Obfuscating Pseudorandom Functions is Post-Quantum Complete

Pedro Branco
Bocconi University

Abhishek Jain
NTT and JHU

Akshayaram Srinivasan
University of Toronto

Abstract

The last decade has seen remarkable success in designing and uncovering new applications of indistinguishability obfuscation (i$\mathcal{O}$). The main pressing question in this area is whether *post-quantum* i$\mathcal{O}$ exists. All current lattice-based candidates rely on new, non-standard assumptions, many of which are known to be broken.

To make systematic progress on this front, we investigate the following question: can general-purpose i$\mathcal{O}$ be reduced, assuming *only* learning with errors (LWE), to obfuscating a smaller class of functions? The specific class of functions we consider are *pseudorandom functions* (PRFs), which constitute a natural functionality of independent interest. We show the following results:

- We construct exponentially-efficient i$\mathcal{O}$ (xi$\mathcal{O}$) for general circuits based on LWE in the pseudorandom oracle model – a variant of the Random Oracle model (Jain et al., CRYPTO'23). Our construction requires the pseudorandom oracle model heuristic to hold for a *specific* pseudorandom function and we prove its security against classical adversaries.
- We construct (post-quantum) i$\mathcal{O}$ for general circuits in the standard model based on (post-quantum) sub-exponentially secure LWE and (post-quantum) sub-exponentially secure *average-case* i$\mathcal{O}$ – a natural notion of i$\mathcal{O}$ for pseudorandom functions that we define.

To obtain these results, we generalize the "encrypt-evaluate-decrypt" paradigm used in prior works by replacing the use of fully homomorphic encryption with succinct secure two-party computation where parties obtain additive output shares (Boyle et al., EUROCRYPT'25 and Abram et al., STOC'25).

1 Introduction

Indistinguishability Obfuscation (i$\mathcal{O}$) [BGI+01, GGH+13] guarantees that the obfuscations of any two functionally equivalent programs are computationally indistinguishable. Despite offering a seemingly weak security guarantee, i$\mathcal{O}$ has emerged as the ultimate cryptography enabler over the past decade. Furthermore, following a long sequence of works [GGH+13, BGK+14, BR14, PST14, AJ15, BV15, LPST16b, LPST16a, BMSZ16, MSZ16, GMM+16, Lin16, LV16, AS17, Lin17, LT17, CVW18, GJK18, JLMS19, Agr19, BIJ+20, AP20, BDGM20, BIJ+20, JLS21, GP21, WW21, BDGM22, JLS22], the question of whether i$\mathcal{O}$ exists was recently settled by the breakthrough work of Jain, Lin and Sahai [JLS21, JLS22] who constructed i$\mathcal{O}$ for all circuits from the sub-exponential hardness of three well-studied problems: Decision Linear assumption on bilinear maps, learning parity with noise (LPN) over large fields and pseudorandom generators in NC^0. The last of these assumptions can be replaced with a sparse variant of LPN over binary field [RVV24].

Post-quantum i$\mathcal{O}$? These schemes, however, are insecure against quantum adversaries. This is because of their reliance on group-based assumptions. To address this situation, an alternative line of work [CVW18, BDGM20, GP21, WW21, DQV+21, BDGM22] has investigated the design of i$\mathcal{O}$ from lattice problems.[1] The current situation, however, is somewhat dire: the assumptions underlying these candidates are known to be broken [HJL21, JLLS23, DJM+25, AMYY25, HJL25]. The only post-quantum construction that can be reduced to a simple-to-state assumption with no known counterexamples was given in a recent work of

[1] There are also candidates [CCMR24] based on alternative approaches that may not be susceptible to quantum attacks; however, their security is not well understood.

Hsieh et al. [HJL25]. This assumption, however, is new and non-standard, and hence necessitates further cryptanalytic research before we can gain greater confidence in it.

The holy grail, naturally, is to build i$\mathcal{O}$ from the learning with errors (LWE) [Reg05] assumption. Unfortunately, this goal remains elusive despite several attempts. The objective of this work is to make systematic progress on this challenge.

1.1 Our Results

We investigate whether the task of realizing general-purpose i$\mathcal{O}$ can be reduced, assuming *only* LWE, to obfuscating a smaller class of functions. The specific class of functions we consider are *pseudorandom functions* (PRFs). We believe this is a natural question because obfuscating PRFs is an important goal in itself, as evidenced by recent work [BDJ+25, AKY24b, AKY24a]. The main contribution of our work is to show that post-quantum i$\mathcal{O}$ for general circuits reduces to post-quantum i$\mathcal{O}$ for PRFs, as defined below.

To state our results, we first recall the notion of exponentially-efficient i$\mathcal{O}$ (xi$\mathcal{O}$) [LPST16a] – a variant of i$\mathcal{O}$ where both the obfuscator and the evaluator are allowed to be inefficient. That is, their running times could grow with the size of the truth table of the function. The key non-trivial efficiency requirement is that the size of the obfuscation should be *sublinear* in the size of the truth table of the function. It is known that assuming sub-exponentially secure LWE, (sub-exponential secure) xi$\mathcal{O}$ can be transformed to standard i$\mathcal{O}$.

xi$\mathcal{O}$ in Pseudorandom Oracle Model. The Pseudorandom Oracle Model (PROM) was introduced by [JLLW23] as a variant of the Random Oracle model that enables non-black-box access to the oracle functionality. At a high level, the pseudorandom oracle is instantiated with a pseudorandom function F. A key for the PRF is mapped to a random handle. Given the handle, anyone can make queries to the oracle to learn the output of the PRF F on arbitrary inputs of their choice. Additionally, by sampling this key, an honest party can cryptographically encode it in a manner that allows private computation of F (e.g., using fully homomorphic encryption). As demonstrated in [JLLW23], this non-black-box access to the function F can enable new applications that are not known in the plain random oracle model.

Our first result is a construction of xi$\mathcal{O}$ under LWE in the Pseudorandom Oracle model.

Theorem 1. *Assuming LWE, there exists xi$\mathcal{O}$ for general circuits in the pseudorandom oracle model.*

A few remarks are in order. First, we note that the above result requires the existence of PROM for a *specific* pseudorandom function. We leave open the question of achieving a similar result assuming PROM for *any* pseudorandom function. Second, we do not claim post-quantum security of this construction as we only prove security against classical adversaries in the PROM model. Specifically, we do not consider adversaries that can make superposition queries to the oracle. We again leave open the problem of analyzing our construction when the adversary makes such superposition queries. Finally, we note that the above result does not extend to i$\mathcal{O}$. Indeed, known transformation from xi$\mathcal{O}$ to i$\mathcal{O}$ requires non-black-box use of the obfuscator, which is not possible when the obfuscator is oracle-aided (as is the case in the above result).

Next, we investigate xi$\mathcal{O}$ in the *standard model*, where it can be upgraded to i$\mathcal{O}$.

i$\mathcal{O}$ in Standard Model. Our second result is a construction of xi$\mathcal{O}$ (and thus, i$\mathcal{O}$) in the standard model assuming LWE and *average-case i$\mathcal{O}$* (avi$\mathcal{O}$) – a natural notion of i$\mathcal{O}$ for pseudorandom functions that we define in this work. Average-case i$\mathcal{O}$ guarantees that if the truth table of a circuit C is pseudorandom even given some simulatable auxiliary input aux, i.e.,

$$(\{C(x)\}_x, \mathsf{aux}) \approx (U, \mathsf{Sim}(U))$$

where Sim is an efficient simulator and U is the uniform distribution over the truth table of C, then

$$(\mathsf{avi}\mathcal{O}(C), \mathsf{aux}) \approx (\mathsf{avi}\mathcal{O}(C'), \mathsf{aux})$$

where C' is a circuit functionally-equivalent to C.

It is easy to see that average-case i$\mathcal{O}$ is weaker than standard i$\mathcal{O}$. On the other hand, it is stronger than obfuscation for identical pseudorandom functions (iPRO) – a notion recently defined by [BDJ+25]. Briefly, iPRO is similar to average-case iO except that it does not allow simulation of the auxiliary input in the pre-condition; that is, its pre-condition requires that $(\{C(x)\}_x, \mathsf{aux}) \approx (U, \mathsf{aux})$.

Theorem 2. *Assuming sub-exponentially secure LWE and sub-exponentially secure average-case i$\mathcal{O}$, there exists an i$\mathcal{O}$ for general circuits.*

Our security reduction in the proof of the above theorem is straight-line and invokes the adversary only once. Thus, assuming post-quantum sub-exponential security of LWE and post-quantum sub-exponentially secure average-case i$\mathcal{O}$, we obtain post-quantum secure i$\mathcal{O}$.

Discussion. As established in prior work, the main challenge in realizing i$\mathcal{O}$ is achieving *output compression*. Despite many years of research, achieving non-trivial output compression solely from LWE has proven to be a formidable challenge. In fact, we do not know this even for the case of degree-two computations.

Let us elaborate. All prior work on post-quantum i$\mathcal{O}$ from lattice assumptions follows a similar "*encrypt-evaluate-decrypt*" template: we first encrypt the circuit C under fully homomorphic encryption (FHE) which allows anyone to obtain an encrypted truth table. To enable output evaluation in the clear, the obfuscator also releases a small decryption "hint". While the FHE secret key trivially constitutes such a hint, it immediately breaks security. The main challenge, therefore, is to design special-purpose hints that hide both the secret key and the encryption randomness. So far, we do not know how to implement this approach using standard assumptions.

In this work, we generalize the encrypt-evaluate-decrypt template by replacing the use of FHE with *succinct* secure two-party computation. Specifically, we consider protocols in the common reference string model where two parties – one holding a large input, and another with a short input – communicate by sending only one message to each other. At the end of the protocol, the parties obtain additive shares of the function output. Importantly, the protocol communication is *independent* of the (large) input and output length of the function. Such protocols were recently constructed based on LWE in [BJSS25, AMR25]. While such a protocol does not yield obfuscation on its own (as the output shares grow with the output size), our key insight is that the output evaluation step of one of the parties can be viewed as the "decrypt" step in the general template. Crucially, as we show in this work, this step can be implemented using an obfuscation of pseudorandom functions. This suffices to achieve output compression for general circuits.

Open Questions. The key challenge arising from our work is to obtain an instantiation of avi$\mathcal{O}$ from well-studied lattice-based assumptions. The recent works of Branco et al. [BDJ+24] and Agrawal et al. [AKY24b] provide a construction of iPRO from the private-coin evasive LWE assumption [Wee22, Tsa22]. However, it remains open to realize iPRO, which is weaker than avi$\mathcal{O}$, from weaker assumptions. On the flip side, it remains also open whether the existing construction of [BDJ+24] (or a variant thereof) can be proven to satisfy avi$\mathcal{O}$. It would also be interesting to explore a strengthening of Theorem 1 that requires PROM for an arbitrary PRF (as opposed to a specific PRF). Finally, obtaining a post-quantum version of Theorem 1 is also an interesting open problem.

1.2 Related Work

Applebaum [App14] showed how to bootstrap virtual black-box (VBB) obfuscation (resp. i$\mathcal{O}$) for the class of circuits containing pseudorandom functions (such as TC^0) into VBB obfuscation (resp. i$\mathcal{O}$) for all circuits. However, our result is different than [App14] as we start with a weaker notion of obfuscation. Specifically, our average-case i$\mathcal{O}$ security is required to hold only if the function table is pseudorandom. In contrast, Applebaum required full fledged i$\mathcal{O}$ security for TC^0 functions.

Obfuscation of pseudorandom functions. More recently, the work of Branco et al. [BDJ+25] introduced new notions of i$\mathcal{O}$ for pseudorandom functions. In particular, they define iPRO (which is weaker than

i$\mathcal{O}$) and show that it suffices to construct i$\mathcal{O}$ for all circuits by additionally assuming standard hardness assumptions on bilinear maps. While their notion of iPRO is weaker than our notion of average-case i$\mathcal{O}$ (see discussion in Section 2.2), they require bilinear maps (that are known to be post-quantum insecure) for bootstrapping to full fledged i$\mathcal{O}$.

Agrawal, Kumari, and Yamada [AKY24a] introduced the notion of pseudorandom functional encryption—a functional encryption scheme specifically for pseudorandom functionalities. In their work, they also present a candidate construction for this primitive. Moreover, it was shown in [AMYY25] that (a slight modification of) this construction yields an obfuscator based on a new, non-standard lattice-based assumption.

Pseudorandom Oracle Model. The pseudorandom oracle model (PROM) was introduced in [JLLW23] as an alternative to the random oracle model, enabling non-black-box use of the pseudorandom function implemented by the oracle. In the same work, it was shown that one can construct VBB obfuscation for all functions with polynomial security of single-key, succinct functional encryption. Currently, known constructions of i$\mathcal{O}$ in the standard model required sub-exponential security of such functional encryption schemes [BV15, AJ15]. Succinct functional encryption is currently only known to be realizable under the combination of assumptions used in [JLS21, JLS22, RVV24].

2 Technical Overview

In this section, we provide an overview of the techniques employed in our construction of indistinguishability obfuscation. In section 2.1, we demonstrate how to construct an exponentially-efficient i$\mathcal{O}$ (xi$\mathcal{O}$) based on the Learning with Errors (LWE) assumption in the Pseudorandom Oracle Model (PROM) [JLLW23]. In section 2.2, we define our average-case indistinguishability obfuscation notion and present a construction of i$\mathcal{O}$ in the standard model using the sub-exponential hardness of this primitive and the sub-exponential hardness of LWE.

2.1 xi$\mathcal{O}$ in PROM

Pseudorandom Oracle Model (PROM) was introduced by Jain et al. [JLLW23] as an idealized primitive to capture scenarios where a non-black-box use of a random oracle is necessary. Let us briefly recall the model.

The model is instantiated with a pseudorandom function F. There is an oracle $\mathcal{O}$ that supports two types of queries:

- $\mathcal{O}(\mathsf{Gen}, k)$ outputs a handle $h = \mathsf{HMap}(k)$ where HMap is a random permutation.
- $\mathcal{O}(\mathsf{Eval}, h, x)$ outputs the evaluation of the PRF $F(\mathsf{HMap}^{-1}(h), x)$.

Both the honest parties and the adversary can make queries to this oracle. The key property of this model is that it enables us to evaluate the pseudorandom function $F(k, \cdot)$ in two distinct ways: either by having access to the key k, or by querying the oracle on $\mathcal{O}(\mathsf{Eval}, h, \cdot)$ where $h = \mathcal{O}(\mathsf{Gen}, k)$. This flexibility allows us to make non-black-box use of the pseudorandom function F by, for instance, encrypting the key k under a fully homomorphic encryption (FHE) scheme and evaluating the function F under the hood of the FHE.

An adversary who gains access to a handle h that represents a key k can only query the oracle $\mathcal{O}$ using this handle. In such scenarios, the oracle $\mathcal{O}$ behaves as a random oracle as long as the key is randomly sampled. However, the non-triviality of this model arises from the fact that the adversary additionally obtains an encoding of the key k, in addition to the handle h. Consequently, proving pseudorandomness of the function F in the presence of this auxiliary information about the key k becomes a challenging task that necessitates a careful security argument.

In this work, we give a construction of xi$\mathcal{O}$ in the PROM model based on Learning with Errors assumption.

Starting Point. The starting point of our work is the recent paper of Boyle et al. [BJSS25] that introduced Simultaneous-Message and Succinct (SMS) secure computation protocols.

