mic00] (building on Kilian's succinct interactive arguments [Kil92]) showed SNARGs (and even SNARKs, succinct non-interactive arguments of knowledge) for all NP languages with security in the random oracle model or under strong, non-falsifiable, knowledge assumptions [Gro10, BCCT12, BCCT13, BCI+13]. This class of SNARG constructions have since then been engineered to have superior concrete efficiency (e.g. in a line of work starting from [PHGR13, BCG+13, BCTV14]) and have been deployed in the real world [BCG+14]. Their very popularity and increasingly widespread use raises the question of whether one can place SNARGs (and SNARKs) on a more solid footing of security, with constructions based on falsifiable assumptions. The rest of this paper will focus on constructions in the plain model (that is, eschewing random oracles).

The second cluster of constructions is really a singular beautiful construction, due to Sahai and Waters [SW14], that builds a SNARG for NP from a sub-exponentially falsifiable assumption, namely the existence of indistinguishability obfuscation (IO) schemes.[1] The Sahai-Waters SNARG comes with very short proofs, essentially $O(\lambda)$ bits for a soundness error of $2^{-\lambda}$, but a long common reference string (CRS) that is as long as the NP witness. A recent work of Jain and Jin [JJ22] shows how to reduce the size of the CRS for a subclass of $\mathsf{NP} \cap \mathsf{coNP}$. While IO can now be based on relatively well-studied (and falsifiable) cryptographic assumptions [JLS21, JLS22], the constructions themselves are complex and inefficient, which motivates the quest for simpler and more direct constructions. An additional issue is that the constructions of [JLS21, JLS22] are not post-quantum secure.

The third cluster of constructions follows the paradigm of Kalai, Raz and Rothblum [KRR13, KRR14] who construct a *designated verifier* SNARG for P from the learning with errors (LWE) assumption. A designed verifier SNARG is a further relaxation where verification of the SNARG proof requires a private verification key. This has since been built upon by a large body of work over the last decade ([KPY19, CCH+19, CJJ21, JKLV24] and many others), obtaining public verifiability and extending the reach to larger and larger classes of languages. However, until now, the most expressive class for which we can construct a SNARG following this line of work (even in the easier designated verifier setting) is a subclass of $\mathsf{NP} \cap \mathsf{coNP}$ [JKLV24].

Several intriguing questions emerge in light of these constructions.

SNARG for NP from LWE? Can we match the expressivity of the Sahai-Waters SNARG, while basing security only on the well-studied (and presumably post-quantum secure) LWE assumption? The question is open even in the easier, designated verifier (dv), setting.

[1] Indistinguishability obfuscation is not a polynomially falsifiable cryptographic assumption but it is sub-exponentially falsifiable. Alternatively, IO constructions can be based on falsifiable assumptions, e.g. [JLS21, JLS22].

Q1: Can we construct a (dv)SNARG for NP (or large subclasses thereof) from the LWE assumption?

Our first contribution is the construction of a *designated verifier* SNARG with *reusable soundness* for UP (unambiguous nondeterministic polynomial time), a subclass of NP where each instance has a unique witness (for example, the set of all numbers that are a product of two distinct prime numbers), from a variant of LWE (see the next paragraph). In a designated verifier SNARG, one could require soundness against (efficient) malicious provers that either have oracle access to the verification algorithm or not. Soundness against the former (stronger) class of malicious provers is referred to as *reusable soundness*, since the verification key can be safely reused to check polynomially many proofs. If the cheating prover is not allowed access to the verification oracle, one obtains the weaker notion of non-reusable soundness (some SNARG constructions, e.g. [KRR13, KRR14] achieve this weaker notion). Henceforth, when we refer to a designated verifier SNARG, we will always mean one with *reusable soundness*.

Our construction is sound under a relatively new, but popular, variant of LWE called *evasive LWE* [Wee22, Tsa22, VWW22, HLL23]. Informally, evasive LWE acts as a "bridge" between the standard learning with errors (LWE) assumption and the existence of indistinguishability obfuscation (IO). Evasive LWE has proved fruitful in constructing several cryptographic primitives such as witness encryption and null-IO [Tsa22, VWW22], optimal broadcast encryption [Wee22], multi-authority ABE [WWW22] and unbounded-depth attribute-based encryption [HLL23] that we have so far been able to construct from IO but not from the LWE assumption alone. These constructions also provide a potential pathway to achieving constructions based on plain LWE. (For a description of the evasive LWE assumption, we refer the reader to Eq. (1) in Section 2.1).

Informal Theorem 1. Assuming sub-exponential LWE and evasive LWE, there exists an reusably sound designated-verifier zero-knowledge SNARG for UP. The length of the SNARG proof is $\lambda + \omega(\log(|x| + |w| + \lambda))$ to achieve a soundness error of $2^{-\lambda}$. The length of the common reference string is $\mathsf{poly}(\lambda, |x|, |w|)$.

We achieve this result via the construction of a new average-case obfuscator for a class of function families with pseudorandom truth tables, called σ-PRF obfuscation, from evasive LWE (see Section 4). We consider the notion and construction of σ-PRF obfuscation of potentially independent interest; for example, our construction captures and generalizes many existing constructions of primitives from LWE such as shift hiding functions and constrained pseudorandom functions.

Adaptive Soundness. A second question relates to the notion of adaptive soundness which allows a cheating prover to pick the instance on which she decides to cheat, based on the common reference string. On the one hand, adaptive soundness is the desired form of security as many applications require one to publish a CRS and being able to handle NP statements chosen after the fact. On the other hand, constructions from the second and third cluster typically do not satisfy adaptive soundness. In particular, the Sahai-Waters construction is non-adaptively sound; and the construction of Jin, Kalai, Lombardi and Vaikuntanathan [JKLV24] for a subclass of $\mathsf{NP} \cap \mathsf{coNP}$ is also inherently non-adaptively sound. This question is interesting even for the easier designated

verifier setting, even with strong (falsifiable) assumptions, and even for subclasses of NP.[2]

> **Q2:** Can we construct an adaptively secure (dv)SNARG for NP (or an interesting subclass thereof)?

Our second contribution is the construction of an *adaptively* secure reusable dvSNARG from evasive LWE. Indeed, we show that the construction from Theorem 1 is adaptively sound *as-is*.

As a third contribution, we show a general lifting theorem, based on careful complexity leveraging, that takes any non-adaptively sound "Sahai-Waters-type" dvSNARG construction, and shows that it is also adaptively sound, *without increasing the proof length* (rather, only the length of the CRS). Our lifting theorem applies to [SW14] as well as to our evasive LWE-based SNARG construction. To the best of our knowledge, this lifting theorem is new, and crucially leverages the designated verifier setting.

Informal Theorem 2. Assuming the sub-exponential hardness of underlying assumptions, SNARGs in the "style of Sahai-Waters" can be made adaptively secure in the designated verifier setting, with proof size $\lambda + \omega(\log(|x| + |w| + \lambda))$ to achieve a soundness error of $2^{-\lambda}$.

In particular, we show in Section 7 that the Sahai-Waters SNARG [SW14] itself can be made adaptively secure in the designated verifier setting by assuming the sub-exponential hardness of the underlying indistinguishability obfuscation scheme and puncturable PRF family.

On the Gentry-Wichs Barrier. It is instructive to pause and ponder why our result does not contradict the Gentry-Wichs (GW) impossibility for SNARGs [GW11]. Gentry and Wichs showed that any construction of an adaptively sound SNARG for a hard language (even a designated verifier one, and even one with a long CRS [CGKS23]) cannot be based on a falsifiable assumption. In slightly more detail, any purported reduction from solving an instance of a falsifiable assumption to breaking the adaptive soundness of such a SNARG can be turned into an algorithm (that runs in the same time as the reduction) that decides the language without any help. Ergo, if the language is hard, such a reduction does not exist. Our construction seems to overcome the GW impossibility for two reasons, one more fundamental than the other: (1) our reduction runs in time exponential in the witness length, and is thus trivially powerful enough to decide the language; and (2) the evasive LWE assumption, on the face of it, is not falsifiable. The second of these two reasons does not appear to be fundamental, again for two reasons. First, to the best of our knowledge, it is plausible that evasive LWE could one day be proved hard under a falsifiable assumption. In such a world, the only reason why our SNARG construction evades the GW impossibility would be the runtime of the reduction. Secondly, it is worth noting that σ-PRF obfuscation, on which our SNARG for UP is directly based, *is* a sub-exponentially falsifiable assumption, in the same sense that IO is a sub-exponentially falsifiable assumption.

The fact that our reduction runs in time exponential in the witness length affects the length of the CRS but not the length of the proof (the same way it plays out in the [SW14] SNARG.)

[2] The recent, and concurrent, work of Waters and Wu [WW24] achieves an adaptively secure (and publicly verifiable) SNARG for NP using indistinguishability obfuscation. We elaborate on this work in Section 1.1.

SNARKs. A third question relates to the notion of succinct non-interactive *arguments of knowledge*, or SNARKs. SNARKs are more directly useful in several applications than plain SNARGs. They also compose better, e.g. in recursive constructions [BCCT13]. However, we know much less about SNARKs; by a recent result, SNARKs in the plain model with black-box extraction do not exist for all of NP [CGKS23, Kal23]. This leads us to the following question:

Q3: Can we construct a (dv)SNARK for an interesting subclass of NP?

We show a general compiler that takes any *adaptively sound* SNARG for UP and converts it into a non-adaptively extractable SNARK. Given the impossibility result for NP mentioned above [CGKS23], our restriction to UP is essential.

Informal Theorem 3. Assuming sub-exponential LWE, any *adaptively sound* SNARG for UP can be compiled into one with *(non-adaptive) black-box knowledge soundness*. If the original SNARG is publicly verifiable, so is the SNARK. Additionally, if the underlying SNARG is zero-knowledge, the resulting SNARK is also zero-knowledge.

We achieve the weaker form of non-adaptive knowledge extraction; the stronger adaptive form is also known to be impossible w.r.t. black-box extraction for essentially any non-trivial language [CGKS23]. Our transformation here builds heavily on the work of [CGKS23] and corrects two issues with their construction (see Remark 8.4). Additionally, the transformation of [CGKS23] relied on a SNARG for NP, whereas a SNARG for UP is sufficient for our transformation. Our compiler gives us zero knowledge for free. We also note that *adaptive security* of the underlying construction seems crucial for both our transformation and that of [CGKS23]. For more details, see Section 8.

Combining this transformation with our SNARG for UP, we obtain a construction of a reusably and adaptively sound, and non-adaptively extractable, SNARK for UP from evasive LWE.

Informal Theorem 4. Assuming sub-exponential LWE and evasive LWE, there exists a reusable and adaptively sound zero-knowledge SNARK for UP in the designated verifier setting, achieving non-adaptive black-box knowledge soundness with proof size $\mathsf{poly}(\lambda)$. The length of the common reference string is $\mathsf{poly}(\lambda, |x|, |w|)$.

Additionally, by applying our compilers to the Sahai-Waters SNARG with sub-exponential hardness of underlying assumptions, we also obtain the following corollary.

Informal Theorem 5. Assuming LWE, sub-exponentially-secure indistinguishability obfuscation and sub-exponentially-secure one-way functions, there exists a reusable and adaptively sound zero-knowledge SNARK system for UP in the designated verifier setting, achieving black-box knowledge soundness with proof size $\mathsf{poly}(\lambda)$. The length of the common reference string is $\mathsf{poly}(\lambda, |x|, |w|)$.

To the best of our knowledge, these are the first SNARK constructions (even in the designated verifier setting) for a non-trivial subset of NP from (sub-exponential) falsifiable assumptions. Both our SNARG to SNARK for UP compiler and that of Campanelli et al. [CGKS23] seem to require *adaptively sound* underlying SNARGs, and we (along with the concurrent work of Waters and Wu [WW24], see Section 1.1) instantiate the first such SNARGs from (sub-exponentially) falsifiable assumptions.

For a roadmap of our results, see Fig. 1.

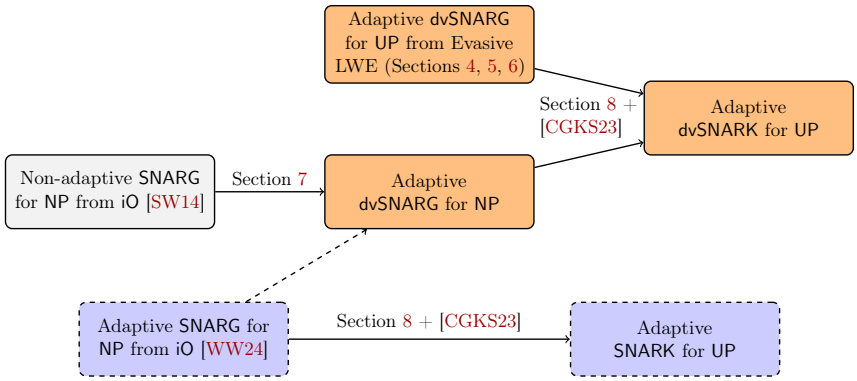

Figure 1: Roadmap of our results and transformations. All of the constructions above are additionally multi-theorem adaptively zero knowledge. Our results are highlighted in orange, and concurrent work and its implications are highlighted in blue. All transformations from our paper are denoted by arrows labeled by the corresponding sections. The dashed arrow represents an immediate consequence.

1.1 Concurrent Work

In concurrent work, Waters and Wu [WW24] modify the Sahai-Waters construction to achieve adaptivity in the *publicly-verifiable setting*. They show the following.

Theorem 1.1 ([WW24, Theorem 1.1])**.** *Assuming (1) either the polynomial hardness of computing discrete logs in a prime-order group or the polynomial hardness of factoring, (2) sub-exponentially-secure indistinguishability obfuscation, and (3) sub-exponentially-secure one-way functions, there exists a* publicly verifiable*, perfectly zero-knowledge* SNARG *for all of* NP*, with proof size* $\mathsf{poly}(\lambda)$*.*

In comparison to our "lifting" theorem (Informal Theorem 3), we note that their transformation introduces a new assumption and requires a *white-box* modification to the Sahai-Waters SNARG. On the other hand, our transformation works for any SNARG in the style of Sahai-Waters (we describe the precise class of SNARGs we mean in Section 2.2), although we are restricted to the (reusable) designated verifier setting.

Combining the Waters-Wu SNARG with our SNARG to SNARK compiler in Informal Theorem 3, or the compiler of [CGKS23], we obtain the corollary that there exists a (publicly verifiable) zkSNARK for all of UP achieving black-box extractability (see Fig. 1). To the best of our knowledge, this is the first publicly verifiable SNARK for a non-trivial subset of NP from (sub-exponential) falsifiable assumptions.

1.2 Organization of the Paper

First, in Section 2, we give high-level descriptions of our SNARG for UP, our adaptive security transformation, and our SNARG-to-SNARK compiler. In Section 3, we state basic definitions and lemmas. In Section 4, we introduce the notion of σ-matrix PRFs, and show how to obfuscate sufficiently secure σ-matrix PRFs using LWE and evasive LWE. In Section 5, we use our σ-matrix PRF obfuscation to construct an adaptively secure witness PRF for UP. In Section 6, we show a generic transformation from a (sufficiently secure) adaptively secure witness PRF for an NP language

L to a reusable, adaptively secure designated-verifier SNARG for L. In Section 7, we show that a class of "Sahai-Waters-type" SNARGs (which includes our SNARG for UP) can be generically upgraded to adaptive soundness *in the designated verifier setting*, if we assume the sub-exponential hardness of the underlying primitives. Finally, in Section 8, we show that any *adaptively sound* SNARG for UP can be generically transformed into a SNARK for UP, with black-box knowledge soundness.

2 Technical Overview

2.1 Our SNARG for UP from Evasive LWE

We first recall the zk-SNARG for NP construction of Sahai and Waters [SW14] from indistinguishability obfuscation and one-way functions. For simplicity, we describe their construction in the designated-verifier model. Given a relation circuit C, the Sahai-Waters common reference string is an obfuscation of the following circuit P. On input x, w, P checks if $C(x, w) = 1$. If yes, P outputs a PRF value $\pi = f_k(x)$, and otherwise it outputs $\perp$. The designated verifier simply stores the PRF key k, and on input (x, π), accepts iff $\pi = f_k(x)$. To show soundness of the SNARG on a non-adaptively chosen $x^* \notin L$, Sahai and Waters use a punctured programming technique to show that $f_k(x^*)$ remains hidden even given an indistinguishability obfuscation of P.

Overview. Our SNARG for UP construction proceeds in two steps, and the second step follows an approach very similar to that of Sahai and Waters. In the first step, we construct a witness PRF [Zha16] for UP. In the second, which we explain next, we generically convert any witness PRF for a NP language L into a SNARG for L. Recall that, intuitively, a witness PRF is a PRF f_k taking instances x as input, along with an evaluation key ek such that

$$\mathsf{Eval}(\mathsf{ek}, x, w) = \begin{cases} f_k(x) & \text{if } R(x, w) = 1 \\ \perp & \text{otherwise.} \end{cases}$$

In particular, ek hides $f_k(x)$ for all $x \notin L$[3]. Given a witness PRF for L, we construct a SNARG for L as follows: the common reference string is the evaluation key ek; the prover on input (x, w) computes $y = f_k(x)$ using ek and w and outputs y, and the designated verifier stores k and on input (x, π), accepts iff $\pi = f_k(x)$. The (adaptive) security of the witness PRF implies the *adaptive* security of our SNARG; for a rough intuition, one can view the witness PRF-based argument as a strengthening of the Sahai-Waters punctured programming argument, where all $x \notin L$ are punctured simultaneously.

Above, the witness PRF evaluation key ek plays the role of an obfuscation of the original PRF; in particular, functionality allows computation of $f_k(x)$ for $x \in L$ given a witness w for x, but security guarantees that the values $f_k(x)$ are hidden for $x \notin L$. The first step—the construction of a witness PRF for UP from LWE and evasive LWE—also relies on an obfuscation notion for PRFs (more precisely, a relaxation of PRFs). To describe the witness PRF construction for UP, it will be helpful

[3]For a reader familiar with constrained PRFs [BW13], a witness PRF is similar to a constrained PRF on a relation circuit $R(x, w)$, except that the output value when $R(x, w) = 1$ depends only on x, not both x and w.

to first describe a simplified construction that does not work. Specifically, it does not work because it requires a strong PRF obfuscation guarantee that we do not know how to obtain.

Simplified Construction of a Witness PRF for UP. First, we instantiate a PRF corresponding to a UP relation R, with the following structure:

$$f_{k_1,k_2}(x,w) = \begin{cases} g_{k_1}(x) & \text{if } R(x,w)=1 \\ g_{k_1}(x)+g_{k_2}(x,w) & \text{otherwise.} \end{cases}$$

where both g_{k_1} and g_{k_2} are PRFs.

Note that if R were not a UP relation, i.e. there exists some $x \in L$ with multiple witnesses, then f_{k_1,k_2} would not be a PRF. In particular, if some instance x had two corresponding witness w_1 and w_2, then we would have $f_{k_1,k_2}(x,w_1) = g_{k_1}(x) = f_{k_1,k_2}(x,w_2)$. This is why our construction only works for UP relations.

Next, we obfuscate f_{k_1,k_2}. For now, we will assume an obfuscation scheme Obf such that $\mathsf{Obf}(f_{k_1,k_2})$ allows the evaluation of f_{k_1,k_2} on any input x, w, while enjoying the strong average-case obfuscation property that there exists a fixed distribution $\mathcal{D}$ such that for random keys k_1, k_2,

$$\mathsf{Obf}(f_{k_1,k_2}) \approx_c \mathcal{D} \ .$$

Notice that this type of obfuscation would certainly not be possible if f_{k_1,k_2} were not a PRF–that is, if the outputs of f_{k_1,k_2} were not indistinguishable from a fixed distribution independent of k_1, k_2.

Given this strong obfuscation, our witness PRF construction is straightforward. The generation algorithm simply samples $k = (k_1, k_2)$ and $\mathsf{ek} = \mathsf{Obf}(f_{k_1,k_2})$; the underlying PRF is $g_{k_1}(x)$, which, for $x \in L$, can be computed given ek and the witness w for x. Security is straightforward, since by obfuscation security, the evaluation key ek can be replaced with an independent sample from $\mathcal{D}$, which completely hides k_1.

From Simplified to Complete Construction However, we cannot quite obtain this strong notion of obfuscation from evasive LWE. In our real construction, we actually obfuscate an object that we call a *σ-matrix PRF*. This is a keyed family of functions $\{f_k\}_k$ that satisfies a relaxation of the definition of a PRF, where each f_k is computable by a matrix branching program. In this technical overview, we will assume for simplicity that all matrix branching programs are read-once; in the full construction, we generalize this to read-c matrix branching programs using more-or-less standard techniques (see Lemma 4.5). Recall that a *(read-once) matrix branching program (MBP)*

$$f := \left(\{\mathbf{M}_{i,b}\}_{i\in[h],b\in\{0,1\}}, \mathbf{u}, \mathbf{v}\right)$$

is a collection of matrices and pair of vectors, all over some ring (in our case, $\mathbb{Z}_q$ for some prime q). Abusing notation, the MBP f is said to compute the function $f : \{0,1\}^h \to \mathbb{Z}_q$ defined by

$$f_k(x) = \mathbf{u}^T \left(\prod_{i\in[h]} \mathbf{M}_{i,x_i} \right) \mathbf{v} \ .$$

The other notion we need is that of a σ-PRF. We define a σ-PRF (in the presence of auxiliary input aux) to be a keyed family of functions $\{f_k\}_k$ satisfying the following pseudorandomness guarantee: letting $\{e_x \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}\}_{x\in\{0,1\}^h}$ be independent Gaussian errors, we have that for a random key k,

$$\{f_k(\mathbf{x}) + e_x\}_{x\in\{0,1\}^h}, \mathsf{aux}(k) \approx_c \{\mathcal{U}(\mathbb{Z}_q)\}_{x\in\{0,1\}^h}, \mathsf{aux}(k) \ .$$

(We note that this is actually the σ-PRF guarantee specialized to adversaries powerful enough to write down the entire truth table. This is the setting we care about, since our proof of obfuscation security will need pseudorandomness against even stronger adversaries.)

Finally, we define a σ-matrix PRF to be a σ-PRF computable by polynomial-sized matrix branching programs. Given a sufficiently secure σ-matrix PRF, our obfuscation will satisfy the strong security guarantee from above: namely, that there exists a fixed distribution $\mathcal{D}$ such that for random keys k,

$$\mathsf{Obf}(f_k) \approx_c \mathcal{D} \ .$$

However, it will only have approximate correctness; that is, knowing $\mathsf{Obf}(f_k)$ will only allow the evaluation of $f_{k_1,k_2}(x,w) + e$, where e is a noise term of width at least σ. This is in fact necessary in general, since otherwise the evaluations would not be pseudorandom, and could be used to break indistinguishability.

Before we describe our obfuscation scheme, let us complete the construction assuming that such a scheme exists. First, notice that approximate correctness has little effect on our witness PRF construction. We simply round the witness PRF values; that is, the underlying PRF becomes $\lfloor g_{k_1}(x) \rceil_p$. Choosing p appropriately, we can guarantee that except with negligible probability, for all $x \in L$ with corresponding witness w,

$$\lfloor f_{k_1,k_2}(x,w) + e \rceil_p = \lfloor g_{k_1}(x) + e \rceil_p = \lfloor g_{k_1}(x) \rceil_p \ ;$$

that is, the witness PRF values $\lfloor g_{k_1}(x) \rceil_p$ can be recovered from the noisy evaluations $f_{k_1,k_2}(x,w)+e$ yielded by the evaluation key $\mathsf{Obf}(f_{k_1,k_2})$.

The only remaining issue is to instantiate a σ-PRF $f_{k_1,k_2}(\cdot,\cdot)$ with the desired structure. In this and in the proof of obfuscation security, we in large part follow the witness encryption construction of Vaikuntanathan, Wee and Wichs [VWW22]. Our starting point is the unrounded BLMR PRF ([BLMR13]). For appropriate parameters, the BLMR PRF is a σ-PRF that has very high levels of security under sub-exponential LWE (see Lemma 3.8). Moreover, since it is defined as a subset product of matrices, it can easily be computed by a matrix branching program. To construct the MBP f_{k_1,k_2}, we carefully combine two BLMR σ-matrix PRFs g_{k_1} and g_{k_2} with a MBP f_R computing the UP relation R. (The latter exists by combining a classical reduction to Circuit-SAT and the classical result due to Barrington that any function computable by logarithmic-depth Boolean circuits can be represented by a read-many MBP.) This completes the construction. Next, we turn to a description of our obfuscation scheme from LWE and evasive LWE.

Constructing an Obfuscation Scheme for σ-Matrix PRFs. To construct our obfuscation scheme, we rely on the evasive LWE assumption, introduced by [Wee22, Tsa22]. We first describe the assumption and then use it to construct our obfuscation scheme. Fix some efficiently samplable

distributions $(\mathbf{S}, \mathbf{P}, \mathsf{aux})$ over $\mathbb{Z}_q^{n' \times n} \times \mathbb{Z}_q^{n \times t} \times \{0,1\}^*$. We want to be able to show statements of the form

$$(\mathbf{SB} + \mathbf{E}, \mathbf{B}^{-1}(\mathbf{P}), \mathsf{aux}) \approx_c (\mathbf{C}, \mathbf{B}^{-1}(\mathbf{P}), \mathsf{aux})$$

where $\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{C} \leftarrow \mathbb{Z}_q^{n' \times m}$ are uniformly random. (The reader should think of parameters $t \geq m = \Omega(n \log q)$ so that $\mathbf{P}$ is wider than $\mathbf{B}$.) There are two distinguishing strategies in the literature:

- distinguish $\mathbf{SB} + \mathbf{E}$ from $\mathbf{C}$ given aux;
- compute $(\mathbf{SB} + \mathbf{E}) \cdot \mathbf{B}^{-1}(\mathbf{P}) = \mathbf{SP} + \mathbf{E} \cdot \mathbf{B}^{-1}(\mathbf{P}) \approx \mathbf{SP}$ and distinguish the latter from uniform, again given aux.

The evasive LWE assumption essentially asserts that these are the only distinguishing attacks. Namely,

$$\textbf{if} \qquad (\mathbf{SB} + \mathbf{E}, \mathbf{SP} + \mathbf{E}', \mathsf{aux}) \approx_c (\mathbf{C}, \mathbf{C}', \mathsf{aux}), \tag{1}$$

$$\textbf{then} \qquad (\mathbf{SB} + \mathbf{E}, \mathbf{B}^{-1}(\mathbf{P}), \mathsf{aux}) \approx_c (\mathbf{C}, \mathbf{B}^{-1}(\mathbf{P}), \mathsf{aux}) \tag{2}$$

where $\mathbf{E}'$ is a fresh noise matrix of not-too-large magnitude relative to $\mathbf{E}$. We refer to Eq. (1) as the *pre-condition*, and Eq. (2) as the *post-condition*. We give a formal definition of Evasive LWE in Section 4.5.

We now use evasive LWE to construct our obfuscation for σ-matrix PRFs $f_k(\mathbf{x}) = \mathbf{u}^T \mathbf{M}_{\mathbf{x}} \mathbf{v}$, where $k = \{\mathbf{M}_{i,b}\}_{i \in [h], b \in \{0,1\}}, \mathbf{u}, \mathbf{v}$. As above, we will assume that f_k is read-once. And, as mentioned above, our construction closely follows the witness encryption construction of Vaikuntanathan, Wichs and Wee [VWW22].

First, we sample matrices with small Gaussian entries $\mathbf{S}_{i,b} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{n \times n}$, and construct the following diagonal matrices

$$\widehat{\mathbf{S}}_{i,b} = \begin{pmatrix} \mathbf{M}_{i,b} & \\ & \mathbf{S}_{i,b.} \end{pmatrix}$$

Additionally, set the "bookend vectors" $\hat{\mathbf{u}} = (\mathbf{u} \mid \mathbf{1}^n)$ and $\hat{\mathbf{v}} = (\mathbf{v} \mid \mathbf{0}^n)$. Note that

$$\hat{\mathbf{u}}^T \widehat{\mathbf{S}}_{\mathbf{x}} \hat{\mathbf{v}} = \mathbf{u}^T \mathbf{M}_{\mathbf{x}} \mathbf{v} = f_k(\mathbf{x}) \ .$$

Using the encoding technique of Gentry, Gorbunov and Halevi [GGH15] (often referred to as the GGH15 encoding), we encode our matrices as follows:

$$\left\{\mathbf{D}_{1,b} := \underset{\sim}{\hat{\mathbf{u}}\widehat{\mathbf{S}}_{1,b}\mathbf{A}_1}\right\}_{b \in \{0,1\}}, \left\{\mathbf{D}_{i,b} := \mathbf{A}_{i-1}^{-1}(\underset{\sim}{\widehat{\mathbf{S}}_{i,b}\mathbf{A}_i})\right\}_{2 \leq i \leq h, b \in \{0,1\}}$$

where $\mathbf{A}_h = \hat{\mathbf{v}}$, $\mathbf{A}_i \leftarrow \mathbb{Z}_q^{(n+w) \times 4(n+w) \log q}$, and $\mathbf{A}_i^{-1}(\cdot)$ denotes a random preimage with small entries. We use curly underlines to indicate noised terms; for example, $\underset{\sim}{\mathbf{A}}$ indicates $\mathbf{A} + \mathbf{E}$ for some small

noise term $\mathbf{E}$. By the correctness of the GGH15 encodings, we have that up to small additive error,

$$\mathbf{D}_{\mathbf{x}} := \prod_{i\in[h]} \mathbf{D}_{i,x_i} \approx \hat{\mathbf{u}}\widehat{\mathbf{S}}_{\mathbf{x}}\hat{\mathbf{v}} = f_k(\mathbf{x}) \ ,$$

that is, the GGH encodings can be used to approximately compute the functionality of f_k.

