# Impossibility of VBB Obfuscation with Ideal Constant-Degree Graded Encodings

Rafael Pass[1] and abhi shelat[2]

[1] Cornell University, rafael@cs.cornell.edu*
[2] U. of Virginia, abhi@virginia.edu**

**Abstract.** A celebrated result by Barak *et al.* (Crypto'01) shows the impossibility of general-purpose *virtual black-box (VBB)* obfuscation in the plain model. A recent work by Canetti, Kalai, and Paneth (TCC'15) extends this impossibility result to the random oracle model (assuming trapdoor permutations).

In contrast, Brakerski-Rothblum (TCC'14) and Barak *et al.* (EuroCrypt'14) show that in *idealized* graded encoding models, general-purpose VBB obfuscation indeed is possible; these constructions require graded encoding schemes that enable evaluating *high-degree* (polynomial in the size of the circuit to be obfuscated) polynomials on encodings.

We show a complementary impossibility of general-purpose VBB obfuscation in idealized graded encoding models that enable only evaluation of *constant-degree* polynomials (assuming trapdoor permutations).

## 1 Introduction

The goal of *program obfuscation* is to "scramble" a computer program in order to hide its implementation details (making it hard to "reverse-engineer") while preserving its functionality (i.e, input/output behavior). The most desirable notion of security—*virtual black-box security* (VBB) [BGI+01]—requires that any bit of information an attacker can learn from the obfuscated code can be simulated using only black-box access to the functionality.[3] The celebrated result of Barak *et al.* [BGI+01], however, demonstrates a strong impossibility result regarding VBB obfuscation: they show the existence of families of functions $\{f_s\}$ for which black-box access to $f_s$ (for a randomly chosen $s$) does not

* Work supported in part by a Microsoft Faculty Fellowship, Google Faculty Award, NSF Award CNS-1217821, NSF Award CCF-1214844, AFOSR Award FA9550-15-1-0262 and DARPA and AFRL under contract FA8750-11-2-0211.

** Work performed while visiting Cornell Tech, and supported by NSF CAREER Award 0845811, NSF TC Award 1111781, NSF TC Award 0939718, DARPA and AFRL under contract FA8750-11-C-0080, Microsoft New Faculty Fellowship, SAIC Scholars Research Award, and Google Faculty Award.

[3] A similar simulation-based, but even stronger, notion of security was previously defined by Hada [Had00]. Even earlier, Canetti [Can97] considered a similar notion of security (without explicitly referring to obfuscation) for the special case of what is now referred to as *point-function obfuscation*.

leak any advantage in guessing even a single bit of $s$, but the code of any program that computes $f_s$ allows recovery of the entire secret $s$. The idea behind their impossibility result is to consider a function $f_s$ that satisfies two properties 1) the function is not learnable (thus given black-box access to it, it is hard to find a concise representation of it), but 2) on input a program $\Pi$ that computes the function $f_s$, $f_s(\Pi)$ reveals some secret. The code of the obfuscated program is thus an input on which the function releases the secret, yet the secret cannot be recovered using just black-box access to the function.

This impossibility result, however, only applies in the *plain* model in which the obfuscated code is a standard circuit that does not make oracle calls to external functionalities (or else, we cannot feed this code as an input to the function). In contrast, Canetti and Vaikuntanathan [CV13] show an obfuscator for $NC^1$ circuits in an idealized composite-order group with special pseudo-free properties. More recently, Brakerski and Rothblum [BR14] and Barak, Garg, Kalai, Paneth and Sahai [BGK$^+$14], following the breakthrough obfuscation construction of Garg, Gentry, Halevi, Raykova, Sahai, and Waters [GGH$^+$13b][4], demonstrate VBB obfuscation for all polynomial-size circuits in the idealized *graded encoding* [GGH13a] (a.k.a. "approximate" multilinear map [BS03,Rot13]) model.

In the *idealized graded encoding model* [BR14,BGK$^+$14], players have black-box access to a field $\mathbb{F}_p$ (where $p$ is a prime), but they can only perform certain restricted operations on field elements and determine whether an expression evaluates to zero. For instance, the simplest form of graded encodings of [GGH13a] enables computing all polynomials of some (a-priori) bounded polynomial degree, and determine whether the polynomial evaluates to zero; this is referred to as a "zero-test query"[5]. Note that a *generic group* [Sho97] model for $\mathbb{Z}_p^*$ where $p$ is a prime can be viewed as a special-case of an idealized graded encoding model in which operations are restricted to be linear (i.e., degree 1 polynomials). Degree two graded encodings capture idealized groups with bilinear maps.