An SMS protocol is a two-party protocol between Alice and Bob in the CRS model. The CRS consists of a hashing key hk. Alice has a private input $\boldsymbol{x} \in \{0,1\}^n$ and Bob has a private input $\boldsymbol{y} \in \{0,1\}^N$. The common input to Alice and Bob is a description of a two-party function $f : \{0,1\}^n \times \{0,1\}^N \to \{0,1\}^m$. We usually consider the case where N and m are much larger than n. The protocol involves a single round of simultaneous message exchange. Specifically, Alice uses her input $\boldsymbol{x}$ and the hashing key hk to derive an encoding key ek along with a trapdoor td. Bob hashes the description of the function f and his input $\boldsymbol{y}$ using hk to obtain a hash value h. Alice sends ek to Bob and Bob sends the hash value h to Alice. Using the messages received from the other party and their secret state, Alice and Bob obtain an additive secret sharing of $f(\boldsymbol{x}, \boldsymbol{y})$. In a bit more detail, Alice applies a decoding function Dec on the hash h using her trapdoor td to derive a bit string $\boldsymbol{d} \in \{0,1\}^m$. Bob applies an encoding function Enc on ek with additional inputs f and $\boldsymbol{y}$ to obtain a bit string $\boldsymbol{e} \in \{0,1\}^m$. SMS correctness guarantees that $\boldsymbol{e} \oplus \boldsymbol{d} = f(\boldsymbol{x}, \boldsymbol{y})$. The security property requires that the encoding key ek provides no information about $\boldsymbol{x}$ (except its length) and the hash h provides no information about $\boldsymbol{y}$ (except its length).

The key efficiency property that SMS needs to satisfy is that the size of the encoding key ek is sub-linear in m and the size of the hash digest h is sub-linear in both N and m. In other words, the total size of the transcript is sub-linear in the size of the larger input N and the output m. Boyle et al.[BJSS25] gave a construction of SMS for computing polynomial sized circuits where the size of the encoding key and the hash value are bounded above by $m^{2/3} + \mathsf{poly} \log N$ (ignoring multiplicative $\mathsf{poly}(\lambda)$ factors). In a concurrent and independent work, Abram et al. [AMR25] gave a construction where their sizes are at most $\mathsf{poly} \log(N + m)$.

One could hope to construct an xi$\mathcal{O}$ scheme from an SMS protocol as follows. Let C be a circuit that maps $[N]$ to $\{0,1\}$. The input of Alice is the circuit C and the input of Bob is the public vector $(1, 2, \ldots, N)$. The function that is computed by the SMS protocol is the Universal circuit U that takes C from Alice and $(1, \ldots, N)$ from Bob and outputs $(C(1), \ldots, C(N))$. The obfuscator generates the ek of Alice and the hash value h of Bob. It then generates $\boldsymbol{d}$ using the hash h and Alice's trapdoor td and outputs $(\mathsf{ek}, \boldsymbol{d})$. To obtain the output, the evaluator computes $\boldsymbol{e}$ from ek and Bob's input $(1, \ldots, N)$ and computes $(C(1), \ldots, C(N)) = \boldsymbol{e} \oplus \boldsymbol{d}$. While this construction satisfies the correctness and security properties of xi$\mathcal{O}$, it fails to satisfy the required efficiency property. In particular, the size of $\boldsymbol{d}$ is as large as the size of the truth table of C. Hence, this construction does not perform any better than the trivial canonical obfuscation construction.

Main Insight. A key property of the SMS protocol is that there exists a sub-linear in N-sized circuit that takes h and td as input and computes the i-th bit of $\boldsymbol{d}$ for each $i \in [N]$. To be a bit more specific, the size of the trapdoor in the SMS protocol of Boyle et al. [BJSS25] is $N^{1/3} \cdot \mathsf{poly}(\lambda)$. We view the trapdoor as LWE secrets $(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{N^{1/3}})$. The size of h is $N^{2/3} \cdot \mathsf{poly}(\lambda)$. We view the hash as a sequence of vectors $(\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{N^{2/3}})$. To compute the i-th bit of $\boldsymbol{d}$ for some $i \in [N]$, we first view i as $(k, j) \in [N^{1/3}] \times [N^{2/3}]$. The i-th bit of $\boldsymbol{d}$ is computed as $\lceil \langle \boldsymbol{s}_k, \boldsymbol{h}_j \rangle \rfloor_2$ where $\lceil \cdot \rfloor_2$ is the rounding function that maps $\mathbb{Z}_p$ to $\{0,1\}$. Thus, the size of this circuit that computes the i-th bit of $\boldsymbol{d}$ is $N^{2/3} \cdot \mathsf{poly}(\lambda)$. This demonstrates that even though the size of $\boldsymbol{d}$ is linear in the truth table size of the circuit C, there exists a succinct circuit F that can be used to generate each bit of $\boldsymbol{d}$. Can we use this property to reduce the size of the obfuscated program?

Using Ideal Obfuscation. As a first step, we demonstrate that in an idealized model where the circuit F is provided as an oracle to the adversary, the proposed construction achieves i$\mathcal{O}$ security. In other words, we show that this achieves i$\mathcal{O}$ security when we provide an ideal obfuscation of F. Let C_0 and C_1 be two functionally equivalent programs. We want to show that obfuscation of C_0 is computationally indistinguishable to the obfuscation of C_1. To argue this, we first switch how the truth table of F is computed. Instead of computing it as $\boldsymbol{d}$, we compute it as $\boldsymbol{e} \oplus (C(1), \ldots, C(N))$. This change is indistinguishable to the adversary from the correctness of the SMS protocol. Now, since the trapdoor of the SMS protocol is not used anywhere, we can rely on its security to switch ek from encoding the circuit C_0 to encoding the circuit C_1. This change is indistinguishable from the security of the SMS protocol and the fact that C_0 and C_1 are functionally equivalent. We can revert the change made initially and switch the truth table of F back to $\boldsymbol{d}$.

Note that the final distribution is identical to an obfuscation of C_1.

Using PROM instead of Ideal Obfuscation. Although we used ideal obfuscation for obfuscating "small" circuits in the above construction, it is a significantly strong assumption. While we are far from constructing obfuscation solely from LWE, an intermediate objective would be to construct obfuscation from LWE in a simpler idealized model. Specifically, can we utilize a pseudorandom oracle model [JLLW23] instead of ideal obfuscation in the aforementioned construction?

The primary obstacle to achieving this objective is demonstrating the pseudorandomness of the truth table of F in the presence of the auxiliary information that contains the encoding key. Given the encoding key ek, the truth table of F is entirely determined. Specifically, it can be computed as $\boldsymbol{e} \oplus (C(1), \ldots, C(N))$ where $\boldsymbol{e}$ is obtained by applying the encoding function Enc on Bob's input $(1, \ldots, N)$ and the encoding key ek. Consequently, there is no pseudorandomness in the truth table of F given ek. Therefore, we must devise an alternative construction that enables us to establish the pseudorandomness of the truth table of the obfuscated function in the presence of the auxiliary information.

An Initial Attempt. Let's first describe a construction that demonstrates the pseudorandomness of the truth table of the obfuscated function in the presence of auxiliary information. However, this construction does not meet the required efficiency property for xi$\mathcal{O}$. Nevertheless, it serves as a stepping stone for our final construction.

Let $C : [N] \to \{0,1\}$ be the circuit to be obfuscated. The obfuscator will first sample a random $\boldsymbol{y}$ from $\{0,1\}^N$. Let $\boldsymbol{y} = (y_1, \ldots, y_N)$ and the obfuscator sets Bob's input to be the vector $((1, y_1), \ldots, (N, y_N))$. Alice's input is a circuit $\mathcal{C}$ that takes (i, y_i) as inputs and outputs $C(i)$. The function that is computed via the SMS protocol is the Universal circuit that outputs $(\mathcal{C}(1, y_1), \ldots, \mathcal{C}(N, y_N))$. The obfuscator generates the encoding key ek using Alice's input $\mathcal{C}$. It generates the hash h of Bob's input. It obfuscates the circuit F using the PROM that takes as input $i \in [N]$ and outputs $\boldsymbol{d}_i$. Note that this circuit F has the hash value h and Alice's trapdoor td hardwired. Looking ahead, these will serve as the key for the function F. The obfuscated program includes $(\boldsymbol{y}, \mathsf{ek}, \mathsf{PROM}(F))$.[2]

The evaluation procedure is exactly same as before. We first compute $\boldsymbol{e}$ using Bob's input $((1, y_1), \ldots, (N, y_N))$ and the encoding key ek. We query the function F on input i to obtain d_i and compute $C(i)$ as $e_i \oplus d_i$. To use the PROM security, we need to show that the truth table of F is pseudorandom given the auxiliary information $(\boldsymbol{y}, \mathsf{ek})$. This is argued as follows.

As before, we first change how the truth table of F is computed. Instead of computing it as $\boldsymbol{d}$, we compute it as $\boldsymbol{e} \oplus (C(1), \ldots, C(N))$. This change is indistinguishable from the correctness of the SMS protocol. Now, notice that the SMS trapdoor is not utilized anywhere. Hence, we switch the circuit that is encoded in ek to $\mathcal{C}'$ where $\mathcal{C}'$ on input (i, y_i) outputs $C(i) \oplus y_i$. This change is indistinguishable from the SMS security. Now, from the correctness of SMS protocol, $\boldsymbol{e} \oplus \boldsymbol{d} = (\mathcal{C}'(1, y_1), \ldots, \mathcal{C}'(N, y_N))$. Therefore, $\boldsymbol{e} \oplus (C(1), \ldots, C(N)) = \boldsymbol{d} \oplus \boldsymbol{y}$. Thus, we switch the truth table of F to be $\boldsymbol{d} \oplus \boldsymbol{y}$ and this switch is indistinguishable from the correctness of the SMS protocol. Now, we observe that we can compute $\boldsymbol{d}$ from the SMS trapdoor td and the hash value h. Furthermore, the hash value h hides information about y (this follows from Bob's security).[3] Thus, the distribution of the truth table $\boldsymbol{d} \oplus \boldsymbol{y}$ is indistinguishable from random. Furthermore, given the truth table, we can compute the encoding key ek as in the last hybrid and we can set $\boldsymbol{y}$ to be the truth table XOR $\boldsymbol{d}$ where $\boldsymbol{d}$ is computed using the SMS trapdoor td and hash h (which is independent of $\boldsymbol{y}$).[4] This shows that the truth table can be switched to random and the auxiliary information can be generated from the truth table.

[2] By $\mathsf{PROM}(F)$, we denote the handle h that is obtained by calling the oracle $\mathcal{O}$ on $(\mathsf{Gen}, (\mathsf{td}, h))$.

[3] We require a stronger property for Bob's security known as the equivocal hash property. In essence, the equivocal hash property asserts that the hash value can be equivocated to any input when provided with a trapdoor. For more details on this property, we refer the reader to the main body of the text. We will omit the formal details in the technical overview.

[4] This is the place where we are crucially using the fact that the output shares of SMS are an additive secret sharing of the function output.

Fixing the Efficiency. The size of the obfuscated program in the above construction is at least N as the size of $\boldsymbol{y}$ is N bits. To fix this efficiency issue, we slice the input domain into $\sqrt{N}$ blocks where the size of each block is $\sqrt{N}$. We sample a single $\boldsymbol{y}$ of size $\sqrt{N}$ randomly and generate $\sqrt{N}$ encoding keys $\mathsf{ek}_1, \ldots, \mathsf{ek}_{\sqrt{N}}$, where ek_i is used for computing the output of C on the i-th block. We then compute h_i as the hash of $\left(\left((i-1)\sqrt{N}+1, y_1\right), \ldots, \left(i\sqrt{N}, y_{\sqrt{N}}\right)\right)$ for each $i \in [\sqrt{N}]$. We generate $\sqrt{N}$ functions $F_1, \ldots, F_{\sqrt{N}}$ where F_i has the i-th trapdoor td_i and h_i hardwired and computes the i-th block of $\boldsymbol{d}$. The obfuscation now consists of $(\boldsymbol{y}, \mathsf{ek}_1, \ldots, \mathsf{ek}_{\sqrt{N}}, \mathsf{PROM}(F_1), \ldots, \mathsf{PROM}(F_{\sqrt{N}}))$. It can be verified that this construction satisfies the required efficiency property of xi$\mathcal{O}$. Via a similar argument as before, we prove the pseudorandomness of a specific F_i by "programming" the value of $\boldsymbol{y}$ from the truth table of F_i. We refer to Section 4 for the details.

A Remark on the Security Guarantee. We note that the PROM model used in our analysis is instantiated with a specific pseudorandom function. The security of our construction holds as long as the PROM guarantees hold for this specific function. This must be contrasted with the VBB construction in Jain et al. [JLLW23] whose security can be argued if there exists *any* pseudorandom function for which the PROM guarantees hold. In the next section, we abstract out the properties needed from the PROM model and formalize a standard model definition that is sufficient to instantiate our construction template.

2.2 Indistinguishability Obfuscation from Average-case i$\mathcal{O}$

The pseudorandom oracle model serves as a preliminary step towards constructing advanced cryptographic primitives, similar to the random oracle model. However, it comes with some caveats. First, it is an idealized model, and just like the random oracle model, there are secure constructions in the PROM model that become insecure when the obfuscation of the pseudorandom function is replaced with any standard model primitive. Second, in our context, the transformation from xi$\mathcal{O}$ to i$\mathcal{O}$ presented in the work of Lin et al. [LPST16a] is not applicable in the pseudorandom oracle model. This is because this transformation makes non-black-box use of the obfuscation procedure, which prevents it from going through with respect to an oracle. Consequently, our next objective is to provide a standard model definition for obfuscating pseudorandom functions that is sufficient to instantiate our construction approach.

Average-case i$\mathcal{O}$. We present a natural definition of obfuscation for pseudorandom functions – a weakening of standard i$\mathcal{O}$ – which we refer to as *average-case* i$\mathcal{O}$. Informally, our definition states that if the truth table of the obfuscated program is pseudorandom – even in the presence of some auxiliary input – then i$\mathcal{O}$-security can be achieved, meaning that if two programs are functionally equivalent, then their obfuscations are computationally indistinguishable.

More formally, our definition states that there exists a simulator Sim such that if

$$(\{C(x)\}_x, \mathsf{aux}) \approx (U, \mathsf{Sim}(U))$$

where U is the uniform distribution over the truth table of C, then for any two functionally equivalent circuits C and C', it holds that

$$\mathsf{Obf}(C), \mathsf{aux} \approx \mathsf{Obf}(C'), \mathsf{aux}.$$

It is easy to see that this definition is weaker than standard i$\mathcal{O}$ because i$\mathcal{O}$ guarantees that the post condition holds regardless of whether the precondition holds or not. Our definition is similar to, but stronger than the notion of obfuscation for identical pseudorandom functions (iPRO) recently defined in [BDJ+25]. The key difference is that iPRO does not allow simulation of the auxiliary input in the pre-condition.

xi$\mathcal{O}$ from average-case i$\mathcal{O}$. We now describe how to modify our construction in the PROM model to build xi$\mathcal{O}$ in the plain model. To keep things simple, we explain a construction where the size of $\boldsymbol{y}$ is as large as the truth table (given by N). To achieve the required efficiency of xi$\mathcal{O}$, we can use the same trick as before of chopping the input domain into $\sqrt{N}$ blocks of size $\sqrt{N}$.

Let $C : [N] \to \{0,1\}$ be the circuit to be obfuscated. The obfuscator starts by sampling a string $\boldsymbol{y} \in \{0,1\}^N$ and Bob's input to the SMS is set to be $((1, y_1), \ldots, (N, y_N))$. On the other hand, we set Alice's input to be the all-zero function, that is, a function that outputs 0 on all inputs. Additionally, we obfuscate the circuit F into $\hat{F}$ that outputs $\boldsymbol{d} + (C(1), \ldots, C(N))$ where $\boldsymbol{d}$ is the SMS decoding value. We make this modification to the construction so that we can use the post condition guarantee that the obfuscations of functionally equivalent programs are computationally indistinguishable.

Evaluation works exactly as before: First, we compute the encoding $\boldsymbol{e}$. Then, we evaluate $\hat{F}$ to obtain $\boldsymbol{d} + (C(1), \ldots, C(N))$. Finally, since ek encodes the all-zero function, we have $\boldsymbol{e} = \boldsymbol{d}$ by the correctness of the SMS protocol. Using this, we can obtain $(C(1), \ldots, C(N))$ by XORing the output of $\hat{F}$ with $\boldsymbol{e}$.

