On White-Box Learning and Public-Key Encryption

Yanyi Liu
Cornell Tech, New York, NY, USA

Noam Mazor
Tel Aviv University, Israel

Rafael Pass
Cornell Tech, New York, NY, USA
Technion, Haifa, Israel
Tel Aviv University, Israel

Abstract

We consider a generalization of the Learning With Error problem, referred to as the *white-box learning problem*: You are given the code of a sampler that with high probability produces samples of the form $y, f(y) + \epsilon$ where ϵ is small, and f is computable in polynomial-size, and the computational task consist of outputting a polynomial-size circuit C that with probability, say, 1/3 over a new sample y' according to the same distributions, approximates $f(y')$ (i.e., $|C(y') - f(y')|$ is small). This problem can be thought of as a generalizing of the Learning with Error Problem (LWE) from linear functions f to polynomial-size computable functions.

We demonstrate that worst-case hardness of the white-box learning problem, conditioned on the instances satisfying a notion of computational shallowness (a concept from the study of Kolmogorov complexity) not only *suffices* to get public-key encryption, but is also *necessary*; as such, this yields the first problem whose worst-case hardness characterizes the existence of public-key encryption. Additionally, our results highlights to what extent LWE "overshoots" the task of public-key encryption.

We complement these results by noting that worst-case hardness of the same problem, but restricting the learner to only get black-box access to the sampler, characterizes one-way functions.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases Public-Key Encryption, White-Box Learning

Digital Object Identifier 10.4230/LIPIcs.ITCS.2025.73

Related Version *Full Version*: `https://eprint.iacr.org/2024/1931` [24]

Funding *Yanyi Liu*: Research partly supported by NSF CNS-2149305.
Noam Mazor: Research partly supported by NSF CNS-2149305, AFOSR Award FA9550-23-1-0312 and AFOSR Award FA9550-23-1-0387 and ISF Award 2338/23.
Rafael Pass: Supported in part by NSF Award CNS 2149305, AFOSR Award FA9550-23-1-0387, AFOSR Award FA9550-23-1-0312 and AFOSR Award FA9550-24-1-0267 and ISF Award 2338/23. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government, or the AFOSR.

1 Introduction

Public-Key Encryption (PKE) [12, 31] is one of the most central cryptographic primitives enabling secure communication on the Internet: it is the primitive that enables entities that have not physically met to engage in confidential conversations and collaborations.

In contrast to private-key primitives, such as symmetric-key encryption and digital signatures, that can be securely build from the minimal primitive of one-way functions (and for which many candidate problems are known), we only know of a handful of candidate hard problems from which public-key encryption can be constructed. More specifically, these include (a) *number-theory problems* based on either factoring [31, 29] or discrete logarithms [12, 13], (b) coding-theory based problems [27], (c) lattice problems as finding shortest/longest vectors in lattices [1, 30, 8], and (d) noisy linear-algebra based problems [2, 5]. Out of these, the number-theory based problems can be efficiently solved by quantum algorithms [32], and the coding-theory, lattice and noisy linear algebra problems are all very related – in essence, they can all be viewed as different instances of solving noisy systems of *linear* equations (on either particular natural distributions, or even in the worst-case, when restricting attention to systems of equations satisfying the condition that appropriate solutions exist.[1]

The main purpose of this paper is to provide an assumption that generalizes all these assumptions (i.e., is implied by all of them), yet suffices for obtaining secure PKE. Indeed, the main result of this paper is that hardness of a notion of *white-box learning* achieves this goal.

White-box Learning

Perhaps the most common noisy linear algebra-based assumption is the hardness of the *Learning With Error (LWE)* problem [30], which, in essence, stipulates the hardness of recovering a vector $\mathbf{x}$ given $\mathbf{A}, \mathbf{A}\mathbf{x} + \mathbf{e}$, where $\mathbf{A}$ is a matrix, $\mathbf{e}$ is some "small" noise vector and all arithmetic is modulo some prime p. In more detail, we typically require an stronger condition: not only that is hard to recover x, but also that it is hard to compute a value "close" to $\mathbf{a}^T\mathbf{x}$ for a random vector $\mathbf{a}$. In other words, we can think of $\mathbf{x}$ as the description of a function $f_{\mathbf{x}}(\mathbf{a}) = \mathbf{a}^T\mathbf{x}$ that we are trying to *improperly* approximately learn given noisy samples – thus the name "learning with error".

In fact, there is also a different way to think about the LWE problem that will be useful for our purposes (which follows from the construction of Regev's PKE [30]): We are given the code of a sampler P_x that enables providing samples of the form $(\mathbf{a}, f_{\mathbf{x}}(\mathbf{a}) + \mathbf{e})$, and the goal is to approximate $f_{\mathbf{x}}(\mathbf{a}')$ on a new fresh sample $\mathbf{a}'$ according to the same distribution as $\mathbf{a}$.[2] We refer to this alternative way of thinking of LWE as an instance of *(improper) white-box learning*, where, more generally, we are given the code of a sampler P that generates noisy samples of the form $(y, f(y) + \epsilon)$ and the goal is to approximate $f(y')$ for a fresh sample y' according to the same distribution, using a polynomial-size circuit. In essence, this problem is generalizing the LWE problem from (improperly) learning *linear functions* from noisy samples, to (improperly) *learning a polynomial-size circuit* from noisy samples, and given white-box access to the sampler[3]; the white-box access feature can be viewed as a generalization of Valiant classic PAC-learning model [35] to a setting where the learner not only gets random samples, but also gets the code of the sampler.

In more detail, given a sampler circuit P that samples "labeled instances" (y, z) (where we think of $z \in \mathbb{N}$ as a, perhaps, noisy label for y), let $\mathsf{Comp}_{\epsilon}^{\Delta}(P)$ denote the set of circuits f that with probability $1 - \epsilon$ over $(y, z) \leftarrow P$ satisfy the property that $|z - f(y)| \leq \Delta$ (when interpreting both z and $f(z)$ as elements in $\mathbb{N}$). For a function $\Delta \colon \{1\}^* \to \mathbb{N}$, let Δ-$\mathsf{WBLearn}$ denote the following learning problem:

[1] In more detail, we require worst-case hardness to hold when conditioning on instances that define a lattice where the shortest vector is long compared to amount of noise.

[2] The reason for this is that given $\mathbf{A}, \mathbf{A}\mathbf{x} + \mathbf{e}$, we can generate noisy samples of $f_{\mathbf{x}}(\mathbf{a}')$ by taking linear combinations. See [30] and the full version of this paper for more details.

[3] As we will discuss in more detail later, it is also more general than the LWE problem in the aspect the LWE *sampler* has a particular form, but we here may consider more general classes of samplers.

- **Input:** Circuit $P \in \{0,1\}^n$, with the promise that there exists a circuit $\widehat{C} \in \{0,1\}^n$ such that $\widehat{C} \in \mathsf{Comp}_{n^{-10}}^{\Delta(1^n)/n^{10}}(P)$.
- **Output:** Circuit $C \in \mathsf{Comp}_{1/3}^{\Delta(1^n)}(P)$.[4]

In other words, we are given a sampler that with very high $(1-n^{-10})$ probability outputs labelled samples where the label is very close $(\Delta(1^n)/n^{10})$ to being correct, and the goal is to, given the code of the sampler P, simply find a circuit C that with probability $2/3$ approximates the label within $\Delta(1^n)$ (i.e., with a factor n^{10} higher error). We use $\mathsf{ExactWBLearn}$ to denote Δ-$\mathsf{WBLearn}$ when $\Delta(1^n) = 0$.

Hardness of Learning v.s. (Public-Key) Cryptography

Roughly speaking, the main result of this paper will be to show that under certain restrictions on samplers P (which come from the study of *time-bounded Kolmogorov complexity* [23, 34, 11, 22, 17, 33] and in particular the notion of *computational depth* [4]), this generalization of the LWE problem not only suffices to realize a PKE, but will also be necessary. In more detail, worst-case hardness of Δ-$\mathsf{WBLearn}$ under these conditions will *characterize* the existence of public-key encryption. As such, our results yield insight on the extent with which LWE "overshoots" the task of public-key encryption.

We highlight that we are not the first ones to consider connections between learning-theory and cryptography; however, as far as we know, all earlier connections were between the hardness of learning and *private-key cryptography* (i.e., the notion of one-way functions), as opposed to public-key cryptography. Indeed, classic results from [21, 7] demonstrate the equivalence of the hardness of a notion of *average-case* PAC-learning of polynomial-size circuits (i.e., black-box learning) and one-way functions.[5] In contrast, our focus here is on PKE, and instead of considering black-box learning, we consider white-box learning. An additional difference is that we consider worst-case hardness, as opposed to average-case hardness, of the learning theory problem. As was recently shown in [18], this issue can be overcome (in the context of characterizing one-way functions) through the use of a (alternative) notion of computational depth (more details on the relationship with this work below). To put out result in context, we additionaly show that exactly the same problem that we demonstrate characterizes PKE, also characterizes one-way functions once modified to only provide the learner black-box access to the sampler.

1.1 Our Results

Towards explaining our results in more detail, let us first recall the notion of *computational depth* [4]. Let the *t-computational depth* of x, denoted $cd^t(x)$, be defined as $cd^t(x) = \mathrm{K}^t(x) - \mathrm{K}(x)$ where $\mathrm{K}(x)$ denotes the Kolmogorov complexity of x and K^t denotes the t-bounded Kolmogorov complexity of x. That is, $cd^t(x)$ measures how much more x could be compressed if the decompression algorithm may be computationally unbounded, as opposed to it running in time bounded by $t(|x|)$.

We will focus on instances P of the Δ-$\mathsf{WBLearn}$ problem having the property that $(P, \widehat{C})$ is "computationally shallow", where $\widehat{C}$ is a circuit that agrees with P with high probability: let Δ-$\mathsf{WBLearn}|_{\mathsf{CS}^t}$ denote Δ-$\mathsf{WBLearn}$ with the additional promise that $cd^t(P, \widehat{C}) \le 2\log n$.[6]

[4] We remark that for efficient algorithms, outputting a circuit that approximates a function f is essentially equivalent to approximating the value of $f(y')$ on a random sample y'. Indeed, the circuit implementation of an algorithm for the latter task, is a valid output for the first task.

[5] And the results of [20] can be thought of as characterizing one-way functions through a different type of learning.