A natural question is whether weaker idealized models such as the generic group model or idealized groups with bilinear maps suffice for obtaining VBB obfuscation for polynomial-size circuits. This question was first addressed by Lynn, Prabhakaran and Sahai [LPS04] who showed positive obfuscation results for specific functions in the Random Oracle model [BR93] where both the obfuscator and the evaluator have oracle access to a truly random function; they left open the question of whether general-purpose obfuscation in the Random Oracle model is possible. This open question was recently answered in an elegant work by Canetti, Kalai and Paneth [CKP15] who show that the impossibility result of [BGI$^+$01] also extends to the Random Oracle Model. [CKP15] in turn left open the questions of whether general-purpose

[4] The construction of [GGH$^+$13b] was proved to satisfy the weaker notion of *indistinguishability obfuscation* in an idealized "matrix-multiplication" model.

[5] The constructions in [BR14,BGK$^+$14] require certain additional "set-based" restrictions on polynomials; we return to this in Section 2.2.

VBB obfuscation in more sophisticated idealized models (such as the generic group model) is possible.

*Our Results* In this work, we show impossibility of VBB obfuscation in idealized graded encoding models that restrict zero-tests to degree-$d$ polynomials, where $d$ is a *constant*.

**Theorem 1 (Informally stated)** *Assuming the existence of trapdoor permutations, there exists a family of functions F for which there do not exist VBB obfuscators for F in idealized degree-d graded encoding models, where d is a constant.*

Our theorem stands in contrast with the results of [BR14] and [BGK$^+$14] which indeed show feasibility of general-purpose VBB obfuscation in an idealized graded encoding model that allows for *high-degree* (polynomial in the size of the circuit being obfuscated) zero-test queries.

The obfuscator construction of [BGK$^+$14] also satisfies *subexponential* VBB security (that is security holds also with respect to subexponential-size attackers). Our main theorem extends to rule out general-purpose VBB obfuscation with subexponential security in idealized graded encoding models that allow for $n^\alpha$-degree zero-test queries (where $\alpha < 1$ and $n$ is the description length of the function being obfuscated).

*Follow-up Work* We note that our proof directly generalizes to any graded encoding scheme that operates on elements in a ring (as opposed to $\mathbb{F}_p$) as long as a) there exists an efficient method for determining the row-rank of a matrix of this ring, and) the row-rank of a matrix is polynomially bounded by the column-rank. In follow-up work, Mahmoody, Mohammed, and Nematihaji [MMN15] have extended our techniques to apply to more general rings.

## 2 Definitions and Preliminaries

### 2.1 Virtual Black-box ObfuscatAion

We recall the definition of approximate VBB obfuscation from Barak *et al*. [BGI$^+$01], and Canetti, Kalai, and Paneth [CKP15], and generalize it for any family of oracles $M$ that are indexed by a security parameter.

**Definition 1 ($\epsilon$-Approximate VBB Obfuscation in an Oracle model [CKP15,BGK$^+$14])** *For a function $\epsilon : \mathbb{N} \to \{0,1\}$, an obfuscator $\mathcal{O}$ is a secure $\epsilon$-approximate virtual black-box (VBB) obfuscation for the family F in the M-oracle model if it satisfies the following properties:*

- *Approximate Functionality: for all $n \in \mathbb{N}$, $k \in \{0,1\}^n$:*

$$\Pr\left[\mathcal{O}^{M_{|k|}}(k)(x) \neq F_k(x)\right] \leq 1 - \epsilon(n)$$

  *where the probability is over the choice of x and the coins of M and $\mathcal{O}$.*

– *Virtual Black-Box (VBB): for every poly-size adversary $A$, there exists a poly-size simulator $S$ and a negligible function $\mu$ such that for every $k \in \{0,1\}^*$:*

$$\left|\Pr\left[A^{M_{|k|}}(\mathcal{O}^{M_{|k|}}(k)) = 1\right] - \Pr\left[S^{F_k}(1^{|k|}) = 1\right]\right| \leq \mu(|k|)$$

*where the probability it taken over the coins of $M$, $\mathcal{O}$, adversary $A$ and the simulator $S$.*

*We simply say that $\mathcal{O}$ is a secure VBB obfuscator if $\epsilon = 1$. We further say that $\mathcal{O}$ is a secure ($\epsilon$-approximate) obfuscation in the plain model for the family $F$ if it is a secure ($\epsilon$-approximate) obfuscation for the family $F$ in the $\perp$-oracle model where the $\perp$-oracle returns $\perp$ on every query.*

*We finally say that $\mathcal{O}$ is subexponentially-secure if the VBB condition holds with respect to any subexponential-size[6] $A$ and a subexponential-size $S$.*

Our definition of subexponentially-secure VBB obfuscation is incomparable to the definition of VBB obfusaction: it is stronger in that we require simulation of subexponential-size attackers, but it is weaker in that we allow the simulator to be subexponential size (even if the attacker is polynomial in size).