Let C' and C be functionally equivalent. To prove security, we need to switch $\hat{F}$ to be an obfuscation of a function that uses C' instead of C. Since these two functions are functionally equivalent, we should be able to use the security of average-case $\mathsf{i}\mathcal{O}$ provided that the precondition holds. To prove that precondition holds, we use a similar argument as before. We first replace the truth table of $\hat{F}$ with $\boldsymbol{e} + (C(1), \ldots, C(N))$. This change is indistinguishable from the correctness of the SMS scheme. Then, we invoke the security of SMS to replace the encoding key ek with one that encodes a function that outputs $C(i) \oplus y_i$ on input (i, y_i). This change is indistinguishable from the security of the SMS protocol. Finally, we replace the truth table with $\boldsymbol{d} + \boldsymbol{y}$ using the correctness of the SMS protocol. From this point onward, we can follow the same argument as in the earlier proof to establish the pseudorandomness of the truth table and show that all auxiliary information (i.e., ek and $\boldsymbol{y}$) can be generated from it. For the details of our construction and the security argument, we refer the reader to Section 6.

Why iPRO doesn't suffice. Note that in the above, the notion of iPRO does not seem sufficient to establish security. The core issue is in arguing pseudorandomness of the truth table. Specifically, when we replace the truth table with uniformly random values, the encoding key ek and the vector $\boldsymbol{y}$—which are treated as auxiliary information—are actually generated from the truth table itself. In other words, the auxiliary information is simulated based on the truth table. This, however, stands in contrast to the precondition in iPRO, which requires pseudorandomness to hold even when given the same auxiliary information across both experiments.

3 Preliminaries

Let λ denote the cryptographic security parameter. We assume that all cryptographic algorithms implicitly take 1^λ as input. A function $\mu(\cdot) : \mathbb{N} \to \mathbb{R}^+$ is said to be negligible if for any polynomial $\mathsf{poly}(\cdot)$, there exists λ_0 such that for all $\lambda > \lambda_0$ we have $\mu(\lambda) < \frac{1}{\mathsf{poly}}$. We will use $\mathsf{negl}(\cdot)$ to denote an unspecified negligible function and $\mathsf{poly}(\cdot)$ to denote an unspecified polynomial function.

Let X and Y be two distributions. Then i) $X \approx_c Y$ means that the distributions are computationally indistinguishable, ii) $X \approx_\varepsilon Y$ means that their statistical distance is at most ε and, iii) $X \approx_s Y$ means that their statistical distance is at most negl.

For a finite set S, we denote $x \leftarrow S$ as the process of sampling x uniformly from the set S. We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$.

3.1 Simultaneous-Message and Succinct (SMS) Secure Computation

Simultaneous-Message and Succinct (SMS) secure computation was introduced in [BJSS25]. It is a two-party protocol between Alice and Bob in the CRS model. The CRS consists of a hashing key hk. Alice has an input $\boldsymbol{x} \in \{0,1\}^n$ and Bob has an input $\boldsymbol{y} \in \{0,1\}^N$. The common input to Alice and Bob is a description of a function $f : \{0,1\}^n \times \{0,1\}^N \to \{0,1\}^m$. Alice uses hk and $\boldsymbol{x}$ as inputs to KeyGen which outputs an encoding key ek and a trapdoor td. Bob hashes his input $\boldsymbol{y}$ and the description of the function f using the hashing key hk to obtain the hash value h. Alice sends the encoding key ek to Bob and Bob sends the hash value h to Alice. Bob and Alice can use the messages exchanged and their secret state to obtain an additive secret sharing of $f(\boldsymbol{x}, \boldsymbol{y})$. The main efficiency property that this scheme should satisfy is that the size of the

exchanged messages should be sublinear in N and m. The security property we need is that the encoding key ek hides information about $\boldsymbol{x}$ and the hash digest h statistically hides information about $\boldsymbol{y}$. Consequently, we require the Hash algorithm to be randomized.

Boyle et al. [BJSS25] and Abram et al. [AMR25] gave a construction of SMS protocol for computing every polynomial sized function from Learning with Errors (LWE) assumption.

Definition 1 (Simultaneous-Message and Succinct secure computation [BJSS25])**.** *Let λ be the security parameter. Let $s = s(\lambda)$, $n = n(\lambda)$, $N = N(\lambda)$ and $m = m(\lambda)$ be polynomial functions. Let $\mathfrak{C}$ be a class of circuits of size s mapping $\{0,1\}^n \times \{0,1\}^N$ to $\{0,1\}^m$ with depth d. A Simultaneous-Message and Succinct (SMS) protocol for computing $\mathfrak{C}$ is a tuple of algorithms $\mathsf{TDH} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Hash}, \mathsf{Enc}, \mathsf{Dec})$ such that*

- $\mathsf{Setup}(1^\lambda, 1^d, 1^s, 1^n, 1^N, 1^m)$ *takes as input the security parameter 1^λ and integers $s, n, N, m \in \mathbb{N}$. It outputs a hashing key hk.*
- $\mathsf{KeyGen}(\mathsf{hk}, \boldsymbol{x})$ *takes as input a hashing key hk and an input $\boldsymbol{x} \in \{0,1\}^n$. It outputs an encoding key ek and trapdoor td.*
- $\mathsf{Hash}(\mathsf{hk}, f, \boldsymbol{y}; r)$ *takes as input a hashing key hk, the description of function $f \in \mathcal{C}$, a vector $\boldsymbol{y} \in \{0,1\}^N$, and it uses r as random coins. It outputs a hash value h.*
- $\mathsf{Enc}(\mathsf{ek}, f, \boldsymbol{y}, r)$ *takes as input an encoding key ek, $f \in \mathfrak{C}$, the vector $\boldsymbol{y} \in \{0,1\}^N$ and random coins r. It outputs a string $\boldsymbol{e} \in \{0,1\}^m$.*
- $\mathsf{Dec}(\mathsf{td}, h)$ *takes as input a trapdoor td and a hash value h. It outputs a string $\boldsymbol{d} \in \{0,1\}^m$.*

An SMS protocol $\mathsf{SMS} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Hash}, \mathsf{Enc}, \mathsf{Dec})$ satisfies the following properties:

- ***Succinctness.***
  1. *For all $\lambda \in \mathbb{N}$ and polynomials $s = s(\lambda)$, $n = n(\lambda)$, $N = N(\lambda)$, $m = m(\lambda)$, all $f \in \mathfrak{C}$ with depth d, $\mathsf{hk} \in \mathsf{Setup}(1^\lambda, 1^d, 1^s, 1^n, 1^N, 1^m)$, $(\mathsf{ek}, \mathsf{td}) \in \mathsf{KeyGen}(\mathsf{hk}, \boldsymbol{x})$, and $h \in \mathsf{Hash}(\mathsf{hk}, f, \boldsymbol{y})$ we have that*
  $$|\mathsf{hk}| = s \cdot \mathsf{poly}(\lambda, d), \quad |\mathsf{ek}| = (n + o(m)) \cdot \mathsf{poly}(\lambda, d) \quad \text{and} \quad |h| = (o(N) + o(m)) \cdot \mathsf{poly}(\lambda, d)$$
  2. *For each $i \in [m]$, there exists a function Dec_i that takes td and h as inputs and outputs $\boldsymbol{d}_i$. The size of Dec_i is $|h| \cdot \mathsf{poly}(\lambda, d)$. We refer to this property as decomposable Dec.*
- ***Correctness.*** *For all $\lambda \in \mathbb{N}$, and polynomials $s = s(\lambda)$, $n = n(\lambda)$, $N = N(\lambda)$, $m = m(\lambda)$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $f \in \mathfrak{C}$ and all $\boldsymbol{x} \in \{0,1\}^n$, $\boldsymbol{y} \in \{0,1\}^N$, we have that*
$$\Pr\left[\boldsymbol{e} \oplus \boldsymbol{d} = f(\boldsymbol{x}, \boldsymbol{y}) : \begin{array}{c} \mathsf{hk} \leftarrow \mathsf{Setup}(1^\lambda, s, n, N, m) \\ (\mathsf{ek}, \mathsf{td}) \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \boldsymbol{x}) \\ h \leftarrow \mathsf{Hash}(\mathsf{hk}, f, \boldsymbol{y}; r) \\ \boldsymbol{e} \leftarrow \mathsf{Enc}(\mathsf{ek}, f, \boldsymbol{y}; r) \\ \boldsymbol{d} \leftarrow \mathsf{Dec}(\mathsf{td}, h) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$
- ***Security.*** *For all $\lambda \in \mathbb{N}$ and polynomials $s = s(\lambda)$, $n = n(\lambda)$, $N = N(\lambda)$, $m = m(\lambda)$, for all $\boldsymbol{x}_0, \boldsymbol{x}_1 \in \{0,1\}^n$ we have that*
$$\left\{(\mathsf{hk}, \mathsf{ek}_0) : \begin{array}{c} \mathsf{hk} \leftarrow \mathsf{Setup}(1^\lambda, s, n, N, m) \\ (\mathsf{ek}_0, \mathsf{td}) \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \boldsymbol{x}_0) \end{array}\right\} \approx_c \left\{(\mathsf{hk}, \mathsf{ek}_1) : \begin{array}{c} \mathsf{hk} \leftarrow \mathsf{Setup}(1^\lambda, s, n, N, m) \\ (\mathsf{ek}_1, \mathsf{td}) \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \boldsymbol{x}_1) \end{array}\right\}.$$
- ***Statistical hiding.*** *For all $\lambda \in \mathbb{N}$ and polynomials $s = s(\lambda)$, $n = n(\lambda)$, $N = N(\lambda)$, $m = m(\lambda)$, for all $f \in \mathfrak{C}$ and $\boldsymbol{y} \in \{0,1\}^N$, we have that*
$$\left\{(\mathsf{hk}, h) : \begin{array}{c} \mathsf{hk} \leftarrow \mathsf{Setup}(1^\lambda, s, n, N, m) \\ r \leftarrow \{0,1\}^\lambda \\ h \leftarrow \mathsf{Hash}(\mathsf{hk}, f, \boldsymbol{y}; r) \end{array}\right\} \approx_s \left\{(\mathsf{hk}, h) : \begin{array}{c} \mathsf{hk} \leftarrow \mathsf{Setup}(1^\lambda, s, n, N, m) \\ h \leftarrow \{0,1\}^k \end{array}\right\}$$
  where k is the output size of $\mathsf{Hash}(\mathsf{hk}, f, \boldsymbol{y}; r)$.

We require an additional property from the SMS. Specifically, there is an alternate sampling procedure $\mathsf{AltSetup}$ that samples the hashing key along with the trapdoor. We require the distribution of the hashing key output by $\mathsf{AltSetup}$ to be statistically close to the hashing key output by Setup. Given the trapdoor output by $\mathsf{AltSetup}$, there is a procedure Equiv that can output the random coins r that "explains" the hash to be a randomly chosen h for an arbitrary $\boldsymbol{y}$ and f. More formally,

Definition 2. *An* $\mathsf{SMS} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Hash}, \mathsf{Enc}, \mathsf{Dec})$ *for computing* $\mathfrak{C}$ *is said to have equivocal hash if there exist two additional algorithms* $(\mathsf{AltSetup}, \mathsf{Equiv})$ *such that:*

- $(\mathsf{hk}, \mathsf{td}') \leftarrow \mathsf{AltSetup}(1^\lambda, s, n, N, m)$ *is an algorithm such that*

$$\{\mathsf{hk} : (\mathsf{hk}, \mathsf{td}') \leftarrow \mathsf{AltSetup}(1^\lambda, s, n, N, m)\} \approx_s \{\mathsf{hk} : \mathsf{hk} \leftarrow \mathsf{Setup}(1^\lambda, s, n, N, m)\}.$$

- $r' \leftarrow \mathsf{Equiv}(\mathsf{td}', h, f, \boldsymbol{y})$ *is an algorithm such that for every* $f \in \mathfrak{C}$ *and* $\boldsymbol{y} \in \{0,1\}^N$*, we have*

$$\left\{(\mathsf{hk}, f, \boldsymbol{y}, r, h) : \begin{array}{c} \mathsf{hk} \leftarrow \mathsf{Setup}(1^\lambda, s, n, N, m) \\ r \leftarrow \{0,1\}^\lambda \\ h \leftarrow \mathsf{Hash}(\mathsf{hk}, f, \boldsymbol{y}; r) \end{array}\right\} \approx_s \left\{(\mathsf{hk}, f, \boldsymbol{y}, r, h) : \begin{array}{c} (\mathsf{hk}, \mathsf{td}') \leftarrow \mathsf{AltSetup}(1^\lambda, s, n, N, m) \\ h \leftarrow \{0,1\}^k \\ r \leftarrow \mathsf{Equiv}(\mathsf{td}', h, f, \boldsymbol{y}) \end{array}\right\}$$

  where k *is the output size of* $\mathsf{Hash}(\mathsf{hk}, f, \boldsymbol{y}; r)$.

Though equivocal hash property was not explicitly mentioned in [BJSS25], we show that this property can be added to their construction. We briefly specify how to do this in Appendix A.1. In Appendix A.2, we show how the construction in [BJSS25] satisfies the decomposable Dec property.

Theorem 3 ([BJSS25])**.** *Let* λ *be the security parameter and* $0 < \varepsilon < 1/2$*. Assuming LWE with sub-exponential modulus-to-noise ratio, an SMS scheme for computing* $\mathfrak{C}$ *of depth* d *with equivocal hash where the correctness error is at most* $2^{-\lambda^\varepsilon}$*, the size of* $|\mathsf{ek}| = (n+m^{2/3}) \cdot \mathsf{poly}(\lambda, d)$*, and* $|h| = m^{2/3} \cdot \mathsf{poly}(\log N, \lambda, d)$*.*

3.2 Indistinguishability Obfuscation

We start by presenting the definitions of indistinguishability obfuscation [BGI+01, GGH+13]. We present the definitions in the oracle model where the obfuscator and the evaluator can make queries to an oracle $\mathcal{O}$.

Definition 3 (Indistinguishability obfuscation [BGI+01])**.** *Let* $\mathcal{O}$ *be an oracle. An indistinguishability obfuscation scheme for a class of circuits* $\mathfrak{C}$ *w.r.t. to* $\mathcal{O}$ *consists of two PPT oracle aided algorithms* $\mathsf{iO}^{\mathcal{O}} = (\mathsf{Obf}^{\mathcal{O}}, \mathsf{Eval}^{\mathcal{O}})$ *such that:*

- $\mathsf{Obf}^{\mathcal{O}}(1^\lambda, \mathcal{C})$ *takes as input the security parameter* 1^λ *and a circuit* $\mathcal{C} \in \mathfrak{C}$*. It outputs an obfuscated circuit* $\hat{\mathcal{C}}$*.*
- $\mathsf{Eval}^{\mathcal{O}}(\hat{\mathcal{C}}, x)$ *takes as input an obfuscated circuit* $\hat{\mathcal{C}}$ *and an input* $x \in \{0,1\}^\eta$*. It outputs a value* y*.*

We require the following properties to hold.

- ***ε-Correctness.*** *For all* $\lambda \in \mathbb{N}$*, all circuits* $\mathcal{C} \in \mathfrak{C}$*, we have that*

$$\Pr\left[\forall x \in \{0,1\}^\eta, \mathcal{C}(x) = \mathsf{Eval}^{\mathcal{O}}(\mathsf{Obf}^{\mathcal{O}}(1^\lambda, \mathcal{C}), x)\right] \geq 1 - \varepsilon.$$

- ***Indistinguishability.*** *For all* $\lambda \in \mathbb{N}$*, all pairs of functionally-equivalent circuits* $\mathcal{C}_0 \equiv \mathcal{C}_1$ *in* $\mathfrak{C}$ *and for every PPT oracle aided adversary* $\mathcal{A}$*, we have that*

$$|\Pr[\mathcal{A}^{\mathcal{O}}(1^\lambda, \mathsf{Obf}^{\mathcal{O}}(1^\lambda, \mathcal{C}_0)) = 1] - \Pr[\mathcal{A}^{\mathcal{O}}(1^\lambda, \mathsf{Obf}^{\mathcal{O}}(1^\lambda, \mathcal{C}_1)) = 1]| \leq \mathsf{negl}(\lambda).$$

Exponentially-efficient i$\mathcal{O}$ (xi$\mathcal{O}$) is a weaker notion of i$\mathcal{O}$ in which we allow the obfuscator to run in polynomial time in the size of the truth table, as long as the size of the resulting obfuscation is slightly sublinear in the size of the truth table.

