# On Strong Simulation and Composable Point Obfuscation*

Nir Bitansky and Ran Canetti

Tel Aviv University and Boston University

January 20, 2013

**Abstract**

The Virtual Black Box (VBB) property for program obfuscators provides a strong guarantee: anything computable by an efficient adversary, given the obfuscated program, can also be computed by an efficient simulator, with only oracle access to the program. However, we know how to achieve this notion only for very restricted classes of programs.

This work studies a simple relaxation of VBB: allow the simulator unbounded computation time, while still allowing only polynomially many queries to the oracle. We demonstrate the viability of this relaxed notion, which we call Virtual Grey Box (VGB), in the context of composable obfuscators for point programs: it is known that, with respect to VBB, if such obfuscators exist, then there exist multi-bit point obfuscators (also known as "digital lockers") and subsequently also very strong variants of encryption that are resilient to various attacks, such as key leakage and key-dependent-messages. However, no composable VBB-obfuscators for point programs have been shown. We show composable *VGB*-obfuscators for point programs under a strong variant of the Decision Diffie Hellman assumption. We show that VGB (instead of VBB) obfuscation still suffices for the above applications, as well as for new applications. This includes extensions to the public key setting and to encryption schemes with resistance to certain related key attacks (RKA).

**Key words.** Obfuscation, Strong Simulation, Composable Point Obfuscation, Strong Encryption, Decision Diffie Hellman.

*Supported by The Check Point Institute for Information Security, Marie Curie grant PIRG03-GA-2008-230640, ISF Grant 1144/09, and NSF grant 1218461. An extended abstract of this paper appears in the proceedings of Crypto'10.

# Contents

# 1 Introduction

Informally, an obfuscator is an algorithm which gets as input a program (e.g., a Turing machine or circuit) and outputs a new program that has the same functionality as the original one, but is otherwise "unintelligible". The rigorous study of obfuscation was initiated in the work of [BGI$^+$01], who introduced the concept of *virtual black box* security (VBB in short). This concept requires the obfuscated program to behave like a "black box", in the sense that it should not leak any information about the program except its input-output behavior. More precisely, any *efficient* adversary with access to an obfuscated program can be simulated by an *efficient simulator* with only oracle access to the program. The same work presented the impossibility of *"universal VBB obfuscation"*, showing a family of programs that cannot be VBB obfuscated.

In light of this negative result, subsequent work has included several research directions. One line of work extends the result of [BGI$^+$01], ruling out obfuscation in various settings [GK05, Wee05]. Another line of work is aimed at constructing obfuscators for specific program families that are not ruled out by the universal impossibility result. Here, if we stick to VBB obfuscators, our knowledge is essentially limited to obfuscating point programs and their extensions [Can97, CMR98, LPS04, DS05, Wee05, CD08, CV09, CRV10]. A point program $P_v : \mathbb{D}_n \to \{0,1\}$ holds a value $v \in \mathbb{D}_n$ in its code and accepts its input $x$ iff $x = v$. We only know how to obfuscate point programs in which the point $v$ is explicitly obtainable from the code. Moreover, the known constructions depend on rather strong hardness assumptions, which was shown to be somewhat inherent in [Wee05].

A third line of work focuses on relaxations of VBB. In this context, [BGI$^+$01] suggested the notion of *indistinguishability obfuscators* (INDO), according to which obfuscations of two related size programs, implementing the same functionality, should be indistinguishable to any *efficient* adversary. Another relaxation, called *best possible obfuscation* (BPO) [GR07], requires that any information which the obfuscation leaks is efficiently learnable from any other program with the same functionality and related size (hence "best possible"). These two notions turn out to be equivalent, when restricted to efficient obfuscators.

Both the INDO and BPO notions are easier to satisfy than VBB; however, the security guarantee they provide is less clear. Unlike VBB, both seem to lose their meaning for a relatively wide range of program classes that are natural candidates for obfuscation. For instance, these notions become meaningless if we allow the obfuscator to work only when the program is given in some *"canonical"* representation, in which case no two programs have the same functionality. Another relaxation requires the obfuscation to be secure only when the program is sampled from some adequate distribution (rather than requiring security for any program in the family). This was done in the context of *perfect one-way hashing* [CMR98], *point proximity testing* [DS05], *re-encryption* [HRSV07] and more [AW07, HMLS07, Had10]. However, in some scenarios such a relaxation does not capture the security properties we would expect from an obfuscation.

A natural goal is thus to come up with a notion of secure obfuscation that is both meaningful and achievable. Here, there is room to consider notions which might be meaningful only for certain program families but not for all.

## 1.1 This Work

We study a new relaxation of VBB security notion for obfuscators. The requirement is that an obfuscation leaks no information about the program, rather than what can also be learned by an *all-powerful learner* that witnesses only a limited number of input-output pairs (at his choice).

More formally, any *efficient* adversary with access to an obfuscated program can be simulated by an *all-powerful simulator* with polynomially many oracle queries to the program (in contrast to poly-time simulation which VBB requires). For lack of better name, we call this notion *virtual grey box* (or VGB in

short). The extra power given to the simulator is intended to allow it to "reverse engineer" the adversary while avoiding technical difficulties that might be irrelevant to the overall goal. In certain cases (such as "highly unlearnable" programs), this could be done without losing too much of the meaningfulness of the guarantee.

**Relationship with existing notions.** VGB obfuscation is clearly weaker than VBB obfuscation. In particular, a VGB obfuscation is allowed to leak information which a VBB obfuscation cannot. Formally, we show that VGB is strictly weaker, demonstrating a family of programs that cannot be VBB obfuscated but is (trivially) VGB obfuscatable. On the other hand, we show that VGB obfuscators are stronger than the indistinguishability obfuscators and best possible obfuscators (INDO and BPO) mentioned above. To do so, we observe that, even if we further weaken the VGB security requirement by allowing the simulator an unlimited number of oracle queries, it still implies INDO and BPO.

For Turing machine obfuscators, the impossibility result of [BGI+01] extends to rule out *"universal VGB obfuscation"*. However, we could not rule out universal VGB *circuit obfuscators* (see more details within regarding this difference). We note that [GR07] show impossibility of strong universal BPO obfuscation that can handle even circuits that use *random oracle* gates. This impossibility applies to the stronger VGB notion.

**A setting where VGB is both meaningful and achievable.** Like INDO and BPO, VGB is not strong enough for some desirable obfuscation tasks; its weakness might be revealed in cases where an all-powerful simulator, even with limited oracle access to the program, has a clear advantage over a bounded simulator (as is often the case for "cryptographic functionalities" such as, say, pseudo-random functions). In general, it seems that VGB is mostly meaningful for program classes which are *unlearnable* with only polynomially many queries *even for learners with unbounded computation time*. We demonstrate concrete obfuscation tasks where VGB obfuscation is both meaningful and achievable (under appropriate hardness assumptions) while VBB is not known to be achievable.

The main task we consider is that of *composable obfuscation of point programs*. A point program obfuscator is $t$-composable if any adversary that has access to $t$ obfuscated programs can be simulated, given only oracle access to the programs. Ideally, $t$ could be any polynomial.

As in other cryptographic settings where composability is studied, here too the goal is to construct obfuscators for more elaborate types of programs from obfuscators of a simpler type, namely, point obfuscators. As an important example, in the context of VBB obfuscation, composable point obfuscators were shown to suffice for obfuscating *multi-bit* point programs (MBPP). An MBPP has two hidden values $(k, m)$ in its code. It returns $m$ on input $k$, and $\perp$ on any other input. MBPP obfuscators (MBPOs) were, in turn, shown to imply strong symmetric encryption schemes that are simultaneously secure against weakly random keys (i.e., keys with any super-log entropy) and key dependent messages (KDM) [CKVW10]. However, as natural and fruitful as the composability property may seem, none of the known point program obfuscators were shown to be composable (with respect to VBB).

We show that, with respect to VGB obfuscation, composable point obfuscators do exist, under appropriate hardness assumptions. Specifically, we show that, under a strong variant of the Decision Diffie Hellman assumption, the point program obfuscator from [Can97] is VGB-composable for any polynomial number of instances. (The mentioned assumption is a natural extension of the one used in [Can97] to show VBB security, without addressing composability).

We then show that VGB composable point obfuscators suffice for constructing MBPOs that are VGB composable on their own. This yields very strong encryption schemes that are resilient to a variety of attacks. This includes the aforementioned KDM and weak keys resilience, as well as new implications to resistance to certain *related key attacks* (RKA) [AHI11]. We also show that, given an extra re-randomization property (that the [Can97] obfuscator has), the encryption schemes can also be extended to the public key setting. We remark that the result for KDM encryption should be contrasted with the

fact that fully KDM-secure encryption schemes cannot be proven secure using *fully black-box reductions* to *efficiently falsifiable assumptions* [HH09]; indeed, the assumption that a given construction is a VGB obfuscator is not an efficiently falsifiable one. (In fact, Wee [Wee05] shows that VBB point obfuscators cannot be constructed using black-box simulators; Wee's result also applies for VGB point obfuscators, as VBB and VGB are equivalent for the case of a single point.)

## 1.2 Our Techniques

Proving composability for point obfuscators encounters several difficulties. We sketch these difficulties, as well as the ideas and techniques we use to overcome them. In particular, we exhibit how the VGB relaxation comes to our aid.

**Simulation and distributional indistinguishability.** Taking a similar approach to [Can97, Wee05] (for non-composable point obfuscation), we first study an appropriate indistinguishability notion of obfuscation and then show how it leads to composable VGB point obfuscation.

Ideally, we might try to require that for fixed sequence of points, the resulting obfuscated point programs would appear to an efficient adversary as a sequence of obfuscated *random* point programs (similarly to the *semantic security* requirement for encryption schemes). This would allow simple simulation, by running the adversary on obfuscations of random hidden values. However, in the context of obfuscation such a requirement is unachievable, since the adversary is able to run the program and verify any guesses it might have; in particular it can have some hardwired values which it can always recognize. Instead, we consider a weaker requirement which we call *Distributional Indistinguishability* (DI in short). We show that: (a) DI is necessary and sufficient for constructing VGB simulators, and (b) it is achievable under appropriate hardness assumptions.

DI is an extension of a notion used in [Can97] in the context of single point (non-composable) obfuscators. The requirement refers to *coordinatewise well spread* (CWS) distributions over tuples, where each coordinate has super-logarithmic min-entropy. In other words, $\left\{(X_n^{(1)} \dots X_n^{(t)})\right\}$ is a CWS distribution ensemble on $\left\{\mathbb{D}_n^t\right\}$ if, for any $a \in \mathbb{D}_n$ and $i \in [t]$, $X_i \neq a$, except with negligible probability.

Essentially, $\mathcal{O}$ is a *$t$-DI obfuscator* if for any CWS distribution $\mathcal{X}$, over $t$-tuples of elements in $\mathbb{D}_n$, no *efficient* adversary can distinguish obfuscations of $t$ uniform values from obfuscations of a tuple of values sampled from $\mathcal{X}$. We show:

**Theorem 1.1** (informal)**.** *If $\mathcal{O}$ is a $t$-DI point obfuscator, then it is a $t$-composable VGB point obfuscator. Moreover, if $\mathcal{O}$ is $t$-DI for any polynomial $t$, then it is a composable VGB point obfuscator, for any polynomial number of point programs.*

The main technical difficulty in this work is in proving Theorem 1.1. We sketch the ideas used in the proof. Our starting point is a result of [Can97] showing that for point obfuscators (i.e. $t = 1$) the notions of DI and VBB obfuscation are equivalent and that DI obfuscation is achievable under certain number theoretic assumptions.

First, we ask whether $t$-DI obfuscators imply $t$-composable VBB obfuscators for $t > 1$. We show that this is the case as long as $t = O(1)$. However, when $t = \omega(1)$, major (and seemingly inherent) difficulties arise. Specifically, recall that, when constructing a simulator, we should deal with the fact that the adversary can run the obfuscated programs and might have some hardwired values that it can always recognize. When the adversary has access only to a single obfuscated point program, [Can97, Wee05, CKVW10] show that, in fact, it cannot do much more than have a polynomial number of such hardwired test elements. We call these the *distinguishing elements*. This allows hardwiring into the simulator the polynomially many distinguishing elements, and having it query its oracle to the circuit only on these elements.

However, in the case of multiple obfuscated points, this plan does not go through. The main difficulty is *adaptivity*. More specifically, while in the case of a single hidden point there is only a single secret, in the case of composable point obfuscators, the adversary might first discover only some of the points, and then use this partial information to make his next choices. In other words, in addition to having an "initial set of hardwired distinguishing elements", the adversary may adaptively compute "new distinguishing elements", after having discovered some of the hidden points.

Fortunately, we can show that, for any partial information already learned (which is, intuitively, the subset of hidden points that the adversary have revealed so far), there is a corresponding poly-size set of "new distinguishing elements". Still, there remains the question of how can the simulator compute these elements ahead of time.

We show that the total number of potentially queried elements is $n^{\Theta(t)}$. Here, when $t = \omega(1)$, VGB comes to our aid. That is, having limited oracle access to the point programs and sufficient power to compute the distinguishing elements allows performing the required simulation.

We remark that a converse statement is also true; that is, DI is necessary for VGB composable obfuscation (and thus also for the stronger VBB notion).

**A $t$-DI point obfuscator.** Finally, we reconsider the point program obfuscator constructed in [Can97]. Under a strong variant of the Decision Diffie Hellman assumption (SDDH), we show that this obfuscator is $t$-DI for any polynomial $t$ and hence is a $t$-composable VGB point obfuscator. As evidence of plausibility, we show that our assumption holds in the *Generic Group Model* [Sho97], where algorithms are only allowed to perform generic group operations and cannot exploit the representation of group elements. We note that there exist well studied group ensembles (e.g. Quadratic Residues modulo a prime, and Elliptic Curves groups) where the best cryptanalytic techniques are in fact generic ones [Bon98].

**Relation to previous POs and MBPOs.** As mentioned above, VBB point obfuscators (POs), for point programs with a single output bit, were constructed in [Can97] and [Wee05]. Also, in [Wee05], the construction was extended to point programs with $\log(n)$ output bits. In contrast, VBB obfuscators for programs with $\mathrm{poly}(n)$ output bits (MBPOs) are only known assuming *composable* point obfuscation [CD08], or for the restricted case that the output $m$ is independent of the key $k$ [CD08, CKVW10]. Likewise, the applications to RKA and KDM encryption mentioned in this work also require composability. However, Composable (VBB) point obfuscators cannot be obtained, in general, from single point obfuscators, even if these satisfy a stronger notion of security with auxiliary input (see Appendix A).

Nevertheless, the point obfuscators mentioned above [Can97, Wee05] may still be composable; the question, however, is under what kind of assumptions. Here, we would clearly like to rely on assumptions that do not go as far as assuming that a given obfuscator is composable (i.e., assuming that a simulator exists). Instead, as done in [Can97, Wee05] for the case of non-composable obfuscation, we formulate a distributional indistinguishability requirement that can be obtained from corresponding indistinguishability assumptions (as the variants of DDH mentioned above). Then, we try to explicitly construct the simulator; however, due to the difficulties explained above we only manage to obtain VGB simulation.

**Organization.** Section 3 recalls the VBB notion and some of its previous relaxations. Section 4 is devoted to the definition and discussion of VGB obfuscation and its relative place in the obfuscation field. Section 5 shows a general way to construct composable VGB point obfuscators for point programs. Section 6 discusses a construction of a composable point obfuscator under a number theoretic assumption. Section 6.1 discusses the nature and plausibility of our hardness assumption. Section 7 demonstrates the applications of composable point obfuscators to obfuscation of set programs and multi-bit point programs as well as to strong encryption schemes. In Appendix A, we discuss the relation between

obfuscation with auxiliary input and composability. In Appendix B, we further explain the relations between some of the point obfuscation definitions provided in this work and those provided in [Can97].

## 2 Preliminaries

**Turing machines, circuits, and adversaries.** TM is shorthand for Turing machine. By circuit we refer to a standard boolean circuit with logical gates (taken from some universal system). Most of the adversaries in this work are represented by circuit families of polynomial size. By PPT we refer to probabilistic polynomial-time Turing machines. We say that an algorithm (circuit family) is efficient if it is PPT (or poly-size in the case of circuits). For a function $f$ we denote by $\mathcal{A}^f$ an algorithm $\mathcal{A}$ with oracle (black-box) access to $f$.

**Distributions, indistinguishability, and min-entropy.** We say that a function $\nu : \mathbb{N} \to \mathbb{N}$ is negligible, if $\nu(n) = n^{-\omega(1)}$ (i.e. it decays faster than any polynomial). Given an ensemble of domains $\mathcal{D} = \{\mathbb{D}_n\}$, we denote by $\mathcal{U}(\mathcal{D}) = \{U(\mathbb{D}_n)\}$ the ensemble of uniform, distributions on this domain (we may omit the brackets when it is clear what is the domain). Given two distribution ensembles $\mathcal{X} = \{X_n\}_{n\in\mathbb{N}}, \mathcal{Y} = \{Y_n\}_{n\in\mathbb{N}}$, where $\mathrm{Supp(X_n)} \cup \mathrm{Supp(Y_n)} \subseteq \{0,1\}^{\mathrm{poly(n)}}$, we say that $\mathcal{X}$ is computationally indistinguishable from $\mathcal{Y}$, if for any poly-size adversary $\mathcal{A}$ there exists a negligible $\nu$ such that for all sufficiently large $n$: $\left|\Pr_{x\leftarrow X_n}[\mathcal{A}(x) = 1] - \Pr_{y\leftarrow Y_n}[\mathcal{A}(y) = 1]\right| \leq \nu(n)$. We denote the latter by $\mathcal{X} \approx_c \mathcal{Y}$. For a distribution ensemble $\mathcal{X} = \{X_n\}_{n\in\mathbb{N}}$ and an algorithm $\mathcal{A}$, $\mathcal{A}(X_n)$ denotes the distribution induced by running (the probabilistic) $\mathcal{A}$ on an input sampled from $X_n$. $\mathcal{A}(\mathcal{X}) = \{\mathcal{A}(X_n)\}_{n\in\mathbb{N}}$ denotes the corresponding distribution ensemble. The min-entropy of a distribution $X$ is defined as $H_\infty(X) = \min_{x\in\mathrm{Supp(X)}} \log \frac{1}{\Pr[X=x]}$.

## 3 Definitions

We recall the *virtual black box* (VBB) definition and two of its previous relaxations. In all following definitions, we consider the task of obfuscating an ensemble $\mathcal{C} = \{\mathcal{C}_n\}$, where each $\mathcal{C}_n$ is a collection of circuits with input length $n$ and $\mathrm{poly}(n)$ size.