We use the following theorem by Bitansky and Paneth [BP13] and its extension which follows by relying on stronger trapdoor permutations. We choose specific constants for simplicity of notation; the theorem holds for any constants.

**Theorem 2 ([BP13])** *Assuming the existence of trapdoor permutations, there exists a family of polynomial-time computable functions $F$ such that a polynomial-size* 0.8*-approximate VBB obfuscator for $F$ does not exist.*

**Theorem 3 (scaled version of [BP13])** *Assuming the existence of sub-exponentially secure[7] trapdoor permutations, there exists a family of polynomial-time computable functions $F$ such that a subexponential-size* 0.8*-approximate subexponentially-secure VBB obfuscator for $F$ does not exist.*

### 2.2 Idealized Graded Encodings

We now define the ideal level-$d$ graded encoding oracle. For simplicity of notation, we consider an oracle that has the size of the field hard-coded. Our model, inspired by the formalism from [PST14,BR14,BGK+14,Sho97], considers a simple idealized graded encoding oracle which enables players to a) encode an element $v$ under a "label" $l$, and receive a random "handle" $h$ in return, and b) to make "legal" *zero-test queries* on these encodings: a zero-test query is a formal polynomial $p$ on variables $\boldsymbol{h}$, which evaluates to `true` if and only if

[6] That is, whose circuit size is bounded by $T(n) = poly(2^{n^\alpha})$ for any $0 < \alpha < 1$.

[7] That is, security holds against all circuits whose size is bounded by $T(n) = poly(2^{n^\alpha})$ for any $0 < \alpha < 1$.

$p(\boldsymbol{v}) = 0$, where for every $i$, $v_i$ is the value encoded under handle $h_i$. The legality of a query is determined by a *legality-predicate* $g$: $g(p, \boldsymbol{l})$ outputs 1 if the query is deemed legal, where $\boldsymbol{l}$ are the labels corresponding to the handles $\boldsymbol{h}$. In this work we consider a natural class of "well-formed" legality predicates, which, as we shall discuss shortly, generalize all previously used notions of legality.

**Definition 2 (Well-formed legality predicate)** *Given a set of multi-sets (legal label sets) $S$ define the predicate $g_S(p, \boldsymbol{l}) = 1$ if and only if for every monomial $x_{j_1} \cdots x_{j_d}$ of $p$, it holds that the multi-set $\{l_{j_1}, \ldots, l_{j_d}\} \in S$. We say that a legality predicate $g$ is* well-formed *if there exists a set $S$ such that $g = g_S$.*

For instance, to capture:

- idealized groups [Sho97] (where we do not allow any multiplications), consider the predicate $g_S$ corresponding to the set $S = \{\{1\}\}$ (and requiring that all encodings are made under the label 1).
- "simple" $d$-level graded encodings of [GGH13a], consider the predicate $g_S$ corresponding to the set $S$ where $\{l_{j_1}, \ldots, l_{j_m}\} \in S$ if and only if $\sum_{i\in[m]} l_{j_i} = d$ (and requiring that all encodings are made under a label $l \in [d]$ that represents the element's "level").
- "set-based" $d$-level graded encodings [GGH13a,BR14,BGK+14], consider the predicate $g_S$ corresponding to the set $S$ where $\{l_{j_1}, \ldots, l_{j_m}\} \in S$ if and only if the disjoint union of labels $l_{j_i}$ where $i \in [m]$ is the set $\{1, 2, \ldots, d\}$, i.e. $\sqcup_{i\in[d]} l_{j_i} = [d]$ (and requiring that all encodings are made under a label $l$ that is a subset of $[d]$).

Additionally, to capture secret-key encodings in which only the obfuscator can create new encodings, we follow [BGK+14] and require that encodings can only occur once upon initialization; after initialization no more encodings can be performed. (In contrast to [BGK+14], however, these encodings can be performed adaptively.)

**Definition 3 (Ideal graded encoding oracle)** *The oracle $M_q^g = (\mathsf{enc}, \mathsf{zero})$ is a stateful oracle, parameterized by integer $q$ and a legality predicate $g$, that responds to queries in the following manner:*

1. *Upon initialization* and only then*, the activator may* adaptively *make any number of queries of the form* $\mathsf{enc}(v, l)$*; for each such query, $M_q^g$ picks a uniformly random "handle" $h \in \{0, 1\}^{3|q|}$, stores the tuple $(v, l, h)$ in a list $\mathcal{L}_O$ and returns $h$.*[8] *This initialization phase ends if any algorithm other than the activating algorithm makes any query to $M^q$, or if the activator makes a non enc query. Any subsequent enc$(\cdot, \cdot)$ queries will be answered with $\perp$.*

[8] In particular, even if the same value $v$ is encoded twice (under the same label), independently random handles are returned for the two encodings. This model thus considers *randomized* graded encodings. Our results also apply to deterministic randomized encodings where the oracle keeps state also during the encoding phase and always returns the *same* handle for an encoding of the value $v$ under the label $l$.