Definition 4 (Exponentially-efficient i$\mathcal{O}$ [LPST16a])**.** *Let $\mathcal{O}$ be an oracle. An exponentially efficient obfuscation scheme $\mathsf{xiO}^{\mathcal{O}} = (\mathsf{Obf}^{\mathcal{O}}, \mathsf{Eval}^{\mathcal{O}})$ scheme is defined identically as above, except that we allow the obfuscation algorithm to run in time polynomial in the size of the truth table (2^η) of $\mathcal{C}$, and we only impose the following requirements:*

- *(Non-Trivial Efficiency) We require that $|\hat{\mathcal{C}}| \leq \mathsf{poly}(\lambda, |\mathcal{C}|) \cdot 2^{\eta(1-\delta)}$ for some constant $\delta > 0$.*

When indistinguishability obfuscation and exponentially-efficient iO are defined in the standard model without any access to the oracles, we simply denote them as iO and xiO respectively.

xi$\mathcal{O}$ to i$\mathcal{O}$. In [LPST16a], it is shown that this weaker notion of xi$\mathcal{O}$ is sufficient to build full-fledged i$\mathcal{O}$ assuming subexponential secure LWE in the standard model.

Lemma 1. *[LPST16a] Let λ be the security parameter. Assuming the sub-exponential LWE assumption and sub-exponentially secure xiO scheme with ε correctness error for obfuscating circuits with $O(\log \lambda)$ input bits, there exists an iO scheme for obfuscating circuits with η bit inputs with correctness error at most $2^\eta \cdot \mathsf{poly}(\eta) \cdot \varepsilon$.*

Remark 1. *Lemma 1 in [LPST16a] is proved assuming the xiO scheme has perfect correctness. We observe that the same proof shows that if xiO scheme as ε-correctness error, then the resultant iO scheme has correctness error $2^\eta \cdot \mathsf{poly}(\eta) \cdot \varepsilon$ for obfuscating circuits that takes η bits of inputs. This follows via a simple union bound over the inputs and the xiOs used in the iO evaluation on a specific input.*

Corollary 1. *Let λ be the security parameter and $0 < \delta < \varepsilon < 1/2$. Assuming sub-exponential LWE assumption and sub-exponentially secure xiO scheme with $2^{-\lambda^\varepsilon}$ correctness error for obfuscating circuits with $O(\log \lambda)$ input bits, there exists an iO scheme for obfuscating circuits with λ^δ bits of input and negligible correctness error.*

3.3 Pseudorandom Oracle Model

We recall the pseudorandom oracle model (PROM) from [JLLW23].

Definition 5 (Pseudorandom oracle model [JLLW23])**.** *Let F be a pseudorandom function. The pseudorandom oracle model (PROM) for F is a model with an oracle that implements a random permutation $\mathsf{HMap} : \{0,1\}^\lambda \to \{0,1\}^\lambda$ and that answers to two different types of queries:*

- $\mathcal{O}(\mathsf{Gen}, k) = \mathsf{HMap}(k)$.
- $\mathcal{O}(\mathsf{Eval}, h, x) = F(\mathsf{HMap}^{-1}(h), x)$.

4 xi$\mathcal{O}$ from LWE in Pseudorandom Oracle Model

In this section, we show how to construct xi$\mathcal{O}$ generically from SMS in the PROM model. We require the security of PROM for a specific PRF family that we define in our construction.

4.1 Construction

Building Blocks. We start by introducing the ingredients needed for our construction.

- Let $\mathfrak{C}$ be the Universal circuit U that takes in a circuit $C : [m] \times \{0,1\} \to \{0,1\}$ (of a fixed size) from Alice and $\boldsymbol{y} \in \{0,1\}^m$ from Bob and outputs $(C(1, y_1), C(2, y_2), \ldots, C(m, y_m))$. We will be using an SMS scheme $\mathsf{SMS} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Hash}, \mathsf{Enc}, \mathsf{Dec})$ with equivocal hash for $\mathfrak{C}$ from Definitions 1 & 2. Let $(\mathsf{AltSetup}, \mathsf{Equiv})$ be the corresponding algorithms for equivocating the hash (see Definition 2).
- A pseudorandom oracle model $(\mathsf{HMap}, \mathcal{O})$ instantiated with a pseudorandom function $\mathcal{D}$ described in the construction.

Construction. We now describe the construction. Let $\ell, m \in \mathbb{N}$ such that $\ell = \sqrt{N}$ and $m = \sqrt{N}$. Note that $N = \ell \cdot m$. Let $\mathcal{C}$ be an arbitrary circuit mapping $[N]$ to $\{0,1\}$ with depth d. We view the inputs to $\mathcal{C}$ as (i,j) where $i \in [\ell]$ and $j \in [m]$.

$\mathsf{xiO.Obf}^{\mathcal{O}}(1^\lambda, \mathcal{C} : [\ell] \times [m] \to \{0,1\})$:

- Sample $\mathsf{hk} \leftarrow \mathsf{Setup}(1^\lambda, 1^{\mathsf{poly}(d)}, 1^{m \cdot |\mathcal{C}| \cdot \mathsf{poly}(\lambda)}, 1^{O(|\mathcal{C}|)}, 1^m, 1^m)$.
- Sample $\boldsymbol{y} = (y_1, \ldots, y_m) \leftarrow \{0,1\}^m$, together with random coins $r \in \{0,1\}^\lambda$ and compute $h := \mathsf{Hash}(\mathsf{hk}, U, \boldsymbol{y}; r)$ where U is the universal circuit defined earlier.
- For all $i \in [\ell]$, compute $(\mathsf{ek}_i, \mathsf{td}_i) \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{F}_i)$ where $\mathcal{F}_i(j, y_j) = \mathcal{C}(i,j)$ for all $j \in [m]$.
- Let $\mathcal{D}_i = \mathcal{D}[i, \mathsf{td}_i, h] : [m] \to \{0,1\}$ be a keyed function with $k_i = (i, \mathsf{td}_i, h)$ as the key. $\mathcal{D}_i$ does the following:
  - Parses $j \in [m]$ as the input.
  - Computes $d_{i,j} \leftarrow \mathsf{Dec}_j(\mathsf{td}_i, h)$.
  - Outputs $d_{i,j}$.
- For all $i \in [\ell]$, it computes $h_i \leftarrow \mathcal{O}(\mathsf{Gen}, k_i)$.
- Output the obfuscated program $\hat{\mathcal{C}} = \left(\mathsf{hk}, \boldsymbol{y}, r, \{\mathsf{ek}_i, h_i\}_{i \in [\ell]}\right)$.

$\mathsf{xiO.Eval}^{\mathcal{O}}(\hat{\mathcal{C}}, (i,j) \in [\ell] \times [m])$:

- Parse $\hat{\mathcal{C}}$ as $\left(\mathsf{hk}, \boldsymbol{y}, r, \{\mathsf{ek}_i, h_i\}_{i \in [\ell]}\right)$.
- Compute $\boldsymbol{e}_i \leftarrow \mathsf{Enc}(\mathsf{ek}_i, U, \boldsymbol{y}, r)$.
- Evaluate $d_{i,j} \leftarrow \mathcal{O}(\mathsf{Eval}, h_i, j)$.
- Parse $\boldsymbol{e}_i$ as $(e_{i,1}, \ldots, e_{i,m})$ and output $c_{i,j} = d_{i,j} \oplus e_{i,j}$.

4.2 Correctness

We now prove correctness of our construction.

Lemma 2 (Correctness)**.** *The* xiO *scheme presented above has $\varepsilon \cdot N$ correctness error assuming that SMS has ε correctness error.*

Proof. Let $\hat{\mathcal{C}} = \left(\mathsf{hk}, \boldsymbol{y}, r, \{\mathsf{ek}_i, h_i\}_{i \in [\ell]}\right)$. Additionally, let $h \leftarrow \mathsf{Hash}(\mathsf{hk}, U, \boldsymbol{y}; r)$.

Fix any $i \in [\ell]$ and $j \in [m]$. Since $h_i \leftarrow \mathcal{O}(\mathsf{Gen}, (i, \mathsf{td}_i, h))$, we have $d_{i,j} = \mathcal{O}(\mathsf{Eval}, h_i, j) := \mathsf{Dec}_j(\mathsf{td}_i, h)$. Additionally, as ek_i encodes $\mathcal{F}_i$, we have from the correctness of SMS that $e_{i,j} \oplus d_{i,j} = \mathcal{F}_i(j, y_j) = \mathcal{C}(i,j)$ except with probability ε where $(e_{i,1}, \ldots, e_{i,m}) \leftarrow \mathsf{Enc}(\mathsf{ek}_i, U, \boldsymbol{y}, r)$. Hence, we have that $c_{i,j} = \mathcal{C}(i,j)$ except with probability at most ε. Lemma now follows by applying an union bound over all inputs. □

4.3 Security

We will now prove the security of our scheme.

Theorem 4. *The above construction is an* xiO *scheme in the PROM model assuming the security of* SMS *with equivocal hash property.*

Proof. Let $\mathcal{C}_0 : [N] \to \{0,1\}$ and $\mathcal{C}_1 : [N] \to \{0,1\}$ be two functionally equivalent circuits. We show that $\mathsf{xiO.Obf}^{\mathcal{O}}(1^\lambda, \mathcal{C}_0 : [N] \to \{0,1\})$ is computationally indistinguishable from $\mathsf{xiO.Obf}^{\mathcal{O}}(1^\lambda, \mathcal{C}_1 : [N] \to \{0,1\})$ via the following sequence of hybrids.

Hybrid $\mathcal{H}_0$: In this hybrid, we obfuscate $\mathcal{C}_0$ using the algorithm $\mathsf{xiO.Obf}^{\mathcal{O}}$ described above.

Now, for all $k \in [\ell]$, we define:

Hybrid $\mathcal{H}_k$: In this hybrid, for each $i \leq k$, we generate $\mathsf{ek}_i \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{G}_i)$ where $\mathcal{G}_i(j, y_j) = \mathcal{C}_1(i, j)$.

Note that $\mathcal{H}_\ell$ is identically distributed to $\mathsf{xiO.Obf}^{\mathcal{O}}(1^\lambda, \mathcal{C}_1 : [N] \to \{0,1\})$.

In Lemma 3, we prove indistinguishability between $\mathcal{H}_k$ and $\mathcal{H}_{k-1}$ for each $k \in [\ell]$. This will conclude the proof of security. □

We now present and prove the lemma that establishes indistinguishability of hybrids in the previous proof.

Lemma 3. *Assume that* SMS *is secure and has equivocal hash property. Then, for all $k \in [\ell]$, hybrids $\mathcal{H}_k$ and $\mathcal{H}_{k-1}$ are computationally indistinguishable in the PROM.*

Proof. Note that the only difference between $\mathcal{H}_k$ and $\mathcal{H}_{k-1}$ is in generation of ek_k. In $\mathcal{H}_{k-1}$, it is generated as $\mathsf{KeyGen}(\mathsf{hk}, \mathcal{F}_k)$, whereas in $\mathcal{H}_k$, it is generated as $\mathsf{KeyGen}(\mathsf{hk}, \mathcal{G}_k)$.

To make use of the PROM, we must first prove that the functionality implemented by the PROM is pseudorandom. This is proved in Claim 1. Given this claim, we prove indistinguishability between $\mathcal{H}_{k-1}$ and $\mathcal{H}_k$ below.

Hybrid $\mathcal{P}_0$: This hybrid corresponds to $\mathcal{H}_{k-1}$ except that we generate h_k uniformly. Indistinguishability of hybrids follows from the fact that HMap is a random permutation and this distribution is statistically close to $\mathcal{H}_{k-1}$.

Hybrid $\mathcal{P}_1$: This hybrid is identical to the previous one except that we answer the queries made to $\mathcal{O}$ on (Eval, h_k, j) differently. Specifically, we set for each $j \in [m]$, we set:

$$\mathcal{O}(\mathsf{Eval}, h_k, j) = e_{k,j} \oplus \mathcal{C}_0(k, j)$$

where $(e_{k,1}, \ldots, e_{k,m}) \leftarrow \mathsf{Enc}(\mathsf{ek}_k, U, \boldsymbol{y}, r)$.

Indistinguishability of hybrids follows from the correctness of the SMS. Specifically, since $\mathsf{ek}_k \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{F}_i)$ where $\mathcal{F}_i(j, y_j) = \mathcal{C}_0(i, j)$, it follows that $e_{k,j} \oplus d_{k,j} = \mathcal{C}_0(i, j)$ for each $j \in [m]$. Thus, $\mathcal{P}_0$ and $\mathcal{P}_1$ are statistically close.

Hybrid $\mathcal{P}_2$: This hybrid is identical to the previous one except that we generate ek_k as $\mathsf{ek}_k \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{G}_k)$ where $\mathcal{G}_k(j, y_j)$ outputs $C_1(k, j)$ for each $j \in [m]$.

Indistinguishability of hybrids $\mathcal{P}_1$ and $\mathcal{P}_2$ is argued in Claim 2 from the security of the SMS.

Hybrid $\mathcal{P}_3$. In this hybrid, we reverse the change made in $\mathcal{P}_1$. Indistinguishability follows directly from the correctness of SMS and the fact that $\mathcal{C}_0(k, j) = \mathcal{C}_1(k, j)$ for each $j \in [m]$.

Note that $\mathcal{P}_3$ is identically distributed to $\mathcal{H}_k$ and this completes the proof of the lemma.

□

It remains to show that the functionality implemented by the PROM is pseudorandom.

Claim 1. *Assuming the security and the equivocal hash property of SMS, $\mathcal{D}_k = \mathcal{D}[k, \mathsf{td}_k, h]$ is a pseudorandom function in $\mathcal{H}_{k-1}$.*

Proof. We prove this via a sequence of hybrids.

Hybrid $\mathcal{P}_0$: This hybrid corresponds to the joint distribution of the truth table $\mathtt{TT}_k$ of $\mathcal{D}[k, \mathsf{td}_k, h]$ along with $(\mathsf{hk}, \boldsymbol{y}, r, \{\mathsf{ek}_i\}_{i \in [\ell]}, \{h_{k'}\}_{k' \neq k})$ that are generated as in $\mathcal{H}_{k-1}$.

Hybrid $\mathcal{P}_1$: This hybrid is identical to the previous one except that

- We generate the truth table $\mathtt{TT}_k$ of $\mathcal{D}[k, \mathsf{td}_k, h]$ differently. Specifically, we set

$$\mathtt{TT}_k = \{e_{k,j} \oplus \mathcal{C}_0(k,j)\}_{j\in[m]}$$

  where $(e_{k,1}, \ldots, e_{k,m}) \leftarrow \mathsf{Enc}(\mathsf{ek}_k, U, \boldsymbol{y}, r)$.

- We additionally generate h_k uniformly.

Indistinguishability of hybrids follows from the correctness of the SMS. Specifically, since $\mathsf{ek}_k \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{F}_i)$ where $\mathcal{F}_i(j, y_j) = \mathcal{C}_0(i, j)$, it follows that $e_{k,j} \oplus d_{k,j} = \mathcal{C}_0(i, j)$ for each $j \in [m]$. Furthermore, the distribution of h_k is uniform given that HMap is a random permutation. Thus, $\mathcal{P}_0$ and $\mathcal{P}_1$ are statistically close.

Hybrid $\mathcal{P}_2$: This hybrid is identical to the previous one except that we generate ek_k as $\mathsf{ek}_k \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{F}'_k)$ where $\mathcal{F}'_k(j, y_j)$ outputs $C_0(k, j) \oplus y_j$ for each $j \in [m]$.

Indistinguishability of hybrids follows from the security of the SMS using an identical argument to Claim 2. Note that in both hybrids $\mathcal{P}_1$ and $\mathcal{P}_2$, the trapdoor associated with ek_k is never used. Hence, we can appeal to the security of the SMS to argue that $\mathsf{ek}_k \approx_c \mathsf{ek}'_k$ where $\mathsf{ek}_k \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{F}_k)$ and $\mathsf{ek}'_k \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{F}'_k)$.