[6] We note that there is nothing special about the constant 2; it can be anything that is strictly larger

Characterizing PKE through Hardness of White-Box Learning

Our main results is that the hardness of this problem *characterizes* the existence of public-key encryption:

▶ **Theorem 1.** *Let $\epsilon > 0$ be a constant and let $\Delta \colon \{1\}^* \to \mathbb{N}$ be an efficiently computable function such that $\Delta(1^n) \leq 2^{n^{(1-\epsilon)}}$, and $t \colon \mathbb{N} \to \mathbb{N}$ be polynomial such that $t(n) \geq n^{1+\epsilon}$. Then, following are equivalent:*

- *PKE exists.*
- $\Delta\text{-}\mathsf{WBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$.

Computational depth is typically thought of as measure of "unnaturality" of strings: strings with low computational depth are considered "natural" and those with high computational depth are considered "unnatural".[7] Given this interpretation, our characterization of public key encryption is thus in terms of the worst-case hardness of white-box learning for "natural" instances.

As far as we know, this thus yields the first problem whose worst-case hardness not only suffices for public-key encryption (such as e.g., [30]) but also is *necessary*.

On the Use of Computational Depth

We note that Antunes and Fortnow [3] elegantly used computational depth to connect worst-case hardness of a problem when restricting attention to elements with small computational depth and *errorless* average-case hardness on sampleable distributions; errorless hardness, however, is not sufficient for cryptographic applications. Nevertheless, inspired by the work of [3], worst-case hardness conditioned on instances with small computational depth was used in [26] (and independently using a variant of this notion in [18]) to characterize one-way functions; additionally, an (interactive) variant of such a notion was also implicitly used in [6] to characterize key exchange protocols. Our techniques are similar to those employed in [26, 6] but instead of applying them to study the hardness of a time-bounded Kolmogorov complexity problem (following [25]), we here instead apply them to study a learning theory problem (namely, white-box learning).

We note that learning theory problems conditioned on small computational depth were recently used in [18] to characterize one-way functions, but our techniques here are more similar to [26, 6]. In particular, [18] does not actually use the standard notion of computational depth but instead define a new alternative variant; in contrast, we here rely on just the standard notion.

Relating Exact and Approximate White-Box Learning

Note that in Theorem 1, the equivalence hold for any (sufficiently small) choice of Δ, as such we directly get as a corollary the equivalence of Exact and Approximate White-box Learning:

than 1.

[7] The reason why low computational depth captures "natural string" is as follows: random strings are known to have low computational depth; furthermore, known results (c.f. slow growth laws [18]) show (at least under derandomization assumptions) that one needs to have a long running time to even find a string with high computational depth. So strings with high computational depth are rare and "hard to find", which is why they can be thought of as "unnatural".

▶ **Corollary 2.** *Let $\epsilon > 0$ be a constant and let $\Delta\colon \{1\}^* \to \mathbb{N}$ be any efficiently computable function such that $\Delta(1^n) \leq 2^{n^{(1-\epsilon)}}$, and let $t\colon \mathbb{N} \to \mathbb{N}$ be any polynomial such that $t(n) \geq n^{1+\epsilon}$. Then the following are equivalent:*

- $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$
- $\Delta\text{-}\mathsf{WBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$

Bounded-Degree Learning and LWE

While in the Δ-$\mathsf{WBLearn}$ problem, we allow the function we are trying to learn to be any polynomial-size circuit, we may also consider a restricted version of the problem, denoted $\Delta\text{-}\mathsf{WBLearn}^d_q$, where we restrict attention to functions that can be computed by a degree d polynomial, and we assume that arithmetic is now over $\mathbb{Z}_q$.

We first remark that our main theorem can next be generalized (basically using padding) to show that it suffices to use learning of degree-n^ϵ polynomials to characterize PKE:

▶ **Theorem 3.** *Let $\epsilon > 0$ be a constant and let $\Delta, q\colon \{1\}^* \to \mathbb{N}$ be efficiently computable functions with $\Delta(1^n) \leq q(1^n)/4$ and $q(1^n) \leq 2^{n^{(1-\epsilon)}}$, and $t\colon \mathbb{N} \to \mathbb{N}$ be polynomial such that $t(n) \geq n^{1+\epsilon}$. The following are equivalent:*

- *PKE exists.*
- $\Delta\text{-}\mathsf{WBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$.
- $\Delta\text{-}\mathsf{WBLearn}^n_q|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$.
- $\Delta\text{-}\mathsf{WBLearn}^{n^\epsilon}_q|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$.

Additionally, as informally discussed above, we note that the hardness of the LWE problem, implies hardness of the $\Delta\text{-}\mathsf{WBLearn}^1_q$ problem (i.e., white-box learning of *linear* functions.)

▶ **Lemma 4** (Informal). *Assuming the hardness of LWE, there exists a polynomial t such that $\Delta\text{-}\mathsf{WBLearn}^1_q|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$ where $\Delta = q/4$.*

Lemma 4 thus shows that white-box learning of linear functions is at least as weak an assumption as LWE. At first sight, one would expect that a converse result may also hold due to known worst-case to average-case reductions for the LWE problem [30, 28, 9] and thus that LWE is equivalent to white-box learning of linear functions. However, the problem with proving the converse direction is that the known worst-case to average-case reductions for LWE only work when the LWE instance defines a lattice where the shortest vector is long compared to amount of noise. Instances sampled from P may not necessarily satisfy this promise, and thus is it not clear how to use an LWE oracle to generally solving white-box learning of linear functions. Thus, it would seem that hardness even of just $\Delta\text{-}\mathsf{WBLearn}^1_q|_{\mathsf{CS}^t}$ is seemingly a weaker (and therefore more general) assumption than LWE. (Of course, it may be that a stronger worst-case to average-case reduction can be established for the LWE problem, in which case equivalence would hold.)

We finally investigate what happens in the regime of "intermediate-degree" polynomials. We remark that using standard linearization techniques, the constant-degree problem is equivalent to the case of degree 1:

▶ **Lemma 5.** *For every constant $d \in \mathbb{N}$, and functions $t, q, \Delta\colon \mathbb{N} \to \mathbb{N}$, there exist $t', q', \Delta'\colon \mathbb{N} \to \mathbb{N}$ such that*

$$\Delta\text{-}\mathsf{WBLearn}^d_q|_{\mathsf{CS}^t} \leq_p \Delta'\text{-}\mathsf{WBLearn}^1_{q'}|_{\mathsf{CS}^{t'}},$$

and

$$\Delta\text{-}\mathsf{WBLearn}^d_q \leq_p \Delta'\text{-}\mathsf{WBLearn}^1_{q'}.$$

Black-box Learning

Finally, to put our results in context, we consider the standard PAC learning model [35] where the learner only get access to samples: Let Δ-BBLearn denote identically the same problem as WBLearn with the exception that the learner gets oracle access (i.e., black-box access) to the sampler (as opposed to white-box access to the sampler). This notion is equivalent to the notion of improper Δ-approximate PAC learning for polynomial-size circuits (and when $\Delta = 0$ to simply improper PAC learning).

As before, let $\Delta\text{-BBLearn}|_{\mathsf{CS}^t}$ denote the problem Δ-BBLearn with the additional promise that $cd^t(P, \widehat{C}) \leq 2 \log n$, and let ExactBBLearn and $\text{ExactBBLearn}|_{\mathsf{CS}^t}$ to denote Δ-BBLearn and $\Delta\text{-BBLearn}|_{\mathsf{CS}^t}$ when $\Delta(1^n) = 0$.

The following theorem can be viewed as the worst-case analog of the classic result of [21, 7] characterizing one-way functions through the hardness of *average-case PAC learning.*

▶ **Theorem 6.** *Let $\epsilon > 0$ be a constant and let $\Delta\colon \{1\}^* \to \mathbb{N}$ be any efficiently computable function such that $\Delta(1^n) \leq 2^{n^{(1-\epsilon)}}$, and let $t\colon \mathbb{N} \to \mathbb{N}$ be any polynomial such that $t(n) \geq n^{1+\epsilon}$. Then the following are equivalent:*

- *One-way function exists*
- $\Delta\text{-BBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$

As mentioned, [18] also recently obtained a worst-case characterization of one-way functions through a learning problem, and using a notion of computational depth. The problems, however, are somewhat different. As opposed to [18], we here consider the standard PAC learning problem (whereas they consider a more general learning problem), and condition on the standard notion of low computational depth (whereas they condition on a new notion of low computational depth that they introduce).[8]

We remark that our Theorem 6 differs from [21, 7] not only in the worst-case condition, but also generalizes those results in the sense that we handle the hardness of Δ-approximate learning for any Δ. As a consequence, we again get an equivalence of approximate and exact black-box learning:

▶ **Corollary 7.** *Let $\epsilon > 0$ be a constant and let $\Delta\colon \{1\}^* \to \mathbb{N}$ be any efficiently computable function such that $\Delta(1^n) \leq 2^{n^{(1-\epsilon)}}$, and let $t\colon \mathbb{N} \to \mathbb{N}$ be any polynomial such that $t(n) \geq n^{1+\epsilon}$. Then the following are equivalent:*

- $\text{ExactBBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$
- $\Delta\text{-BBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$

Open Problems

We leave as an intriguing open problem the question of whether white-box learning of polynomial, or even logarithmic-degree polynomials, also can be collapsed down to the case of constant-degree functions (and thus to linear functions); if this were possible, it would show that PKE, in essence, inherently requires the structure of the LWE problem.

Additionally, as discussed above, even when just restricting attention to learning linear functions, our learning problem generalizes LWE. It would appear that despite this, cryptographic applications of LWE (e.g., to obtain fully homomorphic encryption [14, 10]) nevertheless may still be possible from this generalized version; we leave an exploration of this for future work.

[8] Of course, on a conceptual level, these results are similar; the key point we are trying to make here is that exactly the same learning problem characterizes either one-way functions or PKE, depending on whether the learner gets black-box or white-box access to the sampler.

Finally, it is an intriguing open problem to relate black-box and white-box learning. By our results, doing so is equivalent to relating public-key and secret-key encryption. We note that relating black-box and white-box learning is interesting even just for the case of linear functions (which by our results is equivalent to $O(1)$-degree polynomials). Indeed, Regev's construction of a PKE [30] can be thought of a reduction from black-box learning to white-box learning of linear functions for a *specific* distribution; it is possible that a similar reduction may be applicable more generally.

1.2 Proof Overview

We here provide a detailed proof overview for the proof of Theorem 1. For simplicity, we will show the equivalence between PKE and the worst-case hardness of the exact version, $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$. We start with the construction of PKE based on the hardness of $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$.