2. *On input query* zero$(p)$ *where p is a formal polynomial over variables* $h_1, \dots, h_m$, *each of which is represented as a string of length* $3|q|$ *(corresponding to some handle),* $M_q^g$ *does the following:*
   (a) *For each* $i \in [m]$, *retrieve a tuple* $(v_i, l_i, h_i)$ *from the state* $\mathcal{L}_O$*; if no such tuple exists, it returns* `false`.
   (b) *(Illegal query) If all tuples are retrieved, return* `false` *if* $g(p, \boldsymbol{l}) \neq 1$
   (c) *(Zero test) Finally, return* `true` *iff* $p(v_1, \dots, v_n) = 0 \mod q$, *and* `false` *otherwise.*
3. *(All other queries are answered with* $\perp$*).*

*We say that M is an* ideal graded encoding oracle *if* $M = \{M_{q_1}^{g_1}, M_{q_2}^{g_2}, \dots\}$, *and for every* $n \in \mathbb{N}$, $q_n$ *is a prime,* $|q_n| > n$ *and* $g_n$ *is a well-formed legality predicate. Finally, we say that M is a degree-*$d(\cdot)$ *ideal graded encoding oracle if for all* $n \in \mathbb{N}$, $g_n(p, \boldsymbol{l})$ *returns* `false` *when* $\mathsf{deg}(p) > d(n)$.

*A Remark on the Model* Following [PST14], for simplicity of notation, we do not directly allow players to create new encodings by adding and multiplying old ones as in the definitions of [BR14,BGK+14]. This restriction is without loss of generality since a) an obfuscator "knows" all values it has previously encoded (since it needs to explicitly provide them to the encoding oracle) so instead of operating on old encodings, it can simply operate on the actual values and simply create a new encoding of the resulting value[9], and b) when evaluating the obfuscated code, operations on encodings can be simulated by "bogus" independently random handles[10], and emulating zero-test queries by appropriately modifying the zero-test polynomial $p$ to take into account the previously performed operations.

*Feasibility of VBB obfuscation in idealized graded encoding models* The results of [BR14,BGK+14] demonstrate feasibility of VBB obfuscation in idealized "set-based" graded encoding models that allow zero-test queries with *super-constant* degree.

**Theorem 4 ( [BR14,BGK+14])** *Under the LWE assumption*[11], *for every polynomial* $p(\cdot)$, *there exists a (polynomial-time computable) sequence of well-formed legality predicates* $g_1, g_2, \dots$, *such that for any ideal graded encoding oracle* $M =$

[9] To make this argument it is important that we allow *adaptive* encodings during the initialization phase, as opposed to a single non-adaptive encoding query as in the definition of [BGK+14].

[10] For this emulation with "bogus" random handles to work, it is important that we consider a model of *randomized* graded encodings (where multiple encodings of the same value are given fresh random handles). In case the encoding is deterministic (and thus encodings of the same value need to be given the same handle) the simulation fails: if the result of the operation yields a value that was previously encoded we should output that handle instead. Nevertheless, as we point out at the end of Section 3, our results extend to deal also with deterministic encodings where players can perform operations on the encodings.

[11] [BGK+14] present unconditionally secure obfuscators for $NC1$; the LWE assumption is needed to bootstrap up to polynomial-size circuits.

$\{M_{q_1}^{g_1}, M_{q_2}^{g_2}, \ldots\}$, *there exists a polynomial-size obfuscator* $O$[12] *such that* $O$ *is a VBB obfuscator for the class of* $p(\cdot)$*-sized circuits in the* $M$ *model.*

Their construction also satisfies subexponential VBB security assuming an appropriate subexponential strengthening of LWE.

## 3 Impossibility of VBB Obfuscation

**Theorem 5** *Assuming the existence of trapdoor permutations, there exists a family of functions* $F$ *such that for every constant* $d$ *and every degree-*$d$ *ideal graded encoding oracle* $M$*, a polynomial-size* $0.9$*-approximate VBB obfuscator for* $F$ *does not exist in the* $M$ *oracle model.*

We briefly review the approach of [CKP15] as we will follow the same high-level structure. Their first step is to show that any VBB obfuscator in the Random Oracle model can be transformed into an approximate VBB obfuscator in the plain. They next rely on Theorem 2 to conclude their impossibility result. The first step is achieved by running the original VBB obfuscator in the Random Oracle model by simulating all random oracle queries (with truly random answers). Additionally, to ensure consistency between answers to queries in the obfuscation phase and answers in the execution of the obfuscated code, the obfuscator performs a learning phase in which most *heavy* oracle queries (i.e. oracle queries that are made with high probability when running the obfuscated code on random inputs) are discovered; the answers to the heavy queries are hard-coded into the obfuscated code. This ensures that when the obfuscated code is run on a random input, except with inverse polynomial probability (proportional to the number of random inputs used in the learning phase), the obfuscated code will not make any random oracles queries that were not made during the learning phase (i.e. that are not hard-coded), and as a consequence, the obfuscated code correctly computes the function with high probability. Furthermore, the only difference between the the new (plain-model) obfuscator and the original (random-oracle-model) obfuscator is that the former leaks the set of heavy queries; since this leak is something that can be learned by running the obfuscated code of the random-oracle-model obfuscator, VBB security ensures that the same heavy set can be simulated using only black-box access to the function.