To prove security, we use LWE and evasive LWE to show that the GGH15 encodings look pseudorandom. For notational simplicity, below, each instance of $\mathcal{U}$ will denote an independently sampled, uniformly sampled matrix over $\mathbb{Z}_q$ with appropriate dimensions. For the first step of the argument, we set:

$$\mathbf{S} = \{\hat{\mathbf{u}}\hat{\mathbf{S}}_{\mathbf{x}'}\}_{\mathbf{x}'\in\{0,1\}^{h-1}}, \mathbf{B} = \mathbf{A}_{h-1}, \mathbf{P} = \begin{pmatrix} \widehat{\mathbf{S}}_{h,0}\mathbf{A}_h \\ \widehat{\mathbf{S}}_{h,1}\mathbf{A}_h \end{pmatrix}, \mathsf{aux} = \mathsf{aux}(k)$$

Note that the product $\mathbf{SP}$ is just a noisy version of the pseudorandom truth table, that is:

$$\mathbf{SP} = \{\hat{\mathbf{u}}\widehat{\mathbf{S}}_{\mathbf{x}}\mathbf{A}_h\}_{\mathbf{x}\in\{0,1\}^h} = \{f_k(\mathbf{x})\}_{\mathbf{x}\in\{0,1\}^h} \ .$$

Thus, $\mathbf{SP}$ is pseudorandom in in the presence of $\mathsf{aux}(k)$. It follows $\underline{\mathbf{SP}}$ is too. Additionally, writing $\mathbf{A}_{h-1} = \begin{pmatrix} \overline{\mathbf{A}}_{h-1} \\ \underline{\mathbf{A}}_{h-1} \end{pmatrix}$, we have that

$$\begin{aligned} \underline{\mathbf{SB}} &= \{\hat{\mathbf{u}}\widehat{\mathbf{S}}_{\mathbf{x}'}\mathbf{A}_{h-1}\}_{\mathbf{x}'\in\{0,1\}^{h-1}} \\ &= \{\mathbf{u}\mathbf{M}_{\mathbf{x}'}\overline{\mathbf{A}}_{h-1}\}_{\mathbf{x}'\in\{0,1\}^{h-1}} + \underbrace{\{\mathbf{1}_n\mathbf{S}_{\mathbf{x}'}\underline{\mathbf{A}}_{h-1}\}_{\mathbf{x}'\in\{0,1\}^{h-1}}}_{\text{pseudorandom}} \approx_c \{\mathcal{U}\}_{\mathbf{x}'\in\{0,1\}^{h-1}} \end{aligned}$$

by invoking a BLMR-type argument [BLMR13] with public matrices $\mathbf{S}_{i,b}$ and secret matrix $\underline{\mathbf{A}}_{h-1}$. Because the $\mathbf{S}_{i,b}$ matrices are treated as public, this indistinguishability also holds in the presence of $\underline{\mathbf{SP}}$, and because the $\mathbf{S}_{i,b}$ and $\mathbf{A}_i$ matrices are sampled independently of k, it holds even in the presence of $\mathsf{aux}(k)$. Therefore,

$$(\underline{\mathbf{SB}}, \underline{\mathbf{SP}}, \mathsf{aux}(k)) \approx_c (\mathcal{U}, \mathcal{U}, \mathsf{aux}(k)).$$

Therefore, invoking evasive LWE, we get the following as the post-condition:

$$\begin{aligned} \left(\{\hat{\mathbf{u}}\widehat{\mathbf{S}}_{\mathbf{x}'}\mathbf{A}_{h-1}\}_{\mathbf{x}'\in\{0,1\}^{h-1}}, \{\mathbf{D}_{h,b}\}_b, \mathsf{aux}(k)\right) &= (\underline{\mathbf{SB}}, \mathbf{B}^{-1}(\mathbf{P}), \mathsf{aux}(k)) \\ &\approx_c (\mathcal{U}, \mathbf{B}^{-1}(\mathbf{P}), \mathsf{aux}(k)) \end{aligned}$$

That is, we have "peeled off" the last pair of matrices $\{\mathbf{D}_{h,b}\}_b$ while preserving pseudorandomness. Now we repeat. In the second step of the argument, we use evasive LWE to peel off $\{\mathbf{D}_{h-1,b}\}_b$, by

setting

$$\mathbf{S} = \{\hat{\mathbf{u}}\widehat{\mathbf{S}}_{\mathbf{x}'}\}_{\mathbf{x}'\in\{0,1\}^{h-2}}, \mathbf{B} = \mathbf{A}_{h-2}, \mathbf{P} = \begin{pmatrix} \underbrace{\widehat{\mathbf{S}}_{h-1,0}\mathbf{A}_{h-1}} \\ \underbrace{\widehat{\mathbf{S}}_{h-1,1}\mathbf{A}_{h-1}} \end{pmatrix}, \mathsf{aux} = (\{\mathbf{D}_{h,b}\}, \mathsf{aux}(k))$$

Continuing in this way, applying evasive LWE a total of $h-1$ times, we finally obtain that

$$(\{\mathbf{D}_{1,b}\}_b, \{\mathbf{D}_{i,b}\}_{i\geq 2,b}, \mathsf{aux}(k)) \approx_c (\mathcal{U}, \{\mathbf{D}_{i,b}\}_{i\geq 2,b}, \mathsf{aux}(k)).$$

Now the random matrix $\mathbf{A}_1$ only appears in $\{\mathbf{D}_{2,b}\}_b$. Indeed, we have

$$\begin{pmatrix} \mathbf{D}_{2,0} \\ \mathbf{D}_{2,1} \end{pmatrix} = \begin{pmatrix} \mathbf{A}_1^{-1}(\underbrace{\widehat{\mathbf{S}}_{2,0}\mathbf{A}_2}) \\ \mathbf{A}_1^{-1}(\underbrace{\widehat{\mathbf{S}}_{2,0}\mathbf{A}_2}) \end{pmatrix} = \mathbf{A}_1^{-1}\left(\underbrace{\begin{pmatrix} \widehat{\mathbf{S}}_{2,0}\mathbf{A}_2 \\ \widehat{\mathbf{S}}_{2,0}\mathbf{A}_2 \end{pmatrix}}\right).$$

And, following [CVW18], the random short preimages $\mathbf{A}_i^{-1}(\cdot)$ are sampled in such a way that for any matrix $\mathbf{Z}$, for uniformly random $\mathbf{A}_i$, $\mathbf{A}_i^{-1}(\mathbf{Z})$ is indistinguishable from a discrete Gaussian sample $\mathcal{D}_{\mathbb{Z},\sigma}$ (independent of $\mathbf{Z}$). Thus we can replace $\{\mathbf{D}_{2,b}\}_b$ with a fresh discrete Gaussian sample $\mathcal{D}_{\mathbb{Z},\sigma}$. But now $\mathbf{A}_2$ only appears in $\{\mathbf{D}_{3,b}\}_b$, so we can repeat the argument and replace $\{\mathbf{D}_{3,b}\}_b$ with $\mathcal{D}_{\mathbb{Z},\sigma}$. Continuing in this way, we obtain

$$(\{\mathbf{D}_{1,b}\}_b, \{\mathbf{D}_{i,b}\}_{i\geq 2,b}, \mathsf{aux}(k)) \approx_c (\mathcal{U}, \{\mathcal{D}_{\mathbb{Z},\sigma}\}_{i\geq 2}, \mathsf{aux}(k)).$$

completing the proof. For a formal description of the obfuscation scheme and the formal proof, we point the reader to Section 4.

On Heuristic Counterexamples to Evasive LWE. As shown in [VWW22], the evasive LWE assumption is likely false for general auxiliary input aux. This is due to a heuristic attack in which aux is an (ideal) obfuscation of a program which knows $\mathbf{P}$ along with a trapdoor τ for $\mathbf{P}$, and on input $(\mathbf{C}, \mathbf{D})$ uses τ to decide if $\mathbf{C}\cdot\mathbf{D}$ is close to $\mathbf{S}'\cdot\mathbf{P}$ for some $\mathbf{S}'$. (See [VWW22, Section 8.2] for details.) Such an aux will break the post-condition of evasive LWE while leaving the pre-condition valid. While it is reasonable to conjecture the security of evasive LWE for essentially any "reasonable" aux [VWW22], in our case, caution is warranted.

The auxiliary input we use to argue the security of GGH encodings of σ-matrix PRFs has two components. The first is the sequence of GGH-encoding matrices $(\{\mathbf{D}_{i,b}\}_{i\geq j}, \mathsf{aux}(k))$ arising from our inductive argument. The second is the auxiliary information $\mathsf{aux}(k)$ related to the PRF key k.

In our witness PRF construction, the second component $\mathsf{aux}(k)$ is a list of PRF values $\{\mathsf{PRF}_{\mathsf{fk}}(\mathbf{x})\}_{\mathbf{x}\notin L}$ on inputs not in the language L, which seems quite benign. But the first component seems, at least superficially, quite similar to the auxiliary input used in the heuristic attack. It is a GGH encoding that can be viewed as the obfuscation of a certain program Π, and it is generated using a trapdoor for a matrix related to $\mathbf{P}$. However, we believe the similarity is only superficial. While the trapdoor is used to generate the obfuscation of Π, the *functionality* of Π is independent of the trapdoor for $\mathbf{P}$ (even of $\mathbf{P}$). Indeed, evaluating the obfuscated program involves multiplying $\mathbf{SB}+\mathbf{E}$

by the first matrix that is part of the obfuscated program, simply yielding the matrix $\mathbf{SP} + \mathbf{E}'$ in the evasive LWE precondition. This is a key difference between our aux and the contrived auxiliary input described in [VWW22], indeed, one that makes the existence of a similar attack unlikely.

2.2 Achieving Adaptive Security for "Sahai-Waters-like" SNARGs

As discussed above, Gentry and Wichs [GW11] showed that it is not possible to construct an *adaptively secure* SNARG for all languages in NP through polynomial time reductions to *falsifiable assumptions*, even in the designated verifier setting. In fact, this impossibility result holds for all languages which have sub-exponentially hard-membership problems. Even UP languages such as decisional Diffie-Hellman (DDH) are believed to have such sub-exponentially hard-membership problems.

One interpretation of the Gentry-Wichs barrier is that one has to turn to *sub-exponential hardness assumptions* to be able to argue adaptive soundness based on (sub-exponentially) falsifiable assumptions. Indeed, our SNARG for UP relies on sub-exponential LWE, in addition to evasive LWE, which is seemingly not falsifiable. However, our reduction reduces the soundness of the SNARG to σ-PRF obfuscation, which is in fact falsifiable. Therefore, the main reason our SNARG can be shown to be adaptively sound is that our reduction runs in exponential time and the underlying assumption is sub-exponential.

It is then natural to ask more generally if existing SNARG constructions, in particular the Sahai-Waters SNARG, can be shown to be adaptively secure via similar exponential-time reductions to sub-exponential security assumptions.

Complexity-leveraging. A general strategy to promote a non-adaptively secure scheme to an adaptively secure one is complexity leveraging [BB04]. In the case of SNARGs, the high-level idea is to guess the instance $x^* \leftarrow \overline{L}$ that the adaptive cheating prover is going to cheat on, and proceed with the non-adaptive security reduction. Since the probability of guessing x^* correctly is $1/|\overline{L}|$, the resulting reduction incurs an exponential loss in the security parameter. This approach does immediately work, in a somewhat weak sense—by assuming sub-exponential hardness of the underlying assumptions, one can straightforwardly prove adaptive soundness. However, the difficult part is ensuring that the resulting SNARG proofs remain *succinct*. Indeed, complexity leveraging applied directly to the publicly verifiable Sahai-Waters SNARG [SW14] does not seem to yield succinct proofs.[4]

On the other hand, we show in Section 7 that the Sahai-Waters SNARG construction restricted to the *designated-verifier* setting can in fact be transformed into one that is reusable and adaptively sound while maintaining succinct proofs through a careful complexity leveraging argument.

We rely on the following properties of the Sahai-Waters SNARG construction in our proof:

1. For every statement x and every possible common reference string crs, there is a unique string, which we denote by $\pi_{\mathsf{crs},x} \in \{0,1\}^\ell$, that causes the verifier to accept.

[4]The concurrent work of Waters and Wu [WW24] modifies the Sahai-Waters SNARG in a more sophisticated white-box way to achieve adaptive security in this setting.

2. For all $x^* \notin L$, there exists an efficiently sampleable distribution $\mathcal{D}_{x^*}$ such that the following two distributions are indistinguishable:
   - H_0 : Sample $(\mathsf{crs}, \tau) \leftarrow \mathsf{SNARG.Gen}(1^\lambda)$. Publish crs and give black-box oracle to $\mathsf{Ver}_\tau(\cdot)$.
   - H_1 : Sample $\mathsf{crs}' \leftarrow \mathcal{D}_{x^*}$ and $r \leftarrow \{0,1\}^\ell$. Publish crs' and give black-box access to $\mathsf{Ver}'_{\tau,x^*,r}$ that works as follows:
     - On input (x, π) where $x \neq x^*$, output $\mathsf{Ver}_\tau(x, \pi)$.
     - On input (x^*, π), accept if $\pi = r$, and reject otherwise.
3. A cheating prover can only make $\mathsf{poly}(|x|, |w|, \lambda)$ queries to the verifier. This is the crucial difference between the publicly-verifiable setting and the designated-verifier setting.

We note that our SNARG from evasive LWE also has this structure. For simplicity of the following exposition, suppose $|x|, |w| \leq \mathsf{poly}(\lambda)$, and we only argue $\mathsf{negl}(\lambda)$ soundness error (rather than $2^{-\lambda}$).

Recap of non-adaptive soundness proof. Recall that to prove soundness for a non-adaptively chosen x, one would normally argue in H_1 that if the cheating prover $\mathcal{P}^*$ only gets $\mathsf{poly}(\lambda)$ **black-box** calls to the verifier, it only has a $\mathsf{poly}(\lambda)/2^\ell = \mathsf{negl}(\lambda)$ if $\ell \geq \lambda$ probability of guessing r correctly (since crs' does not contain any information about the independently generated r). On the other hand, a cheating non-adaptive prover $\mathcal{P}^*$ with non-negligible advantage would be able to guess $\pi_{x^*,\mathsf{crs}}$ in H_0 with non-negligible probability. Therefore, $\mathcal{P}^*$ can be used to distinguish the distributions in H_0 and H_1 with non-negligible probability in λ.

Our adaptive soundness proof. To argue adaptive soundness, we consider distributions identical to those before, except with the changes highlighted in blue.

- H'_0 : Sample a uniformly random $x^* \leftarrow \overline{L}$, and $(\mathsf{crs}, \tau) \leftarrow \mathsf{SNARG.Gen}(1^\lambda)$. Publish (crs, x^*), and give black-box oracle to $\mathsf{Ver}_\tau(\cdot)$.
- H'_1 : Sample a uniformly random $x^* \leftarrow \overline{L}$. Sample $\mathsf{crs}' \leftarrow \mathcal{D}_{x^*}$ and $r \leftarrow \{0,1\}^\ell$. Publish (crs', x^*), and give black-box access to $\mathsf{Ver}'_{\tau,x^*,r}$ that works as follows:
  - On input (x, π) where $x \neq x^*$, output $\mathsf{Ver}_\tau(x, \pi)$.
  - On input (x^*, π), accept if $\pi = r$.

It is clear that by the fact that $\mathsf{H}_0 \approx_c \mathsf{H}_1$, the above distributions are also *polynomially* indistinguishable by an averaging argument. However, we additionally need sub-exponential indistingushability of the two hybrids. Suppose that H'_0 and H'_1 are indistinguishable to 2^{ρ^α} time adversaries with $2^{-\rho^\alpha}$ for a parameter $\rho = \lambda + |x| + |w|$ (we can achieve this using sub-exponential hardness assumptions).

Suppose there exists a cheating prover $\mathcal{P}^*$ that breaks adaptive soundness with non-negligible advantage $\epsilon(\lambda)$ by making only $\mathsf{poly}(\lambda)$ queries to the Ver oracle. Then, we show that there exists an $\mathcal{A}$ that can use $\mathcal{P}^*$ to distinguish H'_0 and H'_1 with advantage $p(\lambda)/|\overline{L}|$, for some non-negligible function p. $\mathtt{Adv}$ uses $\mathcal{P}^*$ as follows:

- Upon receiving the tuple (crs, x^*), $\mathcal{A}$ does the following:

– Send crs to $\mathcal{P}^*$, and keep x^* private.
– For all (possibly adaptive) queries (x_i, π_i) from $\mathcal{P}^*$, query the verification oracle and send back the response.
– If any $x_i = x^*$, and (x_i, π_i) was accepted by the oracle, output 0. Otherwise, output 1.

First, note that in H'_0, the prover $\mathcal{P}^*$ does not see x^*. Since x^* is chosen uniformly randomly from $\overline{L}$, the probability that $\mathcal{A}$ outputs 0 is at least $\frac{\epsilon(\lambda)}{|\overline{L}|}$.

Now, we upper bound the probability that $\mathcal{A}$ outputs 0 in H'_1. A naive argument in H'_1 would say that $\mathcal{P}^*$ guesses r with probability at most $\mathsf{poly}(\lambda)/2^\ell$. A more refined argument would be as follows: Since x^* is information-theoretically hidden from $\mathcal{P}^*$ in H'_0, it is *computationally* hidden in H'_1 since $\mathsf{H}'_0 \approx_c \mathsf{H}'_1$ (when $\mathcal{P}^*$ doesn't see x^*). Therefore, $\mathcal{P}^*$ needs to essentially guess *both* x^* and r, which he succeeds in with probability approximately $\frac{\mathsf{poly}(\lambda)}{|\overline{L}|\cdot 2^\ell} = \frac{\mathsf{negl}(\lambda)}{|\overline{L}|}$ since $\mathcal{P}^*$ only gets $\mathsf{poly}(\lambda)$ queries to the verifier. For a more rigorous argument, see Claim 7.5.

Therefore,

$$\Pr[\mathcal{A}^{\mathsf{H}'_0}(1^\rho) = 0] - \Pr[\mathcal{A}^{\mathsf{H}'_1}(1^\rho) = 0] \geq \frac{1}{|\overline{L}|}\epsilon(\lambda) - \frac{1}{|\overline{L}|}\mathsf{negl}(\lambda) \geq \frac{p(\lambda)}{|\overline{L}|} \geq \frac{\mathsf{poly}(\lambda)}{2^{|x|}} \geq 2^{-\rho^\alpha},$$

thereby contradicting the sub-exponential indistinguishability $\mathsf{H}'_0 \approx_c \mathsf{H}'_1$.

2.3 From Adaptive SNARGs for UP to SNARKs for UP

In this section, we outline our transformation from a SNARG to SNARK for UP from Section 8. We take inspiration from the work of Campanelli et al. [CGKS23] (we describe the main differences between our transformation and theirs in Remark 8.7, and we describe some of the issues in their construction and how we fix them in Remark 8.4). Our knowledge extractor relies on the fact that any $x \in L$ has a unique witness w. Our construction will essentially extract a single bit of w at some index i which is hidden from the prover.

Fix a UP language L with relation $R : \mathcal{X} \times \mathcal{W} = \{0,1\}$, where $\mathcal{X}$ is the set of instances, and $\mathcal{W}$ is the corresponding set of potential witnesses. Additionally, fix a fully homomorphic encryption scheme FHE and an public-key encryption scheme PKE. We transform R into a new UP relation $R^{(\mathsf{ek},\mathsf{pk},\mathsf{ct})}$ as follows.

- Sample a secret key $\mathsf{sk}_{\mathsf{FHE}}$ and a evaluation ek from $\mathsf{FHE.Gen}(1^\lambda)$.
- Sample a secret key $\mathsf{sk}_{\mathsf{PKE}}$ and public key pk for some public key encryption scheme PKE.
- Pick an index $i \leftarrow [|w|]$, and compute $\mathsf{ct} = \mathsf{FHE.Enc}_{\mathsf{sk}}(i)$.
- Let C_w be the circuit that takes as input j and outputs the bit $w[j]$.

Now, we define a new relation $R^{(\mathsf{ek},\mathsf{pk},\mathsf{ct})} : \mathcal{X}' \times \mathcal{W}' \to \{0,1\}$

$$R^{(\mathsf{ek},\mathsf{pk},\mathsf{ct})}((x,\rho),(w,r)) = \begin{cases} 1 & \text{if } R(x,w) = 1 \text{ and } \mathsf{PKE.Enc}_{pk}(\mathsf{FHE.Eval}_{\mathsf{ek}}(C_w, \mathsf{ct}); r) = \rho \\ 0 & \text{otherwise.} \end{cases}$$

Here, $\mathsf{FHE.Eval}_{\mathsf{ek}}$ takes as input the circuit C_w and the ciphertext ct and homomorphically evaluates C_w on the ciphertext. Additionally, suppose that PKE has an *injective encryption function*, i.e. $\mathsf{PKE.Enc}_{\mathsf{sk}}$ is injective in the message space and randomness (see Definition 3.24). Assuming that $\mathsf{FHE.Eval}_{\mathsf{ek}}$ is a deterministic function and PKE is injective, one can argue that $R^{(\mathsf{ek},\mathsf{pk},\mathsf{ct})}$ is a UP relation. Moreover, given the secret key $\mathsf{sk}_{\mathsf{FHE}}$ and $\mathsf{sk}_{\mathsf{PKE}}$, one can extract $w[i]$ from the ciphertext ρ.

SNARK Construction. We are now ready to outline our SNARG to SNARK construction (for a more detailed construction, see Section 8). Let Π be an adaptively sound SNARG system for UP languages.

- $\mathsf{SNARK.Gen}(1^\lambda)$:
  - Sample $(\mathsf{sk}_{\mathsf{FHE}}, \mathsf{ek})$ for FHE, and $(\mathsf{sk}_{\mathsf{PKE}}, \mathsf{pk})$ for PKE, and a ciphertext $\mathsf{ct} = \mathsf{FHE.Enc}_{\mathsf{sk}}(i)$, as described earlier.
  - Compute $(\mathsf{crs}, \tau) \leftarrow \Pi.\mathsf{Gen}(1^\lambda)$, for relation $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}$.
  - Set the new common reference string $\mathsf{crs}' = (\mathsf{crs}, \mathsf{ek}, \mathsf{pk}, \mathbf{c})$, and verifier state τ.
  - Output (crs', τ).
- $\mathsf{SNARK.Prove}(\mathsf{crs}', x, w)$:
  - Sample randomness r, and compute $\rho = \mathsf{PKE.Enc}_{pk}(\mathsf{FHE.Eval}_{\mathsf{ek}}(C_w, \mathsf{ct}); r)$, i.e. homomorphically compute the circuit C_w on the FHE ciphertext ct (this will result in an encryption of $w[i]$ under $\mathsf{sk}_{\mathsf{FHE}}$), and encrypt under the public key pk.
  - Compute $\pi \leftarrow \Pi.\mathsf{Prove}(\mathsf{crs}, (x, \rho), (w, r))$.
  - Output ρ, π as the proof for x.
- $\mathsf{SNARG.Verify}(\tau, x, (\rho, \pi))$: Accept iff $\Pi.\mathsf{Verify}(\tau, (x, \rho), \pi)$ accepts, i.e. treat (x, ρ) as the statement under relation $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}$, and π as the SNARG proof.

We sketch the proofs of knowledge extractability and zero-knowledge.

Knowledge soundness. First, we sketch why the above construction is *knowledge-sound*, i.e. one can extract a witness for x by repeatedly querying an *non-adaptive* $\mathcal{P}^*$ on various common reference strings (see Definition 3.19 for a more detailed definition). Suppose $\mathcal{P}^*$ succeeds in creating proofs for x with probability ϵ. Since the language is UP, there is a unique witness w for x. We crucially rely on the uniqueness of w.

By the adaptive soundness of Π, with probability approximately ϵ, $(x, \rho), \pi$ created under $\mathsf{crs}' = (\mathsf{crs}, \mathsf{ek}, \mathsf{pk}, \mathbf{c})$ is accepted only if (x, ρ) has a witness under $R^{(\mathsf{ek},\mathsf{pk},\mathsf{ct})}$. Moreover, given as trapdoor $\mathsf{td} = (\mathsf{sk}_{\mathsf{FHE}}, \mathsf{sk}_{\mathsf{PKE}})$, one can decrypt ρ to obtain $C_w[i] = w[i]$. By repeating this experiment many times, one should eventually collect all bits of the unique witness w. Note that since the index i is encrypted, one can argue that the probability of extracting each $w[i]$ is approximately $\epsilon/|w|$ (up to $\mathsf{negl}(\lambda)$ terms) by IND-CPA security of the FHE scheme. Therefore, by repeating the experiment many times, one can extract all the bits of w.

Here, we crucially rely on the adaptive soundness of Π because even an honest prover can only create an instance for Π after crs' is generated and the parameters $(\mathsf{pk}, \mathsf{ek}, \mathbf{c})$ are publicly available.

Zero-knowledge. To argue that the transformation preserves zero-knowledge, we rely on the public-key encryption scheme. Recall that if $\mathsf{FHE.Eval}_{\mathsf{ek}}$ is deterministic, it is most likely not circuit-private, and might leak information about underlying circuit. Therefore, the additional layer of public-key encryption allows one to invoke IND-CPA security to simulate the ciphertext ρ without the witness. Then, π can be simulated using just (x, ρ) using the zero-knowledge simulator for the underlying zero-knowledge SNARG scheme Π.

3 Preliminaries

3.1 Notation

We write $[n]$ for the set $\{0, 1, \ldots, n\}$. In what follows, q will always denote a prime integer. We denote bitstrings with lowercase bold letters, and given two bitstrings $\mathbf{x}, \mathbf{y} \in \{0,1\}^*$, we write $\mathbf{x} \mid \mathbf{y}$ for their concatenation. For $c \in \mathbb{N}$, we write $|^c(\mathbf{x})$ for the c-fold concatenation of $\mathbf{x}$ with itself, that is, $\mathbf{x} \mid \mathbf{x} \mid \cdots \mid \mathbf{x}$, where $\mathbf{x}$ appears a total of c times.

We denote matrices with uppercase bold letters $\mathbf{M}$, and (column) vectors with lowercase bold letters $\mathbf{v}$. We write $\mathbf{0}$ (resp. $\mathbf{1}$) for the all-zero (resp. all-one) vector when the dimension is clear from context. Given a collection of matrices $\{\mathbf{M}_{i,b}\}_{i\in[h],b\in\{0,1\}}$ and a string $\mathbf{x} \in \{0,1\}^h$, we write

$$\mathbf{M}_{\mathbf{x}} := \prod_{i=1}^{h} \mathbf{M}_{i,x_i} \ .$$

Given matrices $\mathbf{M}_1, \ldots, \mathbf{M}_n$, we write $\mathrm{diag}(\mathbf{M}_1, \ldots, \mathbf{M}_n)$ for the block-diagonal matrix whose blocks are the $\mathbf{M}_i$.

For integers $i, j \in \mathbb{Z}$, $i \bmod j \in \mathbb{Z}$ denotes the representative of the congruence class of i mod j in the interval $[0, j-1]$. However, in expressions that interpret elements of $\mathbb{Z}_q$ as ordinary integers, such as $|x| \in \mathbb{Z}$ for $x \in \mathbb{Z}_q$, we interpret x as an integer in the interval $[-q/2, q/2]$. The rounding operation $\lfloor a \rceil_p : \mathbb{Z}_q \to \mathbb{Z}_p$ is defined by $\lfloor (p/q) \cdot a' \rceil$, where a' is any integer congruent to a mod p, and $\lfloor \cdot \rceil$ denotes rounding to the nearest integer.

We denote distributions with calligraphic letters. We write $x \leftarrow \mathcal{D}$ for drawing a (fresh, independent) sample x from a distribution $\mathcal{D}$. Given a finite set W, $\mathcal{U}(W)$ denotes the uniform distribution over W. We write $x \leftarrow W$ for $x \leftarrow \mathcal{U}(W)$. $\mathcal{D}_{\mathbb{Z},\sigma}$ denotes the discrete Gaussian over $\mathbb{Z}$ with parameter σ; that is, the distribution which assigns mass proportional to $\exp(-\pi x^2/\sigma^2)$ to each $x \in \mathbb{Z}$. And, given a distribution $\mathcal{D}$ and integers $m, n \geq 1$, we write $\mathcal{D}^{n\times m}$ for the distribution over $n \times m$ matrices where each entry is sampled independently from $\mathcal{D}$.

Let $\lambda \in \mathbb{N}$ be a security parameter. Given distributions $\mathcal{D}_1, \mathcal{D}_2$, we write $\mathcal{D}_1 \approx_c \mathcal{D}_2$ to denote computational indistinguishability and $\mathcal{D}_1 \approx_s \mathcal{D}_2$ to denote statistical indistinguishability. By default, these mean that all non-uniform probabilistic $\mathsf{poly}(\lambda)$-time distinguishers and unbounded distinguishers respectively have advantage at most $\mathsf{negl}(\lambda)$. When we wish to specify a different class of distinguishers or a different advantage bound, we will do so explicitly. For explicit statements of statistical indistinguishability, we will sometimes use the statistical (total variation) distance $\mathrm{d}_{\mathrm{TV}}(\cdot, \cdot)$.

3.2 LWE Assumption

Given $n, m, q \in \mathbb{N}$ and $\sigma, \delta > 0$, the subexponential LWE assumption $\mathsf{LWE}^{\delta}_{n,m,q,\sigma}$ asserts that

$$(\mathbf{A}, \mathbf{sA} + \mathbf{e}) \approx_c (\mathbf{A}, \mathbf{b}),$$

with security parameter $\mu = 2^{n^\delta}$, where $\mathbf{s} \leftarrow \mathcal{U}(\mathbb{Z}_q^n)$, $\mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{n \times m})$, $\mathbf{e} \leftarrow \mathcal{D}^m_{\mathbb{Z},\sigma}$, and $\mathbf{b} \leftarrow \mathbb{Z}_q^m$.

Following [VWW22], we rely on the above assumption holding for some $\delta > 0$, for parameters such that $q/\sigma \leq 2^{n^\delta}$.