Weak PKE

First, we use the well-known fact that a PKE is simply a two-rounds key-agreement protocol. Moreover, by the Key-agreement Amplification Theorem of Holenstein [19] and an application of the Goldreich-Levin theorem [15], to obtain (full-fledged) PKE, it suffices to obtain a weak form of two-rounds key-agreemnt, which we simply refer to as *Weak PKE* defined as follows: There exist some $\epsilon = 1/\text{poly}$ such that *agreement* between A and B happens with probability $1 - \epsilon$. *Security* requires that Eve cannot guess the key (output by Alice) with probability better than $1 - 20\epsilon$.

The Weak PKE protocol

We will next define the weak PKE protocol. We note that this construction resembles the universal key-agreement construction from [16], but with some crucial difference that enable our security proof. The parties A and B on input n perform the following steps:

- *Sample random program:* A samples a random length $\lambda \in [2n]$, and a random program Π of length λ.
- *Run random program:* Next, A runs the program Π for at most $t(n)$ steps to get an output, and interprets the output as a pair of circuits $P\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^\ell$ and $C\colon \{0,1\}^k \to \{0,1\}^\ell$. (Think of P as the sampler for a white-box learning instance, and of C as a potential solution to the problem.) If the output of Π is not a valid encoding of such a pair, A sets $P = P_0$ and $C = C_0$ for two fixed circuits P_0, C_0 that always output 0.
- *Agreement estimation:* A estimates the "agreement probability" of P and C (i.e., checking whether C indeed is a good solution): it samples n^{20} random inputs $w_1, \dots, w_{n^{20}}$ for P, and computes $(x_i, s_i) = P(w_i)$. It then lets $\widehat{\alpha} = \Pr_{i \leftarrow [n^{20}]}[C(x_i) = s_i]$. If $\widehat{\alpha} \leq 1 - n^{-9}$, A reset the pair $(P, C) = (P_0, C_0)$.
- *First message:* A sends (the "sampler") P to B.
- *Second message:* B applies P on a random input w, and computes $(x, s) = P(w)$. It then sends x to A.
- *Outputs:* A outputs $C(x)$ and Bob outputs s.

Agreement

We claim that with probability $1 - 1/n^8$, Alice and Bob will agree (i.e., the final outputs are the same). Note that if $(P, C) = (P_0, C_0)$, Alice and Bob always agree. Moreover, let $\alpha = \Pr_{(x,s) \leftarrow P(U_r)}[C(x) = s]$. Then, the probability of Alice and Bob to agree given that

Alice uses (P, C) as the circuits in the protocol, is exactly α. We observe that by the Chernoff bound, the probability that $\widehat{\alpha}$ is far from α is small, and thus Alice uses (P, C) only when α is larger than $1 - n^{-8}$.

Security

We claim that Eve that can guess s with probability $1 - n^{-7}$ can be used to solve $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$. In more detail, consider such Eve, and let P be an input for $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$. The idea is to construct an algorithm Learner, that on input P outputs a circuit C, such that $C(x)$ simply simulates Eve on the messages P and x. C then outputs Eve's output. For simplicity, in the following we assume that Eve (and thus Learner) is deterministic. (We note that this is a *non-black-box* reduction: We are using the code of Eve to generate C – in particular, we are including the code of Eve into this circuit.[9])

Let $\mathbf{P}$ be a random variable distributed according to the same distribution of the first message in the above protocol. By assumption, we have that

$$\Pr[C' = \mathsf{Learner}(\mathbf{P}); (x, s) \leftarrow \mathbf{P}(U_r); C'(x) = s] \geq 1 - n^{-7}.$$

It follows by a simple averaging argument that

$$\Pr_{\mathbf{P}, C'=\mathsf{Learner}(\mathbf{P})}[\Pr_w[(x, s) \leftarrow \mathbf{P}(w); C'(x) = s] \geq 2/3] \geq 1 - 3n^{-7}.$$

Namely, Learner solves ExactWBLearn with probability at least $1 - 3n^{-7}$ over the distribution of $\mathbf{P}$. We next use ideas from [26, 6] to show this implies that Learner solves $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$ in the worst case.

Indeed, assume that Learner fails on some instance P of $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$. By the promise of the problem, there exists a circuit $\widehat{C}$ such that $cd^t(P, \widehat{C}) \leq 2 \log n$, and $\widehat{C}$ agrees with P with probability at least $1 - n^{-10}$. Let $\ell = \mathrm{K}^t(P, \widehat{C})$. Our goal is to show that $\mathrm{K}(P, \widehat{C}) < \ell - 2 \log n$, which is a contradiction.

Toward this goal, let Let $\mathcal{S}_\ell$ be the set of all pairs of circuits $(P', \widehat{C'})$ with $\mathrm{K}^t(P', \widehat{C'}) = \ell$ that agree with probability at least $1 - n^{-10}$, and on which Learner fails, so that $(P, \widehat{C}) \in \mathcal{S}_\ell$. Fix $(P', \widehat{C'}) \in \mathcal{S}_\ell$, and let Π be the length ℓ program that outputs $(P', \widehat{C'})$ in time t. Observe that the probability of A to sample $(P', \widehat{C'})$ in the first step of the above protocol is at least its probability to sample the program Π, which is $1/2n \cdot 2^{-\ell}$. Since $(P', \widehat{C'})$ agree with high probability, the equality test it the third step of the protocol will pass with high probability, and thus A will send P' to B with probability at least $1/4n \cdot 2^{-\ell}$. In other words,

$$\Pr[\mathbf{P} = P'] \geq 1/4n \cdot 2^{-\ell}$$

for every $(P', \widehat{C'}) \in \mathcal{S}_\ell$, and thus

$$\Pr[(\mathbf{P}, \cdot) \in \mathcal{S}_\ell] \geq |\mathcal{S}_\ell| \cdot 1/4n \cdot 2^{-\ell}$$

(here we say that $(P, \cdot) \in \mathcal{S}_\ell$ if there is some C such that $(P, C) \in \mathcal{S}_\ell$).

On the other hand, by definition of $\mathcal{S}_\ell$, Learner fails on every $(P', \widehat{C'}) \in \mathcal{S}_\ell$. Since Learner fails with probability at most $3n^{-7}$ on $\mathbf{P}$, it must holds that

$$3n^{-7} \geq \Pr[(\mathbf{P}, \cdot) \in \mathcal{S}_\ell].$$

[9] This particular non-black usage of Eve is not inherent. We could have considered a different formalization of the learning theory problem which simply requires the attacker to succeed on a randomly sampled instance. Subsequent parts of the argument, however, will use non-black-box access to Eve more inherently.

Combining the above, we get that

$$|\mathcal{S}_\ell| \le 12n^{-6} \cdot 2^\ell.$$

We can now use the bound on $\mathcal{S}_\ell$ to bound the Kolmogorov complexity of $(P, \widehat{C})$. To describe $(P, \widehat{C})$, it is enough to describe the set $\mathcal{S}_\ell$, and the index of the pair $(P, \widehat{C})$ in this set. That is,

$$\mathrm{K}(P, \widehat{C}) \le \mathrm{K}(\mathcal{S}_\ell) + \log|\mathcal{S}_\ell| + O(\log\log n) \le \mathrm{K}(\mathcal{S}_\ell) + \ell - 6\log n + O(\log\log n).$$

We conclude the proof with the observation that to describe $\mathcal{S}_\ell$ it is enough to describe n (which can be done using $\log n + O(\log\log n)$ bits), ℓ ($\log n$ bits) and Eve (that can be described with constant many bits[10]). Thus, $\mathrm{K}(\mathcal{S}_\ell) \le 3\log n$, and we get that

$$\mathrm{K}(P, \widehat{C}) < \ell - 2\log n,$$

as we wanted to show.

Hardness of $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$ from PKE

We next show that the existence of PKE implies the hardness of $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$. We now sketch the proof. Let Gen be the algorithm the generate pair (pk, sk) of public and secret key, and let $\mathsf{Enc}, \mathsf{Dec}$ be the encryption and decryption protocols. For a random pair of keys $(pk, sk) \leftarrow \mathsf{Gen}(r)$, we construct two circuits, $P(w, s) = (x = \mathsf{Enc}_{pk}(s; w), s)$, and $\widehat{C}(x) = \mathsf{Dec}_{sk}(x)$.

By the security of the PKE scheme, it follows that with high probability over the randomness of Gen, it is hard to learn the function of C, as the circuit P only uses the public key, and the function C computes is the decryption of a random encryption. This already implies that $\mathsf{ExactWBLearn}$ is hard, but we still need to show that P is inside the promise of the problem $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$.

It follows by the correctness of the PKE scheme that $\widehat{C}$ computes the function that is sampled by P. Thus, to be in the promise of $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$, we only need that $(P, \widehat{C})$ has small computational depth. This, however, is not necessarily the case. To solve this, we simply pad $\widehat{C}$ with the randomness used by Gen. That is, we let $\widehat{C'}$ be a functionally equivalent circuit to $\widehat{C}$, with r encoded to it, where r is such that $\mathsf{Gen}(r) = (pk, sk)$. It follows that when t is large enough, $\mathrm{K}^t(P, \widehat{C'}) \le |r| + O(1)$ (as we can describe them by simply describing the randomness and the algorithm Gen). On the other hand, $\mathrm{K}(P, \widehat{C'}) \ge \mathrm{K}(r)$ (since r can be obtained from $\widehat{C'}$), which is at least $|r| - O(1)$ with high probability. Together, we conclude that with high probability over the randomness of Gen, the circuits $(P, \widehat{C'})$ are in the promise of $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$.

Comparison with [6]

We remark that at a high-level, our construction of a two-message key-agreement protocol shares many similaries with the key-agreement protocol developing in [6], relying on the hardness of an interactive notion of time-bounded Kolmogorov complexity, conditionned on an analog of computational shallowness. They central difference is that the protocol in [6] requires at least 3 rounds due to the use of an equility check protocol to determine whether

[10] This non-black-box usage of Eve (which is taken from [26, 6]) is seemingly inherent to our proof technique. Note that we are here relying on the fact that Eve is a uniform algorithm, but as we discuss in the formal section, the argument can be extended to work also in the non-uniform setting.

Alice and Bob managed to agree on a key. In contrast, our protocol does not rely on such an equality check step (and thus it can be executed in only two rounds, which is crucial to get PKE); indeed, we replaced the equality protocol with a step where Alice on her own can determine whether the message she sends will enable agreement to happen. Of course, the reason why we can do this is that we are reducing security from a different problem.

Other features of the protocol are quite similar; this is because we also rely on the worst-case hardness of a problem "conditioning on computationally shallow instances". Indeed, as described above, our security proof shares many features with those of [25, 6].