Hybrid $\mathcal{P}_3$: This hybrid is identical to the previous one except that we set

$$\mathtt{TT}_k = \{d_{k,j} \oplus y_j : d_{k,j} \leftarrow \mathsf{Dec}_j(\mathsf{td}_k, h)\}_{j\in[m]}$$

and $h \leftarrow \mathsf{Hash}(\mathsf{hk}, U, \boldsymbol{y}; r)$.

Indistinguishability of hybrids follows from the correctness of the SMS. Note from the correctness of the SMS, we have that $e_{k,j} \oplus d_{k,j} = \mathcal{C}_0(k, j) \oplus y_j$ for each $j \in [m]$. Hence, we have that $e_{k,j} \oplus \mathcal{C}_0(k, j) = d_{k,j} \oplus y_j$ for each $j \in [m]$. Thus, $\mathcal{P}_2$ and $\mathcal{P}_3$ are statistically close.

Hybrid $\mathcal{P}_4$: In this hybrid, we proceed as in previous hybrid except that:

- Sample $(\mathsf{hk}, \mathsf{td}') \leftarrow \mathsf{AltSetup}(1^\lambda, 1^{\mathsf{poly}(d)}, 1^{m\cdot|\mathcal{C}|\cdot\mathsf{poly}(\lambda)}, 1^{O(|\mathcal{C}|)}, 1^m, 1^m)$.
- Sample $h \leftarrow \{0,1\}^*$
- Set $r \leftarrow \mathsf{Equiv}(\mathsf{td}', h, U, \boldsymbol{y})$.

Statistical indistinguishability of $\mathcal{P}_3$ and $\mathcal{P}_4$ is argued in Claim 3 from equivocal hash property of SMS (see Definition 2).

Hybrid $\mathcal{P}_5$: In this hybrid, we reverse the order in which $\mathtt{TT}_k$ and $\boldsymbol{y}$ are generated. Specifically, we do the following

- Sample $\mathtt{TT}_k = \{z_{k,j}\}_{j\in[m]}$ uniformly at random.
- Sample $(\mathsf{hk}, \mathsf{td}') \leftarrow \mathsf{AltSetup}(1^\lambda, 1^{\mathsf{poly}(d)}, 1^{m\cdot|\mathcal{C}|\cdot\mathsf{poly}(\lambda)}, 1^{O(|\mathcal{C}|)}, 1^m, 1^m)$.
- Sample $(\mathsf{ek}_k, \mathsf{td}_k) \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{F}'_k)$.
- Sample the hash digest $h \leftarrow \{0,1\}^*$.
- Generate $d_{k,j} \leftarrow \mathsf{Dec}_j(\mathsf{td}_k, h)$ for all $j \in [m]$.
- Compute $y_j = z_{k,j} \oplus d_{k,j}$ for all $j \in [m]$.
- Sample $r \leftarrow \mathsf{Equiv}(\mathsf{td}', h, U, \boldsymbol{y})$.

Perfect indistinguishability of hybrids follows the fact that the joint distribution of $\mathtt{TT}_k$ along with the other components is exactly the same in both hybrids. This follows directly from the definition of conditional probability.

Note that in $\mathcal{P}_5$, the distribution of $\mathtt{TT}_k$ is completely random and this completes the proof of the claim. □

Claim 2. *Assuming Alice's security in the SMS protocol, we have $\mathcal{P}_1 \approx_c \mathcal{P}_2$.*

Proof. Assume for the sake of contradiction that there is an adversary $\mathcal{A}$ that distinguishes $\mathcal{P}_1$ and $\mathcal{P}_2$ with non-negligible advantage. We will use this adversary to break the security of SMS protocol.

The reduction interacts with the SMS challenger and gives $\mathcal{F}_k$ and $\mathcal{G}_k$ as the challenge inputs of Alice. The challenger replies with $(\mathsf{hk}, \mathsf{ek}_k)$ where ek_k is an encoding of either $\mathcal{F}_k$ or $\mathcal{G}_k$. It uses hk to sample the rest of $\mathsf{ek}_{k'}$ along with $\mathsf{td}_{k'}$ for each $k' \neq k$. It samples $\boldsymbol{y}$ uniformly from $\{0,1\}^n$ and $r \leftarrow \{0,1\}^\lambda$ and computes $h = \mathsf{Hash}(\mathsf{hk}, U, \boldsymbol{y}; r)$. It generates $h_{k'} \leftarrow \mathcal{O}(\mathsf{Gen}, (k', \mathsf{td}_{k'}, h))$ for each $k' \neq k$. It samples h_k uniformly. It gives $\left(\mathsf{hk}, \boldsymbol{y}, r, \{\mathsf{ek}_i, h_i\}_{i\in[\ell]}\right)$ to $\mathcal{A}$. For each oracle call with input (Eval, h_k, j) made by $\mathcal{A}$, the reduction computes $(e_{k,1}, \ldots, e_{k,m}) \leftarrow \mathsf{Enc}(\mathsf{ek}_k, U, \boldsymbol{y}, r)$ and outputs $e_{k,j} \oplus \mathcal{C}_0(k, j)$. The reduction finally outputs whatever $\mathcal{A}$ outputs.

Note that if ek_k is generated as an encoding of $\mathcal{F}_k$, then the view of the adversary emulated by the above reduction is identical to $\mathcal{P}_1$. Else, it is identically distributed to $\mathcal{P}_2$. This contradicts the security of the SMS protocol. □

Claim 3. *Assuming the equivocal hash property of SMS, we have $\mathcal{P}_3 \approx_s \mathcal{P}_4$*

Proof. Assume for the sake of contradiction that there exists an adversary $\mathcal{A}$ that can distinguish $\mathcal{P}_3$ and $\mathcal{P}_4$ with non-negligible advantage. We give a reduction that breaks the equivocal hash property of SMS.

The reduction interacts with the SMS challenger and provides U and $\boldsymbol{y}$ as the challenge inputs. It obtains $(\mathsf{hk}, U, \boldsymbol{y}, r, h)$ from the challenger. It uses hk to sample the rest of $\mathsf{ek}_{k'}$ along with $\mathsf{td}_{k'}$ for each $k' \neq k$. It generates $h_{k'} \leftarrow \mathcal{O}(\mathsf{Gen}, (k', \mathsf{td}_{k'}, h))$ for each $k' \neq k$. It samples $(\mathsf{ek}_k, \mathsf{td}_k) \leftarrow \mathsf{Enc}(\mathsf{hk}, \mathcal{F}'_k)$ and generates the truth table $\mathtt{TT}_k$ as $\{d_{k,j} \oplus y_j\}_{j\in[m]}$ where $d_{k,j} \leftarrow \mathsf{Dec}_j(\mathsf{td}_k, h)$ for each $j \in [m]$. It provides the adversary $\mathcal{A}$ with $(\mathtt{TT}_k, \mathsf{hk}, \boldsymbol{y}, r, \{\mathsf{ek}_i\}_{i\in[\ell]}, \{h_{k'}\}_{k'\neq k})$ as input. The reduction finally outputs whatever the adversary outputs.

Note that if the output of the challenger corresponds to the LHS of the equivocal hash property, the view of the adversary as emulated by the reduction is identical to $\mathcal{P}_3$. Else, it is identically distributed to $\mathcal{P}_4$. Thus, the reduction breaks the equivocal hash property and this is a contradiction. □

4.4 Efficiency

We now analyze the size of the obfuscated circuit. Let d be the depth of $\mathcal{C}$. The obfuscated circuit is composed by $\hat{\mathcal{C}} = \left(\mathsf{hk}, \boldsymbol{y}, r, \{\mathsf{ek}_i, h_i\}_{i\in[\ell]}\right)$

- $|\mathsf{hk}| = m \cdot |\mathcal{C}| \cdot \mathsf{poly}(\lambda, d)$.
- $|\boldsymbol{y}| = m$ and $|r| = \lambda$.
- $|\mathsf{ek}_i| = (|\mathcal{F}_i| + m^{2/3}) \cdot \mathsf{poly}(\lambda, d) = (|\mathcal{C}| + m^{2/3}) \cdot \mathsf{poly}(\lambda, d)$ by the definition of SMS.
- $|h_i| = |i| + |\mathsf{td}_i| + |h| = m^{2/3} \cdot \mathsf{poly}(\lambda, d)$ as $|h| = m^{2/3} \cdot \mathsf{poly}(\lambda, d)$.

Hence, we have that

$$\begin{aligned}|\hat{\mathcal{C}}| &= m \cdot |\mathcal{C}| \cdot \mathsf{poly}(\lambda, d) + \ell \cdot (|\mathcal{C}| + m^{2/3}) \cdot \mathsf{poly}(\lambda, d) + \ell \cdot m^{2/3} \cdot \mathsf{poly}(\lambda, d) \\ &= (m \cdot |\mathcal{C}| + \ell \cdot |\mathcal{C}| + \ell \cdot m^{2/3}) \cdot \mathsf{poly}(\lambda, d).\end{aligned}$$

As $\ell = m = \sqrt{N}$, we have that the size $|\hat{\mathcal{C}}|$ of $\hat{\mathcal{C}}$ is upper-bounded $N^{1-\delta} \cdot \mathsf{poly}(\lambda, |\mathcal{C}|)$ for a small constant $\delta > 0$.

5 Average-case Indistinguishability Obfuscation

Average-case i$\mathcal{O}$ (avi$\mathcal{O}$) is a primitive used for obfuscating for (keyed) pseudorandom functions. Unlike indistinguishability obfuscation whose security needs to hold for worst-case choice of circuits, the security of average-case obfuscation is required to hold for randomly chosen keys. Furthermore, the security is required to hold only if the truth table on a random choice of the key is pseudorandom.

Definition 6 (Pseudorandom Obfuscation)**.** *An average-case i$\mathcal{O}$ scheme (avi$\mathcal{O}$) for a family of keyed circuits $\mathcal{C} : \{0,1\}^k \times \{0,1\}^\eta \to \{0,1\}^\ell$ consists of two PPT algorithms* $(\mathsf{avi}\mathcal{O}.\mathsf{Obf}, \mathsf{avi}\mathcal{O}.\mathsf{Eval})$ *with the following syntax.*

- $\mathsf{avi}\mathcal{O}.\mathsf{Obf}(1^\lambda, \mathcal{C}, K)$ *takes as input the security parameter 1^λ, a circuit $\mathcal{C}$, and a key $K \in \{0,1\}^k$. It ouputs an obfuscated circuit $\hat{\mathcal{C}}$.*
- $\mathsf{avi}\mathcal{O}.\mathsf{Eval}(\hat{\mathcal{C}}, x)$ *takes as input an obfuscated circuit $\hat{\mathcal{C}}$, and an input x. It outputs y.*

We require the following correctness property to hold.

- ***ε-Correctness:*** *For all $\lambda \in \mathbb{N}$, all $K \in \{0,1\}^k$, and all circuits $\mathcal{C}$ it holds that*

$$\Pr_{\hat{C} \leftarrow \mathsf{avi}\mathcal{O}.\mathsf{Obf}(1^\lambda, \mathcal{C}, K)} \left[\forall\, x \in \{0,1\}^\eta : \mathcal{C}(K, x) = \mathsf{avi}\mathcal{O}.\mathsf{Eval}(\hat{\mathcal{C}}, x) \right] \geq 1 - \varepsilon.$$

- ***Average-case i$\mathcal{O}$:*** *Assume there are PPT algorithms* $\mathsf{KeySamp}$ *and* Sim *such that*

$$\left(\{\mathcal{C}(K,x)\}_{x \in \{0,1\}^\eta}, \mathsf{aux}_K \right) \approx_c \left(\left\{ u_x : u_x \leftarrow \{0,1\}^\ell \right\}_{x \in \{0,1\}^\eta}, \mathsf{Sim}\left(\{u_x\}_{x \in \{0,1\}^\eta} \right) \right) \tag{5.1}$$

where $(K, \mathsf{aux}_K) \leftarrow \mathsf{KeySamp}(1^\lambda)$. Then it holds that

$$\left(\mathsf{avi}\mathcal{O}.\mathsf{Obf}(1^\lambda, \mathcal{C}, K), \mathsf{aux}_K \right) \approx_c \left(\mathsf{avi}\mathcal{O}.\mathsf{Obf}(1^\lambda, \mathcal{C}', K), \mathsf{aux}_K \right) \tag{5.2}$$

where $(K, \mathsf{aux}_K) \leftarrow \mathsf{KeySamp}(1^\lambda)$ and $\mathcal{C}'$ is such that $\mathcal{C}(K, \cdot)$ and $\mathcal{C}'(K, \cdot)$ are functionally equivalent. We denote Equation 5.1 as the precondition and Equation 5.2 as the post condition.

- ***Efficiency:*** *We require $|\hat{\mathcal{C}}|$ to be $|\mathcal{C}| \cdot \mathsf{poly}(\lambda)$.*

We make a couple of remarks about this definition.

Remark 2. *Note that this definition is trivially implied by i$\mathcal{O}$ as i$\mathcal{O}$ security is guaranteed to hold irrespective of whether the precondition holds or not. The post condition essentially says that obfuscation of two functionally equivalent circuits is computationally indistinguishable.*

Remark 3 (Comparison to iPRO)**.** *Branco et al. [BDJ+25] introduced an average-case notion of obfuscation called iPRO. The difference between their notion and our average-case obfuscation notion is in the precondition. In their definition, the right-hand side of the precondition is given by $\left(\{u_x : u_x \leftarrow \{0,1\}^\ell\}_{x \in \{0,1\}^\eta}, \mathsf{aux}_K \right)$ where $(K, \mathsf{aux}_K) \leftarrow \mathsf{KeySamp}(1^\lambda)$. In our definition, we allow aux_K to be generated by Sim based on $\{u_x\}_{x \in \{0,1\}^\eta}$. Their definition implies our definition as we can set Sim to sample an independent aux_K.*

Remark 4. *We note that any avi$\mathcal{O}$ where the size of $|\hat{\mathcal{C}}|$ is $\mathsf{poly}(|\mathcal{C}|, \lambda)$ can be transformed into a avi$\mathcal{O}$ where the size of $|\hat{\mathcal{C}}|$ is $|\mathcal{C}| \cdot \mathsf{poly}(\lambda)$. We do this by decomposing this obfuscation into $|\mathcal{C}|$ obfuscations where each obfuscation is responsible for generating the garbled gate entries of a particular gate in blind garbled circuits [BMR90, BLSV18]. The randomness needed for generating the garbled circuit and the garbled input is obtained using a puncturable PRF. The pseudorandomness of the truth table of each of the $|\mathcal{C}|$ obfuscations follows from the blindness of the garbling scheme. To prove indistinguishability between obfuscations of $\mathcal{C}$ and $\mathcal{C}'$ that are functionally equivalent, we use the standard approach of going over each input one at a time, puncturing the PRF key at that input, switching from garbling $\mathcal{C}$ to $\mathcal{C}'$ using the security of blind garbled circuits for that input, and then unpuncturing the PRF key. We sketch this transformation in Appendix B.*

6 Indistinguishability Obfuscation from LWE and avi$\mathcal{O}$

In this section, we show how to construct xi$\mathcal{O}$ generically from SMS and avi$\mathcal{O}$. We note that using Lemma 1, we can bootstrap from sub-exponentially secure xi$\mathcal{O}$ to i$\mathcal{O}$ assuming sub-exponentially secure LWE.

6.1 Construction

Building Blocks. We start by introducing the ingredients needed for our construction.

- Let $\mathfrak{C}$ is given by the Universal circuit U that takes in a circuit $C : [m] \times \{0,1\} \to \{0,1\}$ from Alice and $\boldsymbol{y} \in \{0,1\}^m$ from Bob. The output of U is $(C(1, y_1), \ldots, C(m, y_m))$. We will be using an SMS scheme $\mathsf{SMS} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Hash}, \mathsf{Enc}, \mathsf{Dec})$ with equivocal hash for $\mathfrak{C}$ from Definitions 1 & 2. Let $(\mathsf{AltSetup}, \mathsf{Equiv})$ be the corresponding algorithms for equivocating the hash (see Definition 2).
- A avi$\mathcal{O}$ scheme $(\mathsf{avi}\mathcal{O}.\mathsf{Obf}, \mathsf{avi}\mathcal{O}.\mathsf{Eval})$ from Definition 6.