The following useful lemma shows that LWE with small-norm public matrices (with entries sampled from the error distribution) is as hard as LWE.

Lemma 3.1 ([CVW18, Lemma 3.9]). *For $q \leq 2^{\mathsf{poly}(n)}$, $m = \mathsf{poly}(n)$, $n' \leq n/(2 \log q)$, there is a $\mathsf{poly}(n)$-time reduction from $\mathsf{LWE}_{n',\mathsf{poly}(n),q,\sigma}$ to the problem of distinguishing $(\mathbf{A}, \mathbf{sA} + \mathbf{e})$ from uniform, where $m = \mathsf{poly}(n)$, $\mathbf{A} \leftarrow \mathcal{D}^{n \times m}_{\mathbb{Z},\sigma}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, and $\mathbf{e} \leftarrow \mathcal{D}^m_{\mathbb{Z},\sigma}$.*

3.3 Security Parameters

In this paper, we will need to reason about two different parameter regimes; one concerned with *programs* P mapping bitstrings $\mathbf{x} \in \{0,1\}^h$ to values $P(\mathbf{x}) \in \mathbb{Z}_q$, and the other with the *collections of outputs* $\{P(\mathbf{x})\}_{\mathbf{x}}$ of such programs. To keep track of the different efficiency and security requirements in each regime, it will be useful to introduce two related security parameters, λ and $\mu = \mu(\lambda)$. As is standard, we will require that all cryptographic algorithms run in time $\mathsf{poly}(\lambda)$, and require that their security guarantees cannot be broken by $\mathsf{poly}(\lambda)$-time adversaries, except with probability $\mathsf{negl}(\lambda)$. The second security parameter μ roughly corresponds to collections of outputs of these algorithms, and it is much larger. In particular, we will always assume that parameters $n = n(\lambda), h = h(\lambda) \leq \mathsf{poly}(\lambda)$ and $q = q(\lambda)$ satisfy $h \leq n^{\delta/20}$ and $q = 2^{n^\delta}$, and we will set $\mu = \mu(\lambda) := 2^{n^\delta}$. Notice that $\{P(\mathbf{x})\}_{\mathbf{x}}$ has size $2^h \log q = \mathsf{negl}(\mu)$, so $\mathsf{poly}(n)(\mu)$-time distinguishers can perform arbitrary polynomial-time computations on such collections. For a function $f = f(\lambda)$, we will say $\mathcal{D}_1 \approx_c \mathcal{D}_2$ *with security parameter f* to mean that all non-uniform probabilistic $\mathsf{poly}(f)$-time distinguishers have advantage at most $\mathsf{negl}(f)$. We will often use $f = \mu$. We will also often work with pairs of distributions for which all $\mathsf{poly}(\mu)$-time distinguishers have advantage $\mathsf{negl}(2^{h^2\lambda})$; this is a relaxation of indistinguishability with security parameter μ.

3.4 Trapdoor and Pre-image Sampling

Given a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ for $m \geq 2n \log q$, a vector $\mathbf{y} \in \mathbb{Z}_q^n$, and $\sigma > 0$, we use $\mathbf{A}^{-1}(\mathbf{y}, \sigma)$ to denote the distribution of a vector $\mathbf{d}$ sampled from $\mathcal{D}^m_{\mathbb{Z},\sigma}$ conditioned on $\mathbf{Ad} = \mathbf{y} \pmod q$. (Vectors satisfying the condition exist except with probability $\mathsf{negl}(\mu)$.) We extend this notation to matrices $\mathbf{Y} \in \mathbb{Z}_q^{n \times k}$ in the natural way (i.e., columnwise). We sometimes suppress σ when it is clear from context. The following convenient lemma statements are adapted from [CVW18].

Lemma 3.2 ([CVW18, Lemma 3.10], originally [Ajt96, AP11, MP12]). *Assume $\mathsf{LWE}^{\delta}_{n,m,q,\sigma}$. There is a PPT algorithm $\mathsf{TrapSam}(1^n, 1^m, q)$ that, on input the modulus $q \geq 2$ and dimensions n, m such*

that $m \geq 2n \log q$, outputs a matrix $\mathbf{A}$ such that $\mathbf{A} \approx_s \mathcal{U}(\mathbb{Z}_q^{n \times m})$ with security parameter μ, along with a trapdoor τ (referenced in the next lemmas).

Lemma 3.3 ([CVW18, Lemma 3.11])**.** *There is a PPT algorithm that, on input $\mathbf{y} \leftarrow \mathbb{Z}_q^n$ and $(\mathbf{A}, \tau) \leftarrow \mathsf{TrapSam}(1^n, 1^m, q, \sigma)$ for some $\sigma \geq 2\sqrt{n \log q}$, outputs a sample from $\mathbf{A}^{-1}(\mathbf{y}, \sigma)$. We will abuse notation and denote this algorithm by $\mathbf{A}^{-1}(\mathbf{y}, \sigma)$.*

Lemma 3.4 ([CVW18, Lemma 4.4])**.** *Assume $\mathsf{LWE}^{\delta}_{2n,k,q,\sigma}$. Let $n, m, k, q \in \mathbb{N}$, $\sigma, \sigma' \in \mathbb{R}$ satisfy $n, m, k \in \mathsf{poly}(\lambda)$, $m \geq 4n \log q$, and $\sigma' \geq \sigma \geq 2\sqrt{n \log q}$. Then, for any fixed matrix $\mathbf{Z} \in \mathbb{Z}_q^{n \times k}$, we have that with security parameter μ,*

$$\mathbf{A}^{-1}(\mathbf{Z} + \mathbf{E}, \sigma) \approx_c \mathcal{D}_{\mathbb{Z},\sigma}^{m \times k} ,$$

where $\mathbf{A}, \tau \leftarrow \mathsf{TrapSam}(1^n, 1^m, q)$ and $\mathbf{E} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{n \times k}$.

Note that [CVW18, Lemma 4.4] had $\sigma' = \sigma$, but we state a generalization to $\sigma' \geq \sigma$. Indeed, the generalization can be easily proved by absorbing some of the variance of $\mathbf{E}$ into $\mathbf{Z}$ (this was used already implicitly in [VWW22]).

3.5 Primitives

In this section, we define and instantiate the various basic cryptographic primitives we will need.

3.5.1 Pseudorandom Functions

Puncturable PRFs.

Definition 3.5 (Puncturable PRF)**.** A puncturable PRF family on key space $\mathcal{K} = \{\mathcal{K}_\lambda\}_\lambda$, domain $\mathcal{X} = \{\mathcal{X}_\lambda\}_\lambda$, output space $\mathcal{Y} = \{\mathcal{Y}_\lambda\}_\lambda$ and polynomial $s(\lambda)$ is a tuple of PPT algorithms $(\mathsf{PPRF.Gen}, \mathsf{PPRF.Punc}, \mathsf{PPRF.Eval})$ with the following interface and properties.

- $\mathsf{PPRF.Gen}(1^\lambda)$: On input the security parameter 1^λ, outputs a key $k \in \mathcal{K}_\lambda$.
- $\mathsf{PPRF.Punc}(k, S)$: On input a key $k \in \mathcal{K}_\lambda$ and a set $S \subseteq \mathcal{X}_\lambda$ such that $|S| \leq s(\lambda)$, outputs a new key $k(\{S\}) \in \mathcal{K}_\lambda$.
- $\mathsf{PPRF.Eval}(k, x)$: On input $k \in \mathcal{K}_\lambda$ and $x \in \mathcal{K}_\lambda$, outputs some value $y \in \mathcal{Y}_\lambda$.
- **Functionality preserving:** For all sets $S \subseteq \mathcal{X}$ of size $|S| \leq s(\lambda)$, for all $x \notin S$,

$$\Pr[k \leftarrow \mathsf{Gen}(1^\lambda), k\{S\} \leftarrow \mathsf{Punc}(k, S) : \mathsf{Eval}(k\{S\}, x) = \mathsf{Eval}(k, x)] = 1 .$$

- **Polynomial pseudorandomness at punctured points:** Consider the following experiment $\mathtt{EXP}^R_{\mathcal{A}}(b, \lambda)$ between a challenger $\mathcal{C}$ and adversary $\mathcal{A}$:
  - First, $\mathcal{A}$ chooses a set $S \subseteq \mathcal{X}_\lambda$ such that $|S| \leq s(\lambda)$, and sends it to $\mathcal{C}$.

- $\mathcal{C}$ samples $k \leftarrow \mathsf{PPRF.Gen}(1^\lambda)$, and computes $k\{S\} \leftarrow \mathsf{PPRF.Punc}(k, S)$.
  - If $b = 0$, compute $L_0 = \{x_i, \mathsf{PPRF.Eval}(k, x_i)\}_{x_i \in S}$.
  - If $b = 1$, compute $L_1 = \{x_i, y_i\}_{x_i \in S}$ where each $y_i \leftarrow \mathcal{Y}$.

  Give $k\{S\}, L_b$ to $\mathcal{A}$.
- $\mathcal{A}$ produces a bit b'.

Let W_b be the event that the adversary outputs 1 in experiment b, and define the adversary's advantage to be $\mathtt{Adv}^R_{\mathcal{A}}(\lambda) = |\Pr[W_0] - \Pr[W_1]|$. We say that the PPRF is pseudorandom at punctured points if for all PPT adversaries $\mathcal{A}$, $\mathtt{Adv}^R_{\mathcal{A}}(\lambda) = \mathsf{negl}[\lambda]$.

We additionally say that the PPRF is sub-exponentially pseudorandom at punctured points if, for some $\alpha > 0$, for all $\mathsf{poly}\big(2^{\lambda^\alpha}\big)$-time adversaries $\mathcal{A}$, $\mathtt{Adv}^R_{\mathcal{A}}(\lambda) = \mathsf{negl}(2^{\lambda^\alpha})$.

LWE-Based PRFs. The following lemma is a simple consequence of Banaczyzck's tail bound (see, e.g., [SD17, Corollary 1.3.11]).

Lemma 3.6. *For any $\sigma > 0$ and $n \in \mathbb{N}$,*

$$\Pr_{x \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}}[|x| \geq \sigma\sqrt{n}] \leq 2^{-n} .$$

The following simple noise flooding lemma is integral to our uses of subexponential LWE.

Lemma 3.7 ([CVW18, Lemma 3.2])**.** *For all $y \in \mathbb{Z}$ and $\sigma \in \mathbb{R}$, the statistical distance between $\mathcal{D}_{\mathbb{Z},\sigma}$ and $\mathcal{D}_{\mathbb{Z},\sigma} + y$ is at most y/σ.*

The following lemma adapts the BLMR PRF [BLMR13, Theorem 5.1] to the setting of subexponential LWE.

Lemma 3.8. *Let $n, n', q, h \in \mathbb{N}$ and $f, k, \sigma, \sigma' \in \mathbb{R}$ be functions of λ satisfying*

- $\lambda = 2^{(n')^\delta}$
- $\lceil (2 \log q) \cdot n' \rceil \leq n \leq \mathsf{poly}(n')$,
- $\sigma' \geq k \cdot (n^2\sigma)^{h+1}$,
- $2^{-n^\delta} \leq 1/k(n) \leq \mathsf{negl}(2^h \cdot n \cdot f(n))$.

Let

$$\mathbf{a} \leftarrow \mathbb{Z}_q^n, \{\mathbf{e}_\mathbf{x} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}\}_{x \in \{0,1\}^h}.$$

Then, assuming $\mathsf{LWE}^\delta_{n',\mathsf{poly}(n'),q,\sigma}$,

$$\{\mathbf{S}_{i,b} \leftarrow \mathcal{D}^{n\times n}_{\mathbb{Z},\sigma}\}_{\substack{i\in[h]\\ b\in\{0,1\}}}, \left\{\left(\prod_{i=1}^{h} \mathbf{S}_{i,\mathbf{x}_i}\right) \cdot \mathbf{a} + \mathbf{e}_\mathbf{x}\right\}_{\mathbf{x}\in\{0,1\}^h} \approx_c \{\mathbf{S}_{i,b} \leftarrow \mathcal{D}^{n\times n}_{\mathbb{Z},\sigma}\}_{\substack{i\in[h]\\ b\in\{0,1\}}}, \{\mathcal{U}\}_{\mathbf{x}\in\{0,1\}^h} .$$

where all $\mathsf{poly}(\mu)$*-time distinguishers have advantage at most* $\mathsf{negl}(f(n))$*. In particular, one can take* $q = 2^{n^\delta}$ *(to rely on subexponential LWE with* $q/\sigma \le 2^{n^\delta}$*),* $h = n^c$ *for some* $c < \delta/20$*,* $k = 2^{h^3\lambda}$*, and* $f(n) = 2^{-h^2\lambda}$*.*

We defer the details of the proof to Appendix A.1.

Remark 3.9. Notice that the statement is still true if $\mathbf{a}$ is replaced by $\mathbf{a}y_{\mathbf{x}}$, where $\{y_{\mathbf{x}} \in \mathbb{Z}_q\}_{\mathbf{x}}$ is arbitrary. Indeed, exactly the same proof works; the only step that changes is the LWE hybrid that relies on Lemma 3.1, and this lemma still applies since $\mathbf{a}y_{\mathbf{x}}$ is still uniform in $\mathbb{Z}_q$.

To prove a rounded version of Lemma 3.8, we will need a few simple lemmas, which will also be useful elsewhere in the paper. The first is a simple lemma about rounding (see, e.g., [BPR12]).

Lemma 3.10. *For integers* $q \ge p \ge 2$*,* $\mathrm{d_{TV}}(\lfloor \mathcal{U}(\mathbb{Z}_q) \rceil_p, \mathcal{U}(\mathbb{Z}_p)) \le p/q$*.*

The second is a simple condition for computational indistinguishability implying statistical indistinguishability.

Lemma 3.11. *Suppose* $\{a_\lambda\}, \{b_\lambda\}$ *are random variables,* $g(\lambda) \ge 1/\mathsf{poly}(\lambda)$*, and* $a \approx_c b$*, where all* $\mathsf{poly}(\lambda)$*-time distinguishers have advantage* $\mathsf{negl}(g(\lambda))$*. Suppose further that* f *is a* $\mathsf{poly}(\lambda)$*-time computable function taking values in a finite set of cardinality* $\mathsf{poly}(\lambda)$*. Then* $f(a) \approx_s f(b)$*, where all* $\mathsf{poly}(\lambda)$*-time distinguishers have advantage* $\mathsf{negl}(g(\lambda))$*.*

Proof. The optimal statistical distinguisher just checks membership in a (polynomial-sized) subset of the image of f, so it runs in polynomial time. Hence there is a trivial advantage-preserving reduction: given a challenge c to the computational indistinguishability, compute $f(c)$ and run the optimal statistical distinguisher. □

The third is a key step in justifying learning-with-rounding approaches that allows one to introduce or remove error terms inside a rounding operation at the cost of negligible statistical distance, for appropriate parameters.

Lemma 3.12. *Suppose* $\{a_\lambda\}, \{b_\lambda\}$ *are random variables,* $n \le \mathsf{poly}(\lambda)$ *and* $q \le \mathsf{poly}(\lambda)$*,* $g(\lambda) \ge 1/\mathsf{poly}(\lambda)$*, and*

$$\{a_i\}_{i\in[n]} \approx_c \{\mathcal{U}(\mathbb{Z}_q)\}_{i\in[n]} ,$$

where all $\mathsf{poly}(\lambda)$*-time distinguishers have advantage* $\mathsf{negl}(g(\lambda))$*. Suppose further that* $B \in \mathbb{R}$*,* $p \in \mathbb{Z}$ *satisfy* $B < p < q$*. Then for any* $\{e_i\}_i$ *such that except with probability* $\mathsf{negl}(\lambda)$*,* $|e_i| \le B$ *for all* i*, we have*

$$\mathrm{d_{TV}}(\{\lfloor a_i + e_i \rceil_p\}_i, \{\lfloor a_i \rceil_p\}_i) \le n \cdot (2B/p) + g(\lambda) + \mathsf{negl}(\lambda) .$$

Proof. As a first step, we may assume that $|e_i| \le B$ for all i, at the price of an additive $\mathsf{negl}(\lambda)$ in the statistical distance. Next, notice that the condition E that for all i, $\lfloor a_i + e_i \rceil_p = \lfloor a_i \rceil_p$ can be checked in polynomial time. Moreover, for truly uniform a_i, $\Pr[E] \ge 1 - n \cdot (2B/p)$. Hence, applying Lemma 3.11, $\Pr[E] \ge 1 - n \cdot (2B/p) - g(\lambda) - \mathsf{negl}(\lambda)$, completing the proof. □

Lemma 3.13. *Assuming the hypotheses of Lemma 3.12, no* $\mathsf{poly}(\lambda)$*-time adversary can distinguish* $\{\lfloor a_i \rceil_p\}_i$ *from* $\{\mathcal{U}(\mathbb{Z}_p)\}_i$ *with advantage greater than* $n \cdot ((2B/p) + (p/q)) + g(\lambda) + \mathsf{negl}(\lambda)$*.*

Proof. This follows immediately from the conclusion of Lemma 3.12 along with the hypothesis that $\{a_i\}_{i\in[n]} \approx_c \{\mathcal{U}(\mathbb{Z}_q)\}_{i\in[n]}$ with advantage bound $g(\lambda)$ and the fact that $\mathrm{d_{TV}}(\lfloor\mathcal{U}(\mathbb{Z}_q)\rceil_p, \mathcal{U}(\mathbb{Z}_p)) \leq p/q$ (Lemma 3.10). □

Finally, we present a "rounded" version of Lemma 3.8.

Lemma 3.14. *With parameters as in Lemma 3.8, let $p \in \mathbb{N}$ be such that $k \cdot 2^h \cdot (2\sigma'\sqrt{n}+1) \leq p \leq q/k$. (Setting $p = 2^{h^7\lambda}$ along with the other "in particular" parameters of Lemma 3.8 satisfies this.) Then, assuming $\mathsf{LWE}^{\delta}_{n',\mathsf{poly}(n'),q,\sigma}$,*

$$\{\mathbf{S}_{i,b} \leftarrow \mathcal{D}^{n\times n}_{\mathbb{Z},\sigma}\}_{\substack{i\in[h]\\ b\in\{0,1\}}}, \left\{ \left\lfloor \left(\prod_{i=1}^{h} \mathbf{S}_{i,\mathbf{x}_i}\right) \cdot \mathbf{a} \right\rceil_p \right\}_{\mathbf{x}\in\{0,1\}^h} \approx_c \{\mathbf{S}_{i,b} \leftarrow \mathcal{D}^{n\times n}_{\mathbb{Z},\sigma}\}_{\substack{i\in[h]\\ b\in\{0,1\}}}, \{\mathcal{U}\}_{\mathbf{x}\in\{0,1\}^h}$$

where all $\mathsf{poly}(\mu)$-time distinguishers have advantage at most $\mathsf{negl}(f(n))$.

We defer the proof to Appendix A.2.

3.5.2 SNARGs, SNARKs and NIZKs

A *designated verifier non-interactive argument* system (Definition 3.16) Π consists of a triple of efficient algorithms $\Pi = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Verify})$:

- $\Pi.\mathsf{Gen}(1^\lambda, R)$: Probabilistic algorithm that outputs a common reference string crs and a private state τ for the verifier.
- $\Pi.\mathsf{Prove}(\mathsf{crs}, x, w)$: Given the common reference string crs, a statement x and a witness w, outputs a proof π.
- $\Pi.\mathsf{Verify}(\tau, x, \pi)$: Takes as input the private state τ, a statement x, and proof π and outputs either 0 or 1. An argument system is *publicly verifiable* if $\tau = \mathsf{crs}$ or $\tau \subseteq \mathsf{crs}$.

We say that such an argument is also *succinct* (Definition 3.17) if the proof size π is small. We say that such a system is also *non-interactive zero-knowledge* if it satisfies the zero-knowledge as described in Definition 3.18.

Remark 3.15. In the definitions below, we use the notation $\Pr[E : \mathcal{D}]$ to denote the probability of event E over the distribution of $\mathcal{D}$ (sometimes notated as $\Pr_{\mathcal{D}}[E]$). We denote conditional probability of event E conditioned on event C as $\Pr[E \mid C]$.

Definition 3.16 (Non-Interactive Argument)**.** We say that $\Pi = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Verify})$ is a designated verifier non-interactive argument for a language $L \in \mathsf{NP}$ if L has an NP relation R such that Π satisfies the following three properties:

- **Completeness:** For all x, w such that $R(x, w) = 1$,

$$\Pr\left[\mathsf{Verify}(\tau, x, \pi) = 0 : \begin{array}{c}(\mathsf{crs}, \tau) \leftarrow \mathsf{Gen}(1^\lambda, R)\\ \pi \leftarrow \mathsf{Prove}(\mathsf{crs}, x, w)\end{array}\right] = \mathsf{negl}(\lambda).$$

- **Adaptive soundness:** For all p.p.t. algorithms $\overline{\mathcal{P}}$,

$$\Pr\left[\begin{array}{c}\mathsf{Verify}(\tau,x,\pi)=1\\ \wedge\ x\notin L\end{array}\ :\ \begin{array}{c}(\mathsf{crs},\tau)\leftarrow\mathsf{Gen}(1^\lambda,R)\\ (x,\pi)\leftarrow\overline{\mathcal{P}}(1^\lambda,\mathsf{crs})\end{array}\right]=\mathsf{negl}(\lambda).$$

  We additionally say that Π is *reusable* if soundness holds even against malicious provers $\overline{\mathcal{P}}$ with oracle access to $\mathsf{Verify}(\tau,\cdot,\cdot)$.

- **Non-adaptive soundness:** For every $x\notin L$, and every p.p.t. adversary $\overline{\mathcal{P}}$,

$$\Pr\left[\mathsf{Verify}(\tau,x,\pi)=1\ :\ \begin{array}{c}(\mathsf{crs},\tau)\leftarrow\mathsf{Gen}(1^\lambda,R)\\ \pi\leftarrow\overline{\mathcal{P}}(1^\lambda,\mathsf{crs})\end{array}\right]=\mathsf{negl}(\lambda).$$

An Π is *publicly verifiable* if $\tau=\mathsf{crs}$ or $\tau\subseteq\mathsf{crs}$.

Definition 3.17 (SNARG)**.** We say that $\Pi=(\mathsf{Gen},\mathsf{Prove},\mathsf{Verify})$ is a (designated verifier) **succinct** non-interactive argument (SNARG) for a language $L\in\mathsf{NP}$ if Π is a (designated verifier) non-interactive argument with the following additional succinctness condition:

- **Succinctness:** For proofs $\pi\leftarrow\mathsf{Prove}(\mathsf{crs},x,w)$ where $R(x,w)=1$, the proof length $|\pi|$ is at most $\mathsf{poly}(\lambda)\cdot(|x|+|w|)^{o(1)}$.[5]

Definition 3.18 (Verifier NIZK)**.** We say that $\Pi=(\mathsf{Gen},\mathsf{Prove},\mathsf{Verify})$ is a non-interactive **zero-knowledge** argument (NIZK) for a language $L\in\mathsf{NP}$ if Π is a non-interactive argument with one of the following zero knowledge properties.

- **Adaptive multi-theorem zero-knowledge:** There exists a p.p.t. simulator $\mathcal{S}=(\mathcal{S}_1,\mathcal{S}_2)$ that satisfies the following: For all (stateful) p.p.t. adversaries $\mathcal{A}$, we have that for experiments $\mathtt{EXP}^{\mathtt{Real}}_{\mathcal{A}}(1^\lambda)$ and $\mathtt{EXP}^{\mathtt{Ideal}}_{\mathcal{A}}(1^\lambda)$ as in Experiments 1 and 2,

$$\left|\Pr[\mathtt{EXP}^{\mathtt{Real}}_{\mathcal{A}}(1^\lambda)=1]-\Pr[\mathtt{EXP}^{\mathtt{Ideal}}_{\mathcal{A}}(1^\lambda)=1]\right|=\mathsf{negl}(\lambda)$$

  if $\mathcal{A}$ is limited to querying only (x,w) such that $R(x,w)=1$.

Experiment 1 $\mathtt{EXP}^{\mathtt{Real}}_{\mathcal{A}}(1^\lambda)$.

$(\mathsf{crs},\tau)\leftarrow\mathsf{Gen}(1^\lambda)$
$(x,w)\leftarrow\mathcal{A}(1^\lambda,\mathsf{crs},\tau)$
while $(x,w)\neq\bot$ and $R(x,w)=1$ **do**
  $\pi\leftarrow\mathsf{Prove}(\mathsf{crs},x,w)$
  $(x,w)\leftarrow\mathcal{A}(1^\lambda,\pi)$
end while
return $b\leftarrow\mathcal{A}\left(1^\lambda\right)$

Experiment 2 $\mathtt{EXP}^{\mathtt{Ideal}}_{\mathcal{A}}(1^\lambda)$.

$(\mathsf{crs},\tau,\mathsf{st})\leftarrow\mathcal{S}_1(1^\lambda)$
$(x,w)\leftarrow\mathcal{A}(1^\lambda,\mathsf{crs},\tau)$
while $(x,w)\neq\bot$ and $R(x,w)=1$ **do**
  $\pi\leftarrow\mathcal{S}_2(\mathsf{crs},\mathsf{st},x)$
  $(x,w)\leftarrow\mathcal{A}(1^\lambda,\pi)$
end while
return $b\leftarrow\mathcal{A}\left(1^\lambda\right)$

Additionally, we say Π is **statistically** adaptively multi-theorem zero-knowledge if the above indistinguishability holds even for all stateful *unbounded* adversaries $\mathcal{A}$ making $\mathsf{poly}(\lambda)$ queries.

[5] In fact, we obtain proof size $O(\lambda)$.

- **Adaptive single-theorem zero-knowledge:** This is defined similarly to adaptive multi-theorem zero-knowledge, except that $\mathcal{A}$ is allowed to make only a single query.

We follow the definition of black-box knowledge soundness of Campanelli et al. [CGKS23].

Definition 3.19 (Black-box knowledge soundness)**.** A designated verifier non-interactive argument system as in Definition 3.16 is non-adaptive black-box $\epsilon(\lambda)$-*knowledge sound* for a relation R if there exists a non-uniform PPT extracter Ext such that for any non-uniform PPT prover $\mathcal{P} = (\mathcal{P}_{inp}, \mathcal{P}_{chall})$,

$$\Pr\left[\begin{array}{c|c} \mathsf{Ver}(\tau, x, \pi) = 1 \\ \wedge\ R(x, w) \neq 1 \end{array} : \begin{array}{c} (x, \mathsf{st}) \leftarrow \mathcal{P}_{inp}(1^\lambda) \\ (\mathsf{crs}, \tau) \leftarrow \mathsf{Gen}(1^\lambda) \\ \pi \leftarrow \mathcal{P}_{chall}(\mathsf{st}, \mathsf{crs}) \\ w \leftarrow \mathsf{Ext}^{\mathcal{P}_{chall}(\mathsf{st}, \cdot)}(\mathsf{crs}, \tau, x, \pi) \end{array}\right] \leq \epsilon(\lambda).$$

We say that the argument system is non-adaptively black-box knowledge sound if $\epsilon(\lambda) = \mathsf{negl}(\lambda)$.

Intuitively, the prover selects the instance x on which she generates a proof, and the extractor is permitted to query the prover on many possible crs values (possibly with corresponding trapdoors) to reconstruct a witness for x.

Definition 3.20 (Non-Adaptive SNARK)**.** A (designated verifier) SNARG system (Definition 3.17) is also a *non-adaptive succinct argument of knowledge* (SNARK) if it is non-adaptively black-box knowledge sound, as in Definition 3.19.

3.5.3 Fully Homomorphic Encryption

Definition 3.21 (Fully homomorphic encryption)**.** A leveled fully homomorphic encryption scheme FHE with ciphertext size $\ell = \ell(\lambda) = \mathsf{poly}(\lambda)$ for a fixed polynomial ℓ is a tuple of PPT algorithms $(\mathsf{FHE.Gen}, \mathsf{FHE.Enc}, \mathsf{FHE.Dec}, \mathsf{FHE.Eval})$ with the following interface and properties.

Interface:

- $\mathsf{FHE.Gen}(1^\lambda, 1^d)$: Probabilistic algorithm that outputs a $(\mathsf{sk}, \mathsf{ek})$ where sk is the secret key, ek is a public evaluation key.
- $\mathsf{FHE.Enc}_{\mathsf{sk}}(b)$: Probabilistic algorithm that takes as input the secret key sk and the message bit $b \in \{0,1\}$, and outputs a ciphertext $\mathsf{ct} \in \{0,1\}^\ell$.
- $\mathsf{FHE.Dec}_{\mathsf{sk}}(\mathsf{ct})$: Deterministic algorithm that takes as input the secret key sk and a ciphertext ct, and outputs a bit $b \in \{0,1\}$.
- $\mathsf{FHE.Eval}_{\mathsf{ek}}(C, \mathsf{ct}_1, \ldots, \mathsf{ct}_m)$: Deterministic algorithm that takes as input the evaluation key ek, an m-input circuit C of size at most d, and ciphertexts $\{\mathsf{ct}_i\}_{i \in m}$, and outputs a ciphertext $\mathsf{ct}^* \in \{0,1\}^\ell$.

Properties:

- **Encryption correctness:** For all $\mathsf{sk} \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^d)$ and $b \in \{0,1\}$, we have that

$$\Pr[\mathsf{FHE.Dec}_{\mathsf{sk}}(\mathsf{FHE.Enc}_{\mathsf{sk}}(b)) = b] = 1$$

- **IND-CPA security:** or any PPT adversary $\mathcal{A}$ it holds that

$$\begin{aligned}|&\Pr_{\mathsf{sk}\leftarrow\mathsf{FHE.Gen}(1^\lambda)}[\mathcal{A}^{\mathsf{FHE.Enc}_{\mathsf{sk}}(\cdot)}(\mathsf{ek}, \mathsf{FHE.Enc}_{\mathsf{sk}}(0)) = 1] \\ &- \Pr_{\mathsf{sk}\leftarrow\mathsf{FHE.Gen}(1^\lambda)}[\mathcal{A}^{\mathsf{FHE.Enc}_{\mathsf{sk}}(\cdot)}(\mathsf{ek}, \mathsf{FHE.Enc}_{\mathsf{sk}}(1)) = 1]| \leq \mathsf{negl}(\lambda),\end{aligned}$$

  where $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^d)$.