Comparison with Universal Constructions

Universal constructions of PKE are known (see [16]); that is, constructions having the property that they are secure if (any) PKE exist. We emphasize that while the details of the protocol from [16] are somewhat different, the "agreement estimation" step performed there is very similar to what we do. Furthermore, we can interpret our protocol as an alternative (variant) universal PKE protocol in which Alice chooses a random (key-generation) program Gen and executes it to get an encryption scheme Enc_{pk} and description scheme Dec_{sk}, with the keys pk, sk hardcoded to the scheme.[11] Alice then estimates the agreement probability of the encryption, and if it is high enough, she sends the encryption scheme as the public-key, and uses the decryption scheme as the private-key. If PKE exists, with a noticeable probability over the choice of Gen, this scheme will be secure. We emphasize that since we base the security of our protocol (and thus also the above universal one) on the hardness of the white-box learning problem, it enables an approach for measuring the concrete security of the protocol by relating it to the security of the learning theory problem.

2 Preliminaries

2.1 Notations

All logarithms are taken in base 2. We use calligraphic letters to denote sets and distributions, bold uppercase for random variables, and lowercase for values and functions. Let poly stand for the set of all polynomials. Let PPT stand for probabilistic poly-time, and n.u.-poly-time stand for non-uniform poly-time. An n.u.-poly-time algorithm A is equipped with a (fixed) poly-size advice string set $\{z_n\}_{n\in\mathbb{N}}$ (that we typically omit from the notation), and we let A_n stand for A equipped with the advice z_n (used for inputs of length n). For a randomized algorithm A, we denote by $\mathsf{A}(\cdot; r)$ the algorithm A with fixed randomness $r \in \{0,1\}^*$. Let neg stand for a negligible function. Given a vector $v \in \Sigma^n$, let v_i denote its i^{th} entry, let $v_{<i} = (v_1, \ldots, v_{i-1})$ and $v_{\leq i} = (v_1, \ldots, v_i)$. Similarly, for a set $\mathcal{I} \subseteq [n]$, let $v_{\mathcal{I}}$ be the ordered sequence $(v_i)_{i\in\mathcal{I}}$. For $x, y \in \{0,1\}^*$, we use $x||y$ to denote the concatenation of x and y. For a set $\mathcal{S} \subseteq \{0,1\}^*$, we use $\mathcal{S}||y$ to denote the set $\{x||y \colon x \in \mathcal{S}\}$.

2.2 Distributions and Random Variables

When unambiguous, we will naturally view a random variable as its marginal distribution. The support of a finite distribution $\mathcal{P}$ is defined by $\mathrm{Supp}(\mathcal{P}) := \{x \colon \Pr_{\mathcal{P}}[x] > 0\}$. For a (discrete) distribution $\mathcal{P}$, let $x \leftarrow \mathcal{P}$ denote that x was sampled according to $\mathcal{P}$. Similarly, for a set $\mathcal{S}$, let $x \leftarrow \mathcal{S}$ denote that x is drawn uniformly from $\mathcal{S}$. For $m \in \mathbb{N}$, we use $\mathbf{U}_m$ to denote

[11] In the protocol of [16], Alice would instead choose random programs for Alice and Bob to run; we do not know how to prove the security of such a protocol under our assumption.

a uniform random variable over $\{0,1\}^m$ (that is independent from other random variables in consideration). The statistical distance (also known as, variation distance) of two distributions $\mathcal{P}$ and $\mathcal{Q}$ over a discrete domain $\mathcal{X}$ is defined by $\mathrm{SD}(\mathcal{P},\mathcal{Q}) := \max_{\mathcal{S}\subseteq\mathcal{X}}|\mathcal{P}(\mathcal{S}) - \mathcal{Q}(\mathcal{S})| = \frac{1}{2}\sum_{x\in\mathcal{S}}|\mathcal{P}(x) - \mathcal{Q}(x)|$.

The following lemma is proven in the full version of this paper.

▶ **Lemma 8.** *There exists an efficient oracle-aided algorithm* A *such that the following holds. Let* $(\mathbf{X},\mathbf{Y})$ *be a pair of jointly distributed random variables over* $\{0,1\}^* \times \{0,1\}^n$*, and let* $\mathbf{R}$ *be a uniform independent random variable over* $\{0,1\}^n$*. Let* E *be an algorithm such that* $\Pr[\mathsf{E}(\mathbf{X},\mathbf{R}) = \langle \mathbf{Y},\mathbf{R}\rangle] \geq 1-\epsilon$*, for* $0 \leq \epsilon$*. Then* $\Pr\left[\mathsf{A}^{\mathsf{E}}(1^n,\mathbf{X}) = \mathbf{Y}\right] \geq 1 - 8\epsilon - \mathrm{neg}(n)$*.*

2.3 Circuits

In this paper we consider circuits over the De-Morgan Basis, which contains the following gates: $\wedge$ ("and" gate with fan-in 2), $\vee$ ("or" gate with fan-in 2), and $\neg$ ("not" gate with fan-in one). The size of a circuit C, is the number of gates in C.

We consider encoding of a circuit as a string over $\{0,1\}^*$ in the following natural way: Given a circuit C, we first encode the length of C using a prefix-free encoding, and then for every gate g, according to a topological order, we encode its type (input, output, $\wedge$,$\vee$, or $\neg$), and the (up to two) gates wired into g.

Observe that every circuit C of size s can be encoded to a string of length $O(s\log s)$. Moreover, given the encoding and an input x to C, $C(x)$ can be computed efficiently (in polynomial time in the encoding length).

We identify a string in $\{0,1\}^*$ with the circuit it encodes. For example, for $C \in \{0,1\}^n$ we use $C(x)$ to denote the value of the circuit encoded by C applied on the input x. We denote by $|C|$ the length of the encoding of C.

Finally, by encoding first the size of C, we assume that for every encoding of a circuit C, and for every $z \in \{0,1\}^*$, the string $C||z$ encodes the same circuit as C.[12]

2.4 Entropy

For a random variable $\mathbf{X}$, let $\mathrm{H}(\mathbf{X}) = \mathrm{E}[\log\frac{1}{\Pr[\mathbf{X}=x]}]$ denote the (Shannon) entropy of $\mathbf{X}$, and let $\mathrm{H}_\infty(\mathbf{X}) = \min_{x\in Supp(\mathbf{X})}\log\frac{1}{\Pr[\mathbf{X}=x]}$ denote the *min-entropy* of $\mathbf{X}$.

$$\min_{x\in\mathrm{Supp}(\mathbf{X})}\log\frac{1}{\Pr[\mathbf{X}=x]}.$$

For a random variable $\mathbf{X}$ and an event E, we use $\mathrm{H}_\infty(\mathbf{X} \mid E)$ to denote the min-entropy of the distribution $\mathbf{X}|_E$. We will use the following facts.

▶ **Fact 9.** *Let* $\mathbf{X}$ *and* $\mathbf{Y}$ *be independent random variables. Then* $\mathrm{H}_\infty(\mathbf{X},\mathbf{Y}) = \mathrm{H}_\infty(\mathbf{X}) + \mathrm{H}_\infty(\mathbf{Y})$*.*

2.5 Complexity Classes

We define the complexity classes FBPP and ioP/poly.

[12] We actually only use the fact that we can encode z into the circuit. This can be done by adding dummy gates that do not change the output of C, where each gate g_i is either $\wedge$ or $\vee$ according to the ith bit of z.

▶ **Definition 10** (Infinitely-often FBPP (ioFBPP)). *A binary relation $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$ is in* ioFBPP *if there exists* PPT *algorithms* A *such that the following holds for infinitely many n's:*

For every $x \in \{0,1\}^n$ such that there exists $y \in \{0,1\}^*$ with $(x, y) \in \mathcal{R}$,

$$\Pr\left[(x, \mathsf{A}(x, 1^k)) \in \mathcal{R}\right] \geq 1 - 1/k,$$

for every $k > 0$. $\mathcal{R}$ is in ioP/poly *if the above holds with respect to n.u. − poly − time algorithms* A.

2.6 One-Way Function

▶ **Definition 11** (One-way function). *A polynomial-time computable function $f : \{0,1\}^* \to \{0,1\}^*$ is called a* one-way function *if for every polynomial-time algorithm* A,

$$\Pr_{x \leftarrow \{0,1\}^n}\left[\mathsf{A}(1^n, f(x)) \in f^{-1}(f(x))\right] = \operatorname{neg}(n)$$

▶ **Definition 12** (Weak one-way function). *Let $m \in \mathrm{poly}$ be a polynomial-time computable function. A polynomial-time computable function $f : \{0,1\}^{m(n)} \to \{0,1\}^*$ is called α-*weak one-way function *if for every polynomial-time algorithm* A, *for every large enough n,*

$$\Pr_{x \leftarrow \{0,1\}^{m(n)}}\left[\mathsf{A}(1^n, f(x)) \in f^{-1}(f(x))\right] \leq 1 - \alpha(n)$$

f is a weak one-way function *if it is $1/p$-weak one way function, for some $p \in \mathrm{poly}$.*

▶ **Theorem 13** (Weak to strong OWFs, [36]). *One-way functions exist if and only if weak-one way functions exist.*

2.7 Public-Key Encryption

▶ **Definition 14** (Public-key encryption scheme (PKE)). *A triplet of randomized, efficiently computable functions* (Gen, Enc, Dec) *is a $(\alpha(n), \beta(n))$-*public-key encryption scheme (PKE) *if the following holds:*

- *Correctness: For every large enough $n \in \mathbb{N}$ and any $m \in \{0,1\}$,*

$$\Pr_{(sk,pk) \leftarrow \mathsf{Gen}(1^n)}[\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m] \geq 1 - \alpha(n)$$

- *Security: For every* PPT Eve, *for every large enough $n \in \mathbb{N}$,*

$$\Pr_{(sk,pk) \leftarrow \mathsf{Gen}(1^n), m \leftarrow \{0,1\}}[\mathsf{Eve}(pk, \mathsf{Enc}(pk, m)) = m] \leq 1/2 + \beta(n).$$

Such a scheme is a PKE *if it is $(1/n^c, 1/n^c)$-PKE for every $c \in \mathbb{N}$.*

The following lemma shows that it is possible to amplify an 1-bit weak key-agreement protocol into a key-agreement. This lemma is a simple case of the more general result of [19].

▶ **Lemma 15** (PKE amplification, [19]). *The following holds for every constants $c_1 > c_2$. Assume there exists an $(n^{-c_1}, 1/2 - n^{-c_2})$-PKE. Then, there exists a PKE.*

We also define weak-PKE.