**Definition 3.1** (*Obfuscator* [BGI$^+$01] )**.** *A PPT $\mathcal{O}$ is a VBB obfuscator for $\mathcal{C}$, if it satisfies:*

- **Functionality.** *For any $n \in \mathbb{N}$, $C \in \mathcal{C}_n$, $\mathcal{O}(C)$ is a circuit that computes the same function as $C$.*
- **Polynomial slowdown.** *There is a polynomial $q$ such that, for any $n \in \mathbb{N}$, $C \in \mathcal{C}_n$, $|\mathcal{O}(C)| \leq q(|C|)$.*
- **Virtual black-box**.[1] *For any poly-size adversary $\mathcal{A}$, and polynomial $p$, there is a poly-size simulator $\mathcal{S}$ such that for all sufficiently large $n \in \mathbb{N}$ and $C \in \mathcal{C}_n$:*

$$\left|\Pr_{\mathcal{A},\mathcal{O}}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{\mathcal{S}}[\mathcal{S}^C(1^{|C|}) = 1]\right| \leq \frac{1}{p(n)} \ .$$

**Definition 3.2** (*Indistinguishability Obfuscation* [BGI$^+$01])**.** *$\mathcal{O}$ is said to be an* indistinguishability obfuscator *(INDO) for $\mathcal{C}$, if it satisfies the* functionality *and* polynomial slowdown*, and for any ensemble of circuit pairs $\mathcal{C}^{(1)} \times \mathcal{C}^{(2)} = \{(C_n^{(1)}, C_n^{(2)}) \in \mathcal{C}_n \times \mathcal{C}_n\}$, where the two circuits in each pair are of the same size and functionality, it holds that:*

$$\mathcal{O}(\mathcal{C}^{(1)}) \approx_c \mathcal{O}(\mathcal{C}^{(2)}) \ .$$

[1] As noted by [BGI$^+$01], the following can be replaced with the equivalent requirement that $\left|\Pr[\mathcal{A}(\mathcal{O}(C) = \pi(C)] - \Pr[\mathcal{S}^C(1^{|C|}) = \pi(C)]\right| \leq \frac{1}{p(n)}$, for any predicate $\pi : \mathcal{C}_n \to \{0,1\}$. Also the size of the simulator can depend on $p(n)$, namely the required simulation quality.

Another relaxation of VBB is *Best Possible Obfuscation* (BPO) [GR07]. Here, the requirement is that any information that the obfuscation leaks is efficiently learnable from any other circuit with the same functionality and related size (hence it is "best possible"). The two definitions are equivalent when the obfuscator is required to be a PPT [GR07].

Before presenting our definition, in the next section, we make the following preliminary observation regarding the nature of the above relaxations. The INDO definition is in fact equivalent to a weak black-box definition that allows an unbounded simulator with unlimited number of oracle queries .

**Proposition 3.1.** *$\mathcal{O}$ is an indistinguishability obfuscator, for an ensemble of circuits $\mathcal{C} = \{\mathcal{C}_n\}$, iff for any efficient distinguisher $\mathcal{A}$ and polynomial $p$, there is a (possibly inefficient) simulator $\mathcal{S}$, such that for all large enough $n$ and $C \in \mathcal{C}_n$:*

$$\left| \Pr_{\mathcal{A},\mathcal{O}}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{\mathcal{S}}[\mathcal{S}^C(1^{|C|}) = 1] \right| \leq \frac{1}{p(n)} \ .$$

*Proof.* Assume $\mathcal{O}$ is an obfuscator for $\mathcal{C} = \{C_n\}$ satisfying the unbounded simulation property. Let $\mathcal{A}$ be an efficient distinguisher and let $\mathcal{C}^{(1)} \times \mathcal{C}^{(2)} = \{(C_n^{(1)}, C_n^{(2)}) \in \mathcal{C}_n \times \mathcal{C}_n\}_{n\in\mathbb{N}}$ be an ensemble of circuit pairs of the same functionality and of the same size, $c_n$. Then for any $c \in \mathbb{N}$, there exists a simulator $\mathcal{S} = \mathcal{S}_c$, such that for any large enough $n \in \mathbb{N}$, $i \in \{1, 2\}$:

$$\left| \Pr[\mathcal{A}(\mathcal{O}(C_n^{(i)})) = 1] - \Pr[\mathcal{S}^{C_n^{(i)}}(1^{c_n}) = 1] \right| \leq n^{-c} \ .$$

Moreover, since $C_n^{(1)}, C_n^{(2)}$ compute the same function, obviously:

$$\Pr[\mathcal{S}^{C_n^{(1)}}(1^{c_n}) = 1] = \Pr[\mathcal{S}^{C_n^{(2)}}(1^{c_n}) = 1] \ ,$$

implying that for any $c \in \mathbb{N}$ and large enough $n$:

$$\left| \Pr[\mathcal{A}(\mathcal{O}(C_n^{(1)})) = 1] - \Pr[\mathcal{A}(\mathcal{O}(C_n^{(2)})) = 1] \right| \leq 2n^{-c} \ .$$

For the converse , assume $\mathcal{O}$ is an *indistinguishability obfuscator* for $\mathcal{C}$. Consider the unbounded simulator that gets as input $1^c$ as well as oracle access to a function $f$, and operates as follows. It first learns the function and produces a circuit $\tilde{C} \in \mathcal{C}_n$ of size $|\tilde{C}| = c$ that computes the function. Then it computes an obfuscation $\mathcal{O}(\tilde{C})$ and feeds it as input to the simulated adversary. The result follows directly from the indistinguishability condition. □

The definitions above concern obfuscators for circuits. That is, both the input program and the output of the obfuscator are given by circuits. One can naturally adjust these definitions to fit the case of *Turing Machine obfuscators* (both input and output are given by a description of a TM). We next give the VBB TM definition. In what follows we assume all TM's have some canonical description. By $\mathcal{A}(M)$ we mean that the algorithm $\mathcal{A}$ gets as input the description of $M$. In addition, all TM's discussed have some timeout mechanism (i.e. they always halt and output).

**Definition 3.3** (*VBB TM obfuscator* [BGI$^+$01])**.** *A PPT $\mathcal{O}$ is a VBB obfuscator for a TM family $\mathcal{M}$, if it satisfies:*

- **Functionality.** *For any $M \in \mathcal{M}$, $\mathcal{O}(M)$ is a TM that computes the same function as $M$.*
- **Polynomial slowdown.** *There is a polynomial $q$ such that, for any $M \in \mathcal{M}$, $|\mathcal{O}(M)| \leq q(|M|)$, and for any $x \in \{0,1\}^*$, if $M(x)$ performs at most $t$ steps, then $\mathcal{O}(M)(x)$ performs at most $q(t)$ steps.*
- **Virtual black-box.** *For any poly-size adversary $\mathcal{A}$ and polynomial $p$, there is a poly-size simulator $\mathcal{S}$ such that, for all sufficiently large $n \in \mathbb{N}$ and $M \in \mathcal{M}$ of description size $|M| = n$:*

$$|\Pr[\mathcal{A}(\mathcal{O}(M)) = 1] - \Pr[\mathcal{S}^M(1^n) = 1]| \leq \frac{1}{p(n)} \ ,$$

*where the probability is taken over the coins of $\mathcal{A}, \mathcal{S}$ and $\mathcal{O}$.*

# 4 VGB Obfuscation

In this section, we formalize the notion of *virtual grey box obfuscation with strong simulators*, and explore its relation to existing notions. The new definition relaxes the VBB security requirement by allowing the simulator to have more computational power. However, we still restrict the number of oracle queries it is allowed to make. The *functionality* and *polynomial slowdown* requirements should be satisfied as in Definition 3.1. The VBB requirement is replaced by the following. Denote by $C[q]$ an oracle to the circuit (function) $C$ that allows at most $q$ queries.

**Definition 4.1** (*Virtual Grey Box - obfuscation with a strong simulator*)**.** *A PPT $\mathcal{O}$ has the VGB property, if for any PPT adversary $\mathcal{A}$ and polynomial $p$, there is a (possibly* inefficient*) simulator $\mathcal{S}$ and a polynomial $q$ such that for all sufficiently large $n \in \mathbb{N}$ and any $C \in \mathcal{C}_n$:*

$$\left| \Pr_{\mathcal{A},\mathcal{O}}[\mathcal{A}(\mathcal{O}(C)) = 1] - \Pr_{\mathcal{S}}[\mathcal{S}^{C[q(n)]}(1^{|C|}) = 1] \right| \leq \frac{1}{p(n)} \ .$$

*Remark* 4.1. This definition can be naturally adjusted to the case of *Turing Machine obfuscators*, by replacing the simulator in (Definition 3.3) with an unbounded simulator with polynomially many queries.

**When Is VGB Meaningful?** Like INDO and BPO, VGB obfuscation does not seem strong enough for some desirable obfuscation tasks. Examples include: transforming private key encryption schemes to public ones and constructing homomorphic encryption schemes[2]. Informally, the problem in these scenarios is that the obfuscated program computes some kind of cryptographic functionality that does not remain secure in the presence of unbounded simulators. In general, it seems that VGB is mostly meaningful for program classes that are *unlearnable* with only polynomially many queries, even for learners with unbounded computation time. For other program families, VGB might not guarantee the required security (in the proof of Proposition 4.1, we describe such a family).

## 4.1 VGB Vs. VBB and INDO

**VGB is strictly weaker than VBB.** The VGB definition is clearly implied by the VBB definition. We show that, in fact, it is strictly weaker. That is, we show a family that cannot be obfuscated according to the VBB definition but is (trivially) obfuscatable under the weaker VGB definition. To do so, we use a slight variation of the family constructed in the [BGI$^+$01] impossibility result.

**Proposition 4.1.** *Assuming the existence of one-way permutations, there exists a family of programs that is not VBB obfuscatable but is VGB obfuscatable.*

To prove the above proposition, we use the notion of TM obfuscation. This choice is only for the sake of simplicity; indeed, constructing TM families that are not VBB-obfuscatable is technically much simpler than constructing such circuit families as reflected in [BGI$^+$01]. The separation can be extended to case of circuit families using *pseudo random functions* that are *exactly learnable* for unbounded adversaries with polynomially many queries (which can also be easily constructed from one-way permutations).

We recall the definition of one way permutations and then turn to prove Proposition 4.1

**Definition 4.2** (*One Way Permutation*)**.** *A family of permutations $f = \{f_n : \{0,1\}^n \to \{0,1\}^n\}_{n\in\mathbb{N}}$ is a one-way permutation if $f$ is* efficiently *computable and for any poly-size $\mathcal{A}$:*

$$\Pr_{x \xleftarrow{U} \{0,1\}^n}[\mathcal{A}(f_n(x)) = x] = n^{-\omega(1)} \ .$$

[2] See the section on applications in [BGI$^+$01] for more details.

*Proof of Proposition 4.1.* We describe a family of TM's that is not VBB obfuscatable but is (trivially) VGB obfuscatable. We use a slight variation of the family constructed in the negative result of [BGI$^+$01] (for TM's). For $n \in \mathbb{N}, \alpha, \beta \in \{0,1\}^n$ , consider TM's $C_{\alpha,\beta}, D_{\alpha,\beta}$ with the following functionality:

$$C_{\alpha,\beta}(x) = \begin{cases} \beta & x = \alpha \\ \perp & \text{otherwise} \end{cases}$$

$$D_{\alpha,\beta}(M, 1^t) = \begin{cases} 1 & M \text{ halts on input } \alpha \text{ with output } \beta \text{ after at most } t \text{ steps} \\ \perp & \text{otherwise} \end{cases} .$$

We assume that both TM's have descriptions of size $\Theta(n)$, that $C_{\alpha,\beta}$ runs at most $\delta n$ steps for some constant $\delta$, and $D_{\alpha,\beta}$ runs in poly-time (in its input length). We also assume that given $(\alpha, \beta)$, both TM's can be generated (by another TM) in time $\mathrm{poly}(n)$. Let $f = \{f_n : \{0,1\}^n \to \{0,1\}^n\}_{n\in\mathbb{N}}$ be a *one way permutation* family. For any $n \in \mathbb{N}, \alpha, \beta, \beta' \in \{0,1\}^n$ define another TM:

$$F_{\alpha,\beta,\beta'}(i, s) = \begin{cases} C_{\alpha,\beta}(s) & i = 1 \\ D_{\alpha,\beta'}(s) & i = 2 \\ f_n(\alpha), f_n(\beta') & i = 3 \end{cases} .$$

Consider the corresponding families $\mathcal{F}_n = \{F_{\alpha,\beta,\beta'}\}$ and $\mathcal{F} = \bigcup_{n\in\mathbb{N}} \mathcal{F}_n$. We first claim that $\mathcal{F}$ is trivially VGB obfuscatable, the obfuscator is just the identity function. An unbounded simulator can invert $f_n$, retrieve $\alpha, \beta'$, use its oracle to compute $\beta = C_{\alpha,\beta}(\alpha)$ and run the simulated adversary on the corresponding $F_{\alpha,\beta,\beta'}$. We now show that $\mathcal{F}$ is not VBB obfuscatable. Indeed, let $\mathcal{O}$ be any candidate for obfuscation, and let $c$ be the polynomial slowdown constant such that for any $F \in \cup\mathcal{F}_n$, it holds that $|\mathcal{O}(F)| \leq |F|^c$ and if $F(x)$ halts after $t$ steps then $(\mathcal{O}(F))(x)$ halts after at most $t^c$ steps with the same output. Let $\mathcal{A}$ be the adversary that, given a program $P$ as input, where $|P| = k$, first computes the code of a new program $C = P(1, .)$ that, given any input $x$, runs $P(1, x)$. Then, the adversary computes $P(2, C, (\delta k)^c)$. $\mathcal{A}$ runs in poly time for any input in $\mathcal{F}$ [3]. Furthermore, for any $\alpha, \beta, \beta' \in \{0,1\}^n$, it holds that $\mathcal{A}(\mathcal{O}(F_{\alpha,\beta,\beta'})) = 1$ iff $\beta = \beta'$ [4]. On the other hand, for a **randomly chosen** $\alpha, \beta, \beta' \in \{0,1\}^n$ any *efficient* simulator, with nothing but black box access to $F_{\alpha,\beta,\beta}$, fails to determine whether $\beta = \beta'$, except with negligible probability. Indeed, the simulator fails to learn anything from its oracle except for $f_n(\alpha), f_n(\beta')$ (i.e. sees only $\perp$) as long as it never queries on $(1, \alpha)$ nor on $(2, C, 1^t)$, where $C$ is the code of a TM which on input $\alpha$ returns $\beta'$ in time $t$. The latter happens only with negligible probability; otherwise, we could construct a poly-size adversary which inverts $f$. Indeed, given an adversary $\mathcal{A}$ that on input $f_n(\alpha), f_n(\beta')$ (for random $\alpha, \beta'$) produces a program $C$ that on input $\alpha$ outputs $\beta'$ in time $\mathrm{poly}(n)$, we could invert $f$ as follows. On input $f_n(\beta')$ choose a random $\alpha$ and compute $f_n(\alpha)$, then use $\mathcal{A}$ to create the program $C$ and run it on $\alpha$. Ruling out an adversary that on input $f_n(\alpha), f_n(\beta')$ outputs $\alpha$ is straight forward. It follows that $\mathcal{F}$ is not VBB obfuscatable. □

**VGB implies INDO (BPO).** The relation between VGB obfuscation and INDO (BPO) follows from Proposition 3.1. That is, even when VGB is further weakened by allowing the (unbounded) simulator unlimited oracle access, it still implies INDO and (for *efficient* obfuscators) BPO.

## 4.2 Impossibility Results

We consider the possibility of *"universal VGB obfuscation"*. That is, could there exists a VGB obfuscator for the class of all programs? We observe that for TM's obfuscators the impossibility result of

[3] Formally, one should also set a time out mechanism to deal with other inputs.

[4] Note that $k = |\mathcal{O}(F_{\alpha,\beta,\beta'})| \geq n$ so $\mathcal{A}$ is allowed $\mathrm{poly}(n)$ steps

[BGI+01] extends and also applies for VGB obfuscation. However, for circuits obfuscators, the separation shown in [BGI+01] no longer holds. Essentially, the reason for this difference is that the *VBB unobfuscatable circuit family* constructed by [BGI+01] includes cryptographic functionalities (such as *encryption schemes* and *pseudo random functions*) that fail to remain secure in the presence of unbounded simulators (even with limited oracle access). We could not rule out universal VGB obfuscation in the circuit case.

We note that [GR07] show impossibility of universal BPO obfuscation for circuits that are allowed to use *random oracle* gates. Their result also applies for the stronger VGB notion; however, the meaning of an impossibility result in such a setting is somewhat less clear.

### 4.3 VGB Obfuscation with Auxiliary Information

A more general notion of obfuscation is obfuscation with auxiliary information (for either VBB or VGB). In this setting, the adversary also has some prior information regarding the obfuscated circuit. This notion was previously studied in [GK05] who showed impossibility results for VBB obfuscation with auxiliary input. While, for VBB, obfuscation with auxiliary input seems to be a stronger notion than plain VBB obfuscation, we show that, for VGB, it actually does not add any extra power. In Appendix A we give the formal definitions and prove this result (Proposition A.3).

## 5 Composable Point Obfuscators

In this section, we define composable VGB point obfuscators and study the relation between the natural simulation-based definition and an indistinguishability-based definition. In Section 6, we study a concrete construction satisfying the indistinguishability-based definition, under appropriate number-theoretic assumptions.

### 5.1 Composition of Obfuscators

One central question in the context of obfuscation is the question of *composition*, which asks when and whether is it secure to obfuscate a sequence of programs by obfuscating each program on its own and combining the obfuscated programs. There are several forms of *composition* one could consider, in this work we consider one specific form, namely *composition by concatenation* [LPS04].

**Definition 5.1** (*$t$-Composable Obfuscation* [LPS04])**.** *A PPT $\mathcal{O}$ is a* $t$-composable obfuscator *for a circuit ensemble $\mathcal{C} = \{\mathcal{C}_n\}$ if it satisfies the* functionality *and* polynomial slow-down *requirements, as in Definition 3.1, and for any poly-size binary adversary $\mathcal{A}$ and polynomial $p$, there is a simulator $\mathcal{S}$, such that for any sequence of circuits $C^1, \ldots, C^t \in \mathcal{C}_n$ (where $t = \mathrm{poly}(n)$), and any sufficiently large $n$:*

$$\left|\Pr[\mathcal{A}(\mathcal{O}(C^1), \ldots, \mathcal{O}(C^t)) = 1] - \Pr[\mathcal{S}^{C^1,\ldots,C^t}(1^{|C^1|}, \ldots, 1^{|C^t|}) = 1]\right| \leq \frac{1}{p(n)} \ ,$$

*where $C^1, \ldots, C^t$ gets as input $(x, i)$ and returns $C^i(x)$.*

*Remark* 5.1. A special case of *$t$-composability* is *$t$-self-composability*, where $C^1 = C^2 = \cdots = C^t$. This captures the requirement that multiple obfuscations of the same point would not reveal more information than a single obfuscation of that point.