As mentioned, we follow the same high-level approach. Our main result (Lemma 6 below) shows how to transform any VBB obfuscator in the constant-degree graded encoding model into an approximate VBB obfuscator in the plain model. The proof of Theorem 5 is then concluded by applying Theorem 2. Just as [CKP15], we run the original (graded-encoding-model) obfuscator and simulate its oracle queries. But it no longer suffices to simply learn all the heavy queries: the obfuscated code may only ask "light" queries (i.e., each query has negligible probability) yet the answer to those queries are correlated

[12] The only non-uniform advice needed is the prime $q_n$.

(in fact, even determined by) the queries made during the obfuscation phase. For instance, assume that the obfuscator encodes two elements $v_1$ and $v_2$, and later the evaluator makes a zero-test query of the form $p(v_1, v_2) = av_1 + bv_2$ where $a$ and $b$ are chosen from some distribution with high min-entropy.

Rather, we show that by running the obfuscated code on sufficiently many random inputs and honestly emulating answers to oracle queries, we can recover a set of linearly independent polynomials in the values $v_1, \ldots, v_\ell$ encoded during the obfuscation phase such that, except with inverse polynomial probability, when the obfuscated code is run on a random input, every zero-test query can be correctly emulated by simply determining whether the zero-test polynomial is a linear combination of polynomials in the stored set. Since the oracle is restricted to answering constant-degree $d$ polynomials, there can be at most $(\ell + 1)^d$ monomials in the values $v_1, \ldots, v_\ell$, and thus at most $(\ell + 1)^d$ linearly independent polynomials in those values. If we record all zero-test polynomials that evaluate to zero, then after sufficiently many samples, we have either recovered the full basis (which allows one to correctly answer all remaining zero-test queries), or it is unlikely that a new sample will add another linearly independent polynomial, which in turn means that when the obfuscated code is run on a random input, our emulation only fails with small probability. We finally observe that, just as in [CKP15], leaking the set of linearly independent polynomials does not challenge VBB security because this set (just as the set of heavy random oracle queries in the case of [CKP15]) can be learned from just observing the obfuscated code and can thus be simulated.

We now turn to state and formally prove our main lemma, which combined with Theorem 2 directly concludes our main result (i.e, Theorem 5).

**Lemma 6 (Main)** *For every constant $d$ and every degree-$d$ ideal graded encoding oracle $M$, if a family of functions $F$ indexed by $k$ has a polynomial-size $\epsilon(|k|)$-approximate VBB obfuscator in the $M$ oracle model, then there exists a polynomial-size $(\epsilon(|k|) + 1/|k|)$-approximate[13] VBB obfuscator for $F$ in the plain model.*

*Proof.* Let $M = \{M^{g_1}_{q_1}, M^{g_2}_{q_2}, \ldots\}$ be a degree-$d$ ideal graded encoding oracle for some constant $d$. Let $\mathcal{O}$ be an $\epsilon$-approximate obfuscator for family $F$ in the $M$ oracle model that requests encodings of at most $\ell(|k|)$ elements where $k$ is the index for family $F$; we assume without loss of generality that $\ell(n) \geq 1$. We construct a (non-uniform[14]) polynomial-size $(\epsilon(n) + 1/n)$-approximate VBB obfuscator $\mathcal{O}'$ for $F$ in the plain model below.

*New obfuscator $\mathcal{O}'(k)$:*

[13] $1/|k|$ can be replaced by any inverse polynomial by appropriately adjusting the parameters in our proof.

[14] The non-uniformity in our construction is to encode the sequence of primes $q_1, q_2, \ldots$ that is implicit in the oracle $M^g$. If we model the oracle $M$ with a uniform algorithm that picks the field for each security parameter, then our construction below can also be uniform.