▶ **Definition 16** (Weak Public-key encryption scheme (weak-PKE)). *For an efficiently computable function $d \colon \{1\}^* \to \mathbb{N}$, a triplet of randomized, efficiently computable functions* (Gen, Enc, Dec) *is a $(\alpha(n), \beta(n), \gamma(n))$-*weak-public-key encryption scheme (weak-PKE) *if the following holds:*

- *For every $n \in \mathbb{N}$, given pk and randomness r,* $\mathsf{Enc}(pk;r)$ *outputs a message $m(pk;r)$ and an output $o(pk;r) \in \mathbb{N}$.*
- *Correctness: For every large enough $n \in \mathbb{N}$*

$$\Pr_{(sk,pk)\leftarrow\mathsf{Gen}(1^n),r}[|\mathsf{Dec}(sk,m(pk,r)) - o(pk,r)| \le d(1^n)] \ge 1-\alpha(n)$$

- *Security: For every* PPT Eve, *for every large enough $n \in \mathbb{N}$,*

$$\Pr_{(sk,pk)\leftarrow\mathsf{Gen}(1^n),r}[|\mathsf{Eve}(pk,m(pk,r)) - o(pk,r)| \le \gamma(n)\cdot d(1^n)] \le \beta(n).$$

In the full version of this paper we prove the following lemma, stating that weak-PKE can be used to construct PKE.

▶ **Lemma 17** (Weak-PKE amplification)**.** *The following holds for every constants $c_1 > c_2$. Assume there exists an $(n^{-c_1}/2, 1-10n^{-c_2}, 2n^{c_1})$-weak-PKE. Then, there exists a PKE.*

2.8 Kolmogorov Complexity and Computational Depth

Roughly speaking, the *t-time-bounded Kolmogorov complexity*, $\mathrm{K}^t(x)$, of a string $x \in \{0,1\}^*$ is the length of the shortest program $\Pi = (M, y)$ such that, when simulated by an universal Turing machine, Π outputs x in $t(|x|)$ steps. Here, a program Π is simply a pair of a Turing Machine M and an input y, where the output of P is defined as the output of $M(y)$. When there is no running time bound (i.e., the program can run in an arbitrary number of steps), we obtain the notion of Kolmogorov complexity.

In the following, fix universal TM U with polynomial simulation overhead, and let $\mathsf{U}(\Pi, 1^t)$ denote the output of Π when emulated on U for t steps. We now define the notion of Kolmogorov complexity with respect to the universal TM U.

▶ **Definition 18.** *Let t be a polynomial. For all $x \in \{0,1\}^*$, define the t-*bounded Kolmogorov complexity *of x*

$$\mathrm{K}^t(x) = \min_{\Pi\in\{0,1\}^*}\{|\Pi| : \mathsf{U}(\Pi, 1^{t(|x|)}) = x\}$$

where $|\Pi|$ is referred to as the description length *of Π. When there is no time bound, the* Kolmogorov complexity *of x is defined as*

$$\mathrm{K}(x) = \min_{\Pi\in\{0,1\}^*}\{|\Pi| : \exists t \in \mathbb{N}\mathsf{U}(\Pi, 1^t) = x\}.$$

The computational depth *of x [4], denoted by $cd^{t,\infty}(x)$, is defined to be*

$$cd^{t,\infty}(x) = \mathrm{K}^t(x) - \mathrm{K}(x).$$

We use $\mathsf{K}(x,y)$ to denote the Kolmogorov complexity of some generic self-delimiting encoding of the pair x, y. Recall that we use $\mathsf{K}(x||y)$ to denote the complexity of the concatenation of x and y. We will use the following well-known fact:

▶ **Fact 19.** *For every $x, y \in \{0,1\}^*$,*

$$\mathsf{K}(x,y) \le \mathsf{K}(x) + \mathsf{K}(y) + \log(\mathsf{K}(x)) + 2\log\log(\mathsf{K}(x)) + O(1).$$

We will also use the following bound on the Kolmogorov complexity of strings sampled from distributions with high min-entropy.

▶ **Lemma 20.** *For every $n \in \mathbb{N}$, and every distribution $\mathcal{D}$, it holds that*

$$\Pr_{x \leftarrow \mathcal{D}}[\mathsf{K}(x) \geq \mathrm{H}_\infty(\mathcal{D}) - \log n] \geq 1 - 1/n.$$

Proof. There are at most $2^{\mathrm{H}_\infty(\mathcal{D}) - \log n}$ strings x with $\mathsf{K}(x) < \mathrm{H}_\infty(\mathcal{D}) - \log n$. Thus,

$$\Pr_{x \leftarrow \mathcal{D}}[\mathsf{K}(x) < \mathrm{H}_\infty(\mathcal{D}) - \log n] \leq 2^{\mathrm{H}_\infty(\mathcal{D}) - \log n} \cdot 2^{-\mathrm{H}_\infty(\mathcal{D})} = 1/n.$$ ◀

We will also use the well-known Chernoff bound in our proof.

▶ **Fact 21** (Hoeffding's inequality). *Let $\mathbf{A}_1, ..., \mathbf{A}_n$ be independent random variables s.t. $\mathbf{A}_i \in \{0,1\}$. Let $\widehat{\mathbf{A}} = 1/n \cdot \Sigma_{i=1}^n \mathbf{A}_i$ and $\mu = \mathrm{E}\left[\widehat{\mathbf{A}}\right]$. For every $\epsilon \in [0,1]$ it holds that:*

$$\Pr\left[\left|\widehat{\mathbf{A}} - \mu\right| \geq \epsilon\right] \leq 2 \cdot e^{-\epsilon^2 \cdot n}.$$

3 White-Box Distributional Learning

Let $P\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^\ell$ be a circuit that, given r bits of randomness, samples labeled instances (x, s). In the following we view s as a binary representation of a number in $\mathbb{N}$ (and respectively all the operations below are over $\mathbb{N}$, and we use $|\cdot|$ to denote the absolute value). We define the set of all circuits that approximate P with high probability,

$$\mathsf{Comp}_\epsilon^\Delta(P) = \left\{C\colon \{0,1\}^k \to \{0,1\}^\ell \colon \Pr_{(x,s) \leftarrow P(\mathbf{U}_r)}[|C(x) - s| \leq \Delta] \geq 1 - \epsilon\right\}.$$

We define the following white-box learning problem $\mathsf{WBLearn}$:

▶ **Definition 22** (Δ-$\mathsf{WBLearn}$). *For a function $\Delta\colon \{1\}^* \to \mathbb{N}$, let Δ-$\mathsf{WBLearn}$ be the following learning problem:*

- *Input: Circuit $P \in \{0,1\}^n$, with the promise that there exists a circuit $\widehat{C} \in \{0,1\}^n$ such that $\widehat{C} \in \mathsf{Comp}_{n^{-10}}^{\Delta(1^n)/n^{10}}(P)$.*
- *Output: Circuit $C \in \mathsf{Comp}_{1/3}^{\Delta(1^n)}(P)$*

In this work we are focusing on inputs P for which the circuit $\widehat{C}$ that agree with P with high probability, is such that the description of $\widehat{C}$ and P is computational shallow. Formally, for a time function $t\colon \mathbb{N} \to \mathbb{N}$, we denote by Δ-$\mathsf{WBLearn}|_{\mathsf{CS}^t}$ the problem Δ-$\mathsf{WBLearn}$ with the additional promise that $cd^t(P, \widehat{C}) \leq 2 \log n$.

We use $\mathsf{ExactWBLearn}$ and $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t}$ to denote Δ-$\mathsf{WBLearn}$ and Δ-$\mathsf{WBLearn}|_{\mathsf{CS}^t}$ when $\Delta(1^n) = 0$. Note that $\mathsf{ExactWBLearn}$ and Δ-$\mathsf{WBLearn}|_{\mathsf{CS}^t}$ are incomparable: While in Δ-$\mathsf{WBLearn}|_{\mathsf{CS}^t}$ we only need to find a circuit that approximates P, the promise in $\mathsf{ExactWBLearn}$ is stronger. Yet, there is a simple reduction from $\mathsf{ExactWBLearn}$ to Δ-$\mathsf{WBLearn}|_{\mathsf{CS}^t}$.

▶ **Lemma 23.** *For every $\Delta\colon \{1\}^* \to \mathbb{N}$ such that $\Delta \in 2^{o(n/\log n)}$, it holds that*

$$\mathsf{ExactWBLearn} \leq_p \Delta\text{-}\mathsf{WBLearn}.$$

Similarly, for any such Δ and a function $t\colon \mathbb{N} \to \mathbb{N}$, there exists $t'\colon \mathbb{N} \to \mathbb{N}$, such that

$$\mathsf{ExactWBLearn}|_{\mathsf{CS}^t} \leq_p \Delta\text{-}\mathsf{WBLearn}|_{\mathsf{CS}^{t'}}.$$

Proof. We start with the first reduction $\mathsf{ExactWBLearn} \leq_p \Delta\text{-}\mathsf{WBLearn}$. Given a circuit $P\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^\ell$ of length n, the reduction outputs a circuit $P'\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^{\ell+n}$, which is equivalent to P, with additional n output gates that always output 0. That is, $P'(w) = (x, s')$, where $P(w) = (x, s)$ and $s' = 2^n \cdot s$. As we only added n gates to P, P' can be encoded using $O(n \log n)$ bits. Using padding we assume that $|P'| \in \Theta(n \log n)$.

For correctness, observe that if there exists $\widehat{C} \in \mathsf{Comp}^0_{n-10}(P)$, then there exists such $\widehat{C'} \in \mathsf{Comp}^0_{n-10}(P')$ (where $\widehat{C'}$ is defined similarly to P' using $\widehat{C}$). Moreover, an approximation of the output s' of P' within a distance of $2^n/4$ is equivalent to the exact output s of P. Indeed, given an $2^n/4$-approximation of s' we can find s by simply dividing s' by 2^n and rounding to the closest integer.

Finally, to see the second reduction, it is enough to show that $cd^{t'}(P', \widehat{C'}) \leq 2\log|P'|$. Since $P', \widehat{C'}$ can be efficiently constructed given $P, \widehat{C}$, and similarly $P, \widehat{C}$ can be efficiently constructed given $P', \widehat{C'}$, it holds that for large enough (polynomial time t', $cd^{t'}(P', \widehat{C'}) \leq cd^t(P, \widehat{C}) + O(1) \leq 2\log n + O(1) \leq 2\log n + 2\log\log n \leq 2\log|P'|$. ◀

We prove the following theorem.