*Remark* 5.2. [LPS04] naturally refer to VBB obfuscation, i.e. the simulator $\mathcal{S}$ is polynomially bounded. We consider the definition also for VGB obfuscators; i.e., we allow the simulator to be unbounded with polynomially many oracle queries.

## 5.2 Point Obfuscators

**Point circuits.** For a security parameter $n \in \mathbb{N}$ and a domain $\mathbb{D}_n$, a point circuit $C_x : \mathbb{D}_n \to \{0,1\}$ returns 1 on input $x$ and 0 on all other inputs. The point circuits we discuss are given in some "canonical" form where the point $x$ is explicit. As the size of the canonical circuits is determined by the parameter $n$, we simplify our notation by letting the simulator take input $1^n$ (instead of the circuit size). The natural choice for the domain is $\mathbb{D}_n = \{0,1\}^n$. However, to avoid confusion when discussing tuples of points in $\mathbb{D}_n^t$, we shall stick to the more general notation. We refer to obfuscators for point circuits as point obfuscators.

**Is any point obfuscator composable?** Point obfuscators have been constructed, both in the plain model and in the random oracle model. A natural question is whether any VBB secure point obfuscator is also guaranteed to be composable (as in Definition 5.1). [LPS04] conjectured that the answer is negative. To support their conjecture they give a point obfuscator in the *Random Oracle model* that is not even 2-composable. In the standard model, it can be shown that if point obfuscators exist, then there are also point obfuscators which are not $\Omega(n)$-composable [CD08] (see further discussion in Appendix A) . In general, none of the constructions of point obfuscators were known to be composable.

**Does obfuscation with auxiliary information imply composability?** In the context of cryptographic protocols, auxiliary information is known to be tightly related to composability. A natural question is whether the same holds for obfuscation; in particular, whether point obfuscators with auxiliary input would imply composable point obfuscation. This was partially answered in [CD08] who showed that such an implication does not hold for a certain type of auxiliary information. In Appendix A, we extend this to a more general setting (Proposition A.1). However, we show that point obfuscation with auxiliary information does imply a more restricted notion of composability, namely constant-self-composability (Proposition A.2).

## 5.3 Distributional Indistinguishability and Composable Point Obfuscation

To overcome the difficulties in achieving composable point obfuscators, we explore in this section an additional property of point obfuscators called *Distributional Indistinguishability (or DI in short)*.[5] We will show that this additional property is necessary for composable obfuscation, even under the VGB notion, just as it is necessary for the stronger VBB notion. More importantly, we will show that, in fact, it suffices for VGB point obfuscation. The definition we present generalizes the DI definition presented in [Can97].

**Definition 5.2** (*Coordinatewise Well Spread Distribution*)**.** *Let $\mathcal{X} = \{X_n\}$ be an ensemble where each $X_n$ is a distribution on $\mathbb{D}_n^{t(n)}$ for a domain ensemble $\{\mathbb{D}_n\}$. We say that $\mathcal{X}$ is CWS if:*

$$\max_{a \in \mathbb{D}_n} \Pr_{\bar{x} \leftarrow X_n} [\exists i \in [t] : x_i = a] = n^{-\omega(1)} \ .$$

That is, any element has only a negligible chance of being picked within a vector sampled from the distribution. Equivalently, in a CWS ensemble the distributions $X_n^{(i)}$ all have super-log *min-entropy*; i.e., $\min_{i \in [t]} H_\infty(X_n^{(i)}) = \omega(\log n)$.

**Definition 5.3** (*Distributional Indistinguishability*)**.** *$\mathcal{O}$ is $t$-DI if for any CWS distribution ensemble, $\mathcal{X} = \{X_n = \langle X_n^{(1)}, \ldots, X_n^{(t)} \rangle\}$, it holds that:*

$$\mathcal{O}(C_{\mathcal{X}^{(1)}}), \ldots, \mathcal{O}(C_{\mathcal{X}^{(t)}}) \approx_c \mathcal{O}(C_{\mathcal{U}^{(1)}}), \ldots, \mathcal{O}(C_{\mathcal{U}^{(t)}}) \ ,$$

[5] DI should not be confused with *Indistinguishability Obfuscators* of [BGI+01], which were presented in Definition 3.2.

*where each $\mathcal{O}(C_{\mathcal{X}^{(i)}})$ is an ensemble of distributions on point obfuscations, and the hidden point is drawn from $\mathcal{X}^{(i)}$ and $\mathcal{U}^{(1)}, \ldots, \mathcal{U}^{(t)}$ are ensembles of independent uniform distributions over $\{\mathbb{D}_n\}$.*

We note that, for the case $t = 1$, Definition 5.3 is equivalent to the DI definition in [Can97] (see Appendix B). There, it is shown that for $t = 1$, DI and VBB are in fact equivalent. The proof there does not follow through for larger $t$. Nevertheless, we show:

**Theorem 5.1** (Restatement of Theorem 1.1)**.** *Any $t$-DI point obfuscator is a $t$-composable VGB point obfuscator. Moreover, for $t = O(1)$, it is VBB composable. Conversely, any $t$-composable VGB point obfuscator is $t$-DI.*

We first prove the second part of Theorem 5.1, which is simpler, and then prove the more involved second part. We start by introducing preliminary notation.

**Notations.** Given a vector of $t$ points $\bar{x} = \langle x_1, \ldots, x_t \rangle$, we abuse notation and denote by $C_{\bar{x}}$ the vector of point circuits $\langle C_{x_1}, \ldots, C_{x_t} \rangle$. We also denote by $\mathcal{O}(C_{\bar{x}})$ the composition $\mathcal{O}(C_{x_1}), \ldots, \mathcal{O}(C_{x_t})$. Speaking of vectors, we shall often be interested in the (unordered) set of their elements. Whenever we use set operators such as $\in, \cap, \cup$ on vectors, it should be interpreted as operating on the corresponding sets. For integers $s \leq t$ we denote by $\binom{[t]}{s}$ the family of subsets of $[t]$ of size $s$. For vectors $\bar{x}, \bar{z}$ of dimensions $s$ and $t-s$, and a set of indices $I \subseteq [t]$ of size $|I| = s$, we denote by $\text{CMB}_I(\bar{x}, \bar{z})$ the $t$-vector with the elements of $\bar{x}$ in coordinates $I$ and those of $\bar{z}$ in coordinates $[t] - I$ (the mapping is according to ascending order of indices).[6]

*Proof - Any $t$-composable VGB point obfuscator is $t$-DI.* Let $\mathcal{X}$ be a CWS distribution ensemble over vectors in $\mathbb{D}_n^t$ and let $\mathcal{A}$ be a binary poly-size adversary. By the VGB assumption, for any polynomial $p$, there exists an (unbounded) simulator $\mathcal{S}$ that is allowed $q = \text{poly}(n)$ many oracle queries and satisfies, for all $\bar{x} \in \mathbb{D}_n^t$ and sufficiently large $n$:

$$\left| \Pr_{\mathcal{A},\mathcal{O}}[\mathcal{A}(\mathcal{O}(C_{\bar{x}})) = 1] - \Pr_{\mathcal{S}}[\mathcal{S}^{C_{\bar{x}}[q]}(1^n) = 1] \right| \leq 1/4p \ . \tag{1}$$

It follows that, for large enough $n$:

$$\left| \Pr_{\substack{\bar{x} \leftarrow X_n \\ \mathcal{O},\mathcal{A}}}[\mathcal{A}(\mathcal{O}(C_{\bar{x}})) = 1] - \Pr_{\substack{\bar{u} \xleftarrow{U} \mathbb{D}_n^t \\ \mathcal{O},\mathcal{A}}}[\mathcal{A}(\mathcal{O}(C_{\bar{u}})) = 1] \right| \leq$$

$$\left| \Pr_{\bar{x} \leftarrow X_n}[\mathcal{S}^{C_{\bar{x}}[q]}(1^n) = 1] - \Pr_{\bar{u} \xleftarrow{U} \mathbb{D}_n^t}[\mathcal{S}^{C_{\bar{u}}[q]}(1^n) = 1] \right| + \frac{1}{2p} \ .$$

We can assume WLOG that $\mathcal{S}$ is deterministic (by fixing its coins to those that maximize the above difference). To conclude the claim observe that:

$$\left| \Pr_{\bar{x} \leftarrow X_n}[\mathcal{S}^{C_{\bar{x}}[q]}(1^n) = 1] - \Pr_{\bar{u} \xleftarrow{U} \mathbb{D}_n^t}[\mathcal{S}^{C_{\bar{u}}[q]}(1^n) = 1] \right| = n^{-\omega(1)} \ . \tag{2}$$

Indeed, for any CWS distribution $\mathcal{Y} = \{Y_n\}$ on vectors in $\mathbb{D}_n^t$, the probability that $\mathcal{S}$ queries an element of a vector sampled from $Y_n$ is at most $q \cdot \max_{a \in \mathbb{D}_n} \Pr_{\bar{y} \leftarrow Y_n}[\exists i \in [t] : y_i = a]$, which is negligible when $\mathcal{Y}$ CWS. Thus, $\mathcal{S}$ distinguishes any two CWS distributions, such as $\mathcal{X}$ and $\mathcal{U}$, with negligible probability. □

[6] For example $\text{CMB}_{\{2,5\}}((a,b),(c,d,e)) = (c,a,d,e,b)$.

**Proving the first part of Theorem 5.1 - a road map.** The proof follows the ideas presented in Section 1.2: our eventual goal is to establish that, for any partial information learned by the adversary (intuitively corresponding to hidden points the adversary revealed), there is a corresponding polynomial set of *distinguishing elements* that it may try to identify in its next set of queries. Then, we will construct a simulator that, using its unbounded power (and a polynomially bounded number of oracle queries), will compute the distinguishing elements on the fly in order to simulate.

More concretely, the first Lemma 5.1 deals with the case that no partial information is learned, showing that there is a polynomial set $\mathcal{L}$ of distinguishing elements, such that, as long as the obfuscated vector does not contain any elements from the distinguishing set $\mathcal{L}$, it cannot be distinguished from an obfuscated random vector. For a single point (non-composable obfuscation), this lemma is sufficient for constructing a VBB (i.e., efficient) simulator (as in [Can97]); indeed, the simulator can use its oracle to the hidden point to check whether it is taken from the polynomial distinguishing set, and if not simply simulate the obfuscation using a random point. However, in our setting, this is clearly not enough, as it might be that only some of the elements in the hidden vector are taken from the distinguishing set; this intuitively corresponds to the case that the adversary learns some part of the obfuscated vector, and may adapt its learning strategy accordingly.

To deal with the above, we show in Lemma 5.2 that, for any partial information learned by the adversary, there is still a corresponding polynomial *distinguishing set* that may depend on this partial information. Then, we show in Lemma 5.3 how to deduce a function $\mathcal{F}$ that computes, from any such partial information, a corresponding set of distinguishing elements, where the size of any such set is bounded by some fixed polynomial; however, this function may not be efficiently computable. Finally, we construct the simulator, which will use its unbounded power to compute the function $\mathcal{F}$ on the fly, while performing only a fixed polynomial number of queries. Specifically, the constructed simulator, in each iteration, computes the set of distinguishing elements $\mathcal{L} = \mathcal{F}(I)$ relative to the partial information $I$ it has learned so far (including some of the elements of hidden vector, as well as their positions). Then, it queries its oracle on the elements in $\mathcal{L}$ to try and reveal more elements. If eventually it reveals all the hidden point, it can perfectly simulate the adversary; otherwise, it gets to a point where it revealed information $I$, and non of the unrevealed points are in the relative set $\mathcal{L} = \mathcal{F}(I)$ of distinguishing elements, meaning it can simulate them as random points, without the adversary being able to distinguish.

**Lemma 5.1.** *Assume $\mathcal{O}$ is t-DI, then, for any poly-size $\mathcal{A}$ with binary output and polynomial $p$, there is a poly-size family $\mathcal{L} = \{L_n \subseteq \mathbb{D}_n\}$ such that any vector $\bar{x} \in \mathbb{D}_n^t$ that does not intersect $L_n$ (i.e. $\bar{x} \subseteq \mathbb{D}_n \setminus L_n$) satisfies:*

$$\left| \Pr_{\mathcal{A},\mathcal{O}}[\mathcal{A}(\mathcal{O}(C_{\bar{x}})) = 1] - \Pr_{\mathcal{A},\mathcal{O},\bar{u}\xleftarrow{U}\mathbb{D}_n^t}[\mathcal{A}(\mathcal{O}(C_{\bar{u}})) = 1] \right| \leq \frac{1}{p(n)} \quad . \tag{1}$$

*Proof.* Consider a binary poly-size $\mathcal{A}$ and a polynomial $p$. We describe the corresponding family $\mathcal{L}$. Let $X_n$ be the set of all "identifiable vectors", namely vectors that do not satisfy Equation (1). We treat $X_n$ as the union of two sets, $X_n = X_n^+ \cup X_n^-$, where:

$$X_n^+ = \left\{ \bar{x} \in \mathbb{D}_n^t : \Pr[\mathcal{A}(\mathcal{O}(C_{\bar{x}})) = 1] - \Pr[\mathcal{A}(\mathcal{O}(C_{\bar{u}})) = 1] \geq \frac{1}{p(n)} \right\} \quad ,$$

$$X_n^- = \left\{ \bar{x} \in \mathbb{D}_n^t : \Pr[\mathcal{A}(\mathcal{O}(C_{\bar{u}})) = 1] - \Pr[\mathcal{A}(\mathcal{O}(C_{\bar{x}})) = 1] \geq \frac{1}{p(n)} \right\} \quad .$$

First, we reduce $X_n^+$ to a subset of vectors $Y_n^+ \subseteq X_n^+$ such that: (a) any identifiable vector $\bar{x} \in X_n^+$ shares an element with some vector in $Y_n^+$, i.e. $\bar{x} \cap \bigcup_{\bar{y} \in Y_n} \bar{y} \neq \emptyset$, and (b) any element $a \in \mathbb{D}_n$ appears in at most one vector $\bar{y} \in Y_n^+$. Similarly, reduce $X_n^-$ to $Y_n^-$. Let $Y_n = Y_n^+ \cup Y_n^-$ and define

$$L_n = \bigcup_{\bar{y} \in Y_n} \bar{y} = \{a \in \mathbb{D}_n : \exists \bar{y} \in Y_n, a \in \bar{y}\} \quad .$$

By the construction of $L_n$, any $\bar{x} \subseteq \mathbb{D}_n \setminus L_n$ is not in the set $X_n = X_n^+ \cup X_n^-$, and hence it is not identifiable, i.e., it satisfies Equation (1). Thus, it remains to show that $|L_n| = \text{poly}(n)$. As $|L_n| \leq t|Y_n|$, it suffices to show that $|Y_n| = \text{poly}(n)$. Assume towards contradiction that the latter does not hold. We shall construct a CWS distribution ensemble $\mathcal{Z} = \{Z_n\}$ over $\mathbb{D}_n^t$, such that $\mathcal{A}$ distinguishes $\mathcal{O}(C_{\mathcal{Z}})$ from $\mathcal{O}(C_{\mathcal{U}(\mathcal{D}^t)})$ with advantage $1/p$, contradicting the DI property. By the assumption on the size of $|L_n|$, there exists a function $\ell(n) = n^{\omega(1)}$ such that for infinitely many $n$'s either $|Y_n^+| \geq \ell(n)$ or $|Y_n^-| \geq \ell(n)$. We assume WLOG the first case holds (the proof is similar for the second). For any $n \in \mathbb{N}$ such that $|L_n| \geq \ell(n)$, set $Z_n$ to be uniform on the set $Y_n^+$. For other $n$, let $Z_n$ be uniform on some arbitrary set of size $\ell(n)$ in which any element appears in at most one vector (we can take $\ell = o(|\mathbb{D}_n|)$ to assure such a choice is possible). The resulting ensemble $\mathcal{Z}$ is CWS since any single vector is drawn with probability at most $1/\ell$, and any single element appears in at most one vector. Moreover, for any $n$ such that $Z_n \triangleq U(Y_n^+)$, it holds that:

$$\begin{aligned} \Pr_{\bar{z} \leftarrow Z_n}[\mathcal{A}(\mathcal{O}(C_{\bar{z}})) = 1] \quad - \quad \Pr_{\bar{u} \leftarrow U(\mathbb{D}_n^t)}[\mathcal{A}(\mathcal{O}(C_{\bar{u}})) = 1] \geq \\ \min_{\bar{y} \in Y_n^+} \Pr[\mathcal{A}(\mathcal{O}(C_{\bar{y}})) = 1] \quad - \quad \Pr_{\bar{u} \leftarrow U(\mathbb{D}_n^t)}[\mathcal{A}(\mathcal{O}(C_{\bar{u}})) = 1] \geq \frac{1}{p(n)} \ . \end{aligned}$$

□

The next lemma shows that, for any partial information learned by the adversary, there is still a corresponding polynomial *distinguishing set*.

**Lemma 5.2.** *Assume $\mathcal{O}$ is $t$-DI. Let $s = s(n)$ be any length function such that $s \leq t$ and let $\mathcal{T} = \left\{(\bar{x}_n, I_n) \in \mathbb{D}_n^s \times \binom{[t]}{s}\right\}_{n \in \mathbb{N}}$ be a family of vectors and index sets[7]. Then, for any poly-size $\mathcal{A}$ with binary output and polynomial $p$, there exists a poly-size family $\mathcal{L}^{\mathcal{T}} = \{L_n\}$ such that for any $\bar{y} \in \mathbb{D}_n^{t-s}$ that does not intersect $L_n$:*

$$|\Pr[\mathcal{A}(\text{CMB}_{I_n}(\mathcal{O}(C_{\bar{x}_n}), \mathcal{O}(C_{\bar{y}})) = 1] - \Pr[\mathcal{A}(\text{CMB}_{I_n}(\mathcal{O}(C_{\bar{x}_n}), \mathcal{O}(C_{\bar{u}}))) = 1]| \leq \frac{1}{p(n)} \ ,$$

*where $\bar{u} \overset{U}{\leftarrow} \mathbb{D}_n^{t-s}$ and the probabilities are over the coins of $\mathcal{A}, \mathcal{O}$ and $\bar{u}$.*

To prove the lemma, we shall need the following (rather intuitive) claim.