Construction. We now describe the construction. Let $\ell, m \in \mathbb{N}$ such that $\ell = \sqrt{N}$ and $m = \sqrt{N}$. Note that $N = \ell \cdot m$. Let $\mathcal{C}$ be an arbitrary circuit mapping $[N]$ to $\{0,1\}$ with depth d. We view the inputs to $\mathcal{C}$ as (i, j) where $i \in [\ell]$ and $j \in [m]$.

$\mathsf{xiO}.\mathsf{Obf}(1^\lambda, \mathcal{C} : [\ell] \times [m] \to \{0,1\})$:

- Sample $\mathsf{hk} \leftarrow \mathsf{Setup}(1^\lambda, 1^{\mathsf{poly}(d)}, 1^{m \cdot |\mathcal{C}| \cdot \mathsf{poly}(\lambda)}, 1^{O(|\mathcal{C}|)}, 1^m, 1^m)$.
- Sample $\boldsymbol{y} = (y_1, \ldots, y_m) \leftarrow \{0,1\}^m$, together with random coins $r \in \{0,1\}^\lambda$ and compute $h \leftarrow \mathsf{Hash}(\mathsf{hk}, U, \boldsymbol{y}; r)$ where U is the universal circuit as defined earlier.
- For all $i \in [\ell]$ compute $(\mathsf{ek}_i, \mathsf{td}_i) \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{Z})$ where $\mathcal{Z}$ is the all-zeroes function. That is, $\mathcal{Z}(i, y_i) = 0$ for each $i \in [m]$ and $y_i \in \{0,1\}$. We pad $\mathcal{Z}$ to be of size $\mathcal{G}_i$ in the proof of Lemma 5.
- Let $\mathcal{D}_i = \mathcal{D}_i[\mathsf{td}_i, h, \mathcal{C}] : [m] \to \{0,1\}$ be the circuit that has the trapdoor td_i, the hash value h and the circuit $\mathcal{C}$ hardwired and does the following:
  - Parse $j \in [m]$ as the input.
  - Compute $d_{i,j} \leftarrow \mathsf{Dec}_j(\mathsf{td}_i, h)$.
  - Output $d_{i,j} \oplus \mathcal{C}(i, j)$.
- For all $i \in [\ell]$ compute $\hat{\mathcal{D}}_i \leftarrow \mathsf{avi}\mathcal{O}.\mathsf{Obf}(1^\lambda, \mathcal{D}_i)$.
- Output the obfuscated program $\hat{\mathcal{C}} = \left(\mathsf{hk}, \boldsymbol{y}, r, \left\{\mathsf{ek}_i, \hat{\mathcal{D}}_i\right\}_{i \in [\ell]}\right)$.

$\mathsf{xiO}.\mathsf{Eval}(\hat{\mathcal{C}}, (i, j) \in [\ell] \times [m])$:

- Parse $\hat{\mathcal{C}}$ as $\left(\mathsf{hk}, \boldsymbol{y}, r, \left\{\mathsf{ek}_i, \hat{\mathcal{D}}_i\right\}_{i \in [\ell]}\right)$.
- Compute $\boldsymbol{e}_i \leftarrow \mathsf{Enc}(\mathsf{ek}_i, U, \boldsymbol{y}, r)$.
- Evaluate $w_{i,j} \leftarrow \mathsf{avi}\mathcal{O}.\mathsf{Eval}(\hat{\mathcal{D}}_i, j)$.
- Parse $\boldsymbol{e}_i$ as $(e_{i,1}, \ldots, e_{i,m})$ and output $c_{i,j} = w_{i,j} \oplus e_{i,j}$.

6.2 Correctness

We now prove correctness of our construction.

Lemma 4 (Correctness)**.** *The* xiO *scheme presented above has* $\varepsilon \cdot N$ *correctness assuming that the underlying* $\mathsf{avi}\mathcal{O}$ *is perfectly correct and SMS has* ε*-correctness.*

Proof. Let $\hat{\mathcal{C}} = \left(\mathsf{hk}, \boldsymbol{y}, r, \left\{\mathsf{ek}_i, \hat{\mathcal{D}}_i\right\}_{i\in[\ell]}\right)$. Additionally, let $h \leftarrow \mathsf{Hash}(\mathsf{hk}, U, \boldsymbol{y}; r)$.

Fix an $i \in [\ell]$ and $j \in [m]$. By the correctness of the underlying $\mathsf{avi}\mathcal{O}$ scheme, we have that $w_{i,j} \leftarrow \mathsf{avi}\mathcal{O}.\mathsf{Eval}(\hat{\mathcal{D}}_i, j)$ where $w_{i,j} = d_{i,j} \oplus \mathcal{C}(i,j) \leftarrow \mathcal{D}_i(j)$ and $d_{i,j} \leftarrow \mathsf{Dec}_j(\mathsf{td}_i, h)$. Additionally, by the ε-correctness of SMS, we have that $e_{i,j} = d_{i,j}$ except with probability ε where $(e_{i,1}, \ldots, e_{i,m}) \leftarrow \mathsf{Enc}(\mathsf{ek}_i, U, \boldsymbol{y})$ since ek_i encodes $\mathcal{Z}$. Hence, we have that $c_{i,j} = w_{i,j} \oplus e_{i,j} = d_{i,j} \oplus \mathcal{C}(i,j) \oplus e_{i,j} = \mathcal{C}(i,j)$ except with probability ε. The lemma now follows by a union bound over all inputs. □

6.3 Security

We will now prove the security of our scheme. In the following, we assume that the size of the truth table N is a polynomial in the security parameter. This is sufficient for obtaining iO (see Lemma 1).

Theorem 5. *Assuming the security of SMS with equivocal hash and* $\mathsf{avi}\mathcal{O}$*, the above construction of* $\mathsf{xi}\mathcal{O}$ *is secure.*

Proof. Let $\mathcal{C}_0 : [N] \to \{0,1\}$ and $\mathcal{C}_1 : [N] \to \{0,1\}$ be two functionally equivalent circuits. We show that $\mathsf{xiO.Obf}(1^\lambda, \mathcal{C}_0 : [N] \to \{0,1\})$ is computationally indistinguishable from $\mathsf{xiO.Obf}(1^\lambda, \mathcal{C}_1 : [N] \to \{0,1\})$ via the following sequence of hybrids.

Hybrid $\mathcal{H}_0$**:** In this hybrid, we obfuscate $\mathcal{C}_0$ using the algorithm $\mathsf{xiO.Obf}$ described above.

Now, for all $k \in [\ell]$, we define:

Hybrid $\mathcal{H}_k$**:** In this hybrid, we generate $\hat{\mathcal{D}}_i$ for all $i \le k$ as $\mathsf{avi}\mathcal{O}.\mathsf{Obf}(1^\lambda, \tilde{\mathcal{D}}_i)$ of $\tilde{\mathcal{D}}_i$ defined below:

- Parse $j \in [m]$ as the input.
- Compute $d_{i,j} \leftarrow \mathsf{Dec}_j(\mathsf{td}_i, h)$.
- Output $\underline{d_{i,j} \oplus \mathcal{C}_1(i,j)}$.

Note that $\mathcal{H}_\ell$ is identically distributed to $\mathsf{xiO.Obf}(1^\lambda, \mathcal{C}_1 : [N] \to \{0,1\})$.

In Lemma 5, we prove indistinguishability between $\mathcal{H}_k$ and $\mathcal{H}_{k-1}$ for each $k \in [\ell]$. This will conclude the proof of security. □

We now present and prove the lemma that establishes indistinguishability of hybrids in the previous proof.

Lemma 5. *Assume the security of SMS with equivocal hash and* $\mathsf{avi}\mathcal{O}$*. Then, for all* $k \in [\ell]$*, hybrids* $\mathcal{H}_k$ *and* $\mathcal{H}_{k-1}$ *are computationally indistinguishable.*

Proof. Note that the only difference between $\mathcal{H}_k$ and $\mathcal{H}_{k-1}$ is in generation of $\hat{\mathcal{D}}_k$. In $\mathcal{H}_{k-1}$, it is generated as $\mathsf{avi}\mathcal{O}.\mathsf{Obf}(1^\lambda, \mathcal{D}_k)$, whereas in $\mathcal{H}_k$, it is generated as $\mathsf{avi}\mathcal{O}.\mathsf{Obf}(1^\lambda, \tilde{\mathcal{D}}_k)$. To prove this lemma, we use the security of the underlying $\mathsf{avi}\mathcal{O}$ scheme.

Pre-condition holds. We define aux to be $(\boldsymbol{y}, r, \mathsf{hk}, \mathsf{ek}_k)$. First, we prove that pre-condition for avi$\mathcal{O}$ holds for this aux. That is, we first show that there exists a algorithm Sim such that

$$(\mathtt{TT}_k, (\boldsymbol{y}, r), \mathsf{hk}, \mathsf{ek}_k) \approx_c (U, \mathsf{Sim}(U))$$

where $\mathtt{TT}_k = \{\mathcal{D}_k(j)\}_{j\in[m]}$ and $U \leftarrow \{0,1\}^m$. The proof follows the following sequence of hybrids.

Hybrid $\mathcal{P}_0$: This hybrid corresponds to the LHS of the pre-condition.

Hybrid $\mathcal{P}_1$: This hybrid is identical to the previous one except that the truth table $\mathtt{TT}_k$ is set as

$$\mathtt{TT}_k = \left\{ \ e_{k,j} \oplus \mathcal{C}_0(k,j) \ \right\}_{j\in[m]}$$

where $\{e_{k,j}\}_{j\in[m]} \leftarrow \mathsf{Enc}(\mathsf{ek}_k, U, \boldsymbol{y}, r)$.

Indistinguishability of hybrids follows from the correctness of the SMS. Specifically, since $\mathsf{ek}_k \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{Z})$ where $\mathcal{Z}$ is the zero function, it follows that $e_{k,j} \oplus d_{k,j} = 0$ for each $j \in [m]$. Thus, $\mathcal{P}_0$ and $\mathcal{P}_1$ are statistically close.

Hybrid $\mathcal{P}_2$: This hybrid is identical to the previous one except that we set ek_k as $\mathsf{ek}_k \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{G}_k)$ where $\mathcal{G}_k(j, y_j)$ outputs $C_0(k,j) \oplus y_j$ for each $j \in [m]$.

Indistinguishability of hybrids is proved in Claim 4 from the security of SMS.

Hybrid $\mathcal{P}_3$: This hybrid is identical to the previous one except that we set

$$\mathtt{TT}_k = \{d_{k,j} \oplus y_j : d_{k,j} \leftarrow \mathsf{Dec}_j(\mathsf{td}_k, h)\}_{j\in[m]}$$

and $h \leftarrow \mathsf{Hash}(\mathsf{hk}, U, \boldsymbol{y}; r)$.

Indistinguishability of hybrids follows from the correctness of the SMS. Note that, by the correctness of the SMS, we have that $e_{k,j} \oplus d_{k,j} = \mathcal{C}_0(k,j) \oplus y_j$ for each $j \in [m]$. Hence, we have that $e_{k,j} \oplus \mathcal{C}_0(k,j) = d_{k,j} \oplus y_j$ for each $j \in [m]$. Thus, $\mathcal{P}_2$ and $\mathcal{P}_3$ are statistically close.

Hybrid $\mathcal{P}_4$: In this hybrid, we proceed as in previous hybrid except that:

- Sample $(\mathsf{hk}, \mathsf{td}') \leftarrow \mathsf{AltSetup}(1^\lambda, 1^{\mathsf{poly}(d)}, 1^{m\cdot|\mathcal{C}|\cdot\mathsf{poly}(\lambda)}, 1^{O(|\mathcal{C}|)}, 1^m, 1^m)$.
- Sample $h \leftarrow \{0,1\}^*$
- Set $r \leftarrow \mathsf{Equiv}(\mathsf{td}', h, U, \boldsymbol{y})$.

Indistinguishability of $\mathcal{P}_3$ and $\mathcal{P}_4$ is argued in Claim 5 using the equivocal hash property (see Definition 2).

Hybrid $\mathcal{P}_4$: In this hybrid, we reverse the order in which $\mathtt{TT}_k$ and $\boldsymbol{y}$ are generated. Specifically, we do the following

- Sample $\mathtt{TT}_k = \{z_{k,j}\}_{j\in[m]}$ uniformly at random.
- Sample $(\mathsf{hk}, \mathsf{td}') \leftarrow \mathsf{AltSetup}(1^\lambda, 1^{\mathsf{poly}(d)}, 1^{m\cdot|\mathcal{C}|\cdot\mathsf{poly}(\lambda)}, 1^{O(|\mathcal{C}|)}, 1^m, 1^m)$.
- Sample $(\mathsf{ek}_k, \mathsf{td}_k) \leftarrow \mathsf{KeyGen}(\mathsf{hk}, \mathcal{G}_k)$.
- Sample the hash digest $h \leftarrow \{0,1\}^*$.
- Generate $d_{k,j} \leftarrow \mathsf{Dec}_j(\mathsf{td}_k, h)$ for all $j \in [m]$.
- Compute $y_j = z_{k,j} \oplus d_{k,j}$ for all $j \in [m]$.

- Sample $r \leftarrow \mathsf{Equiv}(\mathsf{td}', h, U, \boldsymbol{y})$.

Perfect indistinguishability of hybrids follows the fact that the joint distribution of $(\mathtt{TT}_k, \mathsf{aux})$ is exactly the same in both hybrids. This follows directly from the definition of conditional probability.

Note that, in this last hybrid $\mathcal{P}_4$, we are able to simulate the aux information given just the truth table. That is, given $\mathtt{TT}_k$, we define Sim to be the algorithm that samples $\mathsf{aux} = (\mathsf{hk}, \mathsf{ek}_k, \boldsymbol{y}, r)$ as in hybrid $\mathcal{P}_4$ depending on $\mathtt{TT}_k$. This proves that precondition holds.

Post-condition reduction. Since the pre-condition holds, and from the fact $\tilde{\mathcal{D}}_k$ and $\mathcal{D}_k$ are functionally equivalent (this follows from the fact that $\mathcal{C}_0$ and $\mathcal{C}_1$ are functionally equivalent), the post condition guarantees that

$$(\mathsf{avi}\mathcal{O}.\mathsf{Obf}(1^\lambda, \tilde{\mathcal{D}}_k), \mathsf{aux}) \approx_c (\mathsf{avi}\mathcal{O}.\mathsf{Obf}(1^\lambda, \mathcal{D}_k), \mathsf{aux})$$

To complete the proof of the lemma, parse aux as $\boldsymbol{y}, r, \mathsf{hk}, \mathsf{ek}_k$. Note that we can generate $\{\hat{\mathcal{D}}_{k'}, \mathsf{ek}_{k'}\}_{k' \neq k}$ as in $\mathcal{H}_{k-1}$ using hk and $h = \mathsf{Hash}(\mathsf{hk}, U, \boldsymbol{y}; r)$. This completes the proof of the lemma. □

Claim 4. *Assuming the security of the SMS protocol, we have $\mathcal{P}_1 \approx_c \mathcal{P}_2$.*

Proof. Assume for the sake of contradiction that there is an adversary $\mathcal{A}$ that distinguishes $\mathcal{P}_1$ and $\mathcal{P}_2$ with non-negligible advantage. We will use this adversary to break the security of SMS protocol.

The reduction interacts with the SMS challenger and gives $\mathcal{Z}$ and $\mathcal{G}_k$ as the challenge inputs of Alice. The challenger replies with $(\mathsf{hk}, \mathsf{ek}_k)$ where ek_k is an encoding of either $\mathcal{G}_k$ or $\mathcal{Z}$. It samples $\boldsymbol{y}$ uniformly from $\{0,1\}^n$ and $r \leftarrow \{0,1\}^\lambda$. It computes $(e_{k,1}, \ldots, e_{k,m}) \leftarrow \mathsf{Enc}(\mathsf{ek}_k, U, \boldsymbol{y}, \boldsymbol{r})$. It generates $\mathtt{TT}_k$ as in $\mathcal{P}_1$. It gives $(\mathtt{TT}_k, , \boldsymbol{y}, r, \mathsf{hk}, \mathsf{ek}_k)$ to $\mathcal{A}$. The reduction finally outputs whatever $\mathcal{A}$ outputs.