▶ **Theorem 24.** *Let $\epsilon > 0$ be a constant and let $\Delta\colon \{1\}^* \to \mathbb{N}$ be any efficiently computable function such that $\Delta(1^n) \leq 2^{n^{(1-\epsilon)}}$, and let $t\colon \mathbb{N} \to \mathbb{N}$ be any polynomial such that $t(n) \geq n^{1+\epsilon}$. Then the following are equivalent:*

- *PKE exists*
- $\Delta\text{-}\mathsf{WBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$

As a corollary, we get a result of independent interest relating exact and approximate white-box learning:

▶ **Corollary 25.** *Let $\epsilon > 0$ be a constant and let $\Delta\colon \{1\}^* \to \mathbb{N}$ be any efficiently computable function such that $\Delta(1^n) \leq 2^{n^{(1-\epsilon)}}$, and let $t\colon \mathbb{N} \to \mathbb{N}$ be any polynomial such that $t(n) \geq n^{1+\epsilon}$. Then the following are equivalent:*

- $\mathsf{ExactWBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$
- $\Delta\text{-}\mathsf{WBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$

Theorem 24 follows by the following two theorems.

▶ **Theorem 26.** *Let $t(n)$ be a polynomial and $\Delta\colon \{1\}^* \to \mathbb{N}$ an efficiently computable function. Then if $\Delta\text{-}\mathsf{WBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$, PKE exist.*

▶ **Theorem 27.** *Assume there exists a PKE. Then for any constant $\epsilon > 0$ and any $t(n) \geq n^{1+\epsilon}$ and any efficiently computable $\Delta\colon \{1\}^* \to \mathbb{N}$ such that $\Delta(1^n) \leq 2^{n^{(1-\epsilon)}}$, $\Delta\text{-}\mathsf{WBLearn}|_{\mathsf{CS}^t} \notin \mathsf{ioFBPP}$.*

Theorem 26 is proven in Section 4, and Theorem 27 is proven in the full version of this paper.

4 Worst-Case hardness of $\Delta\text{-}\mathsf{WBLearn}|_{\mathsf{CS}^t} \implies$ KE

In this section we prove Theorem 26, that states that the worst-case hardness of $\mathsf{WBLearn}|_{\mathcal{Q}_t}$ implies the existence of public-key encryption scheme. In the following, let $C_0\colon \{0,1\} \to \{0,1\}$ be the circuit that always outputs 0, and $P_0\colon \{0,1\} \to \{0,1\} \times \{0,1\}$ be the circuit that always outputs $(0,0)$.

To prove the above theorem, fix $t = t(n)$ and $\Delta = \Delta(1^n)$, and consider the following scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$:

▶ **Algorithm 28** (Gen).
Parameter: function $t\colon \mathbb{N} \to \mathbb{N}$.
Input: 1^n.
Operation:

```
1. Sample $\lambda \leftarrow [3n]$ and $\Pi \leftarrow \{0,1\}^{\lambda}$.
2. Run $\Pi$ for $t(2n)$ steps to get circuits $C\colon \{0,1\}^k \to \{0,1\}^{\ell}, P\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^{\ell}$
   for some $k, r, \ell$. If the output of $\Pi$ is not two such circuits, set $C = C_0$, $P = P_0$,
   $r = k = \ell = 1$.
3. Randomly sample $(x_1, s_1), \ldots, (x_{n^{20}}, s_{n^{20}}) \leftarrow P(U_r)$, and compute
       $\widehat{\alpha} = \Pr_{i \leftarrow [n^{20}]}\left[|C(x_i) - s_i| \leq \Delta(1^n)/n^{10}\right]$.
   If $\widehat{\alpha} < 1 - 2n^{-8}$, reset $C = C_0$ and $P = P_0$.
4. Output $(k, \ell, C)$ as the secret key, and $(r, k, \ell, P)$ as the public key.
```

▶ **Algorithm 29** (Enc).
Input: public-key $(r \in \mathbb{N}, k \in \mathbb{N}, \ell \in \mathbb{N}, P\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^{\ell})$.
Operation:

```
1. Sample randomness $z \leftarrow \{0,1\}^r$.
2. Compute $P(z)$ to get $(x, s)$.
3. Output $m = x$ and $o = s$.
```

▶ **Algorithm 30** (Dec).
Input: secret-key $(k \in \mathbb{N}, \ell \in \mathbb{N}, C\colon \{0,1\}^k \to \{0,1\})$, *cipher* $x \in \{0,1\}^k$.
Operation:

```
1. Compute $C(x) = s'$.
2. Output $s'$.
```

Observe that the size of the circuits C and P sampled by $\mathsf{Gen}(1^n)$ is at most $t(2n)$. Thus, all of the above algorithms can be implemented efficiently. Below we bound the correctness and the security of the above scheme. For every $n \in \mathbb{N}$, let $(\mathbf{K}_n, \mathbf{L}_n, \mathbf{C}_n)$ and $(\mathbf{R}_n, \mathbf{K}_n, \mathbf{L}_n, \mathbf{P}_n)$ be the random variables distributed according to the secret and public keys $(k, \ell, C), (r, k, \ell, P)$ in a random execution of $\mathsf{Gen}(1^n)$. Let $\mathbf{M}_n$ and $\mathbf{O}_n$ be the output of $\mathsf{Enc}(\mathbf{R}_n, \mathbf{K}_n, \mathbf{L}_n, \mathbf{P}_n)$ in a random execution.

4.1 Correctness

We start by analyzing the correctness probability of the scheme.

▶ **Lemma 31.** *For every* $n \in \mathbb{N}$*, it holds that*

$$\Pr\left[|\mathsf{Dec}((\mathbf{K}_n, \mathbf{L}_n, \mathbf{C}_n), \mathbf{M_n}) - \mathbf{O}_n| \leq \Delta(1^n)/n^{10}\right] \geq 1 - n^{-7}.$$

To prove Lemma 31 we will use the following simple claim, which is immediate from the Hoffeding bound.

▷ Claim 32. Let $C\colon \{0,1\}^k \to \{0,1\}^{\ell}$ and $P\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^{\ell}$ be two circuits. Let $\mathbf{A}$ be a random variable distributed according to the following process:

Sample $(x_1, s_1) \ldots, s(x_{n^{20}}, s_{n^{20}}) \leftarrow P(\mathbf{U}_r)$, and let $\mathbf{A} = \Pr_{i \leftarrow [n^{20}]}[|C(x_i) - s_i| \leq \Delta]$. Let $\mu = \Pr_{(x,s) \leftarrow P(\mathbf{U}_r)}[|C(x) - s| \leq \Delta]$. Then $\Pr\left[|\mathbf{A} - \mu| \geq n^{-9}\right] \leq 2^{-n}$.

Proof. Immediate by Fact 21. ◁

Proof of Lemma 31. When $\mathsf{Gen}(1^n)$ outputs C_0 and P_0, the scheme has perfect correctness. Moreover, when the secret and public key are (k, ℓ, C) and (r, k, ℓ, P), it holds that

$$\Pr\left[|\mathsf{Dec}((k,\ell,C),\mathbf{M})) - \mathbf{O}| \le \Delta(1^n)/n^{10}\right] = \Pr_{(x,s)\leftarrow P(\mathbf{U}_r)}\left[|C(x) - s| \le \Delta(1^n)/n^{10}\right].$$

Thus,

$$\begin{aligned}&\Pr\left[|\mathsf{Dec}((\mathbf{K}_n,\mathbf{L}_n,\mathbf{C}_n),\mathbf{M}_n) - \mathbf{O}_n| \le \Delta(1^n)/n^{10}\right]\\ &\ge (1-3n^{-8})\cdot \Pr_{\mathbf{R}_n,\mathbf{K}_n,\mathbf{P}_n,\mathbf{C}_n}\left[\Pr_{(x,s)\leftarrow \mathbf{P}_n(\mathbf{U}_{\mathbf{R}_n})}\left[|\mathbf{C}_n(x) - s| \le \Delta(1^n)/n^{10}\right] \ge 1 - 3n^{-8}\right]\\ &\ge 1 - 3n^{-8} - \Pr_{\mathbf{R}_n,\mathbf{K}_n,\mathbf{P}_n,\mathbf{C}_n}\left[\Pr_{(x,s)\leftarrow \mathbf{P}_n(\mathbf{U}_{\mathbf{R}_n})}\left[|\mathbf{C}_n(x) - s| \le \Delta(1^n)/n^{10}\right] < 1 - 3n^{-8}\right].\end{aligned}$$

The lemma now follows since by Lemma 31, the probability of circuits P, C with

$$\Pr_{(x,s)\leftarrow P(\mathbf{U}_r)}\left[|\mathbf{C}_n(x) - s| \le \Delta(1^n)/n^{10}\right] < 1 - 3n^{-8}$$

to pass the test in Step 3 is at most 2^{-n}. ◀

4.2 Security

We next bound the leakage of the scheme.

▶ **Lemma 33.** *Assume there exists an algorithm* Eve *such that*

$$\Pr[|\mathsf{Eve}(1^n, (\mathbf{R}_n, \mathbf{K}_n, \mathbf{L}_n, \mathbf{P}_n), \mathbf{M}_\mathbf{n}) - \mathbf{O}_n| \le \Delta(1^n)] \ge 1 - n^{-6}$$

for infinitely many n's. Then $\Delta\text{-}\mathsf{WBLearn}|_{\mathsf{CS}^t} \in \mathsf{ioBPP}$.

In the following, let Eve be an algorithm that uses $r_{\mathsf{Eve}}(n)$ bits of randomness and guesses $\mathbf{M}$ with probability at least $1-n^{-6}$. Recall that for $w \in \{0,1\}^{r_{\mathsf{Eve}}(n)}$, $\mathsf{Eve}(1^n, (r, k, \ell, P), m; w)$ denotes the execution of Eve when its randomness is fixed to be w.

▶ **Algorithm 34.**
Input: $P \in \{0,1\}^n$.
Operation:

```
1. Let $r, k, \ell$ be such that $P\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^\ell$.
2. Sample $w \leftarrow \{0,1\}^{r_{\mathsf{Eve}}(n)}$ uniformly at random.
3. Construct a circuit $C'$ such that $C'(x) = \mathsf{Eve}(1^n, (r, k, \ell, P), x; w)$.
4. Return $C'$.
```

We prove the following lemma.