**Claim 5.1.** *If $\mathcal{O}$ is $t$-DI, then it is also $s$-DI for any $s \leq t$. .*

*Proof of claim.* Assume towards contradiction there is an adversary $\mathcal{A}$ and a CWS distribution ensemble $\mathcal{X}$ over $s$-dimensional vectors, such that $\mathcal{A}$ distinguishes $\mathcal{O}(C_{\mathcal{X}})$ from $\mathcal{O}(C_{\mathcal{U}(\mathcal{D}^s)})$ with some non-negligible advantage. We examine a new CWS distribution ensemble $\mathcal{X}' = \mathcal{X} \times \mathcal{U}(\mathcal{D}^{t-s})$ and an adversary $\mathcal{A}'$ that, given an obfuscation of $t$ points, runs $\mathcal{A}$ on the first $s$ obfuscations. Then $\mathcal{A}'$ distinguishes $\mathcal{O}(C_{\mathcal{X}'})$ from $\mathcal{O}(C_{\mathcal{U}(\mathcal{D}^t)})$ (with the same advantage) contradicting the $t$-DI property. □

*Proof of Lemma 5.2.* Consider the function $r = t - s$, then by Claim 5.1, $\mathcal{O}$ is $r$-DI. Consider an adversary $\mathcal{A}'$ (for $r$-compositions) that has $\mathcal{T}$ hardwired and, on input $\bar{w}$ (here $\bar{w} = \mathcal{O}(C_{\bar{y}})$ for some $y_1 \dots y_r$), runs $\mathcal{A}$ on the valid obfuscation $\text{CMB}_{I_n}(\mathcal{O}(C_{\bar{x}_n}), \mathcal{O}(C_{\bar{y}}))$. By Lemma 5.1, this $\mathcal{A}'$ has a family $\mathcal{L}^{\mathcal{T}}$ which satisfies the required property with respect to the original adversary $\mathcal{A}$. □

The next lemma shows that there is a uniform polynomial bound on the size of all *distinguishing sets* (corresponding to any partial information), and hence there exists a *distinguishing function family* that, given any partial information, outputs a poly-size set of all *distinguishing elements* (with respect to this information).

[7] Any pair $(\bar{x}, I)$ should be thought of as partial information on a tuple of size $t$ with the elements of $\bar{x}$ in the indices $I$.

**Lemma 5.3.** *Let $\mathcal{O}$ be a $t$-DI obfuscator. Then for any poly-size $\mathcal{A}$ with binary output, and polynomial $p$, there exists a family of functions $\mathcal{F} = \{F_n\}$ and a $q = \mathrm{poly}(n)$ such that $F_n : \bigcup_{s \le t} \left( \mathbb{D}_n^s \times \binom{[t]}{s} \right) \longrightarrow \bigcup_{s \le q} \binom{\mathbb{D}_n}{s}$ and for any $(\bar{x}, I) \in \mathbb{D}_n^{|I|} \times \binom{[t]}{|I|}$ and any $\bar{y} \in \mathbb{D}_n^{t-|I|}$ which does not intersect the set $F_n(\bar{x}, I)$:*

$$|\Pr[\mathcal{A}(\mathrm{CMB}_{I_n}(\mathcal{O}(C_{\bar{x}}), \mathcal{O}(C_{\bar{y}}))) = 1] - \Pr[\mathcal{A}(\mathrm{CMB}_{I_n}(\mathcal{O}(C_{\bar{x}}), \mathcal{O}(C_{\bar{u}}))) = 1]| \le \frac{1}{p(n)} \ ,$$

*where $\bar{u} \overset{U}{\leftarrow} \mathbb{D}_n^{t-|I|}$ and the probabilities are over the coins of $\mathcal{A}, \mathcal{O}$ and $\bar{u}$.*

*Remark* 5.3. The function $F_n$ is defined for any "partial information"; in particular, the set of indices $I$ is allowed to be the empty set corresponding to no partial information as in Lemma 5.1.

*Proof.* For any $(\bar{x}, I) \in \mathbb{D}_n^{|I|} \times \binom{[t]}{|I|}$, let $F_n(\bar{x}, I) \subseteq \mathbb{D}_n$ be the minimal set that satisfies the above condition (note that such a set always exists as $\mathbb{D}_n$ trivially satisfies the requirement). We show that there exists a polynomial $q$, such that $|F_n| \le q(n)$ (i.e. $q$ is a uniform bound on all images). Let $(\bar{x}_n^*, I_n^*)$ be the pair which maximizes $F_n(\bar{x}, I)$, i.e. $|F_n(\bar{x}_n^*, I_n^*)| = \max_{I \subseteq [t], \bar{x} \in \mathbb{D}_n^{|I|}} |F_n(\bar{x}, I)|$. By Lemma 5.2, there exists a polynomial $q$ for which $|F_n(\bar{x}_n^*, I_n^*)| \le q(n)$ (just by considering the family $\{(\bar{x}_n^*, I_n^*)\}_{n \in \mathbb{N}}$). The result follows. □

To complete the proof of the theorem, we construct a simulator using the family of *distinguishing functions* $\mathcal{F}$. However, as it might not be computable by a poly-size simulator, the result holds only for strong simulators as in the VGB definition.

*Proof - Any $t$-DI point obfuscator is also a $t$-composable VGB point obfuscator (sketch).* Let $\mathcal{A}$ be a binary poly-size adversary and $p$ a polynomial. Let $\mathcal{F}$ be the corresponding family of functions given by Lemma 5.3, and let $q$ be the polynomial bound on the size of the images of $\mathcal{F}$ (which are sets). We construct an unbounded simulator $\mathcal{S}$ that performs at most $q \cdot t$ oracle queries (the full description is given by Algorithm (5.3)). Given oracle access to a tuple of circuits $C_{\bar{x}} = C_{x_1}, \ldots, C_{x_t}$, for some $\bar{x} \in \mathbb{D}_n^t$, $\mathcal{S}$ first runs $F_n$ (on the empty set), retrieves a set $L^{(0)}$ of all *distinguishing elements* with respect to no partial information, and queries its oracle on all the elements in $L^{(0)}$. In case it did not reveal any elements (i.e. $\bar{x} \cap L^{(0)} = \emptyset$), it chooses a uniform vector $\bar{u} \overset{U}{\leftarrow} \mathbb{D}_n^t$, computes obfuscations of the points in $\bar{u}$ and runs $\mathcal{A}$ on their composition. Otherwise, it revealed some elements given by a pair $(\bar{z}^{(0)}, I^{(0)})$. It then computes $L^{(1)} = F_n(\bar{z}^{(0)}, I^{(0)})$, and as in the first step, queries all the values in $L^{(1)}$. In case it did not reveal any new values, it chooses a uniform vector $\bar{u} \overset{U}{\leftarrow} \mathbb{D}_n^{t-|I^{(0)}|}$ and runs $\mathcal{A}$ on an obfuscation $\mathrm{CMB}_{I^{(0)}}(\mathcal{O}(C_{\bar{z}^{(0)}}), \mathcal{O}(C_{\bar{u}})))$. Otherwise, it has updated partial information given by a pair $(\bar{z}^{(1)}, I^{(1)})$. It continues on in this manner. If at any point it revealed all the points in $\bar{x}$, it just runs $\mathcal{A}$ on a random composed obfuscation of the points in $\bar{x}$ performing a perfect simulation. Otherwise, it stops after at most $t$ iterations, guaranteeing a simulation with $1/p$ accuracy. This completes the main part of the proof of Theorem 5.1. □

A more careful analysis shows that we can somewhat "compress" the distinguishing function $\mathcal{F}$ to a set of distinguishing elements. This yields the following.

**Proposition 5.1.** *If $\mathcal{O}$ is a $t$-DI obfuscator, then any binary adversary, given a sequence of $t$ obfuscations, can be simulated by a simulator of size $n^{O(t)}$ and $\mathrm{poly}(n)$ queries. In particular, for $t = O(1)$ this yields a polynomially bounded simulator (VBB).*

**Algorithm 5.1** Simulator $\mathcal{S}^{C_{x_1},\ldots,C_{x_t}}$

$(\bar{z}, I) \leftarrow (\emptyset, \emptyset)$
$L \leftarrow \emptyset$
**while** not all the coordinates of $\bar{x}$ were revealed **do**
  $L \leftarrow F_n(\bar{z}, I)$
  Query all values in $L$
  **if** New elements were revealed **then**
    Update partial information $(\bar{z}, I)$ accordingly
  **else**
    $\bar{u} \overset{U}{\leftarrow} \mathbb{D}_n^{t-|I|} \setminus L$
    **return** $\mathcal{A}(\text{CMB}_I(\mathcal{O}(C_{\bar{z}}), \mathcal{O}(C_{\bar{u}})))$
  **end if**
**end while**
**return** $\mathcal{A}(\mathcal{O}(C_{\bar{x}}))$

*Proof sketch.* Going back to our proof of simulation (in Theorem 5.1), we show that one can replace $\mathcal{F}$ by hardwiring into the simulator sets of distinguishing elements of total size at most $n^{O(t)}$. The main point is to note that the simulator does not need the distinguishing elements corresponding to all partial information sets, but only to some. Formally, let $(\bar{x}, I)$ be some partial information (where $I \subseteq [t], \dim \bar{x} = |I|$). We shall say that the partial information $(\bar{y}, J)$, $F_n$*-extends* $(\bar{x}, I)$ and denote $(\bar{x}, I) \overset{F_n}{\Subset} (\bar{y}, J)$, if the following holds:

$$I \subseteq J \tag{1}$$

$$\bar{y}|I = \bar{x} \tag{2}$$

$$\bar{y}|J \setminus I \subseteq F_n(\bar{x}, I) \ , \tag{3}$$

where $\bar{y}|I$ denotes the restriction of $\bar{y}$ to the coordinates corresponding to $I$. In addition, we define the following $t$ sets of partial information pairs.

$$G_n^{(0)} = \{(\emptyset\, \emptyset)\} \ \text{(no partial information)}$$

$$\forall k \in [t-1] : G_n^{(k)} = \left\{ (\bar{y}, J) : |J| = k, \exists (\bar{x}, I) \in G_n^{(k-1)}, (\bar{x}, I) \overset{F_n}{\Subset} (\bar{y}, J) \right\} \ .$$

We claim that we can construct a simulator by hardwiring into it only the distinguishing sets corresponding to the family $G_n = \bigcup_k G_n^{(k)}$. Indeed, consider a simulator that tries to reveal a single new element at a time; one can think of its query strategy as a tree, where the $k$'th level corresponds to $G_n^{(k)}$, and a concrete run corresponds to a path in the tree (which ends when no distinguishing elements are found). That is, the simulator starts by querying the values in $F_n(\emptyset)$, when it finds an element $x_1$ in coordinate $i_1$, it stops and locates $F_n(\langle x_1 \rangle, \{i_1\})$ (as $(\langle x_1 \rangle, \{i_1\}) \in G_n^{(1)}$). It then continues in the same manner, each time locating the proper extension (in the $j$'th step it finds $x_j$ at coordinate $i_j$ and locates $F_n(\langle x_1 \ldots x_j \rangle, \{i_1 \ldots i_j\})$). If at any point it queries all values in the current set $L$, without revealing any new elements, it completes its partial information with uniform elements that do not intersect $L$, computes the corresponding composition of obfuscations, and runs the adversary on it (just as in the proof of the theorem). The properties of the sets $F_n$ guarantee the required simulation accuracy. It is left to show that the total size of the family (or tree) $G_n$ is at most $n^{O(t)}$. Indeed, any set in the family is of size at most $q$ (where $q$ is the polynomial bound on $\mathcal{F}$ given by Lemma 5.3). Hence, any pair $(\bar{x}, I) \in G_n^{(k-1)}$ has at most $q(t-k) \leq qt$ extensions in $G_n^{(k)}$ (there are at most $q$ elements in $F_n(\bar{x}, I)$

each having $t-k$ possible coordinates). It follows that $|G_n^{(k)}| \leq qt \cdot |G_n^{(k-1)}|$ (the degree is bounded by $qt$) and hence the total number of pairs in $G_n$ is $(qt)^{O(t)}$. Since each corresponding set contains at most $q$ elements, and both $q$ and $t$ are polynomial, the total number of elements is bounded by $n^{O(t)}$. □

### 5.4 On the Possibility of Bounded Simulation (VBB)

We note that our result does not rule out the possibility of bounded simulation for any $t = \mathrm{poly}(n)$; namely, it may still be that any $t$-DI point obfuscator is also a $t$-composable VBB point obfuscator. More specifically, it might be that there always exists a function family $\mathcal{F}$, such as the one required in Theorem 5.1, that is also efficiently computable, or even a "compressed" poly set of distinguishing elements as in Proposition 5.1. Alternatively, there might be other techniques that allow efficient simulation. In this context, we show an example of an adversary whose distinguishing function cannot be compressed into a poly set. We also show that, if bounded simulation exists, then so does an efficiently computable function family $\mathcal{F}$; i.e., simulation can be proven using the same technique we use above.

*Example* 5.1. Intuitively speaking, the reason we can compress the distinguishing function $\mathcal{F}$ for constant dimension $t$ is that the adversary has a relatively limited amount of adaptivity to aid it, while when $t$ grows, there are simply too many adaptive options, which cannot be captured within a polynomial set. This is given by the following example.

For $t = \omega(1)$ consider obfuscating points in $\{0,1\}^n$ (i.e. $\mathbb{D}_n = \{0,1\}^n$). Consider an adversary $\mathcal{A}$ which first checks for any $x \in \{0,1\}^{\log n}$ if $x \circ 0^{n-\log n}$ is one of the obfuscated points (simply by running the obfuscation). Let $b_x$ be the indicator for the case in which $x \circ 0^{n-\log n}$ is indeed one of the points and let $b = (b_x)_{x \in \{0,1\}^{\log n}}$ be the $n$-bit string given by the answers. The adversary now checks whether $b$ is one of the points and returns 1 only if this is indeed the case. We claim that any poly-size family cannot cover all "distinguishing elements". More precisely, we show that for any poly-size family $\mathcal{L} = \{L_n\}$, there are infinitely many $n$'s for which there is some partial information, given by a vector $\bar{x}$ that does not intersect $L_n$, and two possible ways $(\bar{y}, \bar{z})$ to complete it to a vector of $t$ points such that

$$\Pr[\mathcal{A}(\mathcal{O}(C_{\bar{x} \circ \bar{y}})) = 1] - \Pr[\mathcal{A}(\mathcal{O}(C_{\bar{x} \circ \bar{z}})) = 1] = 1 \ ,$$

where $\bar{x} \circ \bar{y} = \mathrm{CMB}_{[\dim \bar{x}]}(\bar{x}, \bar{y})$ is just the vector of $t$ points given by the concatenation of $\bar{x}, \bar{y}$. In particular, for some $\bar{w} \in \{\bar{y}, \bar{z}\}$:

$$|\Pr[\mathcal{A}(\mathcal{O}(C_{\bar{x} \circ \bar{w}})) = 1] - \Pr[\mathcal{A}(\mathcal{O}(C_{\bar{x} \circ \bar{u}})) = 1]| \geq \frac{1}{2} \ ,$$

where $\bar{u}$ is a uniformly chosen vector with elements in $\{0,1\}^n$.

Indeed, define the following set of strings:

$$G_n^t = \left\{ b \in \{0,1\}^n : \sum_{x \in \{0,1\}^{\log n}} b_x \leq t \right\} \ ,$$

and note that [8] :

$$|G_n^t| = \sum_{i \leq t} \binom{n}{i} \geq \left( \frac{n}{\log n} \right)^{\omega(1)} = n^{\omega(1)} \ ;$$

hence, for any large enough $n \in \mathbb{N}$, there must be some $a \in G_n^t - \left( L_n \cup \left( \{0,1\}^{\log n} \times \left\{ 0^{n-\log n} \right\} \right) \right)$. We now consider the set $T = \left\{ x \circ 0^{n-\log n} : x \in \{0,1\}^{\log n}, a_x = 1 \right\}$, a vector $\bar{x}$ consisting of the

[8] If $t \geq \log n$ then $|G_n^t| \geq \binom{n}{\log n} \geq \left( \frac{n}{\log n} \right)^{\log n}$ and otherwise $|G_n^t| \geq \binom{n}{t} \geq \left( \frac{n}{t} \right)^t \geq \left( \frac{n}{\log n} \right)^{\omega(1)}$.

elements in $T$ (in some arbitrary order) and two vectors $\bar{y}, \bar{z}$ of $t - |T|$ elements in $\{0,1\}^n$ that do not intersect $L_n \cup \left(\{0,1\}^{\log n} \times \left\{0^{n-\log n}\right\}\right)$ and satisfy $a \in \bar{y} \setminus \bar{z}$ (e.g. $\bar{y} = a^{t-|T|}, \bar{z} = b^{t-|T|}$ for some $b \notin L_n \cup \left(\{0,1\}^{\log n} \times \left\{0^{n-\log n}\right\}\right)$). Equation (1) follows as required.

It should be noted that although the above example rules out the technique of hardwiring a polynomial set of "distinguishing elements", it does not rule out the possibility of efficient simulation in general. In particular, the adversary described above makes a "black box" attack (i.e. only runs the program) and hence can be easily (and efficiently) simulated.

**Proposition 5.2.** *If $\mathcal{O}$ is a $t$-composable VBB point obfuscator, then there exists an efficient algorithm $\mathcal{B}$ computing a distinguishing function family $\mathcal{F}$ (with the properties given in Lemma 5.3).*

Previously, we showed that, for constant dimensional vectors, bounded simulation is in fact possible. The above proposition shows that in fact, if efficient simulation is possible, then there must be a distinguishing function family $\mathcal{F}$ that is also efficiently computable (i.e. it can be proven using the same techniques we used above).