- **Evaluation correctness:** For all circuits C of size at most $d(\lambda)$, we must have

$$\Pr[\mathsf{FHE.Dec}_{\mathsf{sk}}(\mathsf{FHE.Eval}_{\mathsf{ek}}(C, \mathsf{ct}_1, \ldots, \mathsf{ct}_m)) \neq f(\mu_1, \ldots, \mu_m)] \leq \mathsf{negl}(\lambda)\ ,$$

  where $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(1^\lambda)$ and $\mathsf{ct}_i \leftarrow \mathsf{FHE.Enc}_{\mathsf{sk}}(\mu_i)$.

Theorem 3.22 ([BV11, BGV12, GSW13])**.** *Assuming* $\mathsf{LWE}_{n,\mathsf{poly}(n),q,\sigma}$ *for* $n = \mathsf{poly}(\lambda)$, $q = 2^{n^\varepsilon}$ *for some constant* $\varepsilon > 0$*, and* $\sigma \geq \sqrt{2n}$*, there exists a leveled fully homomorphic encryption scheme with a deterministic evaluation algorithm, and ciphertexts of size* $\ell(\lambda)$*, for a fixed polynomial* ℓ*.*

3.5.4 Public-key Encryption

We additionally need a public key encryption scheme.

Definition 3.23 (Public key encryption)**.** A public key encryption scheme $\mathsf{PKE} = (\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ is a triple of algorithms:

- $\mathsf{PKE.Gen}(1^\lambda, 1^d)$: Probabilistic algorithm that outputs a $(\mathsf{sk}, \mathsf{pk})$ where sk is the secret key, pk is a public key.
- $\mathsf{PKE.Enc}_{\mathsf{pk}}(b)$: Probabilistic algorithm that takes as input the public key pk and the message bit $b \in \{0,1\}$, and outputs a ciphertext $\mathsf{ct} \in \{0,1\}^\ell$.
- $\mathsf{PKE.Dec}_{\mathsf{sk}}(\mathsf{ct})$: Deterministic algorithm that takes as input the secret key sk and a ciphertext ct, and outputs a bit $b \in \{0,1\}$.

The PKE scheme has to additionally sastisfy the following properties:

Encryption correctness: For all $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$ and $b \in \{0,1\}$, we have that

$$\Pr[\mathsf{PKE.Dec}_{\mathsf{sk}}(\mathsf{PKE.Enc}_{\mathsf{pk}}(b)) = b] = 1$$

IND-CPA security: or any PPT adversary $\mathcal{A}$ it holds that

$$\Big| \Pr_{(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{PKE.Gen}(1^\lambda)}[\mathcal{A}(\mathsf{pk},\mathsf{PKE.Enc}_{\mathsf{pk}}(0))=1] - \Pr_{(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{PKE.Gen}(1^\lambda)}[\mathcal{A}(\mathsf{pk},\mathsf{FHE.Enc}_{\mathsf{pk}}(1))=1]\Big| \leq \mathsf{negl}(\lambda),$$

where $(\mathsf{sk},\mathsf{ek}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$.

We need an encryption scheme with the following additional property:

Definition 3.24. We say that a public key encryption scheme PKE is an *injective encryption scheme* if, for $\mathsf{pk} \leftarrow \mathsf{PKE.Gen}(1^\lambda)$, the function $\mathsf{PKE.Enc}_{\mathsf{pk}}(b;r)$ is injective in the message bit b and the randomness r except with probability $\mathsf{negl}(\lambda)$.

Many public-key encryption schemes satisfy this property; in particular, that of Regev, which can be based on $\mathsf{LWE}_{n,\mathsf{poly}(n),q,\sigma}$ for $q \leq \mathsf{poly}(n)$, $\sigma \leq O(\sqrt{n})$ [Reg05].

Theorem 3.25 ([Reg05])**.** *For public keys of size $n \times m$ for $n \geq \lambda$, $m \geq 2n \log q$, Regev's public-key encryption scheme [Reg05] is an injective encryption scheme.*

3.5.5 Indistinguishability Obfuscation

Definition 3.26 (Indistinguishability obfuscation ($i\mathcal{O}$), [BGI+01])**.** An *indistinguishability obfuscator* $i\mathcal{O}$ for a circuit class $\mathcal{C} = \{\mathcal{C}_\lambda\}_\lambda$ is a PPT algorithm that satisfies the following conditions:

- **Correctness:** For all security parameters $\lambda \in \mathbb{N}$, for all $C \in \mathcal{C}_\lambda$, for all inputs x, we have that
$$\Pr[C' \leftarrow i\mathcal{O}(1^\lambda, C) : C'(x) = C(x)] = 1.$$
- **Polynomial Security:** For all pairs of circuits $C_0, C_1 \in \mathcal{C}_\lambda$ such that $C_0(x) = C_1(x)$ for all x,
$$i\mathcal{O}(1^\lambda, C_0) \approx_c i\mathcal{O}(1^\lambda, C_1)\ .$$

We additionally say that the $i\mathcal{O}$ is sub-exponentially secure if, for some $\alpha > 0$, all $\mathsf{poly}(2^{\lambda^\alpha})$-time adversaries have distinguishing advantage at most $\mathsf{negl}(2^{\lambda^\alpha})$.

4 Obfuscating Matrix PRFs with Noise (σ-Matrix PRFs)

In this section, we define a weakening of pseudorandom function (PRF) where the adversary sees the outputs after independent noise is added to each output. Then we show that, if such a σ-PRF has sufficient security and can be computed by a matrix branching program, it can be meaningfully obfuscated using evasive LWE.

4.1 Matrix Branching Programs

We will work with matrix branching programs (MBPs) that compute functions $f_k : \{0,1\}^\ell \to \mathbb{Z}_q$ for some prime q. In this paper we consider MBPs specified by a collection of matrices $\big(\mathbf{M}_{i,b} : i \in [h := c \cdot \ell], b \in \{0,1\}\big)$ and two vectors $\mathbf{u}, \mathbf{v}$ (all over some ring $\mathcal{R}$, which, for us, will always be $\mathbb{Z}_q$ for a prime q) such that for all $\mathbf{x} \in \{0,1\}^\ell$,

$$f(\mathbf{x}) = \mathbf{u}^T \left(\prod_{i=1}^{h} \mathbf{M}_{i,x_{j_i}} \right) \mathbf{v} \ ,$$

where $j_i := (i-1) \bmod \ell + 1$. More explicitly,

$$f(\mathbf{x}) = \mathbf{u}^T \big((\mathbf{M}_{1,x_1} \cdot \ \ldots \ \cdot \mathbf{M}_{\ell,x_\ell}) \cdot (\mathbf{M}_{\ell+1,x_1} \cdot \ \ldots \ \cdot \mathbf{M}_{2\ell,x_\ell}) \cdot \ldots \cdot (\mathbf{M}_{(c-1)\ell+1,x_1} \cdot \ \ldots \ \cdot \mathbf{M}_{c\ell,x_\ell}) \big) \mathbf{v} \ .$$

Such MBPs are called *read-c* MBPs. When $c = 1$, we say the MBP is *read-once*.

The following classical result shows that any function computable by logarithmic-depth Boolean circuits can be represented by a matrix branching program.

Theorem 4.1 (Barrington's Theorem [Bar86])**.** *If $f : \{0,1\}^n \to \{0,1\}$ can be computed by a circuit of depth d, then it can be computed by a matrix branching program*

$$\mathbf{M}_{i,b} : i \in [h := c \cdot \ell], b \in \{0,1\}, \mathbf{u}, \mathbf{v}$$

where $h = O(4^d)$, and all matrices $\mathbf{M}_{i,b} \in \{0,1\}^{5\times 5}$ are permutations.

4.2 σ-Matrix PRFs

We first recall the definition of a pseudorandom function (PRF). Below, $O(\cdot)$ denotes a truly random function.

Definition 4.2. A family of deterministic functions $\mathcal{F} := \{f_k : \mathcal{X}_\lambda \to \mathcal{Y}_\lambda\}$ is called *pseudorandom* if for all PPT adversaries $\mathcal{A}$,

$$\left| \Pr_{k,\mathcal{A}}[\mathcal{A}^{f_k(\cdot)}(1^\lambda) = 1] - \Pr_{k,\mathcal{A}}[\mathcal{A}^{O(\cdot)}(1^\lambda) = 1] \right| \leq \mathsf{negl}(\lambda) \ .$$

Our arguments will crucially rely on the following relaxation of the notion of a pseudorandom function with outputs in $\mathbb{Z}_q$. Informally, it is pseudorandom against adversaries who are only permitted to observe the output values after independent Gaussian noise has been added to each value.

Definition 4.3. Let $q = q(\lambda) \in \mathbb{N}$ and $\sigma = \sigma(\lambda) > 0$. A family of deterministic functions $\mathcal{F} := \{f_k : \mathcal{X}_\lambda \to \mathbb{Z}_{q(\lambda)}\}$ is called *σ-pseudorandom* if for all PPT adversaries $\mathcal{A}$,

$$\left| \Pr_{k,\mathcal{A},O'}[\mathcal{A}^{O'(\cdot)}(1^\lambda) = 1] - \Pr_{k,\mathcal{A}}[\mathcal{A}^{O(\cdot)}(1^\lambda) = 1] \right| \leq \mathsf{negl}(\lambda) \ ,$$

where the function O' is chosen by sampling discrete Gaussian errors $\{e_{\mathbf{x}} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}\}_{\mathbf{x}\in\mathcal{X}_\lambda}$, and on input $\mathbf{x} \in \mathcal{X}_\lambda$, O' outputs $O'(\mathbf{x}) = f_k(\mathbf{x}) + e_{\mathbf{x}}$.

Given a PPT function aux, we say that $\mathcal{F}$ is additionally *pseudorandom in the presence of* aux (resp. *σ-pseudorandom in the presence of* aux) if the inequality in Definition 4.2 (resp. Definition 4.3) holds even when the adversary is additionally given $\mathsf{aux}(k)$ as input.

As is typical, we will abbreviate "efficiently computable pseudorandom function family" as "PRF". Similarly, we will abbreviate "efficiently computable σ-pseudorandom function family" as "σ-PRF".

The σ-PRFs in this paper will have $\mathcal{X}_\lambda = \{0,1\}^h$, where $h = h(\lambda) \leq \mathsf{poly}(\lambda)$, and will be pseudorandom even against adversaries running in time $\mathsf{poly}(\mu)$, where $\mu = 2^{h^2\lambda} \gg 2^h$. Notice that such adversaries have the ability to write down the entire truth table of the σ-PRF (but with Gaussian errors added) and perform arbitrary polynomial-time computations on it.

We will call a σ-PRF computable by a $\mathsf{poly}(\lambda)$-sized MBP a *σ-matrix PRF*.

In this section, we show how to obfuscate σ-matrix PRFs with sufficient security, using evasive LWE. Our main obfuscation construction (Algorithm 3) will use Garg-Gentry-Halevi encodings of the more modern type introduced in their 2015 work [GGH15], or GGH encodings for short. Before we introduce GGH encodings and give our main construction, we need one additional tool, which we provide in the next subsection.

4.3 Transforming Read-c PRFs into Read-Once PRFs

Notice that for general read-c σ-matrix PRFs, it is not the case that all noisy products $\{\mathbf{u}\mathbf{M}_\mathbf{x}\mathbf{v} + e_\mathbf{x}\}_{\mathbf{x}\in\{0,1\}^h}$ are pseudorandom— the σ-PRF guarantee only requires that noisy products corresponding to inputs $\mathbf{x}' \in \{0,1\}^\ell$ (i.e., with $\mathbf{x} = \mathbf{x}' \mid \mathbf{x}' \mid \cdots \mid \mathbf{x}'$) need be pseudorandom. However, our proof techniques will require all products to be pseudorandom. The following construction is a generic transformation that modifies a read-c σ-matrix PRF so that all its products are pseudorandom, without losing functionality.

Construction 4.4. *On input a read-c, height-h MBP $f := (\{\mathbf{M}_{i,b}\}_{i\in[h],b\in\{0,1\}}, \mathbf{u}, \mathbf{v})$:*

- *Define* $\mathbf{C}_{i,b} = \mathrm{diag}(y_1, \ldots, y_{2\ell}, -1, \ldots, -1)$, *where* $\ell = c \cdot h$, *and for* $1 \leq j \leq \ell$ *and* $b \in \{0,1\}$,

$$y_{2j-b} = \begin{cases} 0 & i \bmod \ell = j \text{ and } b' = b \\ 1 & \text{otherwise.} \end{cases}$$

- *Sample matrices* $\mathbf{S}_{i,b} \leftarrow \mathcal{D}^{n\times n}_{\mathbb{Z},2\sqrt{n}}$, *and a vector* $\mathbf{a} \leftarrow \mathbb{Z}_q^n$.
- *Output* $g := (\{\mathbf{M}_{i,b}\}_{i\in[h],b\in\{0,1\}}, \mathbf{u}, \mathbf{v})$, *where*
  - $\mathbf{M}'_{i,b} = \mathrm{diag}(\mathbf{M}_{i,b}, \mathbf{C}_{i,b} \otimes \mathbf{S}_{i,b})$,
  - $\mathbf{u}' := \begin{pmatrix} \mathbf{u} \\ \mathbf{1} \otimes \mathbf{e}_1 \end{pmatrix}^T$,
  - $\mathbf{v}' := \begin{pmatrix} \mathbf{v} \\ \mathbf{1} \otimes \mathbf{a} \end{pmatrix}$.

Lemma 4.5. *Let $g, \{\mathbf{S}_{i,b}\}_{i\in[h],b\in\{0,1\}}$ be as in Construction 4.4 on input f. Then for all $\mathbf{x} \in \{0,1\}^\ell$,*

$$g(\mathbf{x} \mid \mathbf{x} \mid \cdots \mid \mathbf{x}) = f(\mathbf{x}) \ ,$$

where we recall that $\mathbf{x} \mid \mathbf{x} \mid \cdots \mid \mathbf{x}$ is the c-fold concatenation of $\mathbf{x}$ with itself.

Moreover, assuming $\mathsf{LWE}^{\delta}_{n',\mathsf{poly}(n'),q,\sigma}$, and letting

- $\{e_{\mathbf{x}} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}\}_{\mathbf{x}\in\{0,1\}^\ell}$,
- $\{e_{\mathbf{y}} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}\}_{\mathbf{y}\in\{0,1\}^h}$, *and*
- $\mathbf{y}' \in \{0,1\}^h$ *range over all* $\mathbf{y}'$ *not of the form* $\mathbf{x} \mid \mathbf{x} \mid \cdots \mid \mathbf{x}$,

we have that

$$\{g(\mathbf{y}) + e_{\mathbf{y}}\}_{\mathbf{y}\in\{0,1\}^h}, \{\mathbf{S}_{i,b}\}_{i\in[h],b\in\{0,1\}} \approx_c \{f(\mathbf{x}) + e_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^\ell}, \{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{y}'}, \{\mathbf{S}_{i,b}\}_{i\in[h],b\in\{0,1\}}$$

that is, all $\mathsf{poly}(\mu)$-time adversaries have distinguishing advantage at most $\mathsf{negl}(f(n))$.

Proof. It is straightforward to verify that for all $\mathbf{x} \in \{0,1\}^h$, all $b \in \{0,1\}$, and all $j \in [\ell]$, $(\mathbf{C}_{\mathbf{x}})_{2j-b,2j-b} = 1$ if and only if $\mathbf{x}_i = 1 - b$ for all i with $i \bmod \ell = j$; otherwise $(\mathbf{C}_{\mathbf{x}})_{2j-b,2j-b} = 0$. It follows that $\mathbf{1}\mathbf{C}_{\mathbf{x}}\mathbf{1}^T = \sum_{i=1}^{2\ell}(\mathbf{C}_{\mathbf{x}})_{ii} - \ell \leq 0$, with equality iff for all j, $\mathbf{x}_j = \mathbf{x}_{j+\ell} = \cdots = \mathbf{x}_{j+(c-1)\ell}$; that is, $\mathbf{x} = \mathbf{x}' \mid \mathbf{x}' \mid \cdots \mid \mathbf{x}'$.

Notice that, for all $\mathbf{x} \in \{0,1\}^h$,

$$(\mathbf{u}')^T\mathbf{M}'_{\mathbf{x}}\mathbf{v}' = \mathbf{u}^T\mathbf{M}_{\mathbf{x}}\mathbf{v} + (\mathbf{1}^T\mathbf{C}_{\mathbf{x}}\mathbf{1}) \cdot \mathbf{1}^T\mathbf{S}_{\mathbf{x}}\mathbf{a} \ .$$

This implies the claimed functionality; namely, if $\mathbf{x} = \mathbf{x}' \mid \mathbf{x}' \mid \cdots \mid \mathbf{x}'$, then $\mathbf{1}\mathbf{C}_{\mathbf{x}}\mathbf{1}^T = 0$, and

$$(\mathcal{A}(f_k))(\mathbf{x}) = (\mathbf{u}')^T\mathbf{M}'_{\mathbf{x}}\mathbf{v}' = \mathbf{u}^T\mathbf{M}_{\mathbf{x}}\mathbf{v} = f_k(\mathbf{x}') \ .$$

On the other hand, if $\mathbf{x}$ is not a c-fold repetition, then $\mathbf{1}^T\mathbf{C}_{\mathbf{x}}\mathbf{1} \neq 0$. And by BLMR PRF security (Lemma 3.8 along with Remark 3.9) for independent $e_{\mathbf{x}} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}$,

$$\{y_{\mathbf{x}}\mathbf{1}^T\mathbf{S}_{\mathbf{x}}\mathbf{a} + e_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^h}, \{\mathbf{S}_{i,b}\}_{i\in[h],b\in\{0,1\}} \approx_c \{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x}\in\{0,1\}^h}, \{\mathbf{S}_{i,b}\}_{i\in[h],b\in\{0,1\}} \ ,$$

where $y_{\mathbf{x}} := \mathbf{1}^T\mathbf{C}_{\mathbf{x}}\mathbf{1} \in \mathbb{Z}_q$, and all $\mathsf{poly}(\mu)$-time adversaries have distinguishing advantage at most $\mathsf{negl}(f(n))$. By a one-time pad argument, this immediately implies the claimed indistinguishability. □

4.4 GGH Encodings

To construct our obfuscation scheme, we rely on the machinery of Gorbunov, Gentry, and Halevi [GGH15], which we hereby refer to as *GGH encodings*. (This definition is closely related to the definitions given in [CVW18, VWW22].)

Construction 4.6. *Given as input matrices* $\{\mathbf{M}_{i,b} \in \mathbb{Z}_q^{n_{i-1} \times n_i}\}_{i\in[h],b\in\{0,1\}}$,[6] *the randomized algorithm* $\mathsf{ggh.encode}$ *outputs*

$$\{\mathbf{D}_{1,b} := \mathbf{M}_{1,b}\mathbf{A}_1 + \mathbf{E}_{1,b}\}_{b\in\{0,1\}},$$
$$\{\mathbf{D}_{i,b} := \mathbf{A}_{i-1}^{-1}\left(\mathbf{M}_{i,b}\mathbf{A}_i + \mathbf{E}_{i,b}, \sigma\right)\}_{2\le i\le h-1,b\in\{0,1\}},$$
$$\{\mathbf{D}_{h,b} := \mathbf{A}_{h-1}^{-1}(\mathbf{M}_{h,b} + \mathbf{E}_{h,b}, \sigma)\}_{b\in\{0,1\}}\ ,$$

where $\sigma = 2\sqrt{n\log q}$, $\mathbf{A}_i \in \mathbb{Z}_q^{n_i\times m_i}$ *is sampled using Lemma 3.2,* $\mathbf{E}_{i,b} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{n_{i-1}\times m_i}$, *and* $m_i := 4n_i\log q$.

We extend the construction to *MBPs* $P = (\{\mathbf{M}_{i,b}\}_{i\in[h],b\in\{0,1\}}, \mathbf{u}, \mathbf{v})$ via

$$\mathsf{ggh.encode}(P) := \mathsf{ggh.encode}(\{\mathbf{M}'_{i,b}\}_{i,b})\ ,$$

where $\mathbf{M}'_{1,b} := \mathbf{u}\mathbf{M}_{1,b}$, $\mathbf{M}'_{h,b} := \mathbf{M}_{h,b}\mathbf{v}$, and for $2 \le i \le h-1$, $\mathbf{M}'_{i,b} := \mathbf{M}_{i,b}$.

The next lemma, which is similar to [VWW22, Lemma 4.3] and [CVW18, Lemma 5.3], captures the functionality provided by the construction, which is roughly that if the input matrices $\mathbf{M}_{i,b}$ have small entries, then for all $\mathbf{x} \in \{0,1\}^h$, $|\mathbf{D}_\mathbf{x} - \mathbf{M}_\mathbf{x}|$ is small.

Lemma 4.7. *Except with probability* $\mathsf{negl}(\mu)$ *over* $\{\mathbf{D}_{i,b}\}_{i\in[h],b\in\{0,1\}} \leftarrow \mathsf{ggh.encode}(\{\mathbf{M}_{i,b}\}_{i,b}, \sigma)$, *letting* $B := \max\{\sigma\sqrt{n}, \max_{i,b}\|\mathbf{M}_{i,b}\|_\infty\}$ *we have that for all* $\mathbf{x} \in \{0,1\}^h$,

$$\|\mathbf{D}_\mathbf{x} - \mathbf{M}_\mathbf{x}\|_\infty \le h \cdot \left(\prod_{i=1}^{h-1} m_i\right) \cdot B^{h-1}\ ,$$

We defer the simple proof to Appendix A.3.

4.5 Evasive LWE

To argue the security property of GGH encodings that we need, we use the following evasive LWE assumption from [VWW22].

Let $\sigma, \sigma' \in \mathbb{R}_{>0}$, and let Samp be a PPT algorithm that on input 1^λ outputs

$$\mathbf{S} \in \mathbb{Z}_q^{n'\times n}, \mathbf{P} \in \mathbb{Z}_q^{n\times t}, \mathsf{aux} \in \{0,1\}^*\ .$$

We define the following advantage functions:

$$\mathsf{Adv}_{\mathcal{A}_0}^{\mathrm{pre}}(\lambda) := \Pr[\mathcal{A}_0(\mathbf{SB}+\mathbf{E}, \mathbf{SP}+\mathbf{E}', \mathsf{aux}) = 1] - \Pr[\mathcal{A}_0(\mathbf{C}, \mathbf{C}', \mathsf{aux}) = 1],$$

$$\mathsf{Adv}_{\mathcal{A}_1}^{\mathrm{post}}(\lambda) := \Pr[\mathcal{A}_1(\mathbf{SB}+\mathbf{E}, \mathbf{D}, \mathsf{aux}) = 1] - \Pr[\mathcal{A}_1(\mathbf{C}, \mathbf{D}, \mathsf{aux}) = 1]$$

[6] We remark that the "parallel" indices $b \in \{0,1\}$ can be generalized to $j \in [k_i]$ for arbitrary $\{k_i \in \mathbb{Z}\}_{i\in[h]}$, although we consider only binary indices in this paper.

where

$$
\begin{aligned}
&(\mathbf{S}, \mathbf{P}, \mathsf{aux}) \leftarrow \mathsf{Samp}(1^\lambda) \\
&\mathbf{B} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{E} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{n' \times m}, \mathbf{E}' \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{n' \times t}, \\
&\mathbf{C} \leftarrow \mathbb{Z}_q^{n' \times m}, \mathbf{C}' \leftarrow \mathbb{Z}_q^{n' \times t}, \\
&\mathbf{D} \leftarrow \mathbf{B}^{-1}(\mathbf{P}, \sigma) \ .
\end{aligned}
$$

We say that the *evasive LWE* assumption $\mathsf{eLWE}(\mathsf{Samp}, \sigma, \sigma')$ holds if there exists some polynomial $Q(\cdot)$ such that for every PPT $\mathcal{A}_1$ there exists another PPT $\mathcal{A}_0$ such that

$$
\mathsf{Adv}_{\mathcal{A}_0}^{\mathrm{pre}}(\lambda) \geq \mathsf{Adv}_{\mathcal{A}_1}^{\mathrm{post}}(\lambda)/Q(\lambda) - \mathsf{negl}(\lambda)
$$

and $\mathsf{time}(\mathcal{A}_0) \leq \mathsf{time}(\mathcal{A}_1) \cdot Q(\lambda)$. In this work, we will assume evasive LWE with $\sigma = \sigma'$.

4.6 Obfuscating Sufficiently Secure σ-matrix PRFs

Algorithm 3 GGH Encoding for σ-matrix PRFs

Input: A sample f_k from a height-h σ'-matrix PRF $\{f_k\}_{k \in \mathcal{K}_\lambda}$ (in the presence of aux).
Algorithm:

- Use Lemma 4.5 to convert f_k into a read-once σ'-matrix PRF

$$
g_k =: (\{\mathbf{M}_{i,b} \in \mathbb{Z}_q^{w \times w}\}_{i \in [h], b \in \{0,1\}}, \mathbf{u}, \mathbf{v} \in \mathbb{Z}_q^{w \times 1}) \ .
$$

- Sample $\mathbf{S}_{i,b} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{n \times n}$, where $\sigma = 2\sqrt{n}$.[7] Set $\widehat{\mathbf{S}}_{i,b} := \mathrm{diag}(\mathbf{M}_{i,b}, \mathbf{S}_{i,b}) \in \mathbb{Z}_q^{(n+w) \times (n+w)}$.[8]
- Set $\mathbf{u}' = (\mathbf{u}^T \mid \mathbf{1}_n)^T \in \mathbb{Z}_q^{(n+w) \times 1}$ and $\mathbf{v}' = (\mathbf{v}^T \mid \mathbf{0}_n)^T \in \mathbb{Z}_q^{(n+w) \times 1}$.
- Set $P = (\{\widehat{\mathbf{S}}_{i,b}\}_{i,b}, \mathbf{u}', \mathbf{v}')$.
- Compute $\{\mathbf{D}_{i,b}\}_{i \in [h], b \in \{0,1\}} \leftarrow \mathsf{ggh.encode}(P)$.

Output: $\{\mathbf{D}_{i,b}\}_{i \in [h], b \in \{0,1\}}$

The samplers Samp in our evasive LWE assumption will depend on the matrices sampled by $\mathsf{ggh.encode}$, so we write these explicitly. The outputs of $\mathsf{ggh.encode}$ (and our construction) are:

$$
\{\mathbf{D}_{1,b} \in \mathbb{Z}_q^m\}_{b \in \{0,1\}}, \ \{\mathbf{D}_{i,b} \in \mathbb{Z}_q^{m \times m}\}_{2 \leq i \leq h, b \in \{0,1\}}, \ \{\mathbf{D}_{h,b} \in \mathbb{Z}_q^m\}_{b \in \{0,1\}},
$$

$$
\begin{aligned}
&\{\mathbf{D}_{1,b} := \mathbf{u}'\widehat{\mathbf{S}}_{1,b}\mathbf{A}_1 + \mathbf{E}_{1,b}\}_{b \in \{0,1\}}, \\
&\{\mathbf{D}_{i,b} := \mathbf{A}_{i-1}^{-1}\left(\widehat{\mathbf{S}}_{i,b}\mathbf{A}_i + \mathbf{E}_{i,b}, \sigma\right)\}_{2 \leq i \leq h-1, b \in \{0,1\}}, \\
&\{\mathbf{D}_{h,b} := \mathbf{A}_{h-1}^{-1}(\widehat{\mathbf{S}}_{h,b}\mathbf{v}' + \mathbf{E}_{h,b}, \sigma)\}_{b \in \{0,1\}} \ ,
\end{aligned}
$$

where

- $m = 4(w+n)\log q$,
- for $j \in [h-1]$, $\mathbf{A}_i \approx_s \mathcal{U}(\mathbb{Z}_q^{(w+n)\times m})$ with security parameter μ,
- and $\mathbf{E}_{1,b} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{1\times m}, \mathbf{E}_{2,b}, \ldots, \mathbf{E}_{h-1,b} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{(w+n)\times m}, \mathbf{E}_{h,b} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{(w+n)\times 1}$.

Now we can describe the samplers $\mathsf{Samp}_{\mathcal{F},\mathsf{aux},1}, \ldots, \mathsf{Samp}_{\mathcal{F},\mathsf{aux},h-1}$ appearing in our evasive LWE assumptions.

Definition 4.8. Let $\mathcal{F} = \{f_k\}_{k\in\mathcal{K}_\lambda}$ be a height-h MBP, and aux be a PPT algorithm. For $1 \le j \le h-1$, the sampling algorithm $\mathsf{Samp}_{\mathcal{F},\mathsf{aux},j}$ is defined as follows. On input 1^λ, $\mathsf{Samp}_{\mathcal{F},\mathsf{aux},j}$ samples $k \leftarrow \mathcal{K}_\lambda$ and samples matrices as in Algorithm 3 on input f_k. It outputs

$$\begin{aligned}\mathbf{S} &= \{(\mathbf{u}^T \mid \mathbf{1}_n)^T \hat{\mathbf{S}}_{\mathbf{y}}\}_{\mathbf{y}\in\{0,1\}^j},\\ \mathbf{P} &= (\hat{\mathbf{S}}_{j+1,0}\mathbf{A}_{j+1} + \mathbf{E}_{j+1,0} \mid \hat{\mathbf{S}}_{j+1,1}\mathbf{A}_{j+1} + \mathbf{E}_{j+1,1}),\\ \mathsf{aux} &= (\{\mathbf{D}_{i,b}\}_{i\ge j+2,b}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}).\end{aligned}$$

With the necessary samplers defined, we can now state our technical obfuscation result.