1. On input $k$, run $\mathcal{O}(k)$ and simulate the queries to $M^{g_k}_{q_{|k|}}$ (i.e., answer the initial enc queries by creating a list $\mathcal{L}_O$ of encoded elements as in the definition of $M^{g_k}_{q_{|k|}}$, and answer $\mathsf{zero}(p)$ queries by evaluating the polynomial $p$ on the "decoded" elements) to compute the obfuscated program $C_k$.
2. If $\mathcal{O}(k)$ did not make any initial encoding queries, simply modify the code of $C_k$ to honestly emulate the $M$ oracle with some hard-coded uniformly chosen randomness (to generate handles), output this modified code, and halt.
3. Otherwise, set $\mathcal{L}_c$ to empty.
4. *Repeat* until there have been $L = (\ell(|k|) + 1)^d |k|$ iterations without any new additions to $\mathcal{L}_c$:
   (a) Sample random input $x^j$.
   (b) Run $C_k(x^j)$ while simulating zero-test queries to $M$ using the list of encoded elements $\mathcal{L}_O$ from Step 1.
   (c) Additionally, whenever a zero-test query $\mathsf{zero}(p)$ evaluates to `true`, record the formal polynomial $p$ if it is linearly independent with all previously stored polynomials in $\mathcal{L}_c$. Testing whether $p$ is a linear combination of polynomials in $\mathcal{L}_c$ can be performed efficiently through Gaussian elimination (by viewing each monomial as a separate variable).
5. Output a new circuit $C'_k$ that does the following:
   (a) On input $y$, run $C_k(y)$.
   (b) If $C_k(y)$ makes a $\mathsf{zero}(p)$ query to $M$, answer `true` if $p$ is a linear combination of the polynomials in $\mathcal{L}_c$ and otherwise answer `false`.

*Claim.* Obfuscator $\mathcal{O}'$ runs in (non-uniform) polynomial time.

*Proof.* Recall that $\ell(|k|)$ is an upper bound on the number of encodings. As a consequence, there are at most $(\ell(|k|) + 1)^d$ degree-$d$ monomials in the encodings; thus, there can be at most $(\ell(|k|) + 1)^d$ linearly independent zero-test polynomials. Since $\mathcal{O}$ continues iterating until there have been $L$ consecutive iterations with no new additions to $\mathcal{L}_c$, it follows that there can be at most $L \cdot (\ell(|k|) + 1)^d$ iterations, each of which can be implemented in polynomial time.

**Proposition 1.** *The obfuscator $\mathcal{O}'$ is $(\epsilon(n) + 1/n)$-approximately correct.*

*Proof.* Consider a hybrid obfuscator $\tilde{\mathcal{O}}'$ that proceeds just as $\mathcal{O}'$ except that it *always* outputs a program $\tilde{C}'_k$ that *honestly* simulates the $M^g$ oracle using the state $\mathcal{L}_O$ from Stage 1.

Let $\mathsf{Exp}_k$ denote the experiment that consists of running $C_k \leftarrow \mathcal{O}(k)$, picking a uniformly random input $x^* \leftarrow \{0,1\}^{|k|}$, and outputting 1 iff $C_k(x^*) = F_k(x^*)$ (and 0 otherwise). Define $\mathsf{Exp}'_k$ and $\widetilde{\mathsf{Exp}}'_k$ in exactly the same way but using $\mathcal{O}'$ and $\tilde{\mathcal{O}}'$ respectively.

Since $\mathcal{O}$ is $\epsilon(n)$-approximately correct, for every $k \in \{0,1\}^n$, we have

$$\Pr[\mathsf{Exp}_k = 0] \leq \epsilon(n)$$

We also observe that by construction, for every $k \in \{0,1\}^n$,

$$\Pr[\mathsf{Exp}_k = 0] = \Pr[\widetilde{\mathsf{Exp}'}_k = 0]$$

This directly follows from the observation that the only difference between these experiments is that in $\widetilde{\mathsf{Exp}}$, the obfuscator hard-codes the randomness of $M^g$ (needed to generate handles) in the obfuscated code in the event that $\mathcal{O}$ did not make any initial encoding queries. But since in the experiment we only evaluate the obfuscated code on a single input, the outputs of the experiments are identically distributed.

Our goal is now to prove that for $k \in \{0,1\}^n$,

$$\Pr[\mathsf{Exp}'_k = 0] \leq \Pr[\widetilde{\mathsf{Exp}'}_k = 0] + 1/n$$

which concludes the proof of the proposition.

Note that there is only one difference between the program $\tilde{C}'_k$ produced by $\tilde{\mathcal{O}}'$ and the program $C'_k$ produced by $\mathcal{O}'$ when run on the input $x^*$ in the above experiments:

– $C'_k(x^*)$ may make a zero-test query $\mathsf{zero}(p, \boldsymbol{h})$ that should evaluate to `true`, but $p$ is not in the span of $\mathcal{L}_c$ (and thus $C'_k$ emulates the answer as `false`, whereas $\tilde{C}_k$ honestly emulates the answer as `true`.) Let $\mathsf{bad}^i$ denote the event that this happens for the first time when $|\mathcal{L}_c| = i$.