*Proof sketch of Proposition 5.2.* Let $\mathcal{A}$ be a binary poly-size adversary and $p$ a polynomial. We describe the algorithm $\mathcal{B}$. By the VBB property, there exists an efficient simulator $\mathcal{S}$ such that, for any vector $\bar{v} \in \mathbb{D}_n^t$, it holds that:

$$|\Pr[\mathcal{A}(\mathcal{O}(C_{\bar{v}})) = 1] - \Pr[\mathcal{S}^{C_{\bar{v}}}(1^n) = 1]| \leq 1/3p(n) \ ,$$

where the probability is over the coins of $\mathcal{A}, \mathcal{S}, \mathcal{O}$.
Given partial information $(\bar{x}, I)$, let $C_{\bar{x} \circ \bar{0}} : \mathbb{D} \times [t] \to \{0,1\}$ denote the function which returns 1 on input $(z,i)$ if $i \in I$ and $x_i = z$ and 0 otherwise. $\mathcal{B}$ will run $\mathcal{S}^{C_{\bar{x} \circ \bar{0}}}$ and will record its set of queries, it will independently repeat this process $k$ times (where $k = \text{poly}(n, \log|\mathbb{D}_n|)$ will be specified later on). Eventually, it will output the set of all recorded queries $Q = \bigcup_{i \in [k]} Q_i$. We show that there is only a negligible probability (over the coins of $\mathcal{B}$) that there exists a vector $\bar{y}$ of dimension $(t - |I|)$ that does not intersect $Q$ and satisfies:

$$|\Pr[\mathcal{A}(\text{CMB}_I(\mathcal{O}(C_{\bar{x}}), \mathcal{O}(C_{\bar{y}}))) = 1] - |\Pr[\mathcal{A}(\text{CMB}_I(\mathcal{O}(C_{\bar{x}}), \mathcal{O}(C_{\bar{u}}))) = 1] \geq \frac{1}{p(n)} \ , \qquad (1)$$

where $\bar{u}$ is a uniform vector (of the same dimension as $\bar{y}$) and the probabilities are over the coins of $\mathcal{A}, \mathcal{O}$ and the choice of $\bar{u}$.

Concretely, for any vector $\bar{y}$ satisfying Equation (1), we show that $\bar{y} \cap Q = \emptyset$ with probability at most $2^{-n}|\mathbb{D}_n|^{-2t}$. Denote $\bar{x} \circ \bar{y} = \text{CMB}_I(\bar{x}, \bar{y})$. By the simulation property and Equation (1):

$$|\Pr[\mathcal{S}^{C_{\bar{x} \circ \bar{y}}}(1^n) = 1] - \Pr[\mathcal{S}^{C_{\bar{x} \circ \bar{u}}}(1^n) = 1]| \geq 1/3p(n) \ .$$

On the other hand, the probability that $\mathcal{S}^{C_{\bar{x} \circ \bar{u}}}$ queries an element of $\bar{u}$ is at most $\frac{|\mathcal{S}| \cdot t}{|\mathbb{D}_n|} = n^{-\omega(1)}$, and hence:

$$\left|\Pr[\mathcal{S}^{C_{\bar{x} \circ \bar{0}}}(1^n) = 1] - \Pr[\mathcal{S}^{C_{\bar{x} \circ \bar{u}}}(1^n) = 1]\right| \leq n^{-\omega(1)} \ .$$

It follows that:

$$|\Pr[\mathcal{S}^{C_{\bar{x} \circ \bar{0}}} = 1] - \Pr[\mathcal{S}^{C_{\bar{x} \circ \bar{y}}} = 1]| \geq 1/3p(n) - n^{-\omega(1)} \ .$$

As before, conditioning on the event that $\mathcal{S}$ does not query any element of $\bar{y}$, the above probabilities are equal. It follows that $\mathcal{S}$ queries some element in $\bar{y}$ with probability at least $1/4p(n)$. Hence, the probability that $\bar{y} \cap Q_i = \emptyset$ for all $i \in [k]$ is bounded by:

$$\left(1 - \frac{1}{4p(n)}\right)^k \leq 2^{-k/4p} < 2^{-n}|\mathbb{D}_n|^{-t} \ ,$$

where the last inequality holds for $k = \Theta(pt \log|\mathbb{D}_n|)$. Using union bound over all possible vectors $\bar{y}$ (there are at most $|\mathbb{D}_n|^t$ such vectors) we get the required result. □

# 6 A Concrete Composable Point Obfuscator

After establishing the proper framework in the previous, this section is devoted to a concrete construction of composable VGB point obfuscators. We consider the point obfuscator constructed in [Can97] and analyze its security under composition.

**Construction 6.1** (*The $r, r^x$ Point Obfuscator* [Can97])**.** *Let $\mathcal{G} = \{\mathbb{G}_n\}_{n\in\mathbb{N}}$ be a group ensemble, where each $\mathbb{G}_n$ is a group of prime order $p_n \in (2^{n-1}, 2^n)$. We define an obfuscator $\mathcal{O}$, for points in the domain $\mathbb{Z}_{p_n}$ as follows: $C_x \overset{\mathcal{O}}{\longmapsto} \mathcal{C}[r, r^x]$ Where $r \overset{U}{\leftarrow} \mathbb{G}_n^*$ is a random generator of $\mathbb{G}_n$, and $\mathcal{C}[r, r^x]$ is a circuit that has $r, r^x$ hardwired into it, and on input $z$, it checks whether $r^x = r^z$.*

In [Can97], Construction 6.1 is shown to be secure under a strong variant of the Decision Diffie-Hellman assumption. We now present our assumption, which is a generalization of the [Can97] assumption to tuples of points.

**Assumption 6.1** ($t$-*Strong Vector Decision Diffie Hellman I*)**.** *Let $t = \mathrm{poly}(n)$. There exists a group ensemble $\mathcal{G} = \{\mathbb{G}_n : |\mathbb{G}_n| = p_n \text{ is prime}\}$ with efficient representation and operations, such that for any CWS distribution ensemble $\mathcal{X} = \{X_n\}$ over vectors in $\mathbb{Z}_{p_n}^t$ the following holds:*

$$\left\{ \begin{array}{c} g_1, g_1^{a_1} \\ \vdots \\ g_t, g_t^{a_t} \end{array} : \begin{array}{c} \bar{g} \overset{U}{\leftarrow} (\mathbb{G}_n^*)^t \\ \bar{a} \overset{X_n}{\leftarrow} \mathbb{Z}_{p_n}^t \end{array} \right\}_{n\in\mathbb{N}} \approx_c \left\{ \begin{array}{c} g_1, g_1^{u_1} \\ \vdots \\ g_t, g_t^{u_t} \end{array} : \begin{array}{c} \bar{g} \overset{U}{\leftarrow} (\mathbb{G}_n^*)^t \\ \bar{u} \overset{U}{\leftarrow} \mathbb{Z}_{p_n}^t \end{array} \right\}_{n\in\mathbb{N}} .$$

We observe that Assumption 6.1 implies that the $r, r^x$ point obfuscator is $t$-DI with respect to the corresponding group ensemble $\mathcal{G}$, given by the construction. Hence, Theorem 5.1 yields:

**Theorem 6.1.** *Under Assumption 6.1, the $r, r^x$ point obfuscator is a $t$-composable VGB point obfuscator (with respect to the group ensemble $\mathcal{G}$ given by the assumption). Assuming the existence of a "universal" group ensemble that satisfies Assumption 6.1 for any $t = \mathrm{poly}(n)$ implies composable VGB point obfuscators (i.e. $t$-composable for any $t = \mathrm{poly}(n)$).*

In the following subsection, we further discuss our hardness Assumption 6.1.

## 6.1 On the Assumption

As shown in [Wee05], strong hardness assumptions are inherently necessary for point obfuscation (even non-composable). We next discuss the specific nature of our Assumption 6.1, including its relation to previous Decision Diffie Hellman variants. In addition, we show that it holds in the *Generic Group Model*.

**Relation to Previous DDH Assumptions.** We start by presenting another strong variant of DDH for tuples of points, which is in a sense a natural generalization to the standard and strong DDH assumptions [Bon98, Can97].

**Assumption 6.2** ($t$-*Strong Vector Decision Diffie Hellman II*)**.** *Let $t = \mathrm{poly}(n)$. There exists a group ensemble $\mathcal{G} = \{\mathbb{G}_n : |\mathbb{G}_n| = p_n \text{ is prime}\}$ with efficient representation and operations, such that for any CWS distribution ensemble $\mathcal{X} = \{X_n\}$ over vectors in $\mathbb{Z}_{p_n}^t$ the following holds:*

$$\left\{ \begin{array}{c} g_1, g_1^{a_1}, g_1^{b_1}, g_1^{c_1} \\ \vdots \\ g_t, g_t^{a_t}, g_t^{b_t}, g_t^{c_t} \end{array} : \begin{array}{c} \bar{g} \overset{U}{\leftarrow} (\mathbb{G}_n^*)^t \\ \bar{a} \overset{X_n}{\leftarrow} \mathbb{Z}_{p_n} \\ \bar{b}, \bar{c} \overset{U}{\leftarrow} \mathbb{Z}_{p_n}^t \end{array} \right\}_{n\in\mathbb{N}} \approx_c \left\{ \begin{array}{c} g_1, g_1^{a_1}, g_1^{b_1}, g_1^{a_1 b_1} \\ \vdots \\ g_t, g_t^{a_t}, g_t^{b_t}, g_t^{a_t b_t} \end{array} : \begin{array}{c} \bar{g} \overset{U}{\leftarrow} (\mathbb{G}_n^*)^t \\ \bar{a} \overset{X_n}{\leftarrow} \mathbb{Z}_{p_n}^t \\ \bar{b} \overset{U}{\leftarrow} \mathbb{Z}_{p_n}^t \end{array} \right\}_{n\in\mathbb{N}} .$$

Restricting the assumption to $t = 1$ results in the strong DDH (SDDH) assumption in [Can97]. If in addition we restrict $\mathcal{X}$ to be the uniform distribution ensemble, we get the standard DDH assumption. Assumption 6.2 appears as a more familiar and a natural generalization of SDDH and DDH than Assumption 6.1 does. However, 6.1 is somewhat simpler and is clearly weaker (the distributions induced by the last two elements of each foursome in 6.2 are identical to those in 6.1). It is also not hard to see that if 6.1 holds for $2t$ then 6.1 holds for $t$, but in fact the assumptions are equivalent also with the same parameter $t$.

**Proposition 6.1.** *Assumptions 6.1 and 6.2 are equivalent for $t \geq 2$.*

*Proof sketch.* As explained above, Assumption 6.2 trivially implies Assumption 6.1 (for any $t$). To prove that 6.1 implies 6.2 for any $t \geq 2$, we show that the following distribution ensembles are computationally indistinguishable:

$$\left\{ \begin{array}{ccc} g_1, g_1^{a_1}, g_1^{b_1}, g_1^{a_1 b_1} & & \bar{g} \xleftarrow{U} (\mathbb{G}_n^*)^t \\ \vdots & : & \bar{a} \xleftarrow{X_n} \mathbb{Z}_{p_n}^t \\ g_t, g_t^{a_t}, g_t^{b_t}, g_t^{a_t b_t} & & \bar{b} \xleftarrow{U} \mathbb{Z}_{p_n}^t \end{array} \right\}_{n \in \mathbb{N}} \tag{1}$$

$$\left\{ \begin{array}{ccc} g_1, g_1^{a_1}, g_1^{b_1}, g_1^{a_1 b_1} & & \bar{g} \xleftarrow{U} (\mathbb{G}_n^*)^t \\ \vdots & : & (\bar{a}, \bar{b}) \xleftarrow{U} \mathbb{Z}_{p_n}^{t \times 2} \\ g_t, g_t^{a_t}, g_t^{b_t}, g_t^{a_t b_t} & & \end{array} \right\}_{n \in \mathbb{N}} \tag{2}$$

$$\left\{ \begin{array}{ccc} g_1, g_1^{a_1}, g_1^{b_1}, g_1^{c_1} & & \bar{g} \xleftarrow{U} (\mathbb{G}_n^*)^t \\ \vdots & : & (\bar{a}, \bar{b}, \bar{c}) \xleftarrow{U} \mathbb{Z}_{p_n}^{t \times 3} \\ g_t, g_t^{a_t}, g_t^{b_t}, g_t^{c_t} & & \end{array} \right\}_{n \in \mathbb{N}} \tag{3}$$

$$\left\{ \begin{array}{ccc} g_1, g_1^{a_1}, g_1^{b_1}, g_1^{c_1} & & \bar{g} \xleftarrow{U} (\mathbb{G}_n^*)^t \\ \vdots & : & \bar{a} \xleftarrow{X_n} \mathbb{Z}_{p_n}^t \\ g_t, g_t^{a_t}, g_t^{b_t}, g_t^{c_t} & & (\bar{b}, \bar{c}) \xleftarrow{U} \mathbb{Z}_{p_n}^{t \times 2} \end{array} \right\}_{n \in \mathbb{N}} \tag{4}$$

(1) $\approx_c$ (2), since given a distinguisher $\mathcal{A}$ for these two ensembles, we can construct a distinguisher $\mathcal{A}'$ for the ensembles in Assumption 6.2. Given input $g_1, g_1^{a_1}, \ldots, g_t, g_t^{a_t}$, $\mathcal{A}'$ samples $\bar{b} \xleftarrow{U} \mathbb{Z}_{p_n}^t$ and runs $\mathcal{A}$ on $g_1, g_1^{a_1}, g_1^{b_1}, g_1^{a_1 b_1}, \ldots, g_t, g_t^{a_t}, g_t^{b_t}, g_t^{a_t b_t}$. The fact that (2) $\approx_c$ (3) follows from standard DDH by applying a standard hybrid argument, while standard DDH follows from Assumption 6.1 with $t = 2$ and $\mathcal{X} = \mathcal{U}$. Finally, (3) $\approx_c$ (4) is equivalent to Assumption 6.1, as the last two elements in each foursome are uniform over $\mathbb{G}_n^*$ and independent of the first two, hence any distinguisher can simulate these on its own. □

A natural question is whether assumptions 6.1 and 6.2 for $t = 1$ imply the corresponding assumptions for general polynomial $t$ (or even just larger constant $t$). For the case that the distribution ensemble $\mathcal{X}$ is the uniform distribution, this is true (corresponds to showing DDH for any poly number of foursomes from DDH for a single foursome by an hybrid argument). However, when allowing any CWS distribution, such an argument fails to work for two main reasons: (a) dependence among coordinates. (b) the distribution ensemble might not even be efficiently samplable. In general, we do not know whether SDDH implies SVDDH.

## 6.2 SVDDH Holds in the Generic Group Model

We show that Assumption 6.1 holds in the *generic group model* [Sho97] where algorithms cannot exploit the representation of the group elements, other than the fact that each element has a unique representation. As noted in the introduction, there exist well studied group ensembles (e.g. Quadratic Residues

modulo a prime, and Elliptic Curves groups) where the best cryptanalytic techniques are in fact generic ones [6].

Formally, a generic poly-size algorithm $\mathcal{A}$, in $(\mathbb{Z}_p, +)$ takes as input a list of encodings $\sigma(g_1), \ldots, \sigma(g_k)$, where $\sigma$ is a random encoding of $\mathbb{Z}_p$ to bit-strings $\{0,1\}^m$, for $m = \text{poly}(|p|)$. In addition, it has access to two oracles: the first, $\text{ADD}_\sigma$, takes as input two (previously given) encodings $\sigma(g_1), \sigma(g_2)$ and a bit $b$, and returns $\sigma(g_1 + (-1)^b g_2)$, the second $\mathbf{1}_\sigma$, returns $\sigma(1)$ on all inputs.[9] For a vector $\bar{g} = (g_1, \ldots, g_t)$ of group elements, we shall denote by $\sigma(\bar{g})$ the corresponding encodings vector $\sigma(g_1), \ldots, \sigma(g_t)$. For two vectors of elements $(\bar{g}, \bar{h})$, we denote by $\bar{g}\bar{h} = (g_1 h_1, \ldots, g_t h_t)$ the corresponding vector of products. To prove that Assumption 6.2 holds in this model, we show the following.

**Proposition 6.2.** *Let $X_1, X_2$ be two distributions on $\mathbb{Z}_p^t$, such that for both $i \in \{1,2\}$ it holds that $\max_{a \in \mathbb{Z}_p} \Pr_{\bar{v} \leftarrow X_i}[\exists j \in [t] : v_j = a] \leq \nu$ for some $\nu \leq 1$. Let $\mathcal{A}$ be a generic algorithm that makes at most $q$ queries to its oracles and denote:*

$$p_i = \Pr_{\bar{x} \overset{X_i}{\leftarrow} \mathbb{Z}_p^t, \bar{u} \overset{U}{\leftarrow} \mathbb{Z}_p^t} [\mathcal{A}(\sigma(\bar{u}, \bar{u}\bar{x})) = 1] \ ,$$

*where the probability is also taken over $\sigma$ and the coins of $\mathcal{A}$. Then:*

$$|p_1 - p_2| \leq (q + 2t)^2 \left(\nu + \frac{1}{p}\right) \ .$$

In the setting of Assumption 6.1, one distribution is taken from a CWS ensemble $\mathcal{X}$ and the other is taken from the uniform distribution ensemble.

*Proof.* To prove the proposition, we shall need the following simple claim.