Theorem 4.9. *Let $\sigma \ge \sqrt{2n}$ and $B \ge \sigma\sqrt{n}$, and suppose $\mathcal{F} := \{f_k\}_{k\in\mathcal{K}_\lambda}$ is a height-h MBP with all matrix entries bounded by B. Let $\{\mathbf{D}_{i,b}^{(k)}\}_{i\in[h],b\in\{0,1\}}$ be the output of Algorithm 3 on input f_k, and let $m := 4(n+w)\log q$. Then for all $k \in \mathcal{K}_\lambda$, except with probability $\mathsf{negl}(\mu)$, we have that for all $\mathbf{x} \in \{0,1\}^\ell$,*

$$|f_k(\mathbf{x}) - \mathbf{D}_{\mathbf{y}}^{(k)}| \le h(mB)^{h-1} ,$$

where $\mathbf{y} := \mathbf{x} \mid \mathbf{x} \mid \cdots \mid \mathbf{x} \in \{0,1\}^h$.

Moreover, letting $\sigma' = 2^{h^3}\cdot(n^2\sigma)^{h+1}$, and assuming $\mathsf{LWE}_{n,\mathsf{poly}(n),q,\sigma}$ and $\mathsf{eLWE}_{\mathsf{Samp}_{\mathcal{F},\mathsf{aux},j},\sigma',\sigma'}$ for all $1 \le j \le h-1$, if $\mathcal{F}$ is a σ'-matrix PRF such that $\mathsf{poly}(\mu)$-time adversaries achieve distinguishing advantage at most $\mathsf{negl}(2^{h^2\lambda})$, then there is a distribution $\mathcal{D}$ (independent of k) such that for $k \leftarrow \mathcal{K}_\lambda$,

$$\{\mathbf{D}_{i,b}\}_{i\in[h],b\in\{0,1\}}, \mathsf{aux}(k) \approx_c \mathcal{D}, \mathsf{aux}(k) ,$$

such that $\mathsf{poly}(\mu)$-time adversaries achieve distinguishing advantage at most $\mathsf{negl}(2^{h^2\lambda})$.

Proof. The analysis in large part follows that of [VWW22, Theorem 5.1].

First, by Lemma 4.5, we have that $\{g_k\}_k$ is also a height-h σ-matrix PRF with the same security guarantee as $\{f_k\}_k$; namely that all $\mathsf{poly}(\mu)$-time adversaries achieve distinguishing advantage at most $\mathsf{negl}(2^{h^2\lambda})$.

By Lemma 3.6, except with probability $\mathsf{negl}(\mu)$, the matrices of g_k have entries bounded in absolute value by B. Thus for all $\mathbf{y} \in \{0,1\}^h$, we have

$$|\mathbf{D}_{\mathbf{y}} - (\mathbf{u}')^T \widehat{\mathbf{S}}_{\mathbf{y}} \mathbf{v}'| \le h(mB)^{h-1}, \tag{3}$$

by correctness of GGH encodings (Lemma 4.7). Moreover, for all $\mathbf{x} \in \{0,1\}^\ell$, letting $\mathbf{y} = \mathbf{x} \mid \mathbf{x} \mid \cdots \mid \mathbf{x}$, we have

$$(\mathbf{u}')^T \widehat{\mathbf{S}}_{\mathbf{y}} \mathbf{v}' = \mathbf{u}^T \mathbf{M}_{\mathbf{y}} \mathbf{v} = g_k(\mathbf{x}) = f_k(\mathbf{x}) ,$$

where the first equality is algebraic, the second is by definition, and the third is by the guarantee of Lemma 4.5. Substituting this equality into (3) proves the first part of the theorem.

For the second part, we will argue that it suffices to prove the following claim.

Claim 4.10. *Assume* $\mathsf{LWE}_{n,\mathsf{poly}(n),q,\sigma}$ *and* $\mathsf{eLWE}_{\mathsf{Samp}_{\mathcal{F},\mathsf{aux},j},\sigma',\sigma'}$ *for all* $1 \le j \le h-1$. *Set* $d = m$ *for* $p < h$ *and* $d = 1$ *for* $p = h$, *and* $\sigma'' = \sigma$ *for* $p = 1$ *and* $\sigma'' = \sigma'$ *for* $p > 1$. *Sample* $\{\mathbf{e}_{\mathbf{y}} \leftarrow \mathcal{D}^{1\times d}_{\mathbb{Z},\sigma''}\}_{\mathbf{y}\in\{0,1\}^p}$, *and let* $\mathbf{A}_h := \mathbf{v}'$. *Then for all* $1 \le p \le h$,

$$\begin{aligned}&\left(\{(\mathbf{u} \mid \mathbf{1}_n)\widehat{\mathbf{S}}_{\mathbf{y}}\mathbf{A}_p + \mathbf{E}_{\mathbf{y}}\}_{\mathbf{y}\in\{0,1\}^p}, \{\mathbf{D}_{i,b}\}_{i\ge p+1,b\in\{0,1\}}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}_{i\in[h],b\in\{0,1\}}\right)\\ &\approx_c \left(\{\mathcal{U}(\mathbb{Z}_q)^{1\times d}\}_{\mathbf{y}\in\{0,1\}^p}, \{\mathbf{D}_{i,b}\}_{i\ge p+1,b\in\{0,1\}}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}_{i\in[h],b\in\{0,1\}}\right),\end{aligned}$$

where all $\mathsf{poly}(\mu)$*-time adversaries have distinguishing advantage* $\mathsf{negl}(2^{h^2\lambda})$.

Indeed, the claim with $p = 1$ is exactly

$$\begin{aligned}&(\{\mathbf{D}_{i,b}\}_{i\in[h],b\in\{0,1\}}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\})\\ &\approx_c (\{\mathcal{U}(\mathbb{Z}_q)^{1\times m}\}_{b\in\{0,1\}}, \{\mathbf{D}_{i,b}\}_{2\le i\le h,b\in\{0,1\}}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}).\end{aligned} \tag{4}$$

Now, the only place that $\mathbf{A}_1$ appears on the right-hand side of (4) is in $\mathbf{D}_{2,0}$ and $\mathbf{D}_{2,1}$. But

$$\begin{pmatrix}\mathbf{D}_{2,0}\\ \mathbf{D}_{2,1}\end{pmatrix} = \mathbf{A}_1^{-1}\left(\begin{pmatrix}\widehat{\mathbf{S}}_{2,0}\\ \widehat{\mathbf{S}}_{2,1}\end{pmatrix}\mathbf{A}_2 + \begin{pmatrix}\mathbf{E}_{2,0}\\ \mathbf{E}_{2,1}\end{pmatrix}\right).$$

Since $\begin{pmatrix}\mathbf{E}_{2,0}\\ \mathbf{E}_{2,1}\end{pmatrix}$ is distributed as $\mathcal{D}^{2(w+n)\times m}_{\mathbb{Z},\sigma}$, and does not appear anywhere else on the right-hand side of Eq. (4), Lemma 3.4 allows us to replace $\begin{pmatrix}\mathbf{D}_{2,0}\\ \mathbf{D}_{2,1}\end{pmatrix}$ with $\mathcal{D}^{2(w+n)\times m}_{\mathbb{Z},\sigma}$, with security parameter μ. But in the resulting hybrid, $\mathbf{A}_2$ only appears in $\mathbf{D}_{3,0}$ and $\mathbf{D}_{3,1}$. Hence we can repeat the argument to replace $\mathbf{D}_{3,b}$ with Gaussian samples. Continuing in this way, we get that

$$\begin{aligned}&\{\mathbf{D}_{1,b}\}_{b\in\{0,1\}}, \{\mathbf{D}_{i,b}\}_{2\le i\le h,b\in\{0,1\}}, \mathsf{aux}(k)\\ &\approx_c \{\mathcal{U}(\mathbb{Z}_q)^{m\times 1}\}_{b\in\{0,1\}}, \{\mathcal{D}^{(w+n)\times m}_{\mathbb{Z},\sigma}\}_{2\le i\le h,b\in\{0,1\}}, \mathsf{aux}(k)\end{aligned}$$

with security parameter μ, as desired.

It remains to prove Claim 4.10. We proceed by induction in decreasing order of p.

Base case. The base case $p = h$ is

$$\left(\{g_k(\mathbf{x}) + e_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^h}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}\right) \approx_c \left(\{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x}\in\{0,1\}^h}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}\right),$$

where $g_k(\mathbf{x}) = \mathbf{u}' \cdot \widehat{\mathbf{S}}_{\mathbf{x}} \cdot \mathbf{v}'$.

To prove the base case, start with our assumption that $\mathcal{F}$ is a σ-PRF in the presence of aux such that $\mathsf{poly}(\mu)$-time adversaries obtain advantage at most $\mathsf{negl}(2^{h^2\lambda})$; that is, letting $\mathbf{z} = \mathbf{y} \mid \mathbf{y} \mid \cdots \mid \mathbf{y}$,

for random $k \leftarrow \mathcal{K}_\lambda$,

$$\left(\{f_k(\mathbf{z}) + e_{\mathbf{z}}\}_{\mathbf{y}\in\{0,1\}^\ell}, \mathsf{aux}(k)\right) \approx_c \left(\{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{y}\in\{0,1\}^\ell}, \mathsf{aux}(k)\right) .$$

On the other hand, Lemma 4.5 guarantees that for any fixed key k, over the randomness of the $\mathbf{S}_{i,b}$ and the $e_{\mathbf{y}}$,

$$\{g_k(\mathbf{y}) + e_{\mathbf{y}}\}_{\mathbf{y}\in\{0,1\}^h}, \{\mathbf{S}_{i,b}\}_{i\in[h],b\in\{0,1\}} \approx_c \{f_k(\mathbf{x}) + e_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^\ell}, \{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{y}'}, \{\mathbf{S}_{i,b}\}_{i\in[h],b\in\{0,1\}} ,$$

where again, $\mathsf{poly}(\mu)$-time adversaries obtain advantage at most $\mathsf{negl}(2^{h^2\lambda})$. Applying the first indistinguishability to the $\{f_k(\mathbf{x}) + e_{\mathbf{x}}\}_{\mathbf{x}\in\{0,1\}^\ell}$ term of the second and using the fact that the $\mathbf{S}_{i,b}$ are sampled independently of k yields the base case.

Inductive step. Suppose Claim 4.10 is true for $p = j + 1$; we will prove it for $p = j$. For all $i \in [h]$ and all $\mathbf{y} \in \{0,1\}^i$, let $\mathbf{E}_{\mathbf{y}} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{1\times m}$ (previously, $\mathbf{E}_{\mathbf{y}}$ was only defined for $i = h$). Recall that $\mathbf{y} \mid b$ denotes the concatenation of bitstring $\mathbf{y}$ with bit b. We claim that all $\mathsf{poly}(\mu)$-time adversaries obtain advantage at most $\mathsf{negl}(2^{h^2\lambda})$ in distinguishing the following sequence of hybrids:

$$\begin{aligned}
&(\{(\mathbf{u} \mid \mathbf{1}_n)\hat{\mathbf{S}}_{\mathbf{y}}\mathbf{A}_j + \mathbf{E}_{\mathbf{y}}\}_{\mathbf{y}\in\{0,1\}^j},\\
&\qquad \{(\mathbf{u} \mid \mathbf{1}_n)\hat{\mathbf{S}}_{\mathbf{y}} \cdot (\hat{\mathbf{S}}_{j+1,b}\mathbf{A}_{j+1} + \mathbf{E}_{j+1,b}) + \mathbf{E}_{\mathbf{y}|b}\}_{\mathbf{y}\in\{0,1\}^j,b\in\{0,1\}},\\
&\qquad \{\mathbf{D}_{i,b}\}_{i\geq j+2,b}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}) && (\mathsf{H}_1)\\
&= (\{(\mathbf{u} \mid \mathbf{1}_n)\hat{\mathbf{S}}_{\mathbf{y}}\mathbf{A}_j + \mathbf{E}_{\mathbf{y}}\}_{\mathbf{y}\in\{0,1\}^j},\\
&\quad \left\{(\mathbf{u} \mid \mathbf{1}_n)\hat{\mathbf{S}}_{\mathbf{y}|b}\mathbf{A}_{j+1} + \mathbf{E}_{\mathbf{y}|b} + (\mathbf{u} \mid \mathbf{1}_n)\widehat{\mathbf{S}}_{\mathbf{y}}\mathbf{E}_{j+1,b}\right\}_{\mathbf{y}\in\{0,1\}^j,b\in\{0,1\}},\\
&\quad \{\mathbf{D}_{i,b}\}_{i\geq j+2,b}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}) && (\mathsf{H}_2)\\
&\approx_s (\{(\mathbf{u} \mid \mathbf{1}_n)\hat{\mathbf{S}}_{\mathbf{y}}\mathbf{A}_j + \mathbf{E}_{\mathbf{y}}\}_{\mathbf{y}\in\{0,1\}^j},\\
&\qquad \left\{(\mathbf{u} \mid \mathbf{1}_n)\hat{\mathbf{S}}_{\mathbf{y}'}\mathbf{A}_{j+1} + \mathbf{E}_{\mathbf{y}'}\right\}_{\mathbf{y}'\in\{0,1\}^{j+1}},\\
&\qquad \{\mathbf{D}_{i,b}\}_{i\geq j+2,b}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}) && (\mathsf{H}_3)\\
&= (\{\mathbf{u}\mathbf{M}_{\mathbf{y}}\overline{\mathbf{A}_j} + (\mathbf{1}_n)\mathbf{S}_{\mathbf{y}}\underline{\mathbf{A}_j} + \mathbf{E}_{\mathbf{y}}\}_{\mathbf{y}\in\{0,1\}^j},\\
&\qquad \left\{(\mathbf{u} \mid \mathbf{1}_n)\hat{\mathbf{S}}_{\mathbf{y}'}\mathbf{A}_{j+1} + \mathbf{E}_{\mathbf{y}'}\right\}_{\mathbf{y}\in\{0,1\}^{j+1}},\\
&\qquad \{\mathbf{D}_{i,b}\}_{i\geq j+2,b}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}) && (\mathsf{H}_4)\\
&\approx_c (\{\mathbf{u}\mathbf{M}_{\mathbf{y}}\overline{\mathbf{A}_j} + \mathcal{U}(\mathbb{Z}_q)^{1\times m}\}_{\mathbf{y}\in\{0,1\}^j},\\
&\qquad \left\{(\mathbf{u} \mid \mathbf{1}_n)\hat{\mathbf{S}}_{\mathbf{y}'}\mathbf{A}_{j+1} + \mathbf{E}_{\mathbf{y}'}\right\}_{\mathbf{y}\in\{0,1\}^{j+1}},\\
&\qquad \{\mathbf{D}_{i,b}\}_{i\geq j+2,b}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}) && (\mathsf{H}_5)\\
&= (\{\mathcal{U}(\mathbb{Z}_q)^{1\times m}\}_{\mathbf{y}\in\{0,1\}^j},\\
&\qquad \left\{(\mathbf{u} \mid \mathbf{1}_n)\hat{\mathbf{S}}_{\mathbf{y}'}\mathbf{A}_{j+1} + \mathbf{E}_{\mathbf{y}'}\right\}_{\mathbf{y}\in\{0,1\}^{j+1}},\\
&\qquad \{\mathbf{D}_{i,b}\}_{i\geq j+2,b}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}) && (\mathsf{H}_6)
\end{aligned}$$

$$\approx_c (\{\mathcal{U}(\mathbb{Z}_q)^{1\times m}\}_{\mathbf{y}\in\{0,1\}^j}, \{\mathcal{U}(\mathbb{Z}_q)^{1\times m}\}_{\mathbf{y}'\in\{0,1\}^{j+1}},$$
$$\{\mathbf{D}_{i,b}\}_{i\geq j+2,b}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\}) \ . \tag{H_7}$$

We prove each step as follows:

- (H_1) = (H_2) by linear algebra.
- (H_2) $\approx_s$ (H_3) by noise-flooding (Lemma 3.7). In particular, $\mathbf{E}_{\mathbf{y}|b}$ is a discrete Gaussian with parameter σ' satisfying $\sigma' \geq (2^{h^2\lambda})^{\omega(1)}\|(\mathbf{u} \mid \mathbf{1}_n)\widehat{\mathbf{S}}_{\mathbf{y}}\mathbf{E}_{j-1,b}\|_\infty$.
- (H_3) = (H_4) by definition of $\widehat{\mathbf{S}}_{\mathbf{y}}$.
- (H_4) $\approx_c$ (H_5) by BLMR security (Lemma 3.8) of the family $\{(\mathbf{1}_n)\mathbf{S}_{\mathbf{y}}\mathbf{A}_j + \mathbf{E}_{\mathbf{y}}\}$ with secret $\underline{\mathbf{A}_j}$. Note that the parameters of this family are exactly the "in particular" parameters of the lemma statement.
- (H_5) = (H_6) since the additive $\mathcal{U}(\mathbb{Z}_q)^{1\times m}$ acts as a one-time pad.
- (H_6) $\approx_c$ (H_7) by invoking the inductive hypothesis.

Now, we may apply evasive LWE with the following choices of variables, where we view $\{\cdot\}$ as stacking the matrices vertically.

$$\begin{aligned}
\mathbf{S} &= \{(\mathbf{u} \mid \mathbf{1}_n)\hat{\mathbf{S}}_{\mathbf{y}}\}_{\mathbf{y}\in\{0,1\}^j},\\
\mathbf{B} &= \mathbf{A}_j,\\
\mathbf{P} &= (\hat{\mathbf{S}}_{j+1,0}\mathbf{A}_{j+1} + \mathbf{E}_{j+1,0} \mid \hat{\mathbf{S}}_{j+1,1}\mathbf{A}_{j+1} + \mathbf{E}_{j+1,1}),\\
\mathsf{aux} &= (\{\mathbf{D}_{i,b}\}_{i\geq j+2,b}, \mathsf{aux}(k), \{\mathbf{S}_{i,b}\})
\end{aligned}$$

The fact (H_1) $\approx_c$ (H_7) is (up to rearranging matrices) exactly the precondition of $\mathsf{eLWE}_{\mathsf{Samp}_{\mathcal{F},\mathsf{aux},j},\sigma',\sigma'}$:

$$(\mathbf{SB} + \mathbf{E}, \mathbf{SP} + \mathbf{E}', \mathsf{aux}) \approx_c (\mathcal{U}(\mathbb{Z}_q)^{2^j\times m}, \mathcal{U}(\mathbb{Z}_q)^{2^j\times 2m}, \mathsf{aux}),$$

where by definition $\mathbf{E} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{2^j\times m}$, $\mathbf{E}' \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}^{2^j\times 2m}$. Thus, by evasive LWE, we have

$$(\mathbf{SB} + \mathbf{E}, \mathbf{B}^{-1}(\mathbf{P}), \mathsf{aux}) \approx_c (\mathcal{U}(\mathbb{Z}_q)^{2^j\times m}, \mathbf{B}^{-1}(\mathbf{P}), \mathsf{aux}) \ ,$$

which is exactly the statement of the claim with $p = j$. To finish the inductive step, we must write down the security loss from evasive LWE. For $j \in [h]$, let $\mathcal{A}_j$ denote an adversary that breaks the indistinguishability of Claim 4.10 with $p = j$.

Then, evasive LWE with security parameter $\lambda' = 2^j\lambda$ (so that $n' = 2^{j-1}(n + w)$ is bounded by $\mathsf{poly}(\lambda')$) tells us

$$\mathsf{Adv}(\mathcal{A}_j) \geq \mathsf{Adv}(\mathcal{A}_{j-1})/\mathsf{poly}\big(2^j\lambda\big), \qquad \mathsf{time}(\mathcal{A}_j) \leq \mathsf{time}(\mathcal{A}_{j-1}) \cdot \mathsf{poly}\big(2^j\lambda\big)$$

which implies

$$\mathsf{Adv}(\mathcal{A}_h) \geq \mathsf{Adv}(\mathcal{A}_1)/\mathsf{poly}\left(2^{h^2}\lambda^h\right), \qquad \mathsf{time}(\mathcal{A}_h) \leq \mathsf{time}(\mathcal{A}_1) \cdot \mathsf{poly}\left(2^{h^2}\lambda^h\right).$$

Thus we need the base case indistinguishability, that is Claim 4.10 with $p = h$, to hold with security parameter $2^{h^2}\lambda^h$. But the claim in fact holds with security parameter $2^{h^2\lambda} > 2^{h^2}\lambda^h$. This completes the proof. □

5 Witness PRFs for UP

In this section, we first construct witness PRFs for UP, that is, the class of unambiguous non-deterministic polynomial time languages.

5.1 Definitions

We first recall the definition of witness PRFs due to Zhandry [Zha16]. [9]

Definition 5.1 (Witness PRFs)**.** A *witness PRF* is a triple of PPT algorithms $(\mathsf{Gen}, \mathsf{F}, \mathsf{Eval})$ with the following interface and properties.

- $\mathsf{Gen}(1^\lambda, R_\alpha)$: On input 1^λ and a circuit R_α, outputs a secret function key fk and a public evaluation key ek.

  (Here, $\alpha, s \leq \mathsf{poly}(\lambda)$, and $R = \{R_\alpha : \{0,1\}^\alpha \times \{0,1\}^s \to \{0,1\}\}_{\alpha \in \mathbb{N}}$ represents an NP relation. When R and α are clear from context, we will write $\mathcal{X} := \{0,1\}^\alpha$ and $\mathcal{W} := \{0,1\}^s$.)

- $\mathsf{F}(\mathsf{fk}, x)$: On input key fk and statement $x \in \mathcal{X}$, deterministically outputs some $\mathbf{y} \in \mathcal{Y} := \{0,1\}^t$.

- $\mathsf{Eval}(\mathsf{ek}, x, w)$: On input the evaluation key ek, statement $x \in \mathcal{X}$ and witness $w \in \mathcal{W}$, outputs either $\mathbf{y} \in \mathcal{Y}$ or $\perp$.

Correctness: Except with probability $\mathsf{negl}(\lambda)$, for all $(x, w) \in \mathcal{X} \times \mathcal{W}$,

$$\mathsf{Eval}(\mathsf{ek}, x, w) = \begin{cases} \mathsf{F}(\mathsf{fk}, x) & \text{if } R(x,w) = 1 \\ \perp & \text{if } R(x,w) = 0. \end{cases}$$

Security: Consider the following experiment $\mathtt{EXP}^R_{\mathcal{A}}(b, \lambda)$ between a challenger $\mathcal{C}$ and adversary $\mathcal{A}$:

- $\mathcal{C}$ runs $(\mathsf{fk}, \mathsf{ek}) \leftarrow \mathsf{Gen}(1^\lambda, R_\alpha)$, and gives ek to $\mathcal{A}$. $\mathcal{C}$ additionally samples a uniformly random function $g : \mathcal{X} \to \mathcal{Y}$.

- $\mathcal{C}$ additionally initializes two sets $C \leftarrow \varnothing$, and $V \leftarrow \{x\}_{x \in L}$. The set C will be used to keep track of the points where the adversary has "challenged", and the set V will be used to keep

[9] Although Definition 5.1 looks slightly different from the original definition due to Zhandry, the two are easily seen to be equivalent.

track of the points x where the adversary requests $\mathsf{F}(\mathsf{fk}, x)$. The invariant $C \cap V = \varnothing$ will be maintained.

- $\mathcal{A}$ now can make adaptive queries to $x \in \mathcal{X}$ to which $\mathcal{C}$ responds as follows:
  - $\mathcal{A}$ can send an "evaluation query" $(\texttt{``F''}, x)$ for $x \notin C$. $\mathcal{C}$ updates $V \leftarrow V \cup \{x\}$ and returns $\mathsf{F}(\mathsf{fk}, x)$.
  - $\mathcal{A}$ can send a "challenge query" $(\texttt{``Challenge''}, x)$ to $\mathcal{C}$.
    - If $x \notin V$, then $\mathcal{C}$ computes $\mathbf{y}_0 \leftarrow \mathsf{F}(\mathsf{fk}, x)$ and $\mathbf{y}_1 \leftarrow g(x)$, updates $C \leftarrow C \cup \{x\}$, and returns $\mathbf{y}_b$.
    - If $x \in V$, then $\mathcal{C}$ responds with $\mathsf{F}(\mathsf{fk}, x)$ (i.e. always responds consistently with the evaluation query).
- $\mathcal{A}$ produces a bit b'.

Let W_b be the event that the adversary outputs 1 in experiment b, and define the adversary's advantage to be $\mathtt{Adv}^R_{\mathcal{A}}(\lambda) = |\Pr[W_0] - \Pr[W_1]|$. We say that $(\mathsf{Gen}, \mathsf{F}, \mathsf{Eval})$ is *adaptively interactively secure* for a relation R if for all PPT $\mathcal{A}$, $\mathtt{Adv}^R_{\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$.

Definition 5.2 (Unambiguous non-deterministic polynomial time (UP))**.** Let $p : \mathbb{N} \to \mathbb{N}$ be an efficiently computable, polynomially bounded function, and let $L \in \mathsf{NP}$. An efficiently computable relation $R : \{0,1\}^\alpha \times \{0,1\}^{p(\alpha)} \to \{0,1\}$ is a unambiguous non-deterministic polynomial time (UP) relation for L if for all $\alpha \in \mathbb{N}$, for all $\mathbf{x} \in \{0,1\}^\alpha$,

- if $\mathbf{x} \in L$, there is a *unique* $\mathbf{w} \in \{0,1\}^{p(\alpha)}$ such that $R(\mathbf{x}, \mathbf{w}) = 1$;
- if $\mathbf{x} \notin L$, there are no $\mathbf{w} \in \{0,1\}^{p(\alpha)}$ such that $R(\mathbf{x}, \mathbf{w}) = 1$.

If $L \in \mathsf{NP}$ has a UP relation we say that $L \in \mathsf{UP}$.

5.2 Construction

The construction takes inspiration from the witness encryption construction of [VWW22].

Construction 5.3. *Let $\alpha = \alpha(\lambda)$, and let $R_\alpha : \{0,1\}^\alpha \times \{0,1\}^{p(\alpha)} \to \{0,1\}$ be a UP relation. Choose $\alpha(\lambda) \in \mathsf{poly}(\lambda)$ small enough that $\alpha, p(\alpha) \leq n^{\delta/10}$. By a classical reduction to Circuit-SAT, we may assume without loss of generality that $R_\alpha(\mathbf{x}, \mathbf{w})$ is represented by a circuit of depth $O(\log|\alpha + p(\alpha)|) = O(\log \lambda)$. Let $\ell = \alpha + p(\alpha)$.*

The generation algorithm $\mathsf{Gen}(1^\lambda, R_\alpha)$ proceeds as follows. First, it uses Barrington's theorem (Theorem 4.1) to construct a read-c MBP

$$\Gamma_\alpha = \left(\{\mathbf{M}_{i,b} \in \{0,1\}^{v \times v}\}_{i \in [h], b \in \{0,1\}}, \mathbf{u}, \mathbf{v} \in \{0,1\}^v, \right) ,$$

computing $1 - R_\alpha$, where $v = O(1)$, $c, l \in \mathsf{poly}(\lambda)$, and $h := c \cdot l$. Specifically, for all $\mathbf{x} \in \{0,1\}^\alpha$ and $\mathbf{w} \in \{0,1\}^\ell$,

$$1 - R_\alpha(\mathbf{x}, \mathbf{w}) = \mathbf{u}^T \mathbf{M}_{(\mathbf{x},\mathbf{w})|(\mathbf{x},\mathbf{w})|\cdots|(\mathbf{x},\mathbf{w})} \mathbf{v} \, .$$

For $i \in [h]$, it samples $\mathbf{S}_{i,b} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{n\times n}$, and for $i \le \alpha$, it samples $\mathbf{T}_{i,b} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}^{n\times n}$, where $\sigma = \sqrt{2n}$. Next, it samples matrices

$$\mathbf{Q}_{i,b} = \begin{cases} \begin{pmatrix} \mathbf{M}_{i,b} \otimes \mathbf{S}_{i,b} & \\ & \mathbf{T}_{i,b}, \end{pmatrix} & 1 \le i \le \alpha, \\ \begin{pmatrix} \mathbf{M}_{i,b} \otimes \mathbf{S}_{i,b} & \\ & \mathbf{I} \end{pmatrix} & \alpha < i \le h, \end{cases}$$
$$\mathbf{L} = \left((\mathbf{u}^T \otimes (1,0,\ldots,0)) \mid (1,0,\ldots,0)\right),$$
$$\mathbf{R} = \begin{pmatrix} \mathbf{v} \otimes \mathbf{a} \\ \mathbf{b} \end{pmatrix},$$

where $\mathbf{a}, \mathbf{b} \leftarrow \mathbb{Z}_q^n$, and sets $\mathcal{F} := (\mathbf{L}, \{\mathbf{Q}_{i,b}\}_{i\in[h],b\in\{0,1\}}, \mathbf{R})$. Finally, it outputs

$$\mathsf{fk} := \{\{\mathbf{T}_{i,b}\}_{i\in[\alpha],b\in\{0,1\}}, \mathbf{b}\},$$
$$\mathsf{ek} := P := \mathsf{ggh.encode}(\mathcal{F}) .$$

The other two algorithms are as follows, where $p := 2^{h^7\lambda}$ (following Lemma 3.14).

- $\mathsf{F}(\mathsf{fk}, \mathbf{x})$: *Outputs* $\left\lfloor \mathbf{1}^T \mathbf{T}_{\mathbf{x}} \cdot \mathbf{b} \right\rceil_p \in \mathbb{Z}_p$.
- $\mathsf{Eval}(\mathsf{ek}, \mathbf{x}, \mathbf{w})$: *If $R_\alpha(\mathbf{x}, \mathbf{w}) = 0$, output $\perp$. Else, use* ek *to compute*

$$y := \lfloor P(\mathbf{x}, \mathbf{w}) \rceil_p \in \mathbb{Z}_p$$

  and output y.