Let us note that $C'_k(x^*)$ can never err in the other direction; that is, it never answers a zero-test query as `true` when the answer in fact should be `false`. This follows from the fact that if $p$ is in the span of $\mathcal{L}_c$, then a) all input handles to $p$ correspond to some encoding, and $p$ necessarily evaluates to zero given the encoded value corresponding to those handles, and b) by the wellformedness condition of $g$, $g(p, \boldsymbol{l})$ necessarily evaluates to `true` (as $p$ cannot use any monomials not already in use by the polynomials in $\mathcal{L}_c$).

It follows by construction that conditioned on $\mathsf{bad}^i$ not happening for any $i$, experiments $\mathsf{Exp}'_k$ and $\widetilde{\mathsf{Exp}'}_k$ proceed identically.

The proof is concluded by the following two claims which show that the probability of any bad event is small. In the following we focus on experiment $\mathsf{Exp}'$ but the same arguments straighforwardly hold for $\widetilde{\mathsf{Exp}}'$.

*Claim.* For every $i$, $\Pr[\mathsf{bad}^i] \leq 1/L$.

*Proof.* For every *bad* random tape for the experiment that induces event $\mathsf{bad}^i$, we identify at least $L$ unique *good* random tapes obtained by swapping the final run on input $x^*$ with one of the (at least $L$) sampled iterations (using $x_i$); furthermore, we show that any two distinct bad executions lead to disjoint sets of good executions. We conclude the claim based on the fact that the fraction of bad tapes is at most $1/L$ and each random tape is equally likely.

Let us now formally specify the mapping $\Phi$ from bad tapes to good tapes, and specify an *inverse mapping* $\Phi^{-1}$ that given a good tape in the range of $\Phi$

recovers the bad tape it was generated from. The existence of such an inverse map shows that any two distinct bad tapes lead to distinct sets of good tapes, as desired.

Recall that by the proof of Claim 3, $m = L \cdot (\ell(n)+1)^d$ is a bound on the number of iterations in step 4. We define a random tape for the experiment $\mathsf{Exp}_k$ as $(\rho, x_1, \ldots, x_m, x^*)$ where $(x_1, \ldots, x_m)$ are the inputs sampled to be used in step (4a) of $\mathcal{O}'$ (note that not all of those samples may be used), $x^*$ is the final input chosen in the experiment, and $\rho$ is the remaining randomness (i.e., the randomness of underlying $\mathcal{O}$ and randomness of $\mathcal{O}'$ in the event that $\mathcal{O}$ did not make the initial encoding queries). Let $q(R)$ denote the number of samples made in step 3 given the random tape $R$; by construction $L \leq q(R) \leq m$.

We say that a random tape $R = (\rho, x_1, \ldots, x_m, x^*)$ is *bad* if $\mathsf{Exp}'_k(R)$ induces event $\mathsf{bad}^i$; that is, a) in the evaluations of $C'_k(x_j)$ for $j \in [q(R) - L, \ldots, q(R)]$, there are no linearly independent zero-test polynomials that evaluate to 0, b) the evaluation of $C'_k(x^*)$ leads to such a linearly independent polynomial that evaluates to 0, and c) the size of $\mathcal{L}_c$ is $i$.

Define the mapping $\Phi(R)$ as the set of $L$ random tapes $\Phi(R) = \{R_j\}_{j \in [L]}$ where $R_j$ is constructed by swapping the $t^{\text{th}}$ random sample $x_t$, where $t = (q(R) - j + 1)$, with the last sample $x^*$ as follows:

$$R_j = (\rho, x_1, \ldots, x_{t-1}, x^*, x_{t+1}, \ldots, x_m, x_j)$$

Note that $\mathsf{Exp}'_k(R_j)$ does not induce $\mathsf{bad}^i$ since the experiment finds at least $i+1$ linearly independent polynomials that evaluate to zero (and thus $\mathcal{L}_c > i$).

Finally, let $\Phi^{-1}(\cdot)$ be an inverse map that on input a tape $R$, swaps the last sample in the tape with the first sample $x_t$ that leads to $i+1$ linearly independent polynomials in the set $\mathcal{L}_c$ (and if no such $x_t$ exists simply outputs $R$). It follows directly by construction that for every bad $R$, $\Phi^{-1}(\Phi(R)) = R$. (Note that in our definition of the inverse map, we make use of the fact that the event $\mathsf{bad}^i$ is parameterized by $i$.)

By a union bound, it follows from Claim 3 that,

$$\Pr\left[\exists i \text{ s.t. } \mathsf{bad}^i\right] = \Pr\left[\mathsf{bad}^1 \vee \cdots \vee \mathsf{bad}^{\ell'(n)^d}\right] \leq \frac{\ell'(n)^d}{L} = \frac{\ell'(n)^d}{\ell'(n)^d n} = 1/n$$

where $\ell'(n) = \ell(n) + 1$ since as noted in the proof of Claim 3, the maximum size of $\mathcal{L}_c$ is $\ell'(n)^d = (\ell(n)+1)^d$. This concludes that $\mathcal{O}'$ is $\epsilon(n) + 1/n$ approximately correct.