**Claim 6.1.** *Let $P : \mathbb{Z}_p^{2t} \to \mathbb{Z}_p$ be a multivariate polynomial such that $0 \leq \deg P \leq 1$. Then for $i \in \{1,2\}$ and $\bar{x} \overset{X_i}{\leftarrow} \mathbb{Z}_p^t, \bar{u} \overset{U}{\leftarrow} \mathbb{Z}_p^t$, it holds that*

$$\Pr[P(\bar{u}, \bar{u}\bar{x}) = 0] \leq \nu + \frac{1}{p} \ .$$

*Proof of claim.* In case $\deg P = 0$, $P$ is a constant non-zero polynomial and the claim trivially holds. Assume $\deg P = 1$ and write $P(\bar{u}, \bar{u}\bar{x}) = a_0 + \sum_{i \in [t]} a_i u_i + \sum_{i \in [t]} a_{i+t} u_i x_i = a_0 + \sum_{i \in [t]} (a_i + a_{i+t} x_i) u_i$. Since $\deg P = 1$, there is some $j \in [t]$ such that $(a_j, a_{j+t}) \neq (0,0)$. This implies that $a_j + a_{j+t} x_j = 0$ with probability at most $\nu$. Indeed, in case $a_j \neq 0, a_{j+t} = 0$ the above term never vanishes, while if $a_{j+t} \neq 0$, the term vanishes only when $x_j = -a_j a_{j+t}^{-1}$, which occurs with probability at most $\nu$. Given that $a_j + a_{j+t} x_j \neq 0$, $P(\bar{u}, \bar{u}\bar{x}) = 0$ with probability at most $1/p$ as $u_j$ is independent of all other random variables. Overall, $P(\bar{u}, \bar{u}\bar{x})$ vanishes with probability at most $\nu + 1/p$. □

We now prove Proposition 6.2, using the same technique applied in [Sho97, Bon98]. First note that at each step of $\mathcal{A}$'s execution, all the encodings it got so far correspond to some linear polynomial evaluated at $\bar{u}, \bar{x}\bar{u}$. More precisely, its input consists of encodings $\sigma_i = \sigma(P_i(\bar{u}, \bar{x}\bar{u})) : -2t + 1 \leq i \leq 0$, where $P_i(\bar{x}) = x_{i+2t}$ is just a projection polynomial. At its $k$'th query, it either queries $\text{ADD}_\sigma(\sigma_i, \sigma_j, (-1)^b)$ for some $i, j < k$ and is answered with an encoding $\sigma_k = \sigma([P_i + (-1)^b P_j](\bar{u}, \bar{x}\bar{u}))$ or it queries $\mathbf{1}_\sigma$ and is answered with $\sigma_k = \sigma(1)$ (which is just a constant polynomial). Informally, we show that for the algorithm to distinguish the two distributions it must perform queries corresponding to two distinct

[9] Adding such an oracle allows the algorithm to get the encoding of any arbitrary element in $\mathbb{Z}_p$ by applying $\text{ADD}_\sigma$ (in particular it could sample random elements).

polynomials $P_i \neq P_j$ such that $P_i(\bar{u}, \bar{x}\bar{u}) = P_j(\bar{u}, \bar{x}\bar{u})$, otherwise it only sees uniform samples independently of the underlying distribution. Formally, consider an alternative setting in which $x, u$ are disregarded through the entire interaction. Instead, we emulate the interaction by storing a table with values $\sigma_i \in \{0,1\}^m$ and corresponding linear polynomials $P_i$. As input, we give the algorithm random distinct strings $\sigma_{-2t+1}, \ldots, \sigma_0$, and store each with a corresponding projection polynomial $P_i(\bar{x}) = x_{i+2t}$. At the $k$'th query, if $\mathrm{ADD}_\sigma(\sigma_i, \sigma_j, (-1)^b)$ is called (with some $i, j < k$) we compute the corresponding polynomial $P_k = P_i + (-1)^b P_j$, and check whether $P_k = P_\ell$ for some $\ell < k$. In case it does, we return $\sigma_k = \sigma_\ell$, otherwise we choose $\sigma_k$ to be a random value in $\{0,1\}^m \setminus \{\sigma_j\}_{j<k}$. Same goes for queries to $1_\sigma$. Denote by $p^*$ the probability that $\mathcal{A}$ outputs 1 in such an interaction; we show that $|p_i - p^*| \leq \frac{(q+2t)^2}{2}\left(\nu + \frac{1}{p}\right)$. Note that the altered interaction differs from a true interaction (where $\bar{x}, \bar{u}$ are used) only when there are some $i < j$ and $P_i \neq P_j$, such that $P_i(\bar{u}, \bar{x}\bar{u}) = P_j(\bar{u}, \bar{x}\bar{u})$, in which case the true interaction would return $\sigma_j = \sigma_i$, while the altered interaction returns a new random value. Denote by $C_i$ the event in which such an equality occurs, when $\bar{x}$ is sampled from $X_i$ and denote by $p_i|C_i$ the probability that $\mathcal{A}$ outputs 1 in the original (non altered) interaction given that $C_i$ occurs.[10] Then:

$$|p_i - p^*| = \left|\Pr[C_i](p_i|C_i) + \Pr[\overline{C}_i]p^* - p^*\right| = \Pr[C_i]|(p_i|C_i) - p^*| \leq \Pr[C_i] \ ;$$

hence, it is enough to bound $\Pr[C_i]$. Indeed, for any arbitrary $2t + q$ linear polynomials $P_{-2t+1} \ldots P_q$, the probability that for some pair $P_i \neq P_j$ and $[P_i - P_j](\bar{u}, \bar{u}\bar{x}) = 0$ is at most $\nu + 1/p$ by Claim 6.1. Taking union bound over $\binom{q+2t}{2} < (q+2t)^2/2$ pairs yields the required bound. □

# 7 Applications

In this section, we show how composable VGB point obfuscators, can be used to construct VGB set obfuscators and composable VGB point obfuscators for MBPCs. Then we discuss how these can be used to obtain strong encryption schemes that are simultaneously resilient to *key dependent messages* (KDM), *leakage* and *related key attacks* (RKA).

## 7.1 Application to Obfuscation of Set Circuits

Another application is obfuscation of *set membership circuits* (or set circuits in short). A set circuit $C_T : \mathbb{D}_n \to \{0,1\}$, returns 1 for any element in the set $T \subseteq \mathbb{D}_n$ and 0 for all other inputs. Again we deal with set circuits in some canonical form where the set is given explicitly. Set obfuscators have been considered in past work regarding extensions of point obfuscators [CD08, CV09]. We show that a natural construction described at [CD08] implies VGB (VBB) set obfuscators based on $t$-composable VGB (VBB) point obfuscators.

**Proposition 7.1.** *Let $\mathcal{O}$ be a $t$-composable point obfuscator. Consider a new PPT $\mathcal{O}'$, which given a set $T$ of size $|T| = t$, first chooses some random ordering of the elements, applies $\mathcal{O}$ to each circuit and wraps these obfuscations with a circuit that on input $z$ checks if $z$ is one of the obfuscated points (by applying an $\vee$ gate). Then $\mathcal{O}'$ is a set obfuscator.*

*Proof.* As in Section 5, we denote by $\mathcal{O}(C_{\bar{x}})$ the composition $\mathcal{O}(C_{x_1}), \ldots, \mathcal{O}(C_{x_t})$. Let $\mathcal{A}$ be a poly-size adversary (for set obfuscations) and let $p$ be a polynomial. Since $\mathcal{O}$ is a composable obfuscator there exists an efficient simulator $\mathcal{S}$, such that for any vector $\bar{x} \in \mathbb{D}_n^t$:

$$|\Pr[\mathcal{A}(\mathcal{O}(C_{\bar{x}})) = 1] - \Pr[\mathcal{S}^{C_{\bar{x}}}(1^n) = 1]| \leq \frac{1}{p(n)} \ ;$$

[10] formally these are defined on a joint probability space, where both the original and altered interaction are executed and it refers to the polynomials determined by the altered interaction. In particular these polynomials are independent of $\bar{x}, \bar{u}$.

in particular, for any set $T = \{x_1, \ldots, x_t\}$ of size $t$:

$$|\Pr[\mathcal{A}(\mathcal{O}'(C_T)) = 1] - \Pr[\mathcal{S}^{C_{\bar{\sigma}(T)}}(1^n) = 1]| \leq \frac{1}{p(n)} \ ,$$

where $\bar{\sigma}(T) = \langle x_{\sigma(1)}, \ldots, x_{\sigma(t)} \rangle$ is a random ordering of the elements in $T$. We now describe a simulator $\mathcal{S}'$ which simulates $\mathcal{A}$ with oracle access to a set circuit $C_T$. $\mathcal{S}'$ chooses a random permutation $\sigma$ and stores a table of size $t \times 2$, where the first column has indexes $i \in [t]$ and the second will represent corresponding values (at the beginning it is initialized with blanks). $\mathcal{S}'$ runs $\mathcal{S}$ and keeps a counter $c$ of how many distinct values were queried by $\mathcal{S}$ so far (where by distinct we mean distinct elements of $\mathbb{D}_n$, i.e. queries $(x, i), (x, j)$ are not considered distinct). When $\mathcal{S}$ queries $(x, i)$, $\mathcal{S}'$ first checks if $x$ is in the table. If it appears next to the index $i$, $\mathcal{S}'$ answers $\mathcal{S}$ with 1, if it appears but next to another index, $\mathcal{S}'$ answers with 0. In case it does not appear, $\mathcal{S}'$ queries $C_T$ on $x$, if $C_T(x) = 0$ it answers with 0 and continues. Otherwise it sets $c \leftarrow c + 1$ and writes $x$ in the table next to index $\sigma(c)$ and answers 1 only if $\sigma(c) = i$. We now note that for any set $T \subseteq \mathbb{D}_n$ of size $|T| = t$:

$$\Pr[\mathcal{S}'^{C_T}(1^n) = 1] = \Pr[\mathcal{S}^{C_{\bar{\sigma}(T)}}(1^n) = 1] \ ,$$

indeed, by our construction of $\mathcal{S}'$, the emulated $\mathcal{S}$ is experiencing oracle access to a truly random order on $T$. It follows that:

$$|\Pr[\mathcal{A}(\mathcal{O}'(T)) = 1] - \Pr[\mathcal{S}'^{C_T}(1^n) = 1]| \leq \frac{1}{p(n)} \ .$$

The proposition follows. □

## 7.2 Application to Obfuscation of Point Circuits with Multi-bit Output

A *multi-bit point circuit* (or MBPC in short) $C_{x \to y} : \mathbb{D}_n \to \{0,1\}^m$ returns $y$ on input $x$ and $\perp$ on all other inputs (once again we assume $C_{x \to y}$ is given in some *canonical* form where $x, y$ are explicit). MBPC obfuscators were constructed by [CD08] assuming the existence of a composable VBB point obfuscators. However, as explained earlier no known obfuscator has been shown to be composable. We show that applying the [CD08] construction to composable VGB point obfuscators results in VBB (rather than VGB) MBPC obfuscator that is also VGB composable. We remark that existing MBPOs were only shown to be secure for the restricted case that the message $m$ is independent of the key $k$ [CD08, CKVW10]. Moreover, they were not shown to be composable. Both properties are essential for the encryption schemes discussed in the next subsection, in order to get resilience to *key-dependent-messages and related key attacks*.

**Construction 7.1** (*Multibit-bit Output Point Obfuscator* [CD08])**.** *Let $\mathcal{O}$ be a point obfuscator. Define a PPT $\mathcal{O}^{(m)}$ for point circuits with $m$-bit output as follows. For a point $x \in \mathbb{D}_n$ and output $y = y_1 y_2 \ldots y_m \in \{0,1\}^m$, choose a random $s \in \mathbb{D}_n - \{x\}$ and define $\bar{a} = \langle a_0, a_1, \ldots, a_m \rangle$ as follows. $a_0 = x$, and for any $i \in [m]$ $a_i = x$ if $y_i = 1$ and $a_i = s$ otherwise. The output of the obfuscator is:*

$$\mathcal{O}^{(m)}(C_{x \to y}) = \mathcal{C}\left[\mathcal{O}(C_{a_0}), \ldots, \mathcal{O}(C_{a_m})\right] \ ,$$

*where each $\mathcal{O}(C_{a_i})$ is an obfuscation of the point circuit $C_{a_i}$, and $\mathcal{C}\left[\mathcal{O}(C_{a_0}), \ldots, \mathcal{O}(C_{a_m})\right]$ is a circuit that has the obfuscated programs $\mathcal{O}(C_{a_0}), \ldots, \mathcal{O}(C_{a_m})$ hardwired into it; it operates as follows: on input $z$, it first checks if $z = a_0 = x$ (by running the first obfuscated circuit), and if so, it returns $\perp$; otherwise, it finds all other coordinates such that $a_i = z = x$ (by running the rest of the obfuscated circuits) and outputs $y_1 \ldots y_m$, where $y_i = 1$ if $a_i = z = x$ and 0 otherwise.*

**Proposition 7.2.** *if $\mathcal{O}$ is an $(m+1)$-composable VGB point obfuscator, then $\mathcal{O}^{(m)}$ (given by Construction 7.1) is a VBB obfuscator for $m$-bit point circuits. Moreover, for any decomposition, $m+1 = t \times (m'+1)$ $\mathcal{O}^{(m')}$ is a $t$-composable MBPC VGB obfuscator .*

We prove this proposition in two steps. First we claim that if $\mathcal{O}$ is a $(m+1)$-composable VGB point obfuscator, then for any decomposition $m+1 = t \times (m'+1)$, $\mathcal{O}^{(m')}$ is a $t$-composable MBPC VGB obfuscator (in particular for $t = 1$ it is an $m$-bit VGB point obfuscator). The proof basically follows the arguments in [CD08] (replacing the bounded simulator by an unbounded one) and hence, we omit it. In the second step, we show that (putting composability aside) for MBPC VBB and VGB obfuscation are equivalent.

**Proposition 7.3.** *Assume $\mathcal{O}$ is a VGB obfuscator for the family of $m$-bit output point circuits. Then it is also a VBB obfuscator for the family.*

*Proof.* We start by giving the intuition behind the proposition. Note that a simulator that fails to query the hidden point has an output distribution that is actually independent of the hidden point (and corresponding output). In this case, the only concern we might have is that the simulator's output cannot be *efficiently* sampled (as the simulator is unbounded). However, this cannot be the case, as this distribution is strongly related to the one generated by the *efficient* adversary. Moreover, the strong simulator has only polynomially many oracle queries and hence there are relatively few elements that it queries with high probability. Thus, we will be able to perform bounded simulation by hardwiring those elements into our simulator. We remark that applying some of the techniques used in Theorem 5.1 and Proposition 5.1, one could strengthen the above proposition and show that for any $t = O(1)$ and $m = \mathrm{poly}(n)$, a $t$-composable VGB point obfuscator with $m$-bit output is in fact a $t$-composable VBB obfuscator. We present the proof only for the simple case $t = 1$.

We shall first prove two preliminary claims. In what follows, Let $\mathcal{A}$ be a binary poly-size adversary, $p$ a polynomial, and $\mathcal{S} = \mathcal{S}_{\mathcal{A},p}$ the corresponding VGB simulator (which is unbounded). Recall that $\mathcal{S}$ can make only $\mathrm{poly}(n)$ queries to its given oracle. We shall denote the number of allowed queries by $q = q(n)$.

**Claim 7.1.** *Let $Z$ be the function which returns $\perp$ on all inputs. Then there are at most $pq$ elements which $\mathcal{S}^Z$ queries with probability more than $1/p$ (over the coins of $\mathcal{S}$).*

*Proof of claim.* Denote by $X$ the distribution on query vectors (in $\mathbb{D}_n^q$) induced by $\mathcal{S}^Z$ and define

$$G_n^p = \left\{ a \in \mathbb{D}_n : \Pr_{\bar{x} \leftarrow X}[a \in \bar{x}] \geq 1/p \right\} \ .$$

Consider a distribution $\tilde{X}$ defined by first drawing a vector from $X$ and then choosing one of its coordinates uniformly. Then for any $a \in G_n^p$, it holds that $\tilde{X}(a) \geq 1/pq$, hence $|G_n^p| \leq pq$. □

**Claim 7.2.** *For any two values $x_1, x_2$ which $\mathcal{S}^Z$ queries with probability at most $1/p$ and for any $y_1, y_2$ it holds that :*

$$|\Pr[\mathcal{A}(\mathcal{O}(C_{x_1 \to y_1})) = 1] - \Pr[\mathcal{A}(\mathcal{O}(C_{x_2 \to y_2})) = 1]| \leq 4/p \ .$$

*Proof of claim.* Since $\mathcal{S}$ simulates $\mathcal{A}$ (with accuracy $1/p$):

$$\begin{aligned} &|\Pr[\mathcal{A}(\mathcal{O}(C_{x_1 \to y_1})) = 1] - \Pr[\mathcal{A}(\mathcal{O}(C_{x_2 \to y_2})) = 1]| \leq \\ &\quad |\Pr[\mathcal{S}^{C_{x_1 \to y_1}}(1^n) = 1] - \Pr[\mathcal{S}^{C_{x_2 \to y_2}}(1^n) = 1]| + 2/p \ . \end{aligned}$$

Denote by $Q$ the event (set of random tapes) in which $\mathcal{S}^Z$ queries $x_1$ or $x_2$ and note that:

$$\Pr[Q] \leq 2/p$$
$$\Pr[\mathcal{S}^{C_{x_1 \to y_1}}(1^n) = 1 | \overline{Q}] = \Pr[\mathcal{S}^{C_{x_2 \to y_2}}(1^n) = 1 | \overline{Q}] = \Pr[\mathcal{S}^Z(1^n) = 1 | \overline{Q}] \ ,$$

which in turn implies:

$$|\Pr[\mathcal{S}^{C_{x_1 \to y_1}}(1^n) = 1] - \Pr[\mathcal{S}^{C_{x_2 \to y_2}}(1^n) = 1]| =$$
$$|\Pr[Q](\Pr[\mathcal{S}^{C_{x_1 \to y_1}}(1^n) = 1 | Q] - \Pr[\mathcal{S}^{C_{x_2 \to y_2}}(1^n) = 1 | Q])| \leq \Pr[Q] \leq 2/p \ .$$

The claim follows. □

We now return to proving Proposition 7.3. Given an adversary $\mathcal{A}$ and a corresponding unbounded simulator $\mathcal{S}$ with accuracy $1/p$, we show how to construct an alternative simulator $\mathcal{S}'$ which is polynomially bounded and has polynomial accuracy $4/p$. $\mathcal{S}'$ has the set $G_n^p$ (given by the first claim) hardwired. Given oracle access to $C_{x_1 \to y_1}$, it first queries its oracle on all values in $G_n^p$. In case it found the hidden value $x_1$, it retrieves $y_1$, creates an obfuscation $\mathcal{O}(C_{x_1 \to y_1})$ and feeds it to $\mathcal{A}$, performing a perfect simulation. Otherwise, it chooses an arbitrary $x_2 \notin G_n^p$ and an arbitrary $y_2$, and as before runs $\mathcal{A}(\mathcal{O}(C_{x_2 \to y_2}))$. According to the second claim it achieves in this case a simulation with $4/p$ accuracy. The running time (or size) of $\mathcal{S}'$, is proportional to that of $\mathcal{A}$ plus an overhead of $|G_n^p| \leq pq$. □

## 7.3 Application to Strong Encryption Schemes

As noted in [CD08], obfuscation of MBPCs implies a very strong type of symmetric encryption (which they call a *digital locker)*. This usage was further explored by [CKVW10] who showed tight relations between MBPC (VBB) obfuscation and the notions of *weak key encryption* and *key dependent messages encryption* (KDM). Informally, they show that the existence of MBPC VBB obfuscators implies the existence of strong symmetric encryption schemes that are secure for key dependent messages even with weak random keys. We extend their results by showing that using composable VGB MBPC obfuscators (as the ones described above), similar implications still hold, even for the scenario of multiple messages and keys which are correlated (the implications of composable MBPC obfuscation to *related key attacks* resilient encryption (RKA) was not discussed in previous work).

*Remark* 7.1. Consistently with prior work, the encryption schemes discussed in this section are analyzed based on a simulation-based definition (specifically in our case, VGB); however, they can also analyzed directly using the $t$-DI obfuscation definition. The VGB simulation definition (which holds for arbitrary distributions on tuples of points) is more attractive from a definitional point of view, capturing more closely what we expect of obfuscation in general. In addition, one can consider conceptually stronger simulation-based definitions of KDM/RKA encryption, with respect to arbitrary distributions on keys and messages.