Lemma 5.4. *View $(\mathbf{L}, \{\mathbf{Q}_{i,b}\}_{i\in[h],b\in\{0,1\}}, \mathbf{R})$ in Construction 5.3 as a family of MBPs $\{f_k\}_k$ whose keys are the random matrices $k := (\{\mathbf{S}_{i,b}\}_{i,b}, \{\mathbf{T}_{i,b}\}_{i,b})$. Let $\sigma = \sqrt{2n}, \sigma' = 2^{h^3\lambda} \cdot (n^2\sigma)^{h+1}$ (following Lemma 3.8), and let $\mathsf{aux}(k) := \{\lfloor \mathbf{T}_{\mathbf{x}} \cdot \mathbf{B} \rceil_p\}_{\mathbf{x}\notin L}$. Then $\{f_k\}_k$ is a σ'-matrix PRF in the presence of* aux *such that all* $\mathsf{poly}(\mu)$*-time adversaries have distinguishing advantage at most* $\mathsf{negl}(2^{-h^2\lambda})$.

Proof. Let $\{\mathbf{S}_{i,b}\}_{i,b}, \{\mathbf{T}_{i,b}\}_{i,b}$ be sampled as in Construction 5.3. We wish to show that for $\{e_{\mathbf{x}|\mathbf{w}} \leftarrow \mathbf{D}_{\mathbb{Z},\sigma'}\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}}$,

$$\{\mathbf{L}\mathbf{Q}_{(\mathbf{x},\mathbf{w})|(\mathbf{x},\mathbf{w})|\cdots|(\mathbf{x},\mathbf{w})}\mathbf{R} + e_{\mathbf{x}|\mathbf{w}}\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}}, \mathsf{aux} \approx_c \{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}}, \mathsf{aux} . \tag{5}$$

For ease of notation, we let $\mathbf{z} := \mathbf{x} \mid \mathbf{w}$, $\mathcal{Z} := \{(\mathbf{x} \mid \mathbf{w}) \mid \mathbf{x} \in \mathcal{X}, \mathbf{w} \in \mathcal{W}\}$, and let

$$f_{\mathsf{sk}_1}(\mathbf{x}) = \mathbf{1}^T \cdot \mathbf{T}_{\mathbf{x}} \cdot \mathbf{b} \in \mathbb{Z}_q, \qquad \mathsf{sk}_1 = \{\{\mathbf{T}_{i,b}\}, \mathbf{b}\}$$
$$g_{\mathsf{sk}_2}(\mathbf{z}) = \mathbf{1}^T \cdot \mathbf{S}_{\mathbf{z}|\mathbf{z}|\cdots|\mathbf{z}} \cdot \mathbf{a} \in \mathbb{Z}_q, \qquad \mathsf{sk}_2 = \{\{\mathbf{S}_{i,b}\}, \mathbf{a}\}$$

where sk_1 and sk_2 are independently generated according to the distribution in Construction 5.3.

Then, we can rewrite:

$$\{\mathbf{L}\mathbf{Q}_{\mathbf{z}|\mathbf{z}|\cdots|\mathbf{z}}\mathbf{R} + e_{\mathbf{z}}\}_{\mathbf{z}\in\mathbf{Z}} \tag{6}$$
$$= \{(\mathbf{u}^T\mathbf{M}_{\mathbf{z}|\mathbf{z}|\cdots|\mathbf{z}}\mathbf{v}) \cdot (\mathbf{1}^T\mathbf{S}_{\mathbf{z}|\mathbf{z}|\cdots|\mathbf{z}} \cdot \mathbf{a}) + \mathbf{1}^T\mathbf{T}_{\mathbf{x}} \cdot \mathbf{b} + e_{\mathbf{z}}\}_{\mathbf{z}\in\mathbf{Z}} \tag{7}$$
$$= \{\mathbf{1}^T\mathbf{T}_{\mathbf{x}} \cdot \mathbf{b} + e_{\mathbf{x}|\mathbf{w}}\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}:R(\mathbf{x},\mathbf{w})=1} \sqcup \{\mathbf{1}^T\mathbf{S}_{\mathbf{z}|\mathbf{z}|\cdots|\mathbf{z}} \cdot \mathbf{a} + \mathbf{1}^T\mathbf{T}_{\mathbf{x}} \cdot \mathbf{b} + e_{\mathbf{x}|\mathbf{w}}\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}:R(\mathbf{x},\mathbf{w})=0} \tag{8}$$
$$= \{f_{\mathsf{sk}_1}(\mathbf{x}) + e_{\mathbf{x}}\}_{\mathbf{x}\in L} \sqcup \{g_{\mathsf{sk}_2}(\mathbf{x},\mathbf{w}) + f_{\mathsf{sk}_1}(\mathbf{x}) + e_{\mathbf{x}|\mathbf{w}}\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}:R(\mathbf{x},\mathbf{w})=0}. \tag{9}$$

We obtain the first equality by definition of $\mathbf{L}, \mathbf{Q}_{i,b}, \mathbf{R}$. We obtain the second inequality using the assumption that $\mathbf{u}\mathbf{M}_{\mathbf{z}|\mathbf{z}|\cdots|\mathbf{z}}\mathbf{v} = 1 - R(\mathbf{z})$. Finally, we obtain the last equality by using the definition of f_{sk_1} and g_{sk_2}, and by using the fact that the language is in UP to re-index the first term in terms of only $\mathbf{x}$. We can also rewrite $\mathsf{aux} := \{\lfloor \mathbf{T}_{\mathbf{x}} \cdot \mathbf{B} \rceil_p\}_{\mathbf{x}\notin L} = \{\lfloor f_{\mathsf{sk}_1}(\mathbf{x}) \rceil_p\}_{\mathbf{x}\notin L}$.

Now, we proceed in hybrids. Note we allow the reductions below to run in time $\mathsf{poly}(\mu) \geq \mathsf{poly}(|\mathcal{X}|, |\mathcal{W}|, \lambda)$, and bound the distinguishing probability by $\mathsf{negl}(2^{-h^2\lambda})$.

- H_0: This is the real distribution of Construction 5.3. Sample $\mathsf{sk}_1, \mathsf{sk}_2$, and aux as in Construction 5.3, and output

  $$\{f_{\mathsf{sk}_1}(\mathbf{x}) + e_{\mathbf{x}}\}_{\mathbf{x}\in L} \sqcup \{g_{\mathsf{sk}_2}(\mathbf{x},\mathbf{w}) + f_{\mathsf{sk}_1}(\mathbf{x}) + e_{\mathbf{x}|\mathbf{w}}\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}:R(\mathbf{x},\mathbf{w})=0}, \mathsf{aux} = \{\lfloor f_{\mathsf{sk}_1}(\mathbf{x}) \rceil_p\}_{\mathbf{x}\notin L}\ .$$

- H_1: Output

  $$\{f_{\mathsf{sk}_1}(\mathbf{x}) + e_{\mathbf{x}}\}_{\mathbf{x}\in L} \sqcup \{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}:R(\mathbf{x},\mathbf{w})=0}, \mathsf{aux}\ .$$

  Indeed, letting $\{e'_{\mathbf{x}|\mathbf{w}} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma}\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}}$, we have

  $$\begin{aligned}
  &\{g_{\mathsf{sk}_2}(\mathbf{x},\mathbf{w}) + f_{\mathsf{sk}_1}(\mathbf{x}) + e_{\mathbf{x}|\mathbf{w}}\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}:R(\mathbf{x},\mathbf{w})=0} \\
  &\approx_s \{g_{\mathsf{sk}_2}(\mathbf{x},\mathbf{w}) + e'_{\mathbf{x}|\mathbf{w}} + f_{\mathsf{sk}_1}(\mathbf{x}) + e_{\mathbf{x}|\mathbf{w}}\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}:R(\mathbf{x},\mathbf{w})=0} \\
  &\approx_c \{\mathcal{U}(\mathbb{Z}_q) + f_{\mathsf{sk}_1}(\mathbf{x}) + e_{\mathbf{x}|\mathbf{w}}\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}:R(\mathbf{x},\mathbf{w})=0} \\
  &= \{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}:R(\mathbf{x},\mathbf{w})=0}\ .
  \end{aligned}$$

  Here, the first $\approx_s$ follows from a noise-flooding argument, just as in the proof of Lemma 3.8. The $\approx_c$ follows from BLMR PRF security (Lemma 3.8), and the $=$ is by a one-time pad argument.

- H_2: Replace aux as follows: output

  $$\{f_{\mathsf{sk}_1}(\mathbf{x}) + e_{\mathbf{x}}\}_{\mathbf{x}\in L} \sqcup \{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}:R(\mathbf{x},\mathbf{w})=0}, \mathsf{aux} := \{\lfloor f_{\mathsf{sk}_1}(\mathbf{x}) + \mathbf{e}'_{\mathbf{x}} \rceil_p\}_{\mathbf{x}\notin L},$$

  where $\mathbf{e}'_{\mathbf{x}} \leftarrow \mathcal{D}_{\mathbb{Z},\sigma'}$. This follows from Lemma 3.12.

- H_3: Replace $\{f_{\mathsf{sk}_1}(\mathbf{x}) + \mathbf{e}_{\mathbf{x}}\}_{\mathbf{x}\in L} \sqcup \{f_{\mathsf{sk}_1}(\mathbf{x}) + \mathbf{e}'_{\mathbf{x}}\}_{\mathbf{x}\notin L}$ with $\{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x}\in\mathcal{X}}$. That is, output

  $$\{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x}\in L} \sqcup \{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x}\in\mathcal{X},\mathbf{w}\in\mathcal{W}:R(\mathbf{x},\mathbf{w})=0}, \mathsf{aux} := \{\lfloor \mathcal{U}(\mathbb{Z}_q) \rceil_p\}_{\mathbf{x}\notin L},$$

This replacement follows from BLMR PRF security (Lemma 3.8).

- H_4: Replace $\mathsf{aux} = \{\lfloor \mathcal{U}(\mathbb{Z}_q) \rceil_p\}_{\mathbf{x} \notin L}$ with $\mathsf{aux}' = \{\mathcal{U}(\mathbb{Z}_p)\}_{\mathbf{x} \notin L}$. This step is statistical, and follows from Lemma 3.10.

- H_5: Replace aux' with $\{\lfloor f_{\mathsf{sk}_1}(\mathbf{x}) \rceil_p\}_{\mathbf{x} \notin L}$. That is, output

  $$\{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x} \in L} \sqcup \{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x} \in \mathcal{X}, \mathbf{w} \in \mathcal{W} : R(\mathbf{x},\mathbf{w})=0}, \mathsf{aux} := \{\lfloor f_{\mathsf{sk}_1}(\mathbf{x}) \rceil_p\}_{\mathbf{x} \notin L} .$$

  This step follows from rounded BLMR (Lemma 3.14).

H_5 is exactly the distribution on the RHS of (5). Therefore, this completes the proof. □

Theorem 5.5. *Let L be a* UP *language with* UP *relation R, and assume* $\mathsf{LWE}_{n,\mathsf{poly}(n),q,\sigma}$ *and* $\mathsf{eLWE}_{\mathsf{Samp}_{\mathcal{F},\mathsf{aux},j},\sigma',\sigma'}$ *for all $1 \le j \le h-1$, where* $\mathsf{aux}(k) := \{\lfloor \mathbf{T}_{\mathbf{x}} \cdot \mathbf{B} \rceil_p\}_{\mathbf{x} \notin L}$*, and all parameters and matrices are as in Construction 5.3.*

Then Construction 5.3 is an adaptively secure witness PRF for L.

Proof. We show that Construction 5.3 is correct and adaptively secure as in Definition 5.1.

Correctness. It suffices to check that, except with probability $\mathsf{negl}(\lambda)$, for all $\mathbf{x}, \mathbf{w}$ such that $R(\mathbf{x}, \mathbf{w}) = 1$, $\mathsf{Eval}(\mathsf{fk}, \mathbf{x}, \mathbf{w}) = \mathsf{F}(\mathsf{fk}, \mathbf{x})$; that is,

$$\lfloor P(\mathbf{x}, \mathbf{w}) \rceil_p = \lfloor \mathbf{1}^T \mathbf{T}_{\mathbf{x}} \cdot \mathbf{b} \rceil_p .$$

But by Lemma 3.6, except with probability $\mathsf{negl}(\mu)$, all matrices of ek have entries bounded by $B := \sigma\sqrt{n}$. Thus by GGH-encoding correctness (Theorem 4.9), for all $\mathbf{x}, \mathbf{w}$,

$$|\xi_{\mathbf{x},\mathbf{w}} := (P(\mathbf{x}, \mathbf{w}) - \mathbf{LQ}_{(\mathbf{x},\mathbf{w})|(\mathbf{x},\mathbf{w})|\cdots|(\mathbf{x},\mathbf{w})}\mathbf{R})| \le \xi ,$$

where $\xi = (4(\sigma\sqrt{n})(n + O(1)) \log q)^{h+1}$. And for $\mathbf{x}, \mathbf{w}$ such that $R(\mathbf{x}, \mathbf{w}) = 1$,

$$\mathbf{LQ}_{(\mathbf{x},\mathbf{w})|(\mathbf{x},\mathbf{w})|\cdots|(\mathbf{x},\mathbf{w})}\mathbf{R} = \mathbf{1}^T \cdot \mathbf{T}_{\mathbf{x}} \cdot \mathbf{b} ,$$

since $\mathbf{u}^T \mathbf{M}_{(\mathbf{x},\mathbf{w})|(\mathbf{x},\mathbf{w})|\cdots|(\mathbf{x},\mathbf{w})} \mathbf{v} = 0$. It remains to notice that $\xi/p = \mathsf{negl}(2^h \cdot \lambda)$, so by Lemma 3.12, except with probability $\mathsf{negl}(\lambda)$, for all $\mathbf{x}, \mathbf{w}$ such that $R(\mathbf{x}, \mathbf{w}) = 1$,

$$\lfloor \mathbf{1}^T \mathbf{T}_{\mathbf{x}} \cdot \mathbf{b} + \xi_{\mathbf{x},\mathbf{w}} \rceil_p = \lfloor \mathbf{1}^T \mathbf{T}_{\mathbf{x}} \cdot \mathbf{b} \rceil_p .$$

This completes the proof of witness PRF correctness.

Adaptive security. We proceed in a sequence of hybrids to show that $\mathtt{EXP}^R_{\mathcal{A}}(0, \lambda) \approx_c \mathtt{EXP}^R_{\mathcal{A}}(1, \lambda)$

- $\mathsf{H}_{0,b}$: The challenger interacts with the adversary $\mathcal{A}$ according to the original experiment

$\mathtt{EXP}^R_{\mathcal{A}}(b, \lambda)$.

- $\mathsf{H}_{1,b}$: This hybrid follows $\mathsf{H}_{0,b}$, with the following modification to the challenger. The challenger samples $(\mathsf{fk}, \mathsf{ek}) \leftarrow \mathsf{wPRF.Gen}(1^\lambda, R)$, and computes $\mathsf{aux} = \mathsf{aux}_{\mathsf{fk}} := \{\mathsf{F}(\mathsf{fk}, \mathbf{x})\}_{\mathbf{x} \notin L}$. Then, for every challenge query $\mathbf{x}$, the challenger first attempts to compute the witness $\mathbf{w}$ for $\mathbf{x}$ (by searching over all $\mathbf{w} \in \{0,1\}^\ell$). Then, the challenger outputs the following instead of $\mathsf{F}(\mathsf{fk}, \mathbf{x})$:
  - If the challenger finds a corresponding witness $\mathbf{w}$ for $\mathbf{x}$, it outputs $\mathsf{wPRF.Eval}(\mathsf{ek}, \mathbf{x}, \mathbf{w})$.
  - Otherwise, the challenger instead outputs $\mathsf{aux}(\mathbf{x})$. (Note that this is possible since $\mathbf{x} \notin L$.)

  By wPRF correctness, except with probability $\mathsf{negl}(\lambda)$, for all $\mathbf{x}, \mathbf{w}$ such that $R(\mathbf{x}, \mathbf{w}) = 1$, $\mathsf{wPRF.Eval}(\mathsf{ek}, \mathbf{x}, \mathbf{w}) = \mathsf{F}(\mathsf{fk}, \mathbf{x})$. On the other hand, if $\mathbf{x} \notin L$, then $\mathsf{aux}(\mathbf{x}) = \mathsf{F}(\mathsf{fk}, \mathbf{x})$ by construction of aux. Therefore, $\mathsf{H}_{1,b} \approx_s \mathsf{H}_{0,b}$.

- $\mathsf{H}_{2,b}$: Follows $\mathsf{H}_{1,b}$, but replaces ek with a value independent of fk, drawn from the distribution $\mathcal{D}$ guaranteed by Theorem 4.9. This follows from Theorem 4.9, which applies because of Lemma 5.4.

- $\mathsf{H}_{3,b}$: This hybrid is identical to $\mathsf{H}_{2,b}$, except that the challenger replaces aux with $\{\mathcal{U}(\mathbb{Z}_p)\}_{\mathbf{x} \notin L}$. This is justified by Lemma 3.14, now that fk is sampled independently of the rest of the adversary's view. Therefore, it follows that $\mathsf{H}_{3,b} \approx_c \mathsf{H}_{2,b}$.

But now $\mathsf{H}_{3,0} = \mathsf{H}_{3,1}$, since in both hybrids, challenge queries $\mathbf{x}$ for $\mathbf{x} \notin V$ (which satisfy $\mathbf{x} \notin L$) are answered with independent, uniformly random values. Indeed, if $b = 0$, the answer to query $\mathbf{x}$ is $\mathsf{aux}(\mathbf{x})$; if $b = 1$, the answer is $g(\mathbf{x})$, where g is the random function in the wPRF security game definition.

Therefore, putting together the hybrids, we have that

$$\mathtt{EXP}^R_{\mathcal{A}}(0, \lambda) \equiv \mathsf{H}_{0,0} \approx_s \mathsf{H}_{1,0} \approx_c \mathsf{H}_{2,0} \approx_c \mathsf{H}_{3,0} \equiv \mathsf{H}_{3,1} \approx_c \mathsf{H}_{2,1} \approx_s \mathsf{H}_{0,1} \equiv \mathtt{EXP}^R_{\mathcal{A}}(1, \lambda)$$

and hence, $\mathcal{A}$ cannot distinguish $\mathtt{EXP}^R_{\mathcal{A}}(0, \lambda)$ and $\mathtt{EXP}^R_{\mathcal{A}}(1, \lambda)$ with non-negligible probability. □

6 Designated Verifier zk-SNARGs from Witness PRFs

In this section, we show how to construct a designated verifier zero-knowledge SNARG generically from witness PRFs.

6.1 Construction

In this section, we follow the NIZK construction of Sahai and Waters [SW14] (which is also a SNARG construction), and demonstrate how one can use a witness PRF for a relation R to obtain a designated-verifier SNARG for relation R. As a corollary, we obtain a SNARG for UP relations.

Theorem 6.1. *Suppose there exists an adaptively secure witness PRF secure against* $\mathsf{poly}(|\mathcal{X}|, |\mathcal{W}|, \lambda)$ *adversaries for a* NP *language* L *with witness relation* R*. Then, for any* $\kappa = \kappa(\lambda)$ *with* $\kappa \leq (|x| + |w|)^{o(1)}$*, where* $|x|$ *and* $|w|$ *denote the bit-length of the statement and witness in the UP relation,*

there exists a reusable, adaptively secure designated verifier SNARG for R which has statistical adaptive multi-theorem zero-knowledge (see Definition 3.18) and is secure against $\mathsf{poly}(|x| + |w| + \lambda)$*-time adversaries, with:*

- *soundness error* $2^{-\kappa}$*,*
- *proof size* $|\pi| = \kappa + \omega(\log(|x| + |w| + \lambda))$*,*
- *and prover (and verifier) runtime* $\mathsf{poly}(|x| + |w| + \lambda)$*.*

Proof. We show that Algorithm 4 satisfies the properties of Definition 3.17.

Algorithm 4 SNARG for relation R given a witness PRF for R.

Fix a witness PRF family $(\mathsf{wPRF.Gen}, \mathsf{wPRF.F}, \mathsf{wPRF.Eval})$ with domain $\mathcal{X}$ and range $\{0,1\}^\ell$, where $\ell := \kappa + \omega(\log(|x| + |w| + \lambda))$.

- $\mathsf{SNARG.Gen}(1^\lambda)$:
  - Generate $\mathsf{fk}, \mathsf{ek} \leftarrow \mathsf{wPRF.Gen}(1^\lambda, R)$.
  - Set private state $\tau := \mathsf{fk}$, and $\mathsf{crs} := \mathsf{ek}$.
  - Output (crs, τ).
- $\mathsf{SNARG.Prove}(\mathsf{crs}, x, w)$: Output $\pi \leftarrow \mathsf{wPRF.Eval}(\mathsf{crs}, x, w)$.
- $\mathsf{SNARG.Verify}(\tau, x, \pi)$: Output 1 iff $\pi = \mathsf{wPRF.F}(\mathsf{fk}, x)$.

Succinctness. This is clearly satisfied for all $\kappa \leq (|x| + |w|)^{o(1)}$.

Correctness. Clearly satisfied since the witness PRF guarantees that

$$\mathsf{wPRF.Eval}(\mathsf{ek}, x, w) = \mathsf{wPRF.F}(\mathsf{fk}, x)$$

if $R(x, w) = 1$.

Reusable and adaptive soundness.

First, we make the following claim.

Claim 6.2. *No* $\mathsf{poly}(|\mathcal{X}| + |\mathcal{W}| + \lambda)$*-time adversary* $\mathcal{A}$*, given* crs *and access to an oracle* $\mathcal{O}_\tau(\cdot,\cdot)$ *for* $(\mathsf{crs}, \tau) \leftarrow \mathsf{SNARG.Gen}(1^\lambda)$*, can distinguish between the following two cases with advantage better than* $\mathsf{negl}(|\mathcal{X}|, |\mathcal{W}|, \lambda)$*.*

1. H_0*:* $\mathcal{O}_\tau = \mathsf{SNARG.Verify}_\tau(\cdot,\cdot)$
2. H_1*:* $\mathcal{O}_\tau$ *is an oracle* $\mathsf{SNARG.Verify}^*_\tau(\cdot,\cdot)$ *depending on a random function* $g : \mathcal{X} \to \{0,1\}^\ell$ *that behaves as follows. On input* (x, π)*, it (inefficiently) checks whether* $x \in L$*. If so, it accepts iff* $\pi = \mathsf{wPRF.F}(\mathsf{fk}, x)$ *(as in the first case). But if* $x \notin L$*, it accepts iff* $\pi = g(x)$*.*

Proof. Indeed, suppose such an adversary $\mathcal{A}$ exists. We show that there is an adversary $\mathcal{A}'$ running in $\mathsf{poly}(|\mathcal{X}|, |\mathcal{W}|, \lambda)$ time that breaks the adaptive security of the underlying wPRF. On input crs and given oracle access to $\mathsf{SNARG.Verify}_\tau(\cdot,\cdot)$ for $(\mathsf{crs}, \tau) \leftarrow \mathsf{SNARG.Gen}(1^\lambda)$, $\mathcal{A}'$ behaves as follows.

First, $\mathcal{A}'$ simply runs $\mathcal{A}$ on input crs, simulating its oracle queries (x, π) by (1) sending query $(\text{``}\texttt{Challenge}\text{''}, x)$ to the wPRF challenger, receiving as output value ρ, and (2) outputting 1 if $\rho = \pi$, and 0 otherwise. Then, $\mathcal{A}'$ outputs whatever $\mathcal{A}$ outputs.

It is easy to see that if the witness PRF challenger were acting according to $\texttt{EXP}^R_{\mathcal{A}}(0, \lambda)$, then the view of $\mathcal{A}$ is identical to interacting with $\mathsf{SNARG.Verify}_\tau(\cdot)$. But, if the witness PRF challenger were acting according to $\texttt{EXP}^R_{\mathcal{A}}(1, \lambda)$, then the view of $\mathcal{A}$ is identical to interacting with $\mathsf{SNARG.Verify}^*(\cdot)$. Therefore, if $\mathcal{A}$ can distinguish the two oracles, then $\mathcal{A}'$ violates the adaptive security of wPRF. □

Now we show that if there exists a cheating prover $\mathcal{P}^*$ violating adaptive soundness of the SNARG with probability $\epsilon(\lambda)$, then $\epsilon(\lambda) \leq 2^{-\kappa}$, as desired. Consider the following adversary $\mathcal{A}$ running in time $\mathsf{poly}(|\mathcal{W}| + |x| + \lambda)$, that attempts to distinguish the hybrids H_0 and H_1 in Claim 6.2.

- $\mathcal{A}$ obtains crs, and sends the crs to $\mathcal{P}^*$.
- For every query (x_i, π_i) from $\mathcal{P}$ to the verifier oracle:
  - Check if $\mathcal{O}_\tau(x_i, \pi_i) = 1$.
  - If yes, and $x_i \notin L$ (this takes up to $O(|\mathcal{W}|)$ time to decide), then stop the experiment and output 0.
- Output 1.

Note that if $\mathcal{A}$ was interacting with H_0, $\mathcal{P}^*$ is playing the real SNARG game. Therefore, $\Pr[\mathcal{D}^{\mathsf{H}_0}(\mathcal{A}) = 0] \geq \epsilon(\lambda)$.

On the other hand, if $\mathcal{A}$ were interacting with H_1, then $\mathcal{P}^*$ needs to guess a uniformly random value $g(x)$ for some $x \notin L$. Since $\mathcal{P}^*$ only makes $p(|x| + |w| + \lambda)$ queries for some polynomial p, the probability that it finds such an accepting proof (x_i, π_i) for some $x_i \notin L$ is at most $p(|x| + |w| + \lambda)/2^\ell$. Therefore, $\Pr[\mathcal{D}^{\mathsf{H}_1}(\mathcal{A}) = 0] \leq p(|x| + |w| + \lambda)/2^\ell$.

By Claim 6.2, we must have that

$$\mathsf{negl}(|\mathcal{X}|, |\mathcal{W}|, \lambda) = \Pr[\mathcal{D}^{\mathsf{H}_0}(\mathcal{A}) = 0] - \Pr[\mathcal{D}^{\mathsf{H}_1}(\mathcal{A}) = 0] \geq \epsilon(|x| + |w| + \lambda) - p(|x| + |w| + \lambda)/2^\ell.$$

Therefore, we have that

$$\epsilon(|x| + |w| + \lambda) \leq \mathsf{negl}[|\mathcal{X}|, |\mathcal{W}|, \lambda] + p(|x| + |w| + \lambda)/2^\ell \leq 2^{-(\kappa+1)} + 2^{-(\kappa+1)} \leq 2^{-\kappa}$$

as desired.

Statistical Adaptive Multi-Theorem Zero-Knowledge. It suffices to define p.p.t. $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ for Experiment 2.

- $\mathcal{S}_1(1^\lambda)$: Computes $\mathsf{fk}, \mathsf{ek} \leftarrow \mathsf{wPRF.Gen}(1^\lambda, R)$. Sets $\mathsf{crs} = \mathsf{ek}$, $\tau = \mathsf{fk}$ and $\mathsf{st} = \mathsf{fk}$.
- $\mathcal{S}_2(x, \mathsf{st})$: Output $\mathsf{wPRF.F}(\mathsf{st}, x) = \mathsf{wPRF.F}(\mathsf{fk}, x)$.

By statistical correctness of the wPRF, we have that $\mathsf{wPRF.Eval}(\mathsf{crs}, x, w) = \mathsf{wPRF.F}(\tau, x)$ with probability $1 - \mathsf{negl}(\lambda)$. Therefore, by a union bound, we have that $\mathsf{Prove}(\mathsf{crs}, x, w) = \mathcal{S}_2(\tau, x)$, and the views of the $\mathcal{A}$ in the real and ideal experiments are in fact statistically close. □

Therefore, using our witness PRF for UP from Construction 5.3, we have the following corollary.

Corollary 6.3. *Let L be a UP language with UP relation R. Assuming the LWE and evasive LWE assumptions made in Theorem 5.5, there exists a reusable designated-verifer SNARG for L.*

7 Adaptively Secure Designated Verifier SNARG for NP

In this section, we show that the Sahai-Waters SNARG [SW14] is reusably and adaptively sound *in the designated verifier setting* if we assume the sub-exponential hardness of the underlying primitives, namely indistinguishability obfuscation (iO) and puncturable PRFs. Our result is in fact more general: it shows how to "lift" the security of a class of "Sahai-Waters-like" SNARG constructions from non-adaptive to adaptive soundness in the designated verifier setting (see Section 2.2 and Remark 7.7).

Theorem 7.1. *Assuming the existence of subexponentially secure indistinguishability obfuscation and subexponentially hard one-way functions, there exists a reusably and adaptively sound designated verifier SNARG for NP. For any $\kappa \leq (|x| + |w|)^{o(1)}$, to achieve a soundness error of $2^{-\kappa}$, the proof size in our SNARG system is*

$$\kappa + \omega(\log(|x| + |w| + \lambda))$$

bits where $|x|$ and $|w|$ are the bit-length of the statement and witness in the NP relation, respectively, and λ is a security parameter. The common reference string in our system has bit-length $\mathsf{poly}(|x| + |w| + \lambda)$.

Proof. Let L be an NP language with witness relation $R_L : \mathcal{X} \times \mathcal{W} \to \{0,1\}$. We start by describing the Sahai-Waters construction of a $\mathsf{dvSNARG}$ in Algorithm 5.

Let $0 < \alpha < 1$ be some constant and let ρ be a security parameter for the underlying ingredients, namely:

- A 2^{ρ^α}-secure puncturable PRF family $\mathsf{PPRF} = (\mathsf{PPRF.Gen}, \mathsf{PPRF.Punc}, \mathsf{PRF.Eval})$ with domain $\mathcal{X}$ and range $\{0,1\}^\ell$ for some $\ell = \kappa + \omega(\log(|x| + |w| + \lambda))$ (see Definition 3.5).
- A 2^{ρ^α}-secure indistinguishability obfuscation scheme Obf (see Definition 3.26).