Note that if ek_k is generated as an encoding of $\mathcal{Z}$, then the view of the adversary emulated by the above reduction is identical to $\mathcal{P}_1$. Else, it is identically distributed to $\mathcal{P}_2$. This contradicts the security of the SMS protocol. □

Claim 5. *Assuming the equivocal hash property of SMS, we have $\mathcal{P}_3 \approx_s \mathcal{P}_4$*

Proof. Assume for the sake of contradiction that there exists an adversary $\mathcal{A}$ that can distinguish $\mathcal{P}_3$ and $\mathcal{P}_4$ with non-negligible advantage. We give a reduction that breaks the equivocal hash property of SMS.

The reduction interacts with the SMS challenger and provides U and $\boldsymbol{y}$ as the challenge inputs. It obtains $(\mathsf{hk}, U, \boldsymbol{y}, r, h)$ from the challenger. It samples $(\mathsf{ek}_k, \mathsf{td}_k) \leftarrow \mathsf{Enc}(\mathsf{hk}, \mathcal{G}_k)$ and generates the truth table $\mathtt{TT}_k$ as $\{d_{k,j} \oplus y_j\}_{j \in [m]}$ where $d_{k,j} \leftarrow \mathsf{Dec}_j(\mathsf{td}_k, h)$ for each $j \in [m]$. It provides the adversary $\mathcal{A}$ with $(\mathtt{TT}_k, \boldsymbol{y}, r, \mathsf{hk}, \mathsf{ek}_k)$ as input. The reduction finally outputs whatever the adversary outputs.

Note that if the output of the challenger corresponds to the LHS of the equivocal hash property, the view of the adversary as emulated by the reduction is identical to $\mathcal{P}_3$. Else, it is identically distributed to $\mathcal{P}_4$. Thus, the reduction breaks the equivocal hash property and this is a contradiction. □

6.4 Efficiency

We now analyze the size of the obfuscated circuit. Let d be the depth of $\mathcal{C}$. The obfuscated circuit is composed by $\hat{\mathcal{C}} = \left(\mathsf{hk}, \boldsymbol{y}, r, \left\{\mathsf{ek}_i, \hat{\mathcal{D}}_i\right\}_{i \in [\ell]}\right)$ where

- $|\mathsf{hk}| = m \cdot |\mathcal{C}| \cdot \mathsf{poly}(\lambda, d)$.
- $|\boldsymbol{y}| = m$ and $|r| = \lambda$.
- $|\mathsf{ek}_i| = (|\mathcal{G}_i| + m^{2/3}) \cdot \mathsf{poly}(\lambda, d) = (|\mathcal{C}| + m^{2/3}) \cdot \mathsf{poly}(\lambda, d)$ by the definition of SMS, where $\mathcal{G}_i$ is the function described in the proof of Lemma 5.
- By efficiency property of $\mathsf{avi}\mathcal{O}$, we have $|\hat{\mathcal{D}}_i| = |\mathcal{D}_i| \cdot \mathsf{poly}(\lambda)$. Note that $|\mathcal{D}_i|$ is $O(|\mathsf{Dec}_i| + |\mathcal{C}|)$. Since $|\mathsf{Dec}_i|$ is $|h| \cdot \mathsf{poly}(\lambda) = m^{2/3} \cdot \mathsf{poly}(\lambda)$, we have $|\mathcal{D}_i| \leq m^{2/3} \cdot |\mathcal{C}| \cdot \mathsf{poly}(\lambda)$. Therefore, $|\hat{\mathcal{D}}_i| \leq m^{2/3} \cdot |\mathcal{C}| \cdot \mathsf{poly}(\lambda)$.

Hence, we have that

$$\begin{aligned}|\hat{\mathcal{C}}| &\leq m \cdot |\mathcal{C}| \cdot \mathsf{poly}(\lambda, d) + \ell \cdot (|\mathcal{C}| + m^{2/3}) \cdot \mathsf{poly}(\lambda, d) + \ell \cdot m^{2/3} \cdot |\mathcal{C}| \cdot \mathsf{poly}(\lambda) \\ &\leq (m + \ell \cdot m^{2/3}) \cdot |\mathcal{C}| \cdot \mathsf{poly}(\lambda, d).\end{aligned}$$

Since $\ell = m = \sqrt{N}$ we have $|\hat{\mathcal{C}}|$ is at most $N^{1-\varepsilon} \cdot \mathsf{poly}(\lambda, |\mathcal{C}|)$ for some $\varepsilon > 0$.

Remark 5. *We can further reduce the efficiency requirement of aviO scheme such that the size of the obfuscation of a circuit $\mathcal{C}$ is $(|\mathcal{C}| + m^{\delta}) \cdot \mathsf{poly}(\lambda)$ where m is the size of the circuit's truth table and $\delta < 1$.*

By instantiating the SMS scheme from Theorem 3, we get an xiO scheme with $2^{-\lambda^{\varepsilon}}$ correctness error. By corollary 1, we get an iO scheme for obfuscating circuits with λ^{δ} input bits for some $\delta < \varepsilon$.

Acknowledgments

We thank Nico Döttling for invaluable discussions on this project. Pedro Branco is supported by the European Research Council through an ERC Starting Grant (Grant agreement No. 101077455, ObfusQation) and by FONDAZIONE CARIPLO through the project "Trustless - How to Remove Trust Assumptions from Cryptographic Primitives" (Rif. 2025-0813 ex 2024-1165). Abhishek Jain is funded in part by NSF CAREER award (1942789), JP Morgan Faculty award, and gifts from Stellar Development Foundation, Cisco and Ethereum Foundation. Akshayaram Srinivasan is supported in part by a NSERC Discovery Grant RGPIN-2024-03928.

References

[Agr19] Shweta Agrawal. Indistinguishability obfuscation without multilinear maps: New methods for bootstrapping and instantiation. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 191–225. Springer, Cham, May 2019. 1

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 308–326. Springer, Berlin, Heidelberg, August 2015. 1, 4

[AKY24a] Shweta Agrawal, Simran Kumari, and Shota Yamada. Compact pseudorandom functional encryption from evasive LWE. Cryptology ePrint Archive, Paper 2024/1719, 2024. 2, 4

[AKY24b] Shweta Agrawal, Simran Kumari, and Shota Yamada. Pseudorandom multi-input functional encryption and applications. *IACR Cryptol. ePrint Arch.*, page 1720, 2024. 2, 3

[AMR25] Damiano Abram, Giulio Malavolta, and Lawrence Roy. Succinct oblivious tensor evaluation and applications: Adaptively-secure laconic function evaluation and trapdoor hashing for all circuits. *IACR Cryptol. ePrint Arch.*, page 336, 2025. 3, 5, 9

[AMYY25] Shweta Agrawal, Anuja Modi, Anshu Yadav, and Shota Yamada. Evasive LWE: attacks, variants & obfustopia. *IACR Cryptol. ePrint Arch.*, page 375, 2025. 1, 4

[AP20] Shweta Agrawal and Alice Pellet-Mary. Indistinguishability obfuscation without maps: Attacks and fixes for noisy linear FE. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 110–140. Springer, Cham, May 2020. 1

[App14] Benny Applebaum. Bootstrapping obfuscators via fast pseudorandom functions. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 162–172. Springer, Berlin, Heidelberg, December 2014. 3

[ARS24] Damiano Abram, Lawrence Roy, and Peter Scholl. Succinct homomorphic secret sharing. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part VI*, volume 14656 of *LNCS*, pages 301–330. Springer, Cham, May 2024. 26

[AS17] Prabhanjan Ananth and Amit Sahai. Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 152–181. Springer, Cham, April / May 2017. 1

[BDGM20] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Candidate iO from homomorphic encryption schemes. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 79–109. Springer, Cham, May 2020. 1

[BDGM22] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Factoring and pairings are not necessary for IO: Circular-secure LWE suffices. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *ICALP 2022*, volume 229 of *LIPIcs*, pages 28:1–28:20. Schloss Dagstuhl, July 2022. 1

[BDJ+24] Pedro Branco, Nico Döttling, Abhishek Jain, Giulio Malavolta, Surya Mathialagan, Spencer Peters, and Vinod Vaikuntanathan. Pseudorandom obfuscation and applications. *IACR Cryptol. ePrint Arch.*, page 1742, 2024. 3

[BDJ+25] Pedro Branco, Nico Döttling, Abhishek Jain, Giulio Malavolta, Surya Mathialagan, Spencer Peters, and Vinod Vaikuntanathan. Pseudorandom obfuscation and applications. CRYPTO, 2025. 2, 3, 7, 16

[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, Berlin, Heidelberg, August 2001. 1, 10

[BGK+14] Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Protecting obfuscation against algebraic attacks. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 221–238. Springer, Berlin, Heidelberg, May 2014. 1

[BIJ+20] James Bartusek, Yuval Ishai, Aayush Jain, Fermi Ma, Amit Sahai, and Mark Zhandry. Affine determinant programs: A framework for obfuscation and witness encryption. In Thomas Vidick, editor, *ITCS 2020*, volume 151, pages 82:1–82:39. LIPIcs, January 2020. 1

[BJSS25] Elette Boyle, Abhishek Jain, Sacha Servan-Schreiber, and Akshayaram Srinivasan. Simultaneous-message and succinct secure computation. In Serge Fehr and Pierre-Alain Fouque, editors, *Advances in Cryptology - EUROCRYPT 2025 - 44th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Madrid, Spain, May 4-8, 2025, Proceedings, Part V*, volume 15605 of *Lecture Notes in Computer Science*, pages 207–239. Springer, 2025. 3, 5, 8, 9, 10, 25, 26

[BLSV18] Zvika Brakerski, Alex Lombardi, Gil Segev, and Vinod Vaikuntanathan. Anonymous IBE, leakage resilience and circular security from new assumptions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 535–564. Springer, Cham, April / May 2018. 16, 26, 27

[BMR90] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *22nd ACM STOC*, pages 503–513. ACM Press, May 1990. 16, 26

[BMSZ16] Saikrishna Badrinarayanan, Eric Miles, Amit Sahai, and Mark Zhandry. Post-zeroizing obfuscation: New mathematical tools, and the case of evasive circuits. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 764–791. Springer, Berlin, Heidelberg, May 2016. 1

[BR14] Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 1–25. Springer, Berlin, Heidelberg, February 2014. 1

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In Venkatesan Guruswami, editor, *56th FOCS*, pages 171–190. IEEE Computer Society Press, October 2015. 1, 4

[CCMR24] Ran Canetti, Claudio Chamon, Eduardo R. Mucciolo, and Andrei E. Ruckenstein. Towards general-purpose program obfuscation via local mixing. In Elette Boyle and Mohammad Mahmoody, editors, *Theory of Cryptography - 22nd International Conference, TCC 2024, Milan, Italy, December 2-6, 2024, Proceedings, Part IV*, volume 15367 of *Lecture Notes in Computer Science*, pages 37–70. Springer, 2024. 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, Cham, August 2018. 1

[DJM+25] Nico Döttling, Abhishek Jain, Giulio Malavolta, Surya Mathialagan, and Vinod Vaikuntanathan. Simple and general counterexamples for private-coin evasive LWE. CRYPTO, 2025. 1

[DQV+21] Lalita Devadas, Willy Quach, Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. Succinct LWE sampling, random polynomials, and obfuscation. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part II*, volume 13043 of *LNCS*, pages 256–287. Springer, Cham, November 2021. 1

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013. 1, 10

[GJK18] Craig Gentry, Charanjit S. Jutla, and Daniel Kane. Obfuscation using tensor products. Cryptology ePrint Archive, Report 2018/756, 2018. 1

[GMM+16] Sanjam Garg, Eric Miles, Pratyay Mukherjee, Amit Sahai, Akshayaram Srinivasan, and Mark Zhandry. Secure obfuscation in a weak multilinear map model. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 241–268. Springer, Berlin, Heidelberg, October / November 2016. 1

[GP21] Romain Gay and Rafael Pass. Indistinguishability obfuscation from circular security. In Samir Khuller and Virginia Vassilevska Williams, editors, *53rd ACM STOC*, pages 736–749. ACM Press, June 2021. 1

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. 25

[HJL21] Samuel B. Hopkins, Aayush Jain, and Huijia Lin. Counterexamples to new circular security assumptions underlying iO. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part II*, volume 12826 of *LNCS*, pages 673–700, Virtual Event, August 2021. Springer, Cham. 1

[HJL25] Yao-Ching Hsieh, Aayush Jain, and Huijia Lin. Lattice-based post-quantum io from circular security with random opening assumption (part II: zeroizing attacks against private-coin evasive LWE assumptions). *IACR Cryptol. ePrint Arch.*, page 390, 2025. 1, 2

[JLLS23] Aayush Jain, Huijia Lin, Paul Lou, and Amit Sahai. Polynomial-time cryptanalysis of the subspace flooding assumption for post-quantum $i\mathcal{O}$. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part I*, volume 14004 of *LNCS*, pages 205–235. Springer, Cham, April 2023. 1

[JLLW23] Aayush Jain, Huijia Lin, Ji Luo, and Daniel Wichs. The pseudorandom oracle model and ideal obfuscation. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 233–262. Springer, Cham, August 2023. 2, 4, 6, 7, 11

[JLMS19] Aayush Jain, Huijia Lin, Christian Matt, and Amit Sahai. How to leverage hardness of constant-degree expanding polynomials over $\mathbb{R}$ to build $i\mathcal{O}$. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 251–281. Springer, Cham, May 2019. 1

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In Samir Khuller and Virginia Vassilevska Williams, editors, *53rd ACM STOC*, pages 60–73. ACM Press, June 2021. 1, 4

[JLS22] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in NC^0. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part I*, volume 13275 of *LNCS*, pages 670–699. Springer, Cham, May / June 2022. 1, 4

[Lin16] Huijia Lin. Indistinguishability obfuscation from constant-degree graded encoding schemes. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 28–57. Springer, Berlin, Heidelberg, May 2016. 1

[Lin17] Huijia Lin. Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 599–629. Springer, Cham, August 2017. 1

[LPST16a] Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation with non-trivial efficiency. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part II*, volume 9615 of *LNCS*, pages 447–462. Springer, Berlin, Heidelberg, March 2016. 1, 2, 7, 11

[LPST16b] Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Output-compressing randomized encodings and applications. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part I*, volume 9562 of *LNCS*, pages 96–124. Springer, Berlin, Heidelberg, January 2016. 1

[LT17] Huijia Lin and Stefano Tessaro. Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 630–660. Springer, Cham, August 2017. 1

[LV16] Huijia Lin and Vinod Vaikuntanathan. Indistinguishability obfuscation from DDH-like assumptions on constant-degree graded encodings. In Irit Dinur, editor, *57th FOCS*, pages 11–20. IEEE Computer Society Press, October 2016. 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, Berlin, Heidelberg, April 2012. 25

[MSZ16] Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 629–658. Springer, Berlin, Heidelberg, August 2016. 1

[PST14] Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 500–517. Springer, Berlin, Heidelberg, August 2014. 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. 2

[RVV24] Seyoon Ragavan, Neekon Vafa, and Vinod Vaikuntanathan. Indistinguishability obfuscation from bilinear maps and LPN variants. In Elette Boyle and Mohammad Mahmoody, editors, *Theory of Cryptography - 22nd International Conference, TCC 2024, Milan, Italy, December 2-6, 2024, Proceedings, Part IV*, volume 15367 of *Lecture Notes in Computer Science*, pages 3–36. Springer, 2024. 1, 4

[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, Cham, August 2022. 3

[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, Cham, May / June 2022. 3

[WW21] Hoeteck Wee and Daniel Wichs. Candidate obfuscation via oblivious LWE sampling. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part III*, volume 12698 of *LNCS*, pages 127–156. Springer, Cham, October 2021. 1

A SMS with Additional Properties

In Appendix A.1, we show how make the SMS construction in [BJSS25] satisfy equivocal hash property. In Appendix A.2, we show the construction in [BJSS25] has decomposable Dec property.