▶ **Lemma 35.** *Assume that*

$$\Pr[|\mathsf{Eve}(1^n, (\mathbf{R}_n, \mathbf{K}_n, \mathbf{L}_n, \mathbf{P}_n), \mathbf{M}_n) - \mathbf{O}_n| \le \Delta(1^n)] \ge 1 - n^{-6}$$

for infinitely many n's. Then the following holds for infinitely many n's. For every input $P \in \{0,1\}^n \cap \mathcal{Q}_t^\Delta$*, Algorithm 34 outputs* $C \in \mathsf{Comp}_{1/4}^\Delta(P)$ *with probability at least* $1/2$.

We prove Lemma 35 below, but first we use it to prove Lemma 33.

Proof of Lemma 33. Assume there exists an algorithm Eve such that

$$\Pr[|\mathsf{Eve}(1^n, (\mathbf{R}_n, \mathbf{K}_n, \mathbf{L}_n, \mathbf{P}_n), \mathbf{M}_n) - \mathbf{O}_n| \le \Delta(1^n)] \ge 1 - n^{-6}$$

for infinitely many n's.

By Lemma 35, for infinitely many n's, Algorithm 34 outputs $C \in \mathsf{Comp}^{\Delta}_{1/4}(P)$ with probability at least 1/2, for every $P \in \{0,1\}^n \cap \mathcal{Q}^{\Delta}_t$. We want to construct an algorithm Sol that, given P and $1^{1/\delta}$, outputs $C \in \mathsf{Comp}_{1/3}(P)$ with probability at least $1 - \delta$, for every such n.

Let Sol be the algorithm that, given input $(P\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^\ell, 1^{1/\delta})$, first run Algorithm 34 on P for $2\lceil \log 1/\delta \rceil$ times, to get circuits $C_1, \ldots, C_{2\lceil \log 1/\delta \rceil}$. Then, for every circuit C_i, Sol samples $(x_1, s_1), \ldots, (x_{(100\lceil \log 1/\delta \rceil)^2}, s_{(100\lceil \log 1/\delta \rceil)^2}) \leftarrow P(\mathbf{U}_r)$, and computes

$$p_i = \Pr_{j \leftarrow [(100\lceil \log 1/\delta \rceil)^2]}[|C_i(x_j) - s_j| \le \Delta(1^n)].$$

Finally, Sol outputs the circuit C_i for the index i with maximal value of p_i.

We now analyze the success probability of Sol. Using *Fact* 21, for every i, the probability that $\left|p_i - \Pr_{(x,s) \leftarrow P(\mathbf{U}_r)}[|C_i(x) = s| \le \Delta(1^n)]\right| \ge 1/25$ is at most

$$2^{-4(\lceil \log 1/\delta \rceil)^2 + 1} \le 1/(\delta(4\lceil \log 1/\delta \rceil)).$$

Thus, by the union bound, the probability that $\left|p_i - \Pr_{(x,s) \leftarrow P(\mathbf{U}_r)}[|C_i(x) = s| \le \Delta(1^n)]\right| \ge 1/25$ for some i is at most $\delta/2$. Next, by the success probability of Algorithm 34, with probability at least

$$1 - (1 - 1/2)^{2 \log 1/\delta} \ge 1 - \delta/2,$$

at least one of the circuits $C_1, \ldots, C_{2 \log 1/\delta}$ is in $\mathsf{Comp}^{\Delta}_{1/4}(P) \subseteq \mathsf{Comp}^{\Delta}_{1/3}(P)$. Let i^* be the index of such a circuit. Then, with probability at least $1 - \delta/2 - \delta/2 = 1 - \delta$, such i^* exists, and p_{i^*} is at least $(3/4 - 1/25)$, while for every i with $C_i \notin \mathsf{Comp}^{\Delta}_{1/3}(P)$, p_i is at most

$$(2/3 + 1/25) < (3/4 - 1/25) \le p_{i^*},$$

which implies that the output of Sol is in $\mathsf{Comp}^{\Delta}_{1/3}(P)$. ◀

4.3 Proving Lemma 35

To prove Lemma 35, we start with the following claim.

▷ Claim 36. Let $P\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^\ell$ be a circuit, and assume that

$$\Pr_{(x,s) \leftarrow P(\mathbf{U}_r)}[|\mathsf{Eve}(1^n, (r, k, \ell, P), x) - s| \le \Delta(n)] \ge 9/10.$$

Then, on input P, Algorithm 34 outputs $C \in \mathsf{Comp}^{\Delta}_{1/4}(P)$ with probability at least 1/2.

Proof of Claim 36. Let $P\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^\ell$ be a circuit such that

$$\Pr_{(x,s) \leftarrow P(\mathbf{U}_r)}[|\mathsf{Eve}(1^n, (r, k, \ell, P), x) - s| \le \Delta(n)] \ge 9/10.$$

Then, by definition it holds that

$$\mathrm{E}_{w \leftarrow \{0,1\}^{r_{\mathsf{Eve}}(n)}}\left[\Pr_{(x,s) \leftarrow P(\mathbf{U}_r)}[|\mathsf{Eve}(1^n, (r, k, \ell, P), x) - s| > \Delta(n)]\right] \le 1/10.$$

Using Markov's inequality, we gets that

$$\Pr_{w \leftarrow \{0,1\}^{r_{\mathsf{Eve}}(n)}}\left[\Pr_{(x,s) \leftarrow P(\mathbf{U}_r)}[|\mathsf{Eve}(1^n, (r, k, \ell, P), x) - s| > \Delta(n)] > 1/4\right] \le 1/2,$$

which implies that the circuit $C' = \mathsf{Eve}(1^n, (r, k, \ell, P), \cdot; w)$ is in $\mathsf{Comp}^{\Delta}_{1/4}(P)$ with probability at least 1/2 over the choice of w, as we wanted to show. ◁

Given Claim 36, we are now ready to prove Lemma 35.

Proof of Lemma 35. Assume that Eve is such that

$$\Pr[|\mathsf{Eve}(1^n, (\mathbf{R}_n, \mathbf{K}_n, \mathbf{L}_n, \mathbf{P}_n), \mathbf{M}_n) - \mathbf{O}_n| \le \Delta(1^n)] \ge 1 - n^{-6}$$

for infinitely many n's. In the following, fix such large enough $n \in \mathbb{N}$. We show that

$$\Pr_{(x,s)\leftarrow P(\mathbf{U}_r)}[|\mathsf{Eve}(1^n, (r, k, \ell, P), x) - s| \le \Delta(n)] \ge 9/10 \tag{1}$$

for every $P\colon \{0,1\}^r \to \{0,1\}^k \times \{0,1\}^\ell$ with $P \in \mathcal{Q}_t^\Delta \cap \{0,1\}^n$. The proof then follows by Claim 36. To see the above, let $\mathbf{Pk} = (\mathbf{R}_n, \mathbf{K}_n, \mathbf{L}_n, \mathbf{P}_n))$, and notice that

$$\Pr_{\substack{\mathbf{R}_n,\mathbf{K}_n,\mathbf{L}_n,\mathbf{P}_n,\mathbf{C}_n \\ \mathbf{Pk}=(\mathbf{R}_n,\mathbf{K}_n,\mathbf{L}_n,\mathbf{P}_n)}} \left[\Pr_{\mathbf{M}_n,\mathbf{O}_n,\mathbf{W}\leftarrow\{0,1\}^r_{\mathsf{Eve}}}[|\mathsf{Eve}(1^n, \mathbf{Pk}, \mathbf{M}_n; \mathbf{W}) - \mathbf{O}_n| \le \Delta(n)] < 9/10\right] \tag{2}$$
$$\le 10n^{-6}.$$

Indeed, it holds that

$$\begin{aligned}
1 - n^{-6} &\le \Pr[|\mathsf{Eve}(1^n, \mathbf{Pk}, \mathbf{M}_n) - \mathbf{O}_n| \le \Delta(n)] \\
&\le \Pr_{\mathbf{Pk}}[\Pr[|\mathsf{Eve}(1^n, \mathbf{Pk}, \mathbf{M}_n) - \mathbf{O}_n| \le \Delta(n)] \ge 9/10] \\
&\qquad + 9/10 \cdot \Pr_{\mathbf{Pk}}[\Pr[|\mathsf{Eve}(1^n, \mathbf{Pk}, \mathbf{M}_n) - \mathbf{O}_n| \le \Delta(n)] < 9/10] \\
&= (1 - \Pr_{\mathbf{Pk}}[\Pr[|\mathsf{Eve}(1^n, \mathbf{Pk}, \mathbf{M}_n) - \mathbf{O}_n| \le \Delta(n)] < 9/10]) \\
&\qquad + 9/10 \cdot \Pr_{\mathbf{Pk}}[\Pr[|\mathsf{Eve}(1^n, \mathbf{Pk}, \mathbf{M}_n) - \mathbf{O}_n| \le \Delta(n)] < 9/10] \\
&= 1 - 1/10 \cdot \Pr_{\mathbf{Pk}}[\Pr[|\mathsf{Eve}(1^n, \mathbf{Pk}, \mathbf{M}_n) - \mathbf{O}_n| \le \Delta(n)] < 9/10]
\end{aligned}$$

which implies that Equation (2) holds. Next, we use Equation (2) and the upper bound on the computational depth of instances in $\mathcal{Q}_t^\Delta$, to show that Equation (1) holds for every $P \in \mathcal{Q}_t^\Delta$. To do so, fix $P \in \mathcal{Q}_t^\Delta \cap \{0,1\}^n$ and let $C \in \{0,1\}^n \cap \mathsf{Comp}_{n^{-10}}^{\Delta/n^{10}}(P)$ with $cd^{t,\infty}(C, P) \le 2\log n$ be the circuit promised by the definition of $\mathcal{Q}_t^\Delta$. Assume towards a contradiction that

$$\Pr_{(x,s)\leftarrow P(\mathbf{U}_r)}[|\mathsf{Eve}(1^n, (r, k, \ell, P), x) - s| \le \Delta(n)] < 9/10.$$

We want to upperbound $\mathsf{K}(P, C)$, to get a lower bound on the computational depth of (P, C). To this end, let $\mathcal{S}$ be the set of all pairs $(P', C') \in \{0,1\}^n \times \{0,1\}^n$, such that $C' \in \mathsf{Comp}_{n^{-10}}^{\Delta/n^{10}}(P')$, $\mathsf{K}^t(P', C') = \mathsf{K}^t(P, C)$, and on which Eve fails to approximate s with probability more than $1/10$.