We choose ρ to be polynomial in $|x| + |w| + \lambda$. In particular, for some constant $\alpha' < \alpha$, we set $\rho \geq (|x| + |w| + \lambda)^{1/\alpha'}$ so that

$$2^{\rho^\alpha} \geq 2^{\rho^{\alpha'}} = 2^{|x|+|w|+\lambda}$$

In particular,

$$2^{\rho^\alpha}/|\overline{L}| \geq 2^\lambda \cdot 2^{|w|} \geq 2^\lambda \cdot 2^\ell \tag{10}$$

The last inequality is true since $\ell \leq |w|$.

At a high level, we use complexity leveraging to argue adaptive soundness, while carefully maintaining small proof size. First, we consider hybrids $\mathsf{H}_0, \mathsf{H}_1, \mathsf{H}_2$ that we show can only be distinguished with advantage $\mathsf{negl}(2^{\rho^\alpha})$, even by an adversary $\mathcal{A}$ running in time $\mathsf{poly}(2^{\rho^\alpha})$. Then, we show that if there exists a cheating prover P^* that is able to provide an accepting proof on some (adaptively chosen)

Algorithm 5 Sahai-Waters SNARG [SW14] in the designated verifier setting.

$\mathsf{SNARG.Gen}(1^\lambda)$:

- Sample a PRF key $k \leftarrow \mathsf{PPRF.Gen}(1^\rho)$.
- Construct the following circuit P_k:
  - **Input:** x, w
  - **Hardcoded value:** k
  - **Algorithm:** If $R(x, w) = 1$, output $\mathsf{PPRF.Eval}(k, x)$. Else, output $\perp$.
- Compute $\mathsf{crs} := \mathsf{Obf}(P_k)$, and set the verifier state $\tau := k$.
- Output (crs, τ).

$\mathsf{SNARG.Prove}(\mathsf{crs}, x, w)$: Output $\pi \leftarrow \mathsf{Obf.Eval}(\mathsf{crs}, (x, w))$.
$\mathsf{SNARG.Ver}(\tau, x, \pi)$: Accept if $\pi = \mathsf{PPRF.Eval}(\tau, x)$ and reject otherwise.

$x \notin L$ in time $\mathsf{poly}(\lambda)$, then $\mathcal{A}$ can use P^* as a subroutine to distinguish H_0 from H_2 with advantage that is non-negligible in 2^{ρ^α}, a contradiction. We then conclude that such a cheating prover P^* does not exist, completing the proof. The hybrids are as follows.

- H_0: Generate (crs, τ) as in Algorithm 5. Additionally, let x^* be a uniformly random NO instance, i.e. $x^* \leftarrow \overline{L}$. Note that this can be sampled in time $2^{|x|+|w|} \leq 2^{\rho^\alpha}$.

  A distinguisher is given x^*, crs, and oracle access to the $\mathsf{Ver}_\tau(\cdot, \cdot) := \mathsf{SNARG.Ver}(\tau, \cdot, \cdot)$ oracle, but is not given τ.

- H_1: Identical to H_0, except that instead of being generated honestly, crs is generated as the obfuscation of a program punctured on x^*. Specifically:
  - Compute $k \leftarrow \mathsf{PPRF.Gen}(1^\rho)$ and $k\{x^*\} \leftarrow \mathsf{PPRF.Punc}(k, x^*)$.
  - Construct the following circuit $P_{k\{x^*\}}$:
    - **Input:** x, w
    - **Hardcoded value:** $k\{x^*\}$
    - **Algorithm:** If $R(x, w) = 1$, output $\mathsf{PPRF.Eval}(k\{x^*\}, x)$. Else, output $\perp$.
  - Compute $\mathsf{crs} := \mathsf{Obf}(P_{k\{x^*\}})$ and $\tau = k$.

  As before, a distinguisher is given x^*, crs, and oracle access to $\mathsf{Ver}_\tau(\cdot, \cdot)$.

- H_2 : Identical to H_1, except that the $\mathsf{Ver}_\tau(x, \pi)$ oracle is modified into an oracle $\mathsf{Ver}_\tau^{x^*, r}(x, \pi)$ where $r \leftarrow \{0,1\}^\ell$. The modified oracle works as follows: on input (x^*, π), it accepts if $\pi = r$, instead of accepting if $\pi = \mathsf{PPRF.Eval}(k, x^*)$. On all other x, it behaves exactly as before.

  As before, a distinguisher is given x^*, crs, and oracle access to the modified oracle.

Claim 7.2. *Assuming the 2^{ρ^α}-security of Obf, no $\mathsf{poly}(2^{\rho^\alpha})$-time distinguisher has advantage better than $2^{-\rho^\alpha}$ in distinguishing H_0 from H_1.*

Proof. Since $x^* \notin L$, that is, there is no w such that $R(x^*, w) = 1$, P_k and $P_{k\{x^*\}}$ have identical functionalities. Therefore, by invoking iO security, H_0 and H_1 are indistinguishable, preserving the distinguishing advantage. □

Claim 7.3. *Assuming the 2^{ρ^α}-security of* PPRF*, no* $\mathsf{poly}(2^{\rho^\alpha})$*-time distinguisher has advantage better than $2^{-\rho^\alpha}$ in distinguishing H_1 from H_2.*

Proof. Hybrids H_1 (resp. H_2) can be simulated given x^*, $k\{x^*\}$ and $\mathsf{PPRF.Eval}(k, x^*)$ (resp. a uniformly random r). By the punctured-key security of PPRF,

$$(x^*, k\{x^*\}, \mathsf{PPRF.Eval}(k, x^*)) \approx_c (x^*, k\{x^*\}, r).$$

Hence, $\mathsf{H}_2 \approx_c \mathsf{H}_1$. Hence, H_1 and H_2 are computationally indistinguishable, preserving the distinguishing advantage. □

Now, suppose for contradiction that there exists a cheating (adaptive) prover P^* in the SNARG game that runs in time $p(\rho)$ and, with probability $\epsilon(\rho)$, outputs a pair (x', π') such that $\mathsf{Ver}_\tau(x', \pi') = 1$. We will show that this implies a $\mathsf{poly}\big(2^{\rho^\alpha}\big)$-time distinguisher $\mathcal{D}$ that uses P^* as a subroutine to distinguish H_0 from H_2 with advantage $1/\mathsf{poly}\big(2^{\rho^\alpha}\big)$, contradicting the combination of Claims 7.2 and 7.3.

The distinguisher $\mathcal{D}$, on input crs and oracle access to $\mathsf{Ver}_\tau(\cdot,\cdot)$, simply feeds crs to P^* and simulates its oracle queries using $\mathsf{Ver}_\tau(\cdot,\cdot)$. If P^* succeeds in outputting some (x', π'), where $x' = x^*$ and $\mathsf{Ver}_\tau(x', \pi') = 1$, the distinguisher outputs 1; otherwise it outputs 0.

We start with the following claim.

Claim 7.4. *Let Q denote the event that P^* queries the $\mathsf{Ver}_\tau(\cdot,\cdot)$ oracle on $(x^*, \cdot)$ in H_2. Then*

$$\Pr[Q] \leq \frac{p(\rho)}{|\overline{L}|} + \mathsf{negl}(2^{\rho^\alpha}) \ .$$

Proof. In H_0, the claim is true with probability at most $\frac{p(\rho)}{|\overline{L}|}$ since P^* makes at most $p(\rho)$ queries, and x^* is chosen independently from P^*'s input. Thus, by a union bound, the probability that one of its queries hits x^* is at most $\frac{p(\rho)}{|\overline{L}|}$. Since H_0 and H_2 are indistinguishable with advantage better than $\mathsf{negl}(2^{\rho^\alpha})$, it follows that the probability that P^* queries x^* in H_2 is

$$\Pr[Q] \leq \frac{p(\rho)}{|\overline{L}|} + \mathsf{negl}(2^{\rho^\alpha}) \ .$$

□

Claim 7.5. *In H_2, the probability that a 2^{ρ^α}-time distinguisher outputs 1 is*

$$\Pr[\mathcal{D}^{\mathsf{H}_2}(1^\rho) = 1] \leq \frac{2 \cdot (p(\rho))^2}{2^\ell \cdot |\overline{L}|} \ .$$

Proof. In H_2, the distinguisher outputs 1 only when P^* outputs (x^*, r). We will, without loss of generality, assume that if P^* outputs (x^*, r), it has queried the $\mathsf{Ver}_\tau(\cdot,\cdot)$ oracle on (x^*, r).

$$\begin{aligned}\Pr[P^* \text{ outputs } (x^*, r)] &= \Pr[P^* \text{ outputs } (x^*, r)|Q]\Pr[Q] + \Pr[P^* \text{ outputs } (x^*, r)|\overline{Q}]\Pr[\overline{Q}] \\ &\leq \frac{p(\rho)}{2^\ell} \cdot \Pr[Q]\end{aligned}$$

since the second probability term is 0 by our assumption, i.e. in $\overline{Q}$, no query of P^* to the Ver_τ oracle begins with x^*, and therefore with probability 1, P^* does not output (x^*, r).

By Claim 7.4, we have

$$\begin{aligned}\Pr[P^* \text{ outputs } (x^*, r)] &\leq \frac{p(\rho)}{2^\ell} \cdot \Pr[Q] \\ &\leq \frac{p(\rho)}{2^\ell} \cdot \left(\frac{p(\rho)}{|\overline{L}|} + \mathsf{negl}(2^{\rho^\alpha})\right) \\ &\leq \frac{2 \cdot (p(\rho))^2}{2^\ell \cdot |\overline{L}|}\end{aligned}$$

where the last inequality is because $2^{\rho^\alpha} \geq |\overline{L}|$. □

Claim 7.6. *In H_0, the probability that the 2^{ρ^α}-time distinguisher outputs 1 is*

$$\Pr[\mathcal{D}^{\mathsf{H}_0}(1^\rho) = 1] = \frac{\epsilon(\rho)}{|\overline{L}|}.$$

Proof. In H_0, P^* is playing the real SNARG game. Thus by assumption, the probability that P^* succeeds in producing some (x', π') such that $\mathsf{Ver}_\tau(x', \pi') = 1$ is $\epsilon(\rho)$. Additionally, in H_0, x^* is sampled independently from the view of P^*. Therefore, conditioned on the prover's success, $\Pr_{x^*}[x^* = x'] = 1/|\overline{L}|$. It follows that the distinguisher outputs 1 with probability $\epsilon(\rho)/|\overline{L}|$. □

Since H_0 and H_2 are $2 \cdot 2^{-\rho^\alpha}$-indistinguishable by 2^{ρ^α}-time distinguishers, we get that

$$\frac{\epsilon(\rho)}{\overline{L}} - \frac{2 \cdot (p(\rho))^2}{2^\ell \cdot |\overline{L}|} \leq 2 \cdot 2^{-\rho^\alpha}$$

and therefore

$$\epsilon(\rho) \leq \frac{2 \cdot (p(\rho))^2}{2^\ell} + 2 \cdot 2^{-\rho^\alpha} \cdot |\overline{L}|$$

Since $|\overline{L}|/2^{\rho^\alpha} \leq 1/(2^\rho \cdot 2^\ell)$ by equation 10,

$$\epsilon(\rho) \leq \frac{3 \cdot (p(\rho))^2}{2^\ell} \leq 2^{-\kappa}$$

since $\ell = \kappa + \omega(\log(|x| + |w| + \lambda)) \leq \kappa + \omega(\log \rho)$. □

Remark 7.7. Our lifting theorem (Theorem 7.1) is generic, and applies more broadly to a class of SNARGs which share certain structural properties with the Sahai-Waters construction, including our

SNARG for UP construction in Section 6 (although, we do directly show the adaptive soundness of our construction from evasive LWE). For example, a sub-exponentially secure *non-adaptive* witness PRF (i.e. the adversary non-adaptively chooses a single challenge query before the public parameters of the witness PRF are chosen in Definition 5.1) would have also been sufficient to achieve an adaptively secure SNARG. See Section 2.2 for a more detailed discussion.

Remark 7.8. We show that careful complexity leveraging generically gives us an adaptively sound *reusable designated verifier* SNARG for NP. A recent, and concurrent, work of Waters and Wu [WW24] constructs an adaptively sound publicly verifiable SNARG for NP via a direct, white-box, construction from subexponentially secure indistinguishability obfuscation, subexponential one-way functions as well as the polynomial hardness of discrete log or factoring. The comparison is two-fold: they require a specific number-theoretic assumption while our lifting theorem is generic; and their construction is white-box and relies on IO, whereas our lifting theorem applies to *any* designated verifier SNARG. On the other hand, their construction achieves public verifiability. However, we note that the Gentry-Wichs barrier [GW11] on adaptive security seems to apply just as well to the designated verifier setting with a long CRS as it does to publicly verifiable SNARGs [CGKS23].

Remark 7.9. The above discussion makes one wonder if there could exist a lifting theorem of our type for *publicly verifiable* SNARGs. A generic complexity leveraging argument as we do, does not seem to apply in the public verification setting. However, the problem remains intriguingly open.

8 SNARK for UP

In this section, we show that any *adaptively sound* SNARG for UP can be generically transformed into one with black-box knowledge soundness. Our transformation preserves public verifiability and zero knowledge. We use a fully homomorphic encryption scheme (FHE) as well as an injective public-key encryption scheme. Both primitives can be instantiated from the learning with errors (LWE) assumption (see Definition 3.21 and Definition 3.24).

Theorem 8.1. *Assume the existence of a leveled fully homomorphic encryption scheme and an injective public-key encryption scheme (see Definition 3.24). If there exists a reusably and adaptively sound designated-verifier* SNARG *system for* UP*, then there exists a reusably and adaptively sound designated-verifier* SNARK *for* UP *with a non-adaptive black-box knowledge extractor.*

If the underlying SNARG *is publicly verifiable, the resulting* SNARK *is also publicly verifiable. Additionally, if the underlying* SNARG *is a* zk-SNARG*, the resulting* SNARK *is also a* zk-SNARK*.*

Plugging in the zk-SNARG system we constructed in Section 6, and noting that LWE implies a leveled fully homomorphic encryption scheme and an injective public-key encryption scheme (Theorem 3.22 and Theorem 3.25), we get the following corollary.

Corollary 8.2. *Let L be a* UP *language with* UP *relation R. Assuming the LWE and evasive LWE assumptions of Theorem 5.5, there exists a reusably and adaptively sound designated-verifier* zk-SNARK *for L with a non-adaptive black-box knowledge extractor.*

Furthermore, applying both Theorem 7.1 and Theorem 8.1 to the Sahai-Waters SNARG [SW14], we get the same primitive, but from indistinguishability obfuscation instead of evasive LWE.

Corollary 8.3. *Let L be a* UP *language with* UP *relation R. Assuming subexponentially-secure indistinguishability obfuscation, subexponentially-secure one-way functions, and* $\mathsf{LWE}_{n,\mathsf{poly}(n),q,\sigma}$ *for $n = \mathsf{poly}(\lambda)$, $\sigma \geq \sqrt{2n}$, and $q = 2^{n^\varepsilon}$ for some constant $\varepsilon > 0$, there exists a reusably and adaptively sound designated-verifier* zk-SNARK *for L with a non-adaptive black-box knowledge extractor.*

Remark 8.4 (On the work of [CGKS23])**.** Our transformation is inspired by the recent work of Campanelli, Ganesh, Khoshakhlagh and Siim [CGKS23]. However, there are two issues with their claim. First, they claim that their transformation converts a *non-adaptive* SNARG (for UP) into a SNARK. To the best of our knowledge, their transformation as-is seems to require *adaptive soundness* of the underlying SNARG. We elaborate on this in Remark 8.7 after we describe the construction. Secondly, they claim that their construction preserves zero-knowledge, but to the best of our understanding, this is not the case. In our construction, we preserve zero-knowledge by adding a layer of encryption under a public-key encryption scheme. We have contacted the authors and they agree with both of the above points. Coming up with a transformation from *non-adaptive* SNARG for UP/NP to a non-adaptively extractable SNARK for UP is an interesting open problem.

Remark 8.5. One could ask if a similar compiler can be shown for NP instead of merely UP. It turns out that black-box extractable SNARKs for NP in the plain model do not exist [CGKS23, Kal23]; therefore, the restriction to UP is, in a sense, necessary.

We describe our transformation from a SNARG for UP to a SNARK for UP in Section 8.1 and prove that it indeed satisfies the desired properties in Section 8.2.

8.1 Our Transformation from SNARG for UP to SNARK for UP

Our construction builds on [CGKS23] and uses the following building blocks:

- An reusably and *adaptively secure* dvSNARG scheme $\Pi = (\Pi.\mathsf{Gen}, \Pi.\mathsf{Prove}, \Pi.\mathsf{Ver})$ for UP (e.g., Corollary 6.3).
- A leveled fully homomorphic encryption scheme $\mathsf{FHE} = (\mathsf{FHE.Gen}, \mathsf{FHE.Enc}, \mathsf{FHE.Dec}, \mathsf{FHE.Eval})$ such that $\mathsf{FHE.Eval}_{\mathsf{ek}}$ is deterministic (e.g., Theorem 3.22).
- An injective public key encryption scheme $\mathsf{PKE} = (\mathsf{PKE.Gen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ (see Definition 3.24 and Theorem 3.25). When encrypting a string s, we use $\mathsf{PKE.Enc}_{\mathsf{pk}}(s)$ as a shorthand to denote the bit by bit encryption of s.

Construction. Consider $\alpha = \mathsf{poly}(\lambda)$, and consider a UP relation R_α with witnesses of size at most $p(\alpha)$ for some polynomial p. For $w \in \{0,1\}^{p(\alpha)}$, define the function $C_w : [p(\alpha)] \to \{0,1\}$ with the following functionality:

$$C_w[i] = w[i]$$

It is easy to see that such a function can be represented by a circuit of size at most $d := d(\alpha)$, for some polynomial d that depends only on p.

We first define a new relation $R'_\alpha : \mathcal{X}' \times \mathcal{W}' \to \{0,1\}$ as follows:

$$\mathcal{X}' = \mathcal{X} \times \mathcal{C}$$
$$\mathcal{W}' = \mathcal{W} \times \mathcal{R}$$
$$R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}((x,\rho),(w,r)) = \begin{cases} 1 & \text{if } R_\alpha(x,w) = 1 \text{ and} \\ & \rho = \mathsf{PKE.Enc}_{\mathsf{pk}}(\mathsf{FHE.Eval}_{\mathsf{ek}}(C_w,\mathbf{c});r)^{10} \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

where $\mathcal{R}$ is the domain of the randomness used in $\mathsf{PKE.Enc}$ and $\mathcal{C}$ is the output domain of PKE.

Claim 8.6. *The relation $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})} : \mathcal{X}' \times \mathcal{W}' \to \{0,1\}$ (defined in (11)) is a UP relation.*

Proof. It is clear by construction that $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}$ is a polynomial time computable relation. This follows from the fact that R_α, $\mathsf{FHE.Eval}$ and $\mathsf{FHE.Rerand}$ are all polynomial time computable functions.

To show unique witnesses, fix any $(x,\rho) \in \mathcal{X}'$. We now have two cases to consider:

Case 1: There are no witnesses w such that $R_\alpha(x,w) = 1$.

In this case, there is clearly no (w,r) such that $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}((x,\rho),(w,r)) = 1$. Thus (x,ρ) has no witnesses.

Case 2: x has some (unique) witness w such that $R_\alpha(x,w) = 1$.

In this case, clearly all witnesses to (x,ρ) must be of the form $(w,\cdot)$. Moreover, for all r, $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}((x,\rho),(w,r)) = 1$ only if $\rho = \mathsf{PKE.Enc}_{\mathsf{pk}}(s;r)$, where $s := \mathsf{FHE.Eval}_{\mathsf{ek}}(C_w,\mathbf{c})$ is a deterministic function of w and $\mathbf{c}$. But since PKE is an injective encryption scheme, there can be at most one r such that $\mathsf{PKE.Enc}_{\mathsf{pk}}(s;r) = \rho$, so (x,ρ) has at most one witness.

Since every $(x,\rho) \in \mathcal{X}'$ has at most one witness in $\mathcal{W}'$, the relation is in fact a UP relation. □

Now, we describe our SNARK construction in Algorithm 6. Additionally, we present a trapdoor common reference string generation mode in Algorithm 7, which will be vital for our knowledge extractor.

Remark 8.7. Our transformation differs from the construction of [CGKS23] in three ways. Firstly, in the construction of [CGKS23], the homomorphically evaluated output of $\mathsf{FHE.Eval}_{\mathsf{ek}}(C_w,\mathbf{c})$ is not encrypted. However, encrypting under a public key encryption scheme PKE is necessary to argue the fact that the transformation in Algorithm 6 preserves zero-knowledge. Secondly, our transformation only relies on a SNARG for UP, rather than a SNARG for NP, to achieve a SNARK for UP. We are able to do this by choosing our public key encryption scheme PKE to be injective. Finally, we rely on an *adaptively*-secure SNARG system. This seems inherent (even in the construction of [CGKS23]), because even an honest prover needs the public parameters $\mathsf{pk}, \mathsf{ek}, \mathbf{c}$ (generated along with the crs for the underlying SNARG) in Algorithm 6 before being able to generate an instance for the underlying SNARG.

Algorithm 6 Construction of SNARK for UP relation R_α.

We first describe the SNARG scheme, which relies on a leveled fully homomorphic scheme FHE and an injective public key encryption scheme (Definition 3.24).

- $\mathsf{SNARK.Gen}(1^\lambda, R)$:
  - Compute $(\mathsf{sk}_{\mathsf{FHE}}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^d)$, where d was the bound on the size of the circuit C_w.
  - Compute $(\mathsf{sk}_{\mathsf{PKE}}, \mathsf{pk}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$.
  - Let $t = \lceil \log_2(p(\alpha)) \rceil$, and compute t ciphertexts $\mathsf{ct}_j = \mathsf{FHE.Enc}_{\mathsf{sk}_{\mathsf{FHE}}}(0)$ for $j \in [t]$. Let $\mathbf{c} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_t)$.
  - Compute $(\mathsf{crs}, \tau) \leftarrow \Pi.\mathsf{Gen}(1^\lambda, R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})})$, for relation $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}$ as defined in (11).
  - Set the new common reference string $\mathsf{crs}' = (\mathsf{crs}, \mathsf{ek}, \mathsf{pk}, \mathbf{c})$, and verifier state τ.[11]
  - Output (crs', τ).
- $\mathsf{SNARK.Prove}(\mathsf{crs}', x, w)$: If $R(x, w) = 0$, output $\perp$. Else,
  - Sample $r \leftarrow \mathcal{R}$.
  - Compute $\rho \leftarrow \mathsf{PKE.Enc}_{\mathsf{pk}}(\mathsf{FHE.Eval}_{\mathsf{ek}}(C_w, \mathbf{c}); r)$.
  - Compute $\pi \leftarrow \Pi.\mathsf{Prove}(\mathsf{crs}, (x, \rho), (w, r))$.
  - Output $x, \pi' := (\rho, \pi)$.
- $\mathsf{SNARK.Ver}(\tau, x, \pi' = (\rho, \pi))$: Output $\Pi.\mathsf{Ver}(\tau, (x, \rho), \pi)$.

Algorithm 7 Trapdoor common reference string generation for SNARK system in Algorithm 6.

We additionally describe a trapdoor crs generation mode, which we later use in our knowledge extractor construction in Algorithm 8. The trapdoor mode additionally takes as input i representing some index of the witness. All differences from $\mathsf{SNARK.Gen}$ are highlighted in blue.

- $\mathsf{TDGen}(1^\lambda, R, i \in [p(\alpha)])$:
  - Compute $(\mathsf{sk}_{\mathsf{FHE}}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^d)$.
  - Compute $(\mathsf{sk}_{\mathsf{PKE}}, \mathsf{pk}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$.
  - Let $t = \lceil \log_2(p(\alpha)) \rceil$. Let $i_1, i_2, \ldots, i_t$ be the bit decomposition of the index i. Compute t ciphertexts $\mathsf{ct}_j = \mathsf{FHE.Enc}_{\mathsf{sk}_{\mathsf{FHE}}}(i_j)$ for $j \in [t]$. Let $\mathbf{c} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_t)$.
  - Compute $(\mathsf{crs}, \tau) \leftarrow \Pi.\mathsf{Gen}(1^\lambda, R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})})$, for relation $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}$ as defined in (11).
  - Set the new common reference string $\mathsf{crs}' = (\mathsf{crs}, \mathsf{ek}, \mathsf{pk}, \mathbf{c})$, verifier state τ and trapdoor $\mathsf{td} = (\mathsf{sk}_{\mathsf{FHE}}, \mathsf{sk}_{\mathsf{PKE}})$.
  - Output $(\mathsf{crs}', \tau, \mathsf{td})$.

8.2 Proof of Theorem 8.1

We constructed such a SNARK from a SNARG in Algorithm 6. We first show succinctness and soundness, and defer the proof of knowledge soundness to Lemma 8.8 and proof of adaptive multi-theorem zero-knowledge to Lemma 8.11.

Succinctness. Note that the new proof consists of a PKE encryption of a FHE ciphertext of a single bit, and proof produced by $\mathsf{SNARG.Prove}(\cdot)$.

Consider the following parameters.

- By the compactness of FHE, we know that the size of the ciphertext is bounded by $\ell(\lambda)$ for some polynomial ℓ *independent* of the size of C_w (and hence, the size of w).
- The PKE scheme has ciphertexts (encrypting a single bit) of size $p(\lambda)$, and uses $r(\lambda)$ of randomness to encrypt a bit, where p and r fixed polynomials.
- Let $f(|x|, |w|, \lambda)$ be the proof size of the underlying SNARG scheme for $\mathsf{negl}(\lambda)$ soundness.

Then, the new proof consists of a PKE encryption of an FHE ciphertext, which has size $\ell(\lambda) \cdot p(\lambda)$. Also recall that for relation $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}((x,\rho),(w,r))$, $|\rho| \leq \ell(\lambda) \cdot p(\lambda) \leq \mathsf{poly}(\lambda)$ and $|r| \leq \ell(\lambda) \cdot r(\lambda) \leq \mathsf{poly}(\lambda)$, where both are bounded by fixed polynomials independent of x and w. Therefore, the proof π for the relation $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}((x,\rho),(w,r))$, we note that has size $f(|x|+|\rho|, |w|+|r|, \lambda) = f(|x|+\mathsf{poly}(\lambda), |w|+\mathsf{poly}(\lambda))$. Therefore, if $f(|x|, |w|, \lambda) = \mathsf{poly}(\lambda) \cdot (|x|+|w|)^{o(1)}$, then we have that the proof length is:

$$\begin{aligned}
|\rho| + |\pi| &= \ell(\lambda) \cdot p(\lambda) + f(|x| + \mathsf{poly}(\lambda), |w| + \mathsf{poly}(\lambda)) \\
&\leq \mathsf{poly}(\lambda) + \mathsf{poly}(\lambda)(|x| + |w| + \mathsf{poly}(\lambda))^{o(1)} \\
&\leq \mathsf{poly}(\lambda)(|x| + |w|)^{o(1)}
\end{aligned}$$

where all instances of $\mathsf{poly}(\lambda)$ denote polynomials independent of x and w. Therefore, the resulting proof is still succinct if the underlying SNARG is succinct.

Reusable and adaptive soundness. Suppose otherwise, and there exists a cheating prover $\mathcal{P}^*$ with $\epsilon(\lambda)$ advantage (greater than negligible in λ). By an averaging argument, there must exist some $(\mathsf{sk}_{\mathsf{FHE}}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^d)$, $(\mathsf{sk}_{\mathsf{PKE}}, \mathsf{pk}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$ and $\mathbf{c}$ such that such that P^* is able to cheat with probability $\epsilon(\lambda)$ with respect to a SNARK common reference string of the form $(\mathsf{crs}, \mathsf{ek}, \mathsf{pk}, \mathbf{c})$, where $\mathsf{crs} \leftarrow \mathsf{SNARG.Gen}(1^\lambda, R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})})$. Fix such a tuple $(\mathsf{ek}, \mathsf{pk}, \mathbf{c})$. Then, $\mathcal{P}^*$ produces (possibly adaptive) cheating proofs for the underlying SNARG for $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}$ with probability at least $\epsilon(\lambda)$, which contradicts the reusable, and adaptive soundness of the underlying SNARG.

Preserving public verifiability. Suppose that the underlying SNARG system Π is in fact publicly verifiable. In particular, this means that the verifier secret state τ generated in $\mathsf{SNARK.Gen}(1^\lambda)$ in Algorithm 6 in fact equal to the crs. Note that $\mathsf{crs}' = (\mathsf{crs}, \mathsf{pk}, \mathsf{ek}, \mathbf{c})$ in fact contains $\tau = \mathsf{crs}$. Therefore, the resulting SNARK is also publicly verifiable.

Lemma 8.8. *If* Π *is a reusable and adaptively secure* SNARG *scheme, then* SNARK *in Algorithm 6 is non-adaptively black-box knowledge sound.*

Proof. Suppose $\mathcal{P} = (\mathcal{P}_{inp}, \mathcal{P}_{chall})$ such that $\mathcal{P}$ able to produce a verifying proof for a non-adaptively chosen x with non-negligible probability $\frac{1}{h(\lambda)}$, where h is some polynomial. By the adaptive soundness of the underlying SNARG scheme, it must be the case that $x \in L$.

Claim 8.9. *If the* crs *is generated using* $\mathsf{TDGen}(1^\lambda, R, i)$ *(in Algorithm 7) instead of* $\mathsf{SNARK.Gen}(1^\lambda, R)$ *for an arbitrary* $i \in [p(\alpha)]$ *instead, then* $\mathcal{P}$ *still succeeds in producing a verifying proof for* x *with probability at least* $1/h(\lambda) - \mathsf{negl}(\lambda)$.