A.1 SMS with Equivocal Hash

As usual, we define $D_{P,\sigma}$ to be the discrete gaussian distribution over P with parameter σ.

Let $\boldsymbol{B}$ be a matrix of size $n \times m$ with columns $\boldsymbol{b}_1 \ldots, \boldsymbol{b}_m$. We define $\mathbf{Vec}(\boldsymbol{B})$ to be the vectorization of the matrix, that is, the vector $(\boldsymbol{b}_1 \ldots, \boldsymbol{b}_m)$

Lemma 6 ([GPV08])**.** *Let $m \geq 2n \log q$ and $\boldsymbol{A} \leftarrow \mathbb{Z}_q^{n \times m}$. For large enough σ we have that $\boldsymbol{Au} \mod q \approx_s \boldsymbol{v}$ where $\boldsymbol{u} \leftarrow D_{\mathbb{Z}^m,\sigma}$ and $\boldsymbol{v} \leftarrow \mathbb{Z}_q^n$.*

The following lemma states that there are matrices statistically close to uniform and for which we can sample low-norm pre-images with the help of a trapdoor [GPV08, MP12].

Lemma 7 ([MP12])**.** *Let m, n, q be natural numbers. There exists a pair of algorithms $(\mathsf{TrapGen}, \mathsf{PreSamp})$ such that:*

- $(\boldsymbol{B}, \mathsf{td}) \leftarrow \mathsf{TrapGen}(n, m, q, \sigma)$ *takes as input $n, m, q \in \mathbb{Z}$ and $\sigma \mathbb{R}$. It outputs a matrix $\boldsymbol{B} \in \mathbb{Z}_q^{n \times m}$ and a trapdoor td.*
- $\boldsymbol{u} \leftarrow \mathsf{PreSamp}(\mathsf{td}, \boldsymbol{B}, \boldsymbol{v})$ *takes as input a trapdoor td, a matrix $\boldsymbol{B}$ and a vector $\boldsymbol{v} \in \mathbb{Z}_q^n$. It outputs $\boldsymbol{u} \in \mathbb{Z}_q^m$.*

These algorithms are such that, for a large enough σ, the following two distributions are statistically indistinguishable:

$$\left\{ (\boldsymbol{B},\boldsymbol{u},\boldsymbol{v}) : \begin{array}{c} \boldsymbol{B} \leftarrow \mathbb{Z}_q^{n\times m} \\ \boldsymbol{u} \leftarrow D_{\mathbb{Z}^m,\sigma} \\ \boldsymbol{v} = \boldsymbol{B}\boldsymbol{u} \mod q \end{array} \right\} \approx_s \left\{ (\boldsymbol{B},\boldsymbol{u},\boldsymbol{v}) : \begin{array}{c} (\boldsymbol{B},\mathsf{td}) \leftarrow \mathsf{TrapGen}(n,m,q,\sigma) \\ \boldsymbol{v} \leftarrow \mathbb{Z}_q^n \\ \boldsymbol{u} \leftarrow \mathsf{PreSamp}(\mathsf{td},\boldsymbol{A},\boldsymbol{v}) \end{array} \right\}.$$

We show that the SMS construction of [BJSS25] is equivocal. The protocol uses a sVOLE scheme [ARS24] as building block. Let n, m, q be as in Lemmas 6 and 7. This sVOLE scheme contains (among other algorithms) two algorithms $\mathsf{sVOLE.Setup}$ and $\mathsf{sVOLE.Hash}$: i) $\mathsf{sVOLE.Setup}$ creates a hashing key $\mathsf{hk}_{\mathsf{sVOLE}}$ composed of two LWE matrices $\boldsymbol{A},\boldsymbol{B} \in \mathbb{Z}_q^{n\times m}$. ii) $\mathsf{sVOLE.Hash}(\mathsf{hk}_{\mathsf{sVOLE}},\boldsymbol{x})$: To hash a vector $\boldsymbol{x} \in \{0,1\}^m$, first sample a vector $\boldsymbol{u} \leftarrow D_{\mathbb{Z}^m,\sigma}$ and compute $\boldsymbol{h} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \mod q$.

First note that by Lemma 6, the hash value $\boldsymbol{h}$ is statistically close to uniform since $\boldsymbol{B}\boldsymbol{u} \mod q$ is statistically close to uniform.

Equivocal sVOLE. We define equivocal sVOLE similarly to Definition 2. We show that sVOLE of [ARS24] sketch above is equivocal.

The $\mathsf{sVOLE.AltSetup}$ is the same as above, except $(\boldsymbol{B},\mathsf{td}) \leftarrow \mathsf{TrapGen}(n,m,q,\sigma)$. Note that the hashing key $(\boldsymbol{A},\boldsymbol{B})$ is statistically indistinguishable from the real one since $\boldsymbol{B}$ is statistically indistinguishable from uniform by Lemma 7.

We now describe the $\mathsf{sVOLE.Equiv}$ algorithm. It takes $(\mathsf{td},\boldsymbol{h},\boldsymbol{x})$ as input, and computes $\boldsymbol{u} \leftarrow \mathsf{PreSamp}(\mathsf{td},\boldsymbol{B},\boldsymbol{h}-\boldsymbol{A}\boldsymbol{x})$. Lemma 7 guarantees that the distribution of $(\boldsymbol{A},\boldsymbol{B},\boldsymbol{x},\boldsymbol{u},\boldsymbol{h})$ is statistically indistinguishable from the real distribution.

Equivocal SMS from equivocal sVOLE. Finally, we observe that the hash algorithm Hash of the SMS protocol of [BJSS25], on input a circuit C, first computes a function over C and then hashes it using the underlying sVOLE hash algorithm. Hence, to equivoke the hash of the SMS protocol, it is enough to equivoke the hash of the underlying sVOLE protocol.

A.2 Decomposable Dec

To show that Dec has the required decomposability property, we recall how Dec works in [BJSS25]. The Dec algorithm calls the decode algorithm of sVOLE protocol [ARS24] to obtain a vector of $\mathbb{Z}_p$ elements, adds a pseudorandom function output to each component, and then rounds it to a vector of bits. To prove decomposability property of Dec, it is sufficient to prove the decomposability of the decode function of sVOLE. In the construction of sVOLE in [ARS24], the size of the trapdoor is $m^{1/3} \cdot \mathsf{poly}(\lambda, d)$. We view the trapdoor as LWE secrets $(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{m^{1/3}})$ where each secret $\boldsymbol{s}_k$ is a $\mathsf{poly}(\lambda)$-sized vector over $\mathbb{Z}_q$. The size of h is $m^{2/3} \cdot \mathsf{poly}(\lambda, d)$. We view the hash as a sequence of vectors $(\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{m^{2/3}})$ where each $\boldsymbol{h}_j$ is a $\mathsf{poly}(\lambda)$ sized vector over $\mathbb{Z}_q$ (where $q = 2^{\lambda^\varepsilon} \cdot p$). To compute the i-th $\mathbb{Z}_p$ element for some $i \in [N]$, we first view i as $(k, j) \in [m^{1/3}] \times [m^{2/3}]$. The i-th $\mathbb{Z}_p$ element is computed as $\lceil \langle \boldsymbol{s}_k, \boldsymbol{h}_j \rangle \rfloor_p$ where $\lceil \cdot \rfloor_p$ is the rounding function that maps $\mathbb{Z}_q$ to $\mathbb{Z}_p$. Thus, the size of the circuit that computes the i-th $\mathbb{Z}_p$ element in sVOLE construction is $m^{2/3} \cdot \mathsf{poly}(\lambda, d)$.

B Improving the Obfuscation Size of avi$\mathcal{O}$

In this section, we sketch how to improve the size of obfuscation in avi$\mathcal{O}$ from $\mathsf{poly}(|\mathcal{C}|, \lambda)$ to $|\mathcal{C}| \cdot \mathsf{poly}(\lambda)$.

Building Blocks. We use the following building blocks:

- A blind garbled circuits scheme [BMR90, BLSV18] ($\mathsf{GCGen}, \mathsf{GCEval}$). Roughly speaking, a blind garbled circuits is just like standard garbled circuits except that the output of the simulator on a uniform

output is pseudorandom. We will assume that the output of GCGen on input (C, x) is $(\widetilde{x}, \widetilde{g}_1, \ldots, \widetilde{g}_N)$ where $\widetilde{x}$ is the garbled input and $\widetilde{g}_i$ for each $i \in [N]$ is the garbled gate entries. Furthermore, given common randomness r, each $\widetilde{g}_i$ can be generated from the description of g_i using a $\mathsf{poly}(\lambda)$-sized circuit. This is the case for the blind garbled circuits construction based on BMR garbling in Brakerski et al. [BLSV18].

- A puncturable pseudorandom function PRF.

Construction. Let $\mathcal{C} : [N] \to \{0,1\}$ be the circuit that is to be obfuscated. Let $g_1, \ldots, g_N$ be the gates in this circuit. Let $(\mathsf{avi}\mathcal{O}.\mathsf{Obf}, \mathsf{avi}\mathcal{O}.\mathsf{Eval})$ be the average-case $\mathsf{i}\mathcal{O}$ scheme with lower efficiency.

- **Obfuscation.**
  1. Sample a random puncturable PRF key K.
  2. For each $i \in [N]$, generate $\hat{g}_i$ as the output of $\mathsf{avi}\mathcal{O}.\mathsf{Obf}$ of a function that takes as input $x \in [N]$, computes $r = \mathsf{PRF}(K, x)$, and outputs $\widetilde{g}_i$ using randomness r.
  3. Let $\hat{x}$ be the output of $\mathsf{avi}\mathcal{O}.\mathsf{Obf}$ of a function that takes as input $x \in [N]$, computes $r = \mathsf{PRF}(K, x)$ and outputs $\widetilde{x}$.
  4. The output of the new obfuscator is given by $(\hat{x}, \hat{g}_1, \ldots, \hat{g}_N)$.
- **Evaluation.**
  1. On input $x \in [N]$ and obfuscation $(\hat{x}, \hat{g}_1, \ldots, \hat{g}_N)$, run each obfuscated program using $\mathsf{avi}\mathcal{O}.\mathsf{Eval}$ on input x to obtain $(\widetilde{x}, \widetilde{g}_1, \ldots, \widetilde{g}_N)$.
  2. Run $\mathsf{GCEval}(\widetilde{x}, \widetilde{g}_1, \ldots, \widetilde{g}_N)$ and outputs whatever it outputs.

Correctness. The correctness of the construction follows directly from the correctness of the underlying $\mathsf{avi}\mathcal{O}$ and the correctness of blind garbled circuits.

Efficiency. Since the size of each circuit that is obfuscated is of size $\mathsf{poly}(\lambda)$, the total size of the obfuscation is $|\mathcal{C}| \cdot \mathsf{poly}(\lambda)$.

Pre-Condition. We show that whenever the precondition holds for the circuit $\mathcal{C}$ to be obfuscated, the pre-condition holds for each one of the individual functions that are being obfuscated in the above construction. We argue this via a hybrid argument.

- $\mathcal{P}_0$: This corresponds to the truth table of all the functions that are obfuscated in the above construction along with the auxiliary information aux. Note that each entry of the joint truth table is given by $(\widetilde{x}, \widetilde{g}_1, \ldots, \widetilde{g}_N)$.
- $\mathcal{P}_1$: In this hybrid, we switch the randomness used for computing the garbling to be uniform instead of computing it as the output of the PRF. This change is indistinguishable from the security of the PRF.
- $\mathcal{P}_2$: In this hybrid, we generate each entry of the truth table using the simulator for the underlying blind garbled circuits. This change is indistinguishable from the simulation security of garbling scheme.
- $\mathcal{P}_3$: In this hybrid, we switch the truth table of function $\mathcal{C}$ used in generating the simulated garbled circuits to be uniform. We compute the auxiliary input by first evaluating the simulated garbled circuits to obtain the truth table of the circuit and then use Sim (from the pre-condition) on this truth table to generate aux. This change is indistinguishable from the precondition of the circuit $\mathcal{C}$.
- $\mathcal{P}_4$: In this hybrid, we switch each one of the simulated garbled circuits to be uniform. It follows from the blindness of the garbled circuits scheme that this hybrid is computationally indistinguishable to the previous hybrid. This corresponds to the RHS of the precondition of the above construction.

Post condition. Let $\mathcal{C}$ and $\mathcal{C}'$ be two functionally equivalent circuits for which the precondition holds. We show that obfuscations of $\mathcal{C}$ and $\mathcal{C}'$ are computationally indistinguishable. Towards this, we consider an intermediate circuit $\mathcal{C}_i$ which applies $\mathcal{C}'$ for all inputs $x \leq i$ and for $x > i$, it applies $\mathcal{C}$ for some $i \in [N] \cup \{0\}$. Note that $\mathcal{C}_0$ is identical to $\mathcal{C}$ and $\mathcal{C}_N$ is identical to $\mathcal{C}'$. We prove that for any $i \in [N] \cup \{0\}$, obfuscation of $\mathcal{C}_i$ is computationally indistinguishable to obfuscation of $\mathcal{C}_{i+1}$. We prove this via a hybrid argument.

- $\mathcal{H}_0$: This corresponds to an obfuscation of $\mathcal{C}_i$ along with auxiliary input aux.
- $\mathcal{H}_1$: In this hybrid, for each of the $N+1$ obfuscations in the obfuscation of $\mathcal{C}_i$, we make the following changes:
  - We puncture the PRF key K at input $i+1$.
  - We hardwire the output of the function at $i+1$ and output this when the input is $i+1$.

  Note that the resulting function is computing the exact same function as $\mathcal{C}_i$. Furthermore, via an identical argument as done earlier, we can show that the precondition holds for each of $N+1$ functions that are obfuscated in $\mathcal{C}_i$. Thus, the post condition holds and $\mathcal{H}_1$ is computationally indistinguishable to $\mathcal{H}_0$.
- $\mathcal{H}_2$: In this hybrid, we generate the randomness used in computing the hardwired garbled circuit and garbled input corresponding to input $i+1$ in the obfuscations to be truly uniform instead of computing it as the output of the PRF on input $i+1$. It follows from the puncturable PRF security that $\mathcal{H}_1$ and $\mathcal{H}_2$ are computationally indistinguishable.
- $\mathcal{H}_3$: In this hybrid, we generate the hardwired garbled circuit and the garbled input for input $i+1$ using the underlying simultator for the garbling scheme. This hybrid is computationally indistinguishable to $\mathcal{H}_2$ from the simulation security of blind garbling scheme.
- $\mathcal{H}_4$: In this hybrid, we give the output of $\mathcal{C}'$ on $(i+1)$ to the simulator for the garbling scheme instead of giving the output of $\mathcal{C}$ as in the previous hybrid. This hybrid is identical to the previous hybrid from the fact that $\mathcal{C}$ and $\mathcal{C}'$ are functionally equivalent.
- $\mathcal{H}_5$: In this hybrid, we generate the hardwired garbled input and garbled circuit using GCGen on input $(\mathcal{C}', i+1)$. It follows from the simulation security of garbling scheme that this hybrid is computationally indistinguishable to the previous hybrid.
- $\mathcal{H}_6$: In this hybrid, we generate the randomness used in generating the hardwired garbled circuit and garbled input using the output of the PRF on input $i+1$. It follows from the security of the puncturable PRF that this hybrid is computationally indistinguishable to the previous hybrid.
- $\mathcal{H}_7$: This corresponds to an obfuscation of $\mathcal{C}_{i+1}$. Note that the obfuscations in $\mathcal{H}_6$ and $\mathcal{H}_7$ are computing the exact same function. Furthermore, via an identical argument as done earlier, we can show that the precondition holds for each of the $N+1$ functions in the obfuscation of $\mathcal{C}_{i+1}$. Thus, the post condition holds and $\mathcal{H}_6$ is computationally indistinguishable to $\mathcal{H}_7$.