By our assumption on (P, C), it holds that $(P, C) \in \mathcal{S}$. We next bound the size of $\mathcal{S}$. First, we claim that for every $(P', C') \in \mathcal{S}$,

$$\Pr[\mathbf{P}_n = P', \mathbf{C}_n = C'] \ge 1/6n \cdot 2^{-\mathsf{K}^t(C,P)}. \tag{3}$$

Indeed, by definition there exists a program Π of length $\mathsf{K}^t(C, P)$ that outputs (C', P') in at most $t(2n)$ steps. Thus, Gen samples Π with probability at least $1/3n \cdot 2^{-\mathsf{K}^t(C,P)}$. Next, by Claim 32, the test in Step 3 of Gen passes with probability at least $1 - 2^{-n} > 1/2$, since by definition of $\mathsf{Comp}_{n^{-10}}^{\Delta/n^{10}}$, $\Pr_{(x,s)\leftarrow P'(\mathbf{U}_r}\left[|C'(x) - s| \le \Delta(1^n)/n^{10}\right] - n^{-9} \ge 1 - n^{-10} - n^{-9} \ge 1 - 2n^{-7}$. In this case that the test passes, P' and C' are the output of Gen, and thus Equation (3) holds.

By Equation (3) and the definition of $\mathcal{S}$, we get that

$$\Pr_{\substack{\mathbf{R}_n,\mathbf{K}_n,\mathbf{L}_n,\mathbf{P}_n,\mathbf{C}_n\\ \mathbf{Pk}=(\mathbf{R}_n,\mathbf{K}_n,\mathbf{L}_n,\mathbf{P}_n)}}\left[\Pr_{\mathbf{M}_n,\mathbf{O}_n,\mathbf{W}\leftarrow\{0,1\}^r_{\mathsf{Eve}}}[|\mathsf{Eve}(1^n,\mathbf{Pk},\mathbf{M}_n;\mathbf{W})-\mathbf{O}_n|\leq\Delta(n)]<9/10\right]$$
$$\geq|\mathcal{S}|\cdot 1/6n\cdot 2^{-\mathsf{K}^t(C,P)}.$$

Combining the above with Equation (2) yields that

$$|\mathcal{S}|\leq\frac{10n^{-6}}{1/6n\cdot 2^{-\mathsf{K}^t(C,P)}}\leq 2^{\mathsf{K}^t(C,P)+\log n-6\log n+6}.$$

Observe that $\mathcal{S}$ can be (inefficiently) computed given $n, \mathsf{K}^t(C,P)$ and Eve. Thus, to encode (P,C), it is enough to encode $\mathcal{S}$ and the index of (P,C) in $\mathcal{S}$ (according to the lexicographic order). We conclude that,

$$\begin{aligned}\mathsf{K}(P,C)&\leq\mathsf{K}(n,\lambda,\mathsf{Eve})+2\log(\mathsf{K}(n,\lambda,\mathsf{Eve}))+\log|\mathcal{S}|+O(1)\\&\leq 2\log n+4\log\log n+\mathsf{K}^t(C,P)-5\log n+O(1)\\&<\mathsf{K}^t(C,P)-2\log n,\end{aligned}$$

where the last inequality holds for every large enough n. By the above, $\mathsf{K}^t(C,P)-\mathsf{K}(P,C)>2\log n$, in contradiction to the choice of (P,C). This yields that Equation (1) holds for every $P\in\mathcal{Q}_t$, as we wanted to show. ◀

4.4 Proving Theorem 26

We are now ready to use Lemma 15 in order to prove Theorem 26.

Proof of Theorem 26. By Lemmas 31 and 33, $(\mathsf{Gen},\mathsf{Enc},\mathsf{Dec})$ is a $(n^{-7},1-n^{-6},n^{10})$-weak-PKE (for $d(1^n)=\Delta(1^n)/n^{10}$), and thus it is also a $(n^{-6.9}/2,1-10n^{-6.1},2n^{6.9})$-weak-PKE. Thus, by Lemma 15, $(\mathsf{Gen},\mathsf{Enc},\mathsf{Dec})$ can be amplified into a PKE. ◀

▶ Remark 37 (The non-uniform setting). A similar theorem can be proven when assuming that $\Delta\text{-}\mathsf{WBLearn}|_{\mathsf{CS}^t}\notin\mathsf{ioP}/\text{poly}$, and when the PKE is secure against non-uniform adversaries. In this case, we assume that Eve is a non-uniform algorithm that breaks the PKE protocol, and want to construct a non-uniform (randomized) algorithm that decides $\mathsf{WBLearn}|_{\mathcal{Q}_t}$.

The issue with the above proof is that we cannot simply use Eve to bound the Kolmogorov complexity of (C,P) as done in the proof of Lemma 33, as Eve does not have constant size. However, we can find Eve using a small Turing machine: Let M be the (inefficient) Turing machine that, given a constant c such that n^c is a bound on the size of Eve, and an input for Eve, first find the circuit E'_n of size at most n^c that maximize the advantage in predicting M given an encryption of M by Enc, and then execute E' on the input. Observe that M has prediction advantage at least as the advantage of Eve. The theorem now follows using the same proof, by replacing Eve in the proof of Lemma 33 with M, and replacing Eve in Algorithm 34 with $E'=\{E'_n\}_{n\in\mathbb{N}}$.

References

1 Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *stoc29*, pages 284–293, 1997. See also ECCC TR96-065. doi:10.1145/258533.258604.

2 Michael Alekhnovich. More on average case vs approximation complexity. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 298–307. IEEE, 2003. doi:10.1109/SFCS.2003.1238204.

3 Luis Antunes and Lance Fortnow. Worst-case running times for average-case algorithms. In *2009 24th Annual IEEE Conference on Computational Complexity*, pages 298–303. IEEE, 2009. doi:10.1109/CCC.2009.12.

4 Luis Antunes, Lance Fortnow, Dieter Van Melkebeek, and N Variyam Vinodchandran. Computational depth: concept and applications. *Theoretical Computer Science*, 354(3):391–404, 2006. doi:10.1016/J.TCS.2005.11.033.

5 Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 171–180, 2010. doi:10.1145/1806689.1806715.

6 Marshall Ball, Yanyi Liu, Noam Mazor, and Rafael Pass. Kolmogorov comes to cryptomania: On interactive kolmogorov complexity and key-agreement. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 458–483. IEEE, 2023. doi:10.1109/FOCS57990.2023.00034.

7 Avrim Blum, Merrick Furst, Michael Kearns, and Richard J Lipton. Cryptographic primitives based on hard learning problems. In *Annual International Cryptology Conference*, pages 278–291. Springer, 1993. doi:10.1007/3-540-48329-2_24.

8 Andrej Bogdanov, Miguel Cueto Noval, Charlotte Hoffmann, and Alon Rosen. Public-key encryption from homogeneous clwe. In *Theory of Cryptography: 20th International Conference, TCC 2022, Chicago, IL, USA, November 7–10, 2022, Proceedings, Part II*, pages 565–592. Springer, 2022. doi:10.1007/978-3-031-22365-5_20.

9 Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 575–584, 2013. doi:10.1145/2488608.2488680.

10 Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. *Journal of the ACM*, 43(2):831–871, 2014.

11 Gregory J. Chaitin. On the simplicity and speed of programs for computing infinite sets of natural numbers. *J. ACM*, 16(3):407–422, 1969. doi:10.1145/321526.321530.

12 Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, pages 644–654, 1976. doi:10.1109/TIT.1976.1055638.

13 Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Annual International Cryptology Conference (CRYPTO)*, pages 10–18, 1984.

14 Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 169–178, 2009. doi:10.1145/1536414.1536440.

15 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing (STOC)*, pages 25–32, 1989. doi:10.1145/73007.73010.

16 Danny Harnik, Joe Kilian, Moni Naor, Omer Reingold, and Alon Rosen. On robust combiners for oblivious transfer and other primitives. In *Advances in Cryptology–EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005. Proceedings 24*, pages 96–113. Springer, 2005. doi:10.1007/11426639_6.

17 J. Hartmanis. Generalized kolmogorov complexity and the structure of feasible computations. In *24th Annual Symposium on Foundations of Computer Science (sfcs 1983)*, pages 439–445, 1983. doi:10.1109/SFCS.1983.21.

18 Shuichi Hirahara and Mikito Nanashima. Learning in pessiland via inductive inference. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 447–457. IEEE, 2023. doi:10.1109/FOCS57990.2023.00033.

19 Thomas Holenstein. *Strengthening key agreement using hard-core sets.* PhD thesis, ETH Zurich, 2006. doi:10.3929/ETHZ-A-005205852.

20 Russell Impagliazzo and Leonid A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *focs31*, pages 812–821, 1990. doi:10.1109/FSCS.1990.89604.

21 Michael Kearns and Leslie Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of the ACM (JACM)*, 41(1):67–95, 1994. doi:10.1145/174644.174647.

22 Ker-I Ko. On the notion of infinite pseudorandom sequences. *Theor. Comput. Sci.*, 48(3):9–33, 1986. doi:10.1016/0304-3975(86)90081-2.

23 A. N. Kolmogorov. Three approaches to the quantitative definition of information. *International Journal of Computer Mathematics*, 2(1-4):157–168, 1968.

24 Yanyi Liu, Noam Mazor, and Rafael Pass. On white-box learning and public-key encryption. Cryptology ePrint Archive, Paper 2024/1931, 2024. URL: https://eprint.iacr.org/2024/1931.

25 Yanyi Liu and Rafael Pass. On one-way functions and kolmogorov complexity. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1243–1254. IEEE, 2020. doi:10.1109/FOCS46700.2020.00118.

26 Yanyi Liu and Rafael Pass. On one-way functions and the worst-case hardness of time-bounded kolmogorov complexity. *Cryptology ePrint Archive*, 2023.

27 Robert J McEliece. A public-key cryptosystem based on algebraic. *Coding Thv*, 4244:114–116, 1978.

28 Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 333–342, 2009.

29 Michael O Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, Massachusetts Inst of Tech Cambridge Lab for Computer Science, 1979.

30 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):1–40, 2009. doi:10.1145/1568318.1568324.

31 Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978. doi:10.1145/359340.359342.

32 Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999. doi:10.1137/S0036144598347011.

33 Michael Sipser. A complexity theoretic approach to randomness. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pages 330–335, 1983. doi:10.1145/800061.808762.

34 R.J. Solomonoff. A formal theory of inductive inference. part i. *Information and Control*, 7(1):1–22, 1964. doi:10.1016/S0019-9958(64)90223-2.

35 Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984. doi:10.1145/1968.1972.

36 Andrew C. Yao. Theory and applications of trapdoor functions. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 80–91, 1982. doi:10.1109/SFCS.1982.45.