We start by presenting the basic natural transformation between MBPC obfuscators and symmetric encryption schemes.

**Construction 7.2** (MBPC Obfuscator to Symmetric Encryption)**.** *Let $\mathcal{O}$ be an MBPC obfuscator, define (probabilistic) encryption and decryption algorithms:*

$$E_k^{\mathcal{O}}(m) \triangleq \mathcal{O}(C_{k \to m})$$
$$D_k^{\mathcal{O}}(C) = C(k) \ ,$$

*where $C$ is interpreted as an MBPC and $k$ is a key taken from a domain of keys $\mathbb{D}_n$ (key sampling is addressed below).*

There are several definitions regarding KDM, RKA and leakage [BRS02, HK07, BHHO08, App08, AHI11]. We use a variant of the definition in [CKVW10] extended to the setting of multiple related keys. In this definition, $t$ keys are generated from a distribution $\mathcal{X} = \{X_n\}$ on key vectors in $\mathbb{D}_n^t$ and the adversary witnesses $t$ encryptions of predetermined functions of the keys. Any message might depend on any key, and the keys themselves might also be dependent according to the joint distribution $X_n$. The definition considers the case where the distributions $X_n$ are not necessarily uniform but only have certain entropy guarantee.

**Definition 7.1** (Encryption with multi keys-messages dependence - MKM)**.** *An encryption scheme $(E, D)$ is $(m, t)$-MKM secure if for any CWS distribution ensemble $\mathcal{X} = \{X_n\}$ on key vectors in $\mathbb{D}_n^t$, any poly-size $\mathcal{A}$, functions $f_1, \ldots, f_t : \mathbb{D}_n^t \to \{0,1\}^m$ and all large enough $n$:*

$$\left| \Pr_{\substack{\bar{k} \leftarrow X_n \\ E,\mathcal{A}}} [\mathcal{A}(E_{k_1}(f_1(\bar{k})), \ldots, E_{k_t}(f_t(\bar{k}))) = 1] - \Pr_{\substack{\bar{k} \xleftarrow{U} \mathbb{D}_n^t \\ E,\mathcal{A}}} [\mathcal{A}(E_{k_1}(\bar{0}), \ldots, E_{k_t}(\bar{0})) = 1] \right| = n^{-\omega(1)} \ ,$$

*where $m(n), t(n)$ are polynomially bounded length functions and $\bar{0} = 0^m$.*

**Theorem 7.1.** *Let $\mathcal{O}$ be a $t$-composable VGB obfuscator for $m$-bit point circuits, then the encryption scheme $(E^{\mathcal{O}}, D^{\mathcal{O}})$ is $(m, t)$-MKM secure .*

*Proof.* Let $\mathcal{X}$ be a CWS distribution ensemble over $t$-dimensional vectors (of keys) and $\mathcal{A}$ be a binary poly-size adversary. Since $\mathcal{O}$ is a $t$-composable VGB obfuscator for $m$-bit point circuits, for any polynomial $p$ there exists an (unbounded) simulator $\mathcal{S}$ which is allowed $q = \mathrm{poly}(n)$ many oracle queries and satisfies for all $\bar{k}, \bar{y} \in \mathbb{D}_n^t \times \{0,1\}^{m \times t}$ and sufficiently large $n$:

$$\left| \Pr_{\mathcal{A},\mathcal{O}}[\mathcal{A}(\mathcal{O}(C_{k_1 \to y_1}), \ldots, \mathcal{O}(C_{k_t \to y_t})) = 1] - \Pr_{\mathcal{S}}[\mathcal{S}^{C_{\bar{k} \to \bar{y}}[q]}(1^n) = 1] \right| \leq 1/4p \ ;$$

in particular, the above holds if $\bar{y} = f(\bar{k}) = \langle f_1(\bar{k}), \ldots, f_t(\bar{k}) \rangle$ for arbitrary functions $f_i$ with output of length $m$. This implies:

$$\left| \Pr_{\substack{\bar{k} \leftarrow X_n \\ E,\mathcal{A}}} [\mathcal{A}(E_{k_1}(f_1(\bar{k})), \ldots, E_{k_t}(f_t(\bar{k}))) = 1] - \Pr_{\substack{\bar{k} \xleftarrow{U} \mathbb{D}_n^t \\ E,\mathcal{A}}} [\mathcal{A}(E_{k_1}(\bar{0}), \ldots, E_{k_t}(\bar{0})) = 1] \right| =$$

$$\left| \Pr_{\substack{\bar{k} \leftarrow X_n \\ \mathcal{O}}} [\mathcal{A}(\mathcal{O}(C_{k_1 \to f_1(\bar{k})}), \ldots, \mathcal{O}(C_{k_t \to f_t(\bar{k})})) = 1] - \Pr_{\substack{\bar{k} \xleftarrow{U} \mathbb{D}_n \\ \mathcal{O}}} [\mathcal{A}(\mathcal{O}(C_{k_1 \to 0^m}), \ldots, \mathcal{O}(C_{k_t \to 0^m})) = 1] \right| \leq$$

$$\left| \Pr_{\bar{k} \leftarrow X_n} [\mathcal{S}^{C_{k_1 \to f(\bar{k})}, \ldots, C_{k_t \to f(\bar{k})}[q]}(1^n) = 1] - \Pr_{\bar{k} \xleftarrow{U} \mathbb{D}_n} [\mathcal{S}^{C_{k_1 \to 0^m}, \ldots, C_{k_t \to 0^m}[q]}(1^n) = 1] \right| + \frac{1}{2p} \ .$$

Assume WLOG that $\mathcal{S}$ is deterministic (by fixing its coins to those which maximize the above difference). To conclude the claim observe that the left term in the above sum is of negligible size. Indeed, for any CWS distribution $\mathcal{Y} = \{Y_n\}$ on $t$-dimensional vectors, the probability that $\mathcal{S}$ queries an element of a vector sampled from $Y_n$ is at most $q \cdot \max_{a \in \mathbb{D}_n} \Pr_{\bar{y} \leftarrow Y_n}[a \in \bar{y}]$, which is negligible. The latter implies that $\mathcal{S}$ distinguishes any two CWS distributions (such as the two distributions given above) with negligible probability. The result follows. □

*Remark* 7.2. The key dependent messages (KDM) resilience of Construction 7.2 is restricted to a non-adaptive model in which the adversary has to choose in advance the functions of the key which it is interested in[11]. We remark that this restricted setting is still meaningful and captures common KDM resilience such as the classical *circular dependence*.

*Remark* 7.3. There are several definitions of related key attacks (RKA) resilience that can be considered. [App08, AHI11] define a model in which a secret key $k$ is chosen uniformly, and the adversary witnesses encryptions under $t$ new keys $k_1 \dots k_t$ which are derived from $k$ according to some correlation function. The correlation function can be either determined in advance (non-adaptive RKA), or alternatively the adversary can adaptively choose correlation functions according to the encryptions it already witnessed (adaptive RKA). In general, Construction 7.2 only yields non-adaptive RKA. However, considering the instantiation of the scheme with the obfuscator given by Construction 6.1, one gets also adaptive RKA security for the family of affine functions of the key. This follows simply because the construction allows affine homomorphisms of the key (which yields adaptive RKA [AHI11]).

**Extension to asymmetric encryption.**

In case the underlying point obfuscator used in Constructions 7.1,7.2 can be re-randomized, we can in fact get a CPA-secure public key encryption scheme with essentially the same strong properties described above.

**Definition 7.2** (Re-randomizable obfuscator)**.** *Let $\mathcal{O}$ be an obfuscator for a family of circuits $\mathcal{C}$. Denote by $\mathcal{O}_r(C)$ an obfuscation of a circuit $C \in \mathcal{C}$ using random coins $r \in \mathcal{P}$ (for some domain $\mathcal{P}$). Also denote by $\mathcal{O}(C)$ the distribution given by drawing $r \xleftarrow{U} \mathcal{P}$ and computing $\mathcal{O}_r(C)$. We say that $\mathcal{O}$ is re-randomizable if there exists a PPT $\mathcal{R}$, such that for any $C \in \mathcal{C}$ and fixed $r_0 \in \mathcal{P}$, $\mathcal{R}(\mathcal{O}_{r_0}(C)) \triangleq \mathcal{O}(C)$.*

**Construction 7.3** (Re-randomizable point obfuscator to asymmetric bit encryption)**.** *Let $\mathcal{O}$ be a re-randomizable point circuit obfuscator with re-randomization algorithm $\mathcal{R}$. For a distribution $X$ on secret keys from $\mathbb{D}$, we define key generation, encryption and decryption algorithms:*

$$G^{\mathcal{O}}(X) \triangleq (sk, pk) : sk \leftarrow X, pk \leftarrow \mathcal{O}(C_{sk})$$

$$E^{\mathcal{O}}_{pk}(b) \triangleq C : s \xleftarrow{U} \mathbb{D} \setminus \{sk\}, C \leftarrow \begin{cases} \mathcal{R}(pk) & b = 1 \\ \mathcal{O}(C_s) & b = 0 \end{cases}$$

$$D^{\mathcal{O}}_{sk}(C) = C(sk) \ .$$

We can extend the MKM Definition 7.1 to the public key setting. In such an extension, $t$ secret keys are drawn from a CWS distribution, $t$ corresponding public keys are generated as in Construction 7.3, and the encrypted messages are allowed to depend on the keys (according) to pre-determined dependence functions. Using similar arguments as in Theorem 7.3, it follows that using a composable VGB point obfuscator in Construction 7.3 yields an MKM public key encryption scheme (for any polynomial number of secret keys). Finally, we note that the point obfuscator given by Construction 6.1 is indeed re-randomizable as in Definition 7.2.

# Acknowledgements

We thank Sebastian Gajek and Mayank Varia for helpful comments.

[11] This can be equivalently formulated as an adaptive definition where the family of correlation functions is polynomially bounded.

## References


[AHI11] Benny Applebaum, Danny Harnik, and Yuval Ishai. Semantic security under related-key attacks and applications. In *ICS*, pages 45–60, 2011.

[App08] Benny Applebaum. Fast cryptographic primitives based on the hardness of decoding random linear code. Technical Report TR-845-08, Princeton University, 2008. Available at `ftp://ftp.cs.princeton.edu/techreports/2008/845.pdf`.

[AW07] Ben Adida and Douglas Wikström. How to shuffle in public. In *TCC*, pages 555–574, 2007.

[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 *CRYPTO*, pages 1–18, 2001.

[BHHO08] Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision diffie-hellman. In *CRYPTO*, pages 108–125, 2008.

[Bon98] Dan Boneh. The decision diffie-hellman problem. In *ANTS*, pages 48–63, 1998.

[BRS02] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In *Selected Areas in Cryptography*, pages 62–75, 2002.

[Can97] Ran Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *CRYPTO*, pages 455–469, 1997.

[CD08] Ran Canetti and Ronny Ramzi Dakdouk. Obfuscating point functions with multibit output. In *EUROCRYPT*, pages 489–508, 2008.

[CKVW10] Ran Canetti, Yael Tauman Kalai, Mayank Varia, and Daniel Wichs. On symmetric encryption and point obfuscation. In *TCC*, pages 52–71, 2010.

[CMR98] Ran Canetti, Daniele Micciancio, and Omer Reingold. Perfectly one-way probabilistic hash functions (preliminary version). In *STOC*, pages 131–140, 1998.

[CRV10] Ran Canetti, Guy N. Rothblum, and Mayank Varia. Obfuscation of hyperplane membership. In *TCC*, pages 72–89, 2010.

[CV09] Ran Canetti and Mayank Varia. Non-malleable obfuscation. In *TCC*, pages 73–90, 2009.

[DS05] Yevgeniy Dodis and Adam Smith. Correcting errors without leaking partial information. In *STOC*, pages 654–663, 2005.

[GK05] Shafi Goldwasser and Yael Tauman Kalai. On the impossibility of obfuscation with auxiliary input. In *FOCS*, pages 553–562, 2005.

[GL89] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *STOC '89: Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 25–32, New York, NY, USA, 1989. ACM.

[GR07] Shafi Goldwasser and Guy N. Rothblum. On best-possible obfuscation. In *TCC*, pages 194–213, 2007.

[Had10] Satoshi Hada. Secure obfuscation for encrypted signatures. In *Eurocrypt*, 2010.

[HH09] Iftach Haitner and Thomas Holenstein. On the (im)possibility of key dependent encryption. In *TCC*, pages 202–219, 2009.

[HK07] Shai Halevi and Hugo Krawczyk. Security under key-dependent inputs. In *ACM Conference on Computer and Communications Security*, pages 466–475, 2007.

[HMLS07] Dennis Hofheinz, John Malone-Lee, and Martijn Stam. Obfuscation for cryptographic purposes. In *TCC*, pages 214–232, 2007.

[HRSV07] Susan Hohenberger, Guy N. Rothblum, Abhi Shelat, and Vinod Vaikuntanathan. Securely obfuscating re-encryption. In *TCC*, pages 233–252, 2007.

[LPS04] Ben Lynn, Manoj Prabhakaran, and Amit Sahai. Positive results and techniques for obfuscation. In *EUROCRYPT*, pages 20–39, 2004.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In *EUROCRYPT*, pages 256–266, 1997.

[Wee05] Hoeteck Wee. On obfuscating point functions. In *STOC*, pages 523–532, 2005.

# A Obfuscation with Auxiliary Input and Composability

In this section, we discuss obfuscation with auxiliary input. In this setting, the adversary also has some prior information regarding the obfuscated circuit. This notion was previously studied in [GK05] who referred to two variants, obfuscation with "dependent" auxiliary input, and with "independent" auxiliary input. Here, we discuss only the first (stronger) dependent auxiliary input variant. The following definition only concerns the security requirement, functionality and polynomial slow-down are also required as for standard obfuscators.

**Definition A.1** (Obfuscation with Auxiliary Input [GK05])**.** *$\mathcal{O}$ is an obfuscator with auxiliary input for a circuit ensemble $\mathcal{C} = \{\mathcal{C}_n\}$ if for any poly-size adversary $\mathcal{A}$ and polynomials $p, q$ there is a simulator $\mathcal{S}$ such that for all large enough $n \in \mathbb{N}$, $C \in \mathcal{C}_n$ and auxiliary input $z \in \{0,1\}^{q(n)}$:*

$$\left| \Pr_{\mathcal{A},\mathcal{O}}[\mathcal{A}(\mathcal{O}(C), z) = 1] - \Pr_{\mathcal{S}}[\mathcal{S}^C(1^n, z) = 1] \right| \leq \frac{1}{p(n)} \ .$$

*Remark* A.1. Once, again we can consider this definition in the VBB setting with a poly-size simulator or in the VGB setting with an unbounded simulator with polynomially many oracle queries.

**Obfuscation with auxiliary input and composability.** In the context of cryptographic protocols, auxiliary information is known to be tightly related to composability. Here, we question whether the same holds for obfuscation; in particular, whether point obfuscators with auxiliary input would imply composable point obfuscation. This was partially answered in [CD08] who showed that such an implication does not hold in general. They focus on a distributional obfuscation definition with uninvertible auxiliary input. We extend this to the general simulation (Definition A.1). However, we show that point obfuscation with auxiliary information does imply a more restricted notion of composability, namely constant-self-composability.

**Construction A.1** (From Point Obfuscation to Non-composable Point Obfuscation [CD08])**.** *Let $\mathcal{O}$ be a point obfuscator for the domain $\mathbb{D}_n = \{0,1\}^n$. Consider a new algorithm $\mathcal{O}'$ defined as follows:*

$$\mathcal{O}'_{r,s}(C_x) = (\mathcal{O}_r(C_x), s, x \cdot s) \ ,$$

*where $\mathcal{O}_r(C_x)$ denotes an obfuscation of $x$ using random coins $r$, $s \xleftarrow{U} \{0,1\}^n$ and $x \cdot s$ denotes the scalar product* $\bmod 2$.

In [CD08], it is shown that if $\mathcal{O}$ is a VBB point obfuscator then so is $\mathcal{O}'$. However, $\mathcal{O}'$ is not $\Omega(n)$-self-composable as $\Omega(n)$ linear equations in $x$ allow to fully recover it. The proof that $\mathcal{O}'$ remains an obfuscator relies on the fact that $\mathcal{O}(C_x)$ is one-way in $x$ as long as $x$ is taken from a Well Spread distribution $X_n$, i.e. $H_\infty(X_n) = \omega(\log n)$. The latter implies by [GL89] that

$$(\mathcal{O}_r(C_x), s, x \cdot s) \approx_c (\mathcal{O}_r(C_x), s, b) \ ,$$

for $b \stackrel{U}{\leftarrow} \{0,1\}$ and $x$ which is taken from any WS distribution. This in turn implies that $\mathcal{O}'$ is indeed an obfuscator (an easy way to see it would be applying Theorem 5.1). We now question whether the same idea should also work if we also consider auxiliary input. That is, we assume that $\mathcal{O}$ is a point obfuscator with auxiliary input and ask whether $\mathcal{O}'$ is also a point obfuscator with auxiliary input. Now, it is not sufficient that $\mathcal{O}(C_x)$ is one-way, as we should consider $\mathcal{O}(C_x), z$ together. We show the following proposition.

**Proposition A.1.** *Let $\mathcal{O}$ be a VBB point obfuscator with auxiliary input (as in Definition A.1). Then $\mathcal{O}'$ given by Construction A.1 is a VBB point obfuscator with auxiliary input which is not $\Omega(n)$-self-composable.*

*Proof sketch.* The fact that $\mathcal{O}'$ is not composable is shown in the same way as in the standard case (with no auxiliary input). We focus on showing that $\mathcal{O}'$ is indeed a point obfuscator with auxiliary input. Informally, we consider two cases. In the first, the extra bit $x \cdot s$ appears random (even given the obfuscation and auxiliary information), which will allow easy simulation. In the second case, the adversary can predict this extra bit with noticeable chance. Here, the fact that $\mathcal{O}$ is a VBB obfuscator that is secure against binary adversaries implies a simulator that, given the auxiliary input and oracle access to the hidden point circuit $C_x$, can also predict this with noticeable chance. However, this implies that with noticeable chance it can also recover the hidden point $x$ using the well known [GL89] list decoding algorithm. This suffices for perfect simulation.