Proof. The only difference in the two modes of crs generations is that the encryption $\mathbf{c}$ generated in the crs is either an encryption of 0, or an encryption of i. Therefore, by IND-CPA security of FHE, the two modes are indistinguishable and hence the $\mathcal{P}^*$'s success probability differs by at most a negligible amount. □

With this observation in hand, we now present the black-box extractor in Algorithm 8.

Algorithm 8 Black-box extractor Ext for SNARK in Algorithm 6

Input: $(\mathsf{crs}, \tau, x, \pi)$ such that $\mathsf{SNARK.Ver}(x, \pi)$
Oracle access: $\mathcal{P}_{chall}(\mathsf{st}, \cdot)$ which is able to provide accepting proofs for x with probability $\frac{1}{h(\lambda)}$, for some polynomial h.
The algorithm:

- Initial an array W of length $p(\alpha)$ to all $\varnothing$.
- For $i \in \{1, \ldots, p(\alpha)\}$, repeat the following experiment at most $3h(\lambda) \cdot (\log_e p(\alpha) + \lambda)$ times:
  - Restart black-box access to $\mathcal{P}_{chall}$ from state st.
  - Compute $(\mathsf{crs}', \tau, \mathsf{td}) \leftarrow \mathsf{TDGen}(1^\lambda, R, i)$.
  - Send crs' to $\mathcal{P}_{chall}$:
    - Answer all $\mathsf{SNARK.Ver}_\tau(\cdot)$ queries from $\mathcal{P}_{chall}$ with secret state τ.
    - If any query is of the form $(x, (\rho, \pi))$ such that $\mathsf{SNARK.Ver}(\tau, x, (\rho, \pi))$ outputs 1, then do the following:
      - Unpack $\mathsf{td} = (\mathsf{sk}_{\mathsf{FHE}}, \mathsf{sk}_{\mathsf{PKE}})$.
      - Compute $b_i \leftarrow \mathsf{FHE.Dec}_{\mathsf{sk}_{\mathsf{FHE}}}(\mathsf{PKE.Dec}_{\mathsf{sk}_{\mathsf{PKE}}}(\rho))$.
      - Set $W[i] = b_i$.
- Check if $R(x, W) = 1$. If yes, output W. Else, output $\perp$.

Since $x \in L$, there exists some w such that $R_\alpha(x, w) = 1$.

Claim 8.10. *In each iteration of the for loop,* Ext *computes the* i*th bit of the* w *with probability at least* $1 - \frac{1}{2^\lambda p(\alpha)}$.

Proof. By Claim 8.9, $\mathcal{P}_{chall}(\mathsf{ct}, \cdot)$ succeeds in creating a verifying proof (ρ, π) for x with probability $\frac{1}{h(\lambda)} - \mathsf{negl}(\lambda) \geq \frac{1}{2h(\lambda)}$. By adaptive soundness of the underlying SNARG, $\mathcal{P}_{chall}$ creates accepting

proofs for (x, ρ) without a witness if $\mathsf{negl}(\lambda)$. Therefore, with probability $\frac{1}{2h(\lambda)} - \mathsf{negl}(\lambda) \geq \frac{1}{3h(\lambda)}$, it must be the case that (x, ρ) has a witness (w', r') under the corresponding relation $R_\alpha^{(\mathsf{c},\mathsf{ek},\mathsf{pk})}$. By construction, we additionally must have $R_\alpha(x, w') = 1$, and since R_α is a UP relation, $w' = w$. Additionally, by definition of R_α, we have that $\rho = \mathsf{PKE.Enc}_{\mathsf{pk}}(\mathsf{FHE.Eval}_{\mathsf{ek}}(C_w, \mathbf{c}); r)$. Therefore, by the correctness of the PKE scheme, and the correctness of FHE, we have that $W[i] = \mathsf{FHE.Dec}_{\mathsf{sk}_{\mathsf{FHE}}}(\mathsf{PKE.Dec}_{\mathsf{sk}_{\mathsf{PKE}}}(\rho)) = C_w(i) = w[i]$ as desired.

Since the experiment is repeated $3h(\lambda)(\log_e p(\alpha) + \lambda)$ times, we fail to compute the ith bit with probability at most

$$\left(1 - \frac{1}{3h(\lambda)}\right)^{3h(\lambda)(\log_e p(\alpha)+\lambda)} \leq \frac{1}{2^\lambda p(\alpha)}$$

□

By a union bound, the extractor Ext obtains all the bits of w correctly with probability at least $1 - 1/2^\lambda$. Therefore, if $\mathcal{P}^*$ is able to succeed with creating a proof for x with non-negligible probability, than Ext extracts an incorrect witness with probability at most $1/2^\lambda$.

$$\Pr\left[\begin{array}{c} \mathsf{Ver}(\tau, x, \pi) = 1 \\ \wedge\ R(x, w) \neq 1 \end{array} : \begin{array}{c} (x, \mathsf{st}) \leftarrow \mathcal{P}_{inp}(1^\lambda) \\ (\mathsf{crs}, \tau) \leftarrow \mathsf{Gen}(1^\lambda) \\ \pi \leftarrow \mathcal{P}_{chall}(\mathsf{st}, \mathsf{crs}) \\ w \leftarrow \mathsf{Ext}^{\mathcal{P}_{chall}(\mathsf{st},\cdot)}(\mathsf{crs}, \tau, x, \pi) \end{array}\right] \leq \epsilon(\lambda) \cdot 2^{-\lambda} = \mathsf{negl}(\lambda).$$

Hence, Algorithm 6 is in fact non-adaptively black-box knowledge sound. □

Lemma 8.11. *Suppose that the* SNARG Π *is adaptively multi-theorem zero-knowledge. Then,* SNARK *in Algorithm 6 is also (computationally) adaptively multi-theorem zero-knowledge, as in Definition 3.18.*

Proof. Suppose that the underlying SNARG is adaptively multi-theorem zero-knowledge. Then, the resulting scheme is computationally adaptively multi-theorem zero-knowledge. Let $\mathcal{S}^\Pi = (\mathcal{S}_1^\Pi, \mathcal{S}_2^\Pi)$ be the adaptive, multi-theorem p.p.t. simulators for SNARG scheme Π.

Now, we would like to construct simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ for SNARK in Algorithm 6.

- H_0 : The adversary $\mathcal{A}$ interacts with the real experiment $\mathtt{EXP}_{\mathcal{A}}^{\mathtt{Real}}(1^\lambda)$ as in Definition 3.18 with $\mathsf{SNARK.Gen}$ and $\mathsf{SNARK.Prove}$, and the true verifier secret τ.
- H_1 : In this hybrid, we replace both the calls to $\Pi.\mathsf{Gen}$ and $\Pi.\mathsf{Prove}$ in $\mathsf{SNARK.Gen}$ and $\mathsf{SNARK.Prove}$ respectively with the simulator calls to $\mathcal{S}^\Pi$. All changes are highlighted in blue. Concretely:
  - Replace $\mathsf{SNARK.Gen}(1^\lambda, R)$ with $\mathsf{SNARK.Gen}_1(1^\lambda, R)$ as follows:
    - Compute $(\mathsf{sk}_{\mathsf{FHE}}, \mathsf{ek}) \leftarrow \mathsf{FHE.Gen}(1^\lambda, 1^d)$.
    - Compute $(\mathsf{sk}_{\mathsf{PKE}}, \mathsf{pk}) \leftarrow \mathsf{PKE.Gen}(1^\lambda)$.
    - Let $t = \lceil \log_2(p(\alpha)) \rceil$, and compute t ciphertexts $\mathsf{ct}_j = \mathsf{FHE.Enc}_{\mathsf{sk}_{\mathsf{FHE}}}(0)$ for $j \in [t]$. Let $\mathbf{c} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_t)$.

* Compute $(\mathsf{crs}, \tau, \mathsf{st}) \leftarrow \mathcal{S}_1^{\Pi}(1^\lambda, R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})})$, for relation $R_\alpha^{(\mathbf{c},\mathsf{ek},\mathsf{pk})}$ as defined in (11).
* Set the new common reference string $\mathsf{crs}' = (\mathsf{crs}, \mathsf{ek}, \mathsf{pk}, \mathbf{c})$, verifier state τ.
* Output (crs', τ).

– Replace $\mathsf{SNARK.Prove}(\mathsf{crs}, x, w)$ with $\mathsf{SNARK.Prove}_1(\mathsf{crs}, x, w, \mathsf{st})$ as follows: No longer check $R(x, w) = 1$.
* Sample $r \leftarrow \mathcal{R}$.
* Compute $\rho \leftarrow \mathsf{PKE.Enc}_{\mathsf{pk}}(\mathsf{FHE.Eval}_{\mathsf{ek}}(C_w, \mathbf{c}); r)$.
* Compute $\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \mathsf{st}, (x, \rho))$.
* Output $x, \pi' := (\rho, \pi)$.

First, note that the experiment environment already verfies that $R(x, w) = 1$, so removing the check in $\mathsf{SNARK.Prove}$ does not change the view of the adversary. Additionally, we have that by the adaptive multi-theorem zero-knowledge property of the underlying SNARG scheme, we have $\mathsf{H}_0 \approx_c \mathsf{H}_1$ (note that this step can be replaced with a statistical indistinguishability if the underlying scheme has statistical zero-knowledge).

- H_2 : In this hybrid, we replace $\mathsf{SNARK.Prove}_1(\mathsf{crs}, (x, \rho), w)$ with the new prove program $\mathsf{SNARK.Prove}_2(\mathsf{crs}, x, \mathsf{st})$, with the changes highlighted in blue:

  – Sample $r \leftarrow \mathcal{R}$.
  – Compute $\rho \leftarrow \mathsf{PKE.Enc}_{\mathsf{pk}}(0; r)$.
  – Compute $\pi \leftarrow \mathcal{S}_2(\mathsf{crs}, \mathsf{st}, (x, \rho))$.
  – Output $x, \pi' := (\rho, \pi)$.

  Note that $\mathsf{H}_1 \approx_c \mathsf{H}_2$ by the semantic security of the PKE scheme (recall that the $\mathcal{A}$ interacting with this experiment does not see r) and the fact that $\mathcal{S}_1^{\Pi}$ and $\mathcal{S}_2^{\Pi}$ are p.p.t. algorithms.

Now, define $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ such that $\mathcal{S}_1 = \mathsf{SNARK.Gen}_1$ from H_1, and $\mathcal{S}_2 = \mathsf{SNARK.Prove}_2$ (recall that $\mathcal{S}_2$ crucially no longer depends on w) from H_2. Then, one can view H_2 as the ideal experiment $\mathtt{EXP}_{\mathcal{A}}^{\mathtt{Ideal}}(1^\lambda)$ in Definition 3.18 with simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$. Therefore, since $\mathtt{EXP}_{\mathcal{A}}^{\mathtt{Real}}(1^\lambda) \equiv \mathsf{H}_0 \approx_c \mathsf{H}_2 \equiv \mathtt{EXP}_{\mathcal{A}}^{\mathtt{Ideal}}(1^\lambda)$, we indeed have that the real and ideal views are indistinguishable, and the resulting SNARK is also (computationally) adaptively multi-theorem zero-knowledge. □

Acknowledgements. We thank Yael Kalai and Zhengzhong Jin for insightful discussions on SNARGs, and Rahul Ilango for helpful discussions on strong notions of obfuscation for PRFs. SM and VV were supported in part by DARPA under Agreement Number HR00112020023, NSF CNS-2154149, a Simons Investigator award, a Thornton Family Faculty Research Innovation Fellowship from MIT and a Simons Investigator Award. SM was also supported partially by Jane Street. SP was supported in part by the NSF under Grant No. CCF-2122230, and a generous gift from Google. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government or DARPA.

References

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *28th ACM STOC*, pages 99–108. ACM Press, May 1996. 17

[AP11] Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. *Theory of Computing Systems*, 48:535–553, 2011. 17

[Bar86] David A. Mix Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. In *18th ACM STOC*, pages 1–5. ACM Press, May 1986. 26

[BB04] Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, Heidelberg, May 2004. 12

[BCCT12] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In Shafi Goldwasser, editor, *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*, pages 326–349. ACM, 2012. 1

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 111–120. ACM, 2013. 1, 4

[BCG+13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza. Snarks for C: verifying program executions succinctly and in zero knowledge. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 90–108. Springer, 2013. 1

[BCG+14] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014*, pages 459–474. IEEE Computer Society, 2014. 1

[BCI+13] Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. Succinct non-interactive arguments via linear interactive proofs. In Amit Sahai, editor, *Theory of Cryptography - 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, volume 7785 of *Lecture Notes in Computer Science*, pages 315–333. Springer, 2013. 1

[BCTV14] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Succinct non-interactive zero knowledge for a von neumann architecture. In Kevin Fu and Jaeyeon Jung, editors, *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014*, pages 781–796. USENIX Association, 2014. 1

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, Heidelberg, August 2001. 25

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *ITCS 2012*, pages 309–325. ACM, January 2012. 24

[BLMR13] Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic PRFs and their applications. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 410–428. Springer, Heidelberg, August 2013. 8, 10, 19

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 719–737. Springer, Heidelberg, April 2012. 20

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In Rafail Ostrovsky, editor, *52nd FOCS*, pages 97–106. IEEE Computer Society Press, October 2011. 24

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 280–300. Springer, Heidelberg, December 2013. 6

[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 Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1082–1090. ACM, 2019. 1

[CGKS23] Matteo Campanelli, Chaya Ganesh, Hamidreza Khoshakhlagh, and Janno Siim. Impossibilities in succinct arguments: Black-box extraction and more. In *International Conference on Cryptology in Africa*, pages 465–489. Springer, 2023. 3, 4, 5, 14, 23, 47, 48, 49

[CJJ21] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Snargs for $\mathcal{P}$ from LWE. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 68–79. IEEE, 2021. 1

[CVW18] Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. GGH15 beyond permutation branching programs: Proofs, attacks, and candidates. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 577–607. Springer, Heidelberg, August 2018. 11, 17, 18, 19, 28, 29

[GGH15] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 498–527. Springer, Heidelberg, March 2015. 9, 27, 28

[Gro10] Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, volume 6477 of *Lecture Notes in Computer Science*, pages 321–340. Springer, 2010. 1

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Heidelberg, August 2013. 24

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 99–108. ACM, 2011. 1, 3, 12, 47

[HLL23] Yao-Ching Hsieh, Huijia Lin, and Ji Luo. Attribute-based encryption for circuits of unbounded depth from lattices. In *IEEE FOCS*, 2023. 2

[JJ22] Abhishek Jain and Zhengzhong Jin. Indistinguishability obfuscation via mathematical proofs of equivalence. In *63rd FOCS*, pages 1023–1034. IEEE Computer Society Press, October / November 2022. 1

[JKLV24] Zhengzhong Jin, Yael Tauman Kalai, Alex Lombardi, and Vinod Vaikuntanathan. SNARGs under LWE via propositional proofs, 2024. 1, 2

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 60–73. ACM, 2021. 1

[JLS22] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from LPN over Fp, DLIN, and PRGs in NC0. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part I*, volume 13275 of *Lecture Notes in Computer Science*, pages 670–699. Springer, 2022. 1

[Kal23] Yael Tauman Kalai. Personal communication, 2023. 4, 48

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *24th ACM STOC*, pages 723–732. ACM Press, May 1992. 1

[KPY19] Yael Tauman Kalai, Omer Paneth, and Lisa Yang. How to delegate computations publicly. In Moses Charikar and Edith Cohen, editors, *51st ACM STOC*, pages 1115–1124. ACM Press, June 2019. 1

[KRR13] Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. Delegation for bounded space. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 565–574. ACM Press, June 2013. 1, 2

[KRR14] Yael Tauman Kalai, Ran Raz, and Ron D. Rothblum. How to delegate computations: the power of no-signaling proofs. In David B. Shmoys, editor, *46th ACM STOC*, pages 485–494. ACM Press, May / June 2014. 1, 2

[Mic00] Silvio Micali. Computationally sound proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2000. 1

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 700–718. Springer, Heidelberg, April 2012. 17

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 333–342. ACM Press, May / June 2009.

[PHGR13] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*, pages 238–252. IEEE Computer Society, 2013. 1

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005. 25

[SD17] Noah Stephens-Davidowitz. *On the Gaussian Measure Over Lattices.* PhD thesis, New York University, USA, 2017. 19

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014. 1, 3, 5, 6, 12, 40, 43, 44, 47

[Tsa22] Rotem Tsabary. Candidate witness encryption from lattice techniques. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part I*, volume 13507 of *LNCS*, pages 535–559. Springer, Heidelberg, August 2022. 2, 8

[VWW22] Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. Witness encryption and null-IO from evasive LWE. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 195–221. Springer, Heidelberg, December 2022. 2, 8, 9, 11, 12, 17, 18, 28, 29, 31, 36

[Wee22] Hoeteck Wee. Optimal broadcast encryption and CP-ABE from evasive lattice assumptions. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 217–241. Springer, Heidelberg, May / June 2022. 2, 8

[WW24] Brent Waters and David J. Wu. Adaptively-sound succinct arguments for np from indistinguishability obfuscation. Cryptology ePrint Archive, Paper 2024/165, 2024. https://eprint.iacr.org/2024/165. 3, 4, 5, 12, 47

[WWW22] Brent Waters, Hoeteck Wee, and David J. Wu. Multi-authority ABE from lattices without random oracles. In Eike Kiltz and Vinod Vaikuntanathan, editors, *Theory of Cryptography - 20th International Conference, TCC 2022, Chicago, IL, USA, November 7-10, 2022, Proceedings, Part I*, volume 13747 of *Lecture Notes in Computer Science*, pages 651–679. Springer, 2022. 2

[Zha16] Mark Zhandry. How to avoid obfuscation using witness PRFs. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part II*, volume 9563 of *LNCS*, pages 421–448. Springer, Heidelberg, January 2016. 6, 35

A Deferred proofs

A.1 Proof of Lemma 3.8

Proof. It suffices to prove indistinguishability for a pair of modified distributions, identical to those in the theorem statement except that each discrete Gaussian sample $x \leftarrow \mathcal{D}_{\mathbb{Z},s}$ (where $s = \sigma$ or $s = \sigma'$) is replaced with a "clipped" sample $\hat{x}$ defined by $\hat{x} = x$ if $|x| < s\sqrt{n}$, and $\hat{x} = 0$ otherwise. This is justified by Lemma 3.6, which implies that for any $s > 0$, $\hat{\mathcal{D}}_{\mathbb{Z},s} \approx_s \mathcal{D}_{\mathbb{Z},s}$ with security parameter μ, where $\hat{\mathcal{D}}_{\mathbb{Z},s}$ is the distribution of samples clipped as above.

Given $n, h \in \mathbb{N}$, let $a(n,h)$ be the adversary's advantage in the modified distinguishing task. Clearly $a(n,0) = 0$, since $\mathbf{a} + \mathbf{e}_\epsilon = \mathcal{U}(\mathbb{Z}_q)$, where ϵ denotes the empty bitstring. It suffices to show the claim

$$a(n,h) \leq 2a(n,h-1) + \mathsf{poly}\Big(n \cdot 2^h\Big)/k + \mathsf{negl}(\mu) \ ,$$

because a simple induction on h then shows that $a(n,h) \leq h \cdot 2^h \cdot (\mathsf{poly}\big(n \cdot 2^h\big)/k) + \mathsf{negl}(\mu)$. This implies the result since we assumed that $h \leq n^\delta$, $f(n) \leq 2^{n^\delta}$, and $1/k(n) \leq \mathsf{negl}(n, 2^h, f(n))$.

For $b \in \{0,1\}$, let $\mathbf{a}_b \leftarrow \mathbb{Z}_q^n$ and $\mathbf{e}'_b \leftarrow \hat{\mathcal{D}}_{\mathbb{Z},\sigma}$, and for $\mathbf{x} \in \{0,1\}^h$, let $\mathbf{e}''_{\mathbf{x}} \leftarrow \hat{\mathcal{D}}_{\mathbb{Z},\sigma'}$. The claim follows immediately from the following hybrids, which all hold against $\mathsf{poly}(\mu)$-time adversaries with advantage bounds which will be specified below.

$$\begin{aligned}
&\left\{\left(\prod_{i=1}^{h+1} \mathbf{S}_{i,\mathbf{x}_i}\right) \cdot \mathbf{a} + \mathbf{e}_{\mathbf{x}}\right\}_{\mathbf{x}\in\{0,1\}^{h+1}}, && \{\mathbf{S}_{i,b} \leftarrow \hat{\mathcal{D}}_{\mathbb{Z},\sigma}^{n\times n}\}_{b\in\{0,1\},i\in[h+1]} \\
&\approx_s \left\{\left(\prod_{i=1}^{h} \mathbf{S}_{i,\mathbf{y}_i}\right) \cdot (\mathbf{S}_{h+1,b}\mathbf{a} + \mathbf{e}'_b) + \mathbf{e}''_{\mathbf{y},b}\right\}_{\mathbf{y}\in\{0,1\}^h, b\in\{0,1\}}, && \{\mathbf{S}_{i,b} \leftarrow \hat{\mathcal{D}}_{\mathbb{Z},\sigma}^{n\times n}\}_{b\in\{0,1\},i\in[h+1]} \\
&\overset{c}{\approx} \left\{\left(\prod_{i=1}^{h} \mathbf{S}_{i,\mathbf{y}_i}\right) \cdot \mathbf{a}_b + \mathbf{e}''_{\mathbf{y},b}\right\}_{\mathbf{y}\in\{0,1\}^h, b\in\{0,1\}}, && \{\mathbf{S}_{i,b} \leftarrow \hat{\mathcal{D}}_{\mathbb{Z},\sigma}^{n\times n}\}_{b\in\{0,1\},i\in[h+1]} \\
&\approx_c \{\mathcal{U}\}_{\mathbf{y}\in\{0,1\}^{h+1}, b\in\{0,1\}} && \{\mathbf{S}_{i,b} \leftarrow \hat{\mathcal{D}}_{\mathbb{Z},\sigma}^{n\times n}\}_{b\in\{0,1\},i\in[h+1]}
\end{aligned}$$

The $\approx_s$ holds with advantage bound $\mathsf{poly}\big(n \cdot 2^h/k\big)$ and follows from noise flooding. Specifically, since the entries of the $\mathbf{S}_{i,b}$ have absolute value at most $\sigma\sqrt{n}$, and except with probability $n \cdot 2^h \cdot 2^{-n} = \mathsf{negl}(\mu)$

this also holds for all $\mathbf{e}_{\mathbf{x}}$, we have

$$\sigma'' = \max_{i,\mathbf{y},b} \left\{ \left| \left(\left(\prod_{i=1}^{h} \mathbf{S}_{i,\mathbf{y}_i} \right) \mathbf{e}'_b \right)_i \right| \right\} \leq (\sigma n^2)^{h+1} .$$

Thus, for each $\mathbf{y}, b$, we can replace each entry of $\mathbf{e}_{\mathbf{y},b} - \left(\prod_{i=1}^{h} \mathbf{S}_{i,\mathbf{y}_i}\right) \mathbf{e}'_b$ with $\mathbf{e}''_{\mathbf{y},b}$, each replacement within statistical distance $\sigma''/\sigma' \leq 1/k$ by Lemma 3.7, adding up to a total statistical distance of $2^{h+1} \cdot n \cdot (2^{-n} + 1/k) = \mathsf{poly}\big(n, 2^h\big)/k + \mathsf{negl}(\mu) = \mathsf{poly}\big(n, 2^h\big)/k$.

The $\approx_c$ holds with advantage bound $\mathsf{negl}(\mu)$ and follows from Lemma 3.1 in dimension n, with secret $\mathbf{a}$ and public matrix $(\mathbf{S}^T_{h+1,0}, \mathbf{S}^T_{h+1,1})^T$. Finally, the $\approx_c$ holds with advantage bound $2a(n, h-1)$ and follows from inductive hypothesis. This concludes the proof of the claim and thus the proof of the lemma. □

A.2 Proof of Lemma 3.14

Proof. To simplify the notation, we will omit $\{\mathbf{S}_{i,b} \leftarrow \mathcal{D}^{n\times n}_{\mathbb{Z},\sigma}\}_{b\in\{0,1\},i\in[h]}$ from the indistinguishability claims in the proof. (That is, each claimed indistinguishability implicitly includes these matrices on both sides.)

First, from Lemma 3.8 we have

$$\left\{ \left(\prod_{i=1}^{h} \mathbf{S}_{i,\mathbf{x}_i} \right) \cdot \mathbf{a} + \mathbf{e}_{\mathbf{x}} \right\}_{\mathbf{x}\in\{0,1\}^h} \approx_c \{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x}\in\{0,1\}^h} .$$

where all $\mathsf{poly}(\mu)$-time distinguishers have advantage at most $\mathsf{negl}(f(n))$.

Thus we can apply Lemma 3.10 to the above values, with our p, $g = f$, and $B = \sigma'\sqrt{n}$, to obtain

$$\left\{ \left\lfloor \left(\prod_{i=1}^{h} \mathbf{S}_{i,\mathbf{x}_i} \right) \cdot \mathbf{a} \right\rceil_p \right\}_{\mathbf{x}\in\{0,1\}^h} \approx_c \{\mathcal{U}(\mathbb{Z}_q)\}_{\mathbf{x}\in\{0,1\}^h} ,$$

where all $\mathsf{poly}(\mu)$-time distinguishers have advantage at most $\mathsf{negl}(f'(n))$, for

$$f' = f + 2^h \cdot (2\sigma\sqrt{n}/p + (p/q)) + \mathsf{negl}(\lambda) = f + k + 2^h \cdot (p/q) + \mathsf{negl}(\lambda) = \mathsf{poly}(f) .$$

Combining these two indistinguishabilities gives the desired result. □

A.3 Proof of Lemma 4.7

Proof. The proof proceeds in two steps. First, we will prove that for all $d \in [h]$, for all $\mathbf{x} \in \{0,1\}^d$, we have

$$\mathbf{D}_{\mathbf{x}} = \mathbf{M}_{\mathbf{x}}\mathbf{A}_d + \sum_{j=1}^{d} \left(\prod_{i=1}^{j-1} \mathbf{M}_{i,x_i} \right) \cdot \mathbf{E}_{j,x_j} \cdot \left(\prod_{i=j}^{d-1} \mathbf{D}_{i,x_i} \right) , \tag{12}$$

where we define $\mathbf{A}_h := \mathbf{I}^{n_h \times n_h}$. Then, observing that for all $\mathbf{x} \in \{0,1\}^h$, $\mathbf{D}_\mathbf{x} - \mathbf{M}_\mathbf{x}$ is simply the sum in (12), we will finish the proof by showing that the (infinity) norm of this sum is at most the claimed bound. We prove (12) by induction on d. The base case $d = 1$ follows by definition; indeed, for all $x_1 \in \{0,1\}$, $\mathbf{D}_{1,x_1} := \mathbf{M}_{1,x_1}\mathbf{A}_1 + \mathbf{E}_{1,x_1}$. For the inductive step, observe that for all $\mathbf{x} \in \{0,1\}^d$,

$$\begin{aligned}
\mathbf{D}_\mathbf{x} &:= \left(\prod_{i=1}^{d-1} \mathbf{D}_{i,x_i}\right) \cdot \mathbf{D}_{d,x_d} \\
&= \left(\left(\prod_{i=1}^{d-1} \mathbf{M}_{i,x_i}\right) \cdot \mathbf{A}_{d-1} + \sum_{j=1}^{d-1} \left(\prod_{i=1}^{j-1} \mathbf{M}_{i,x_i}\right) \cdot \mathbf{E}_{j,x_j} \cdot \left(\prod_{i=j}^{d-2} \mathbf{D}_{i,x_i}\right)\right) \cdot \mathbf{A}_{d-1}^{-1}(\mathbf{M}_{d,x_d}\mathbf{A}_d + \mathbf{E}_{d,x_d}) \\
&= \mathbf{M}_\mathbf{x}\mathbf{A}_d + \left(\prod_{i=1}^{d} \mathbf{M}_{i,x_i}\right) \cdot \mathbf{E}_{d,x_d} + \left(\sum_{j=1}^{d-1} \left(\prod_{i=1}^{j-1} \mathbf{M}_{i,x_i}\right) \cdot \mathbf{E}_{j,x_j} \cdot \left(\prod_{i=j}^{d-1} \mathbf{D}_{i,x_i}\right)\right) \\
&= \mathbf{M}_\mathbf{x}\mathbf{A}_d + \sum_{j=1}^{d} \left(\prod_{i=1}^{j-1} \mathbf{M}_{i,x_i}\right) \cdot \mathbf{E}_{j,x_j} \cdot \left(\prod_{i=j}^{d-1} \mathbf{D}_{i,x_i}\right) \ ,
\end{aligned}$$

as desired. The second equality follows from the inductive hypothesis, and the others are just algebraic manipulations. We prove the norm bound via

$$\begin{aligned}
&\left\| \sum_{j=1}^{d} \left(\prod_{i=1}^{j-1} \mathbf{M}_{i,x_i}\right) \cdot \mathbf{E}_{j,x_j} \cdot \left(\prod_{i=j}^{d-1} \mathbf{D}_{i,x_i}\right) \right\|_\infty \\
&\leq d \cdot \max_{j \in [d]} \left(\prod_{i=1}^{j-1} \mathbf{M}_{i,x_i}\right) \cdot \mathbf{E}_{j,x_j} \cdot \left(\prod_{i=j}^{d-1} \mathbf{D}_{i,x_i}\right) \\
&\leq d \cdot \left(\prod_{i=1}^{h-1} m_i\right) \cdot B^h \ .
\end{aligned}$$

The first inequality is trivial, and the second repeatedly uses the simple fact that if $\|\mathbf{A} \in \mathbb{Z}_q^{x \times d}\|_\infty \leq a$ and $\|\mathbf{B} \in \mathbb{Z}_q^{d \times y}\|_\infty \leq b$, then $\|\mathbf{AB}\|_\infty \leq dab$, along with Lemma 3.6, which implies that except with probability $\mathsf{negl}(\mu)$, for all $i \in [h]$ and $b \in \{0,1\}$, $\|\mathbf{D}_{i,b}\|_\infty \leq \sigma\sqrt{n}$. □