For the actual proof, let $\mathcal{A}$ be a binary poly-size adversary and let $p, q$ be polynomials. Our goal is to construct a simulator for $\mathcal{A}$ as in Definition A.1. We call $(x, z) \in \{0,1\}^n \times \{0,1\}^{q(n)}$ a *distinguishing pair* if:

$$\left| \Pr_{\mathcal{A},r,s}[\mathcal{A}(\mathcal{O}_r(C_x), s, x \cdot s, z) = 1] - \Pr_{\mathcal{A},r,s,b}[\mathcal{A}(\mathcal{O}_r(C_x), s, b, z) = 1] \right| \geq \frac{1}{2p(n)} \ ,$$

where $r$ are random coins for $\mathcal{O}$, $s \stackrel{U}{\leftarrow} \{0,1\}^n$ and $b \stackrel{U}{\leftarrow} \{0,1\}$. We can now transform the distinguisher $\mathcal{A}$ to a predictor $\mathcal{P}$, such that for any distinguishing pair $(x, z)$:

$$\Pr_{r,s}[\mathcal{P}(\mathcal{O}_r(C_x), s, z) = x \cdot s] \geq \frac{1}{2} + \frac{1}{2p(n)} \ .$$

Since $\mathcal{O}$ is an obfuscator with auxiliary input, there is a poly-size simulator $\mathcal{S}_\mathcal{P}$, such that for all large enough $n$ and any $(x, s, z) \in \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^{q(n)}$:

$$\left| \Pr_{\mathcal{S}_\mathcal{P}}[\mathcal{S}_\mathcal{P}^{C_x}(s, z) = x \cdot s] - \Pr_{r}[\mathcal{P}(\mathcal{O}_r(C_x), s, z) = x \cdot s] \right| \leq \frac{1}{4p(n)} \ .$$

It follows that for all large enough $n$, any distinguishing pair $(x, z) \in \{0,1\}^n \times \{0,1\}^{q(n)}$ and a random $s \stackrel{U}{\leftarrow} \{0,1\}^n$:

$$\Pr_{\mathcal{S}_\mathcal{P},s}[\mathcal{S}_\mathcal{P}^{C_x}(s, z) = x \cdot s] \geq \frac{1}{2} + \frac{1}{4p(n)} \ .$$

By [GL89] we can use $\mathcal{S}_{\mathcal{P}}$ to obtain a poly-size invertor $\mathcal{I}$ such that for the all large enough $n$ and distinguishing pairs $(x, z) \in \{0,1\}^n \times \{0,1\}^{q(n)}$:

$$\Pr_{\mathcal{I}}[\mathcal{I}^{C_x}(z) = x] \geq n^{-O(1)} \ .$$

Since $\mathcal{I}$ can verify a successful inversion using its oracle, we can amplify it to get an invertor which obtains $x$ with overwhelming probability.

Finally, we can construct a simulator $\mathcal{S}_{\mathcal{A}}$ for $\mathcal{A}$. First, we consider an adversary $\mathcal{B}$ that, given an obfuscation $\mathcal{O}_r(C_x)$ and auxiliary input $z$, samples $(s, b) \overset{U}{\leftarrow} \{0,1\}^n \times \{0,1\}$ and runs $\mathcal{A}$ on $\mathcal{O}_r(C_x), s, b, z$. $\mathcal{B}$ can be simulated by a poly-size simulator $\mathcal{S}_{\mathcal{B}}$ with accuracy $1/2p(n)$. Now, to simulate $\mathcal{A}$, given auxiliary input $z$ and oracle access to $C_x$, $\mathcal{S}_{\mathcal{A}}$ first runs $\mathcal{I}^{C_x}(z)$ and records the result $\tilde{x}$. In case $\tilde{x}$ is the hidden point $x$, it can perform a perfect simulation. Otherwise, it runs $\mathcal{S}_{\mathcal{B}}^{C_x}(z)$ . It follows that for any non distinguishing pair $\mathcal{S}_{\mathcal{A}}$ has simulation accuracy $\frac{1}{p(n)}$ and for any distinguishing pair it performs almost perfect simulation (up to the negligible probability that $\mathcal{I}$ fails to obtain $x$). □

**Proposition A.2.** *Any VBB obfuscator with auxiliary input is also $c$-self-composable for any constant $c$.*

*Proof sketch.* For simplicity, we start with the case $c = 2$. Let $\mathcal{O}$ be an obfuscator with auxiliary input for a family of circuits $\mathcal{C} = \{\mathcal{C}_n\}$, and let $\mathcal{A}$ be a binary poly-size adversary and $p$ a polynomial. By obfuscation with auxiliary input, there is a poly-size simulator $\mathcal{S}$, such that for any $C \in \mathcal{C}_n$ and auxiliary input $z \in \{0,1\}^{q(n)}$:

$$\Pr_{\mathcal{A},r,s}[\mathcal{A}(\mathcal{O}_r(C), \mathcal{O}_s(C), z) = 1] - \Pr_{s}[\mathcal{S}^C(\mathcal{O}_s(C), z) = 1] \leq \frac{1}{2p(n)} \ ,$$

where we treated the second obfuscation as auxiliary input. Now, we can consider a poly-size adversary $S'$ that perfectly simulates $\mathcal{S}^C(\mathcal{O}_s(C), z)$ with no oracle access to $C$. Instead, it uses the obfuscation $\mathcal{O}_s(C)$ to evaluate the oracle queries of $\mathcal{S}$. Now since $\mathcal{O}$ is an obfuscator with auxiliary input, it follows that $\mathcal{S}'$ can also be simulated by $\mathcal{S}''^C(z)$ with accuracy $\frac{1}{2p(n)}$ (where $\mathcal{S}''$ is also of poly-size). The result follows for $c = 2$. In general, we can use the above argument to show that for any polynomially bounded function $t = t(n)$, if $\mathcal{O}$ is $t$-self-composable with auxiliary input it is also $t + 1$-self-composable with auxiliary input. In particular we get $c$-self-composability for any constant $c$. □

**On VGB obfuscation with auxiliary input.** When considering point obfuscators, it seems that VBB obfuscators with auxiliary input is a stronger notion than plain point obfuscators (with no auxiliary input). In particular, it implies constant-self-composability, while plain obfuscators are not known to imply it (and are even conjectured not to imply it). In contrast, the following proposition shows that for VGB obfuscators, the notion of VGB obfuscation with auxiliary input is not stronger than plain VGB obfuscation.

**Proposition A.3.** *Let $\mathcal{O}$ be a VGB obfuscator for a circuit ensemble $\mathcal{C} = \{\mathcal{C}_n\}$. Then $\mathcal{O}$ is also a VGB obfuscator with auxiliary input for the ensemble.*

*Proof sketch.* To show this we use a similar idea to the one used in the proof of Theorem 5.1. Roughly speaking, we note that for a fixed auxiliary input VGB implies simulation on all circuits in the family. In general, different auxiliary inputs correspond to different simulators, while we wish to have a single simulator for all auxiliary inputs. However, a VGB simulator which gets some auxiliary input $z$, can compute on its own the best simulator corresponding to $z$ and run it.

More formally, for any adversary $\mathcal{A}$, polynomial $q$, and auxiliary information sequence $\mathcal{Z} = \left\{z_n \in \{0,1\}^{q(n)}\right\}_{n\in\mathbb{N}}$, we can consider a new (non-uniform) adversary $\{\mathcal{A}(\cdot, z_n)\}_{n\in\mathbb{N}}$ which has

$z_n$ hardwired. By VGB (with no auxiliary input) for any polynomial $p$ there is a VGB simulator $\mathcal{S} = \mathcal{S}_{\mathcal{Z}}$ which makes $r(n) = \text{poly}(n)$ queries, such that for all large enough $n \in \mathbb{N}$ and any $C \in \mathcal{C}_n$:

$$\Pr[\mathcal{A}(\mathcal{O}(C), z_n) = 1] - \Pr[\mathcal{S}^{C[r]}(1^n) = 1] \leq \frac{1}{p(n)} \ . \tag{1}$$

Now for any[12] $n \in \mathbb{N}$ and $z \in \{0,1\}^{q(n)}$, let $\mathcal{S}_n = \mathcal{S}_n(z)$ be a circuit with minimal number of oracle queries that satisfies for all $C \in \mathcal{C}_n$:

$$\Pr[\mathcal{A}(\mathcal{O}(C), z) = 1] - \Pr[\mathcal{S}_n^C(1^n) = 1] \leq \frac{1}{p(n)} \ .$$

Consider a family of functions $\mathcal{F} = \{F_n\}$ that, given an input $z \in \{0,1\}^{q(n)}$, returns $\mathcal{S}_n(z)$. We show that there is a uniform bound $r(n) = \text{poly}(n)$ on the number of oracle queries made by the circuits that $\mathcal{F}$ outputs. Indeed, we can consider the auxiliary information sequence $\mathcal{Z}^* = \{z_n^*\}$ which maximizes $\mathcal{S}_n(z)$, and get a polynomial bound given by Equation (1). We can now construct a simulator $\mathcal{S}$ for $\mathcal{A}$ which performs well on any $C \in \mathcal{C}_n$ and $z \in \{0,1\}^{q(n)}$. On input $z$ and oracle access to $C$, $\mathcal{S}$ simply computes $F_n(z)$ and runs the resulting simulator. The proposition follows.

□

# B More on Distributional Indistinguishability and [Can97]

The DI Definition 5.3 presented in this work is formulated differently from the DI definition in [Can97], and is seemingly cleaner and simpler. We show that both formulations are indeed equivalent. We address the single point case, which is the focus of [Can97]. In particular, we deal with well spread (WS) distributions, which is a special case of the CWS distributions given by Definition 5.2 (restricted to a single point).

**Definition B.1** (Distributional Indistinguishability [Can97])**.** *A point obfuscator $\mathcal{O}$ is DI if for any poly-size adversary $\mathcal{A}$ and any WS distribution ensemble $\mathcal{X} = \{X_n\}$ on points in $\{\mathbb{D}_n\}$:*

$$\mathcal{X}_1, \mathcal{A}(\mathcal{O}(C_{\mathcal{X}_1})) \approx_c \mathcal{X}_1, \mathcal{A}(\mathcal{O}(C_{\mathcal{X}_2})) \ ,$$

*where $\mathcal{X}_1, \mathcal{X}_2$ are two independent instances of the distribution $\mathcal{X}$.*

**Proposition B.1.** *Definitions 5.3 and B.1 are equivalent.*

*Proof.* Throughout the proof we use the following notations. For an element $x \in \mathbb{D}_n$, define:

$$P_x = \Pr_{\mathcal{A},\mathcal{O}}[\mathcal{A}(\mathcal{O}(C_x)) = 1] = \mathop{\mathbb{E}}_{\mathcal{A},\mathcal{O}} \mathcal{A}(\mathcal{O}(C_x)) \ ,$$

where $\mathbb{E}$ denotes expectation.

For a distribution $X$ on points in $\mathbb{D}_n$, define

$$P_X = \mathbb{E}_{x \xleftarrow{X} \mathbb{D}_n} P_x \ .$$

We start by showing that if Definition 5.3 holds then so does B.1. Concretely, we show something stronger, the distribution ensembles given in B.1 are statistically indistinguishable.

[12] To be more accurate, we should first note that for all large enough $n \in \mathbb{N}$ and any $z \in \{0,1\}^{q(n)}$ there is indeed a circuit with the required property. Then we can restrict our discussion to all large enough $n$'s

**Claim B.1.** *Assume there exists an adversary $\mathcal{A}$, and a WS distribution ensemble $\mathcal{X}$ such that*

$$\mathrm{SD}(X \circ \mathcal{A}(\mathcal{O}(C_X)), X \circ \mathcal{A}(\mathcal{O}(C_Y))) \geq \epsilon \ ,$$

*for infinitely many $n$'s, where $X, Y$ are two independent instance of $X_n$ and $\epsilon = n^{-O(1)}$. Then for all such $n$'s either $|P_{X_n} - P_{U_n}| \geq \epsilon/4$ or there exists another distribution $X'_n$ with $H_\infty(X'_n) = H_\infty(X_n) - \log \frac{4}{\epsilon}$ for which $|P_{X'_n} - P_{U_n}| \geq \epsilon/4$.*

This claim implies there exist some WS distribution ensemble $\mathcal{X}''$ (consisting of the distributions $X_n$ or $X'_n$) such that $\mathcal{A}$ distinguishes $\mathcal{O}(\mathcal{X})$ from $\mathcal{O}(\mathcal{U}))$ with advantage $\epsilon/4$, contradicting Definition 5.3.

*Proof of claim.* Consider any $n$ for which the assumption holds and let $X, Y$ be two independent instances of $X_n$. Define the set: $G = \{x \in \mathbb{D}_n : |P_x - P_X| \geq \epsilon/2\}$ . We first show that $\Pr[X \in G] \geq \epsilon/2$. Indeed for any set $T \subseteq \mathbb{D}_n$:

$$\begin{aligned}
&|\Pr[X \in T, \mathcal{A}(\mathcal{O}(C_X)) = 1] - \Pr[X \in T, \mathcal{A}(\mathcal{O}(C_Y)) = 1]| \leq \\
&\Pr[X \in G] + \Pr[X \notin G] \Pr[X \in T | X \notin G] \left|\Pr[\mathcal{A}(\mathcal{O}(C_X)) = 1 | X \in T \setminus G] - \Pr[\mathcal{A}(\mathcal{O}(C_Y)) = 1]\right| \leq \\
&\Pr[X \in G] + \left|P_{X|X \in T \setminus G} - P_X\right| \leq \Pr[X \in G] + \epsilon/2 \ ,
\end{aligned}$$

where we used the fact that $X, Y$ are independent and the definition of $G$. It follows that:

$$\begin{aligned}
\Pr[X \in G] \geq \max_{T \subseteq \mathbb{D}_n} |\Pr[X \in T, \mathcal{A}(\mathcal{O}(C_X)) = 1] - \Pr[X \in T, \mathcal{A}(\mathcal{O}(C_Y)) = 1]| - \frac{\epsilon}{2} = \\
\mathrm{SD}(X \circ \mathcal{A}(\mathcal{O}(C_X)), X \circ \mathcal{A}(\mathcal{O}(C_Y))) - \frac{\epsilon}{2} \geq \frac{\epsilon}{2} \ .
\end{aligned}$$

Consider now the sets:

$$\begin{aligned}
G_+ &= \{x \in \mathbb{D}_n : P_x - P_X \geq \epsilon/2\} \\
G_- &= \{x \in \mathbb{D}_n : P_X - P_x \geq \epsilon/2\} \ .
\end{aligned}$$

Then, $G = G_+ \cup G_-$, and hence one of them has density at least $\epsilon/4$, assume WLOG that it holds for $G_+$, and define $X' = X | X \in G_+$. It clearly holds that $H_\infty(X') = H_\infty(X') - \log \frac{4}{\epsilon}$. Moreover, $\frac{\epsilon}{2} \leq P_{X'} - P_X \leq |P_{X'} - P_U| + |P_X - P_U|$. The result follows. □

We now show that Definition B.1 implies that Definition 5.3.

**Claim B.2.** *Assume there exists an adversary $\mathcal{A}$, and a WS distribution ensemble $\mathcal{X}$ such that $\mathcal{A}$ distinguishes $\mathcal{O}(\mathcal{X})$ from $\mathcal{O}(\mathcal{U})$ with advantage $\epsilon$, i.e. $|P_{X_n} - P_{U_n}| \geq \epsilon$ for infinitely many $n$'s. Then there exists a WS distribution ensemble $\mathcal{X}'$ and a poly-size distinguisher $\mathcal{D}$, which distinguishes $\mathcal{X}'_1, \mathcal{A}(\mathcal{O}(C_{\mathcal{X}'_1}))$ from $\mathcal{X}'_1, \mathcal{A}(\mathcal{O}(C_{\mathcal{X}'_2}))$ with advantage $\Omega(\epsilon)$.*

*Proof of claim.* Consider any $n$ for which the assumption holds. Let $Z_n \triangleq \frac{X_n + U_n}{2}$ be the distribution defined by flipping a fair coin, and then drawing a sample from $X_n$ or $U_n$ according to the result. Note that $H_\infty(Z_n) \geq \min\{H_\infty(X_n), H_\infty(U_n)\} \geq H_\infty(X_n)$. Moreover,$P_{Z_n} = \frac{P_{U_n} + P_{X_n}}{2}$ and hence $|P_{Z_n} - P_{X_n}| \geq \epsilon/2$. We first show that at least one of the distributions $X \in \{X_n, Z_n\}$ satisfies $\Pr_{x \xleftarrow{X} \mathbb{D}_n}[|P_x - P_X] \geq \epsilon/4] \geq 1/3$. Indeed, assume this does not hold for $X_n$, then

$$\Pr_{x \xleftarrow{Z_n} \mathbb{D}_n}[|P_x - P_{Z_n}] \geq \epsilon/4] \geq \frac{1}{2} \Pr_{x \xleftarrow{X_n} \mathbb{D}_n}[|P_x - P_{X_n}] \leq \epsilon/4] \geq \frac{1}{2} \cdot \frac{2}{3} \ . \tag{1}$$

Now, construct $\mathcal{X}'$ consisting of the distributions $X_n, Z_n$, according to Equation (1). Then $\mathcal{X}'$ is clearly WS and we can now construct a distinguisher $\mathcal{D}$ as required. Given input $x, b \in \mathbb{D}_n \times \{0, 1\}$, $\mathcal{D}$ first

estimates $P_x$ using $n/\epsilon^2$ samples of $\mathcal{A}(\mathcal{O}(C_x))$. In case its estimate satisfies $|\tilde{P}_x - P_{X'_n}| \geq \epsilon/8$ it outputs $b$, and otherwise it outputs 1. Denote by $F$ the first event (i.e. $\mathcal{D}$ outputs $b$), and by $G$ the event in which $\left|P_x - P_{X'_n}\right| \geq \frac{\epsilon}{4}$. Let $X, Y$ be two independent instances of $X'_n$. Then for the infinitely many $n$'s satisfying $\Pr[G] \geq 1/3$ it holds that:

$$\begin{aligned} |\Pr[\mathcal{D}(X, \mathcal{A}(\mathcal{O}(C_X))) = 1] - \Pr[\mathcal{D}(X, \mathcal{A}(\mathcal{O}(C_Y))) = 1]| \geq \\ \Pr[G, F] \left|P_{X|G,F} - P_{X'_n}\right| - \Pr[\overline{G}, F] \geq \\ \frac{1}{2} \cdot \frac{\epsilon}{4} - e^{-\Omega(n)} \quad , \end{aligned}$$

where we used the fact that given $\overline{F}$, $\mathcal{D}$ outputs the same for both distributions and the fact that both $\Pr[\overline{F}|G]$ and $\Pr[F|\overline{G}]$ are bounded by $\Pr\left[\left|\tilde{P}_x - P_x\right| \geq \epsilon/8\right]$ which is at most $e^{-\Omega(n)}$ by Chernoff inequality. □

This completes the proof of Proposition B.1 □