**Proposition 2.** *Obfuscator $\mathcal{O}'$ satisfies the virtual-black box property.*

*Proof.* This proof is essentially identical to the one given in [CKP15] for a similar statement. We include it here to be self-contained. Fix an index $k$. Given an adversary $A'$ for the new obfuscator $\mathcal{O}'$, we construct a new adversary $A^M$ for the $\mathcal{O}^M$ obfuscator as follows. The new adversary $A^M(C_k)$, on input a circuit $C_k$ produced by the obfuscation $\mathcal{O}^M$ algorithm, simulates steps 2,3, and 4 of the $\mathcal{O}'$ algorithm by answering all queries using its oracle $M$ (whose answers will be consistent with the oracle used by $\mathcal{O}^M$ to produce $C_k$).[15] At the end of this simulation, $A$ thus produces a circuit $C'_k$ with exactly the same distribution as the output of $\mathcal{O}'$. Adversary $A^M$ then runs $A'(C'_k)$ (which does not make any oracle queries) and returns the same output. It therefore follows by construction that

$$\Pr\left[A^{M_{|k|}}(\mathcal{O}^{M_{|k|}}(k)) = 1\right] = \Pr\left[A'(\mathcal{O}'(k)) = 1\right]$$

By the approximate VBB security property of $\mathcal{O}$ for family $F$, it follows that there exists a simulator $S$ and a negligible function $\mu$ such that

$$\Pr\left[A^{M_{|k|}}(\mathcal{O}^{M_{|k|}}(k)) = 1\right] - \Pr\left[S^{F_k}(|k|) = 1\right] \leq \mu(|k|)$$

which immediately implies that

$$\Pr\left[A'(\mathcal{O}'(k)) = 1\right] - \Pr\left[S^{F_k}(|k|) = 1\right] \leq \mu(|k|)$$

and concludes the proposition since $S$ is also a good simulator for $A'$.

We conclude that $\mathcal{O}'$ is a secure $\epsilon(n) + 1/n$-approximate VBB obfuscator for $F$ (in the plain model). This finishes the proof of Lemma 6.

*Remark (extension to "sparse" high-degree zero-test polynomials)* This proof uses the constant-degree restriction on the zero-test queries to argue that the number of monomials in encoded values is polynomial. The theorem thus extends to high-degree polynomials as long as the legality predicate restricts these polynomials to be "sparse" in the sense that the *total* number of monomials over which any legal zero-test query is formed must be (a-priori) polynomially bounded. Note that it does not suffice to require that each zero-test query has a small number of monomials. Rather, we require that there exists a small set of monomials that suffices to represent *all* legal zero-test queries.

*Remark (extension to "multi-slot" graded encodings)* Our result directly extend to "multi-slot" graded encodings (as in [AB15]), which are a model of composite-order graded encodings. In this model, an encoding is a vector of elements; operations on elements are performed component-wise and finally a zero-test can be performed which determines whether the whole vector is 0. Our proof directly extends also to this setting (by simply viewing each component as a separate variable).

[15] Step 2 (i.e., checking whether the initial encoding queries have been made) can be simulated by making a "dummy" $enc(0,0)$ query and checking whether $M$ returns $\perp$.

*Remark (extension to deterministic encodings)* Our graded encoding oracle models an idealized *randomized* graded encodings scheme: even if the same value $v$ is encoded twice (under the same label), we get independently random handles for the two encodings. Our proof, however, works in exactly the same way also for *deterministic* randomized encodings, where the oracle keeps state also during the encoding phase and always returns the *same* handle for an encoding of the value $v$ under the label $l$. This trivially follows since our oracle does not allow players to perform any operations on encodings but simply zero-test queries. As previously mentioned, for the case of randomized graded encodings, it is without loss of generality since operations on encodings can be simulated by "bogus" independently random handles. For the case of deterministic encodings, however, this simulation no longer works: if the result of the operation yields a value that was previously encoded we should output that handle instead. But for the purpose of our proof, we can make the simulation work: Modify the learning phase to keep track of also all handles $h$ "seen", adding them to $\mathcal{L}_c$; additionally, for every operation, make a zero-test query to check whether the value to be encoded after the operation equals the value encoded under any previously stored handle. Next, during the evaluation of the obfuscated code, emulate operations on encodings by first checking (using a zero-test query, which is emulated as before) whether the value to be encoded after the operation equals the value encoded under any previously stored handle, and, if so, outputting this handle, and otherwise outputting a random handle. It follows using the same argument as above that this emulation only fails with inverse polynomial probability.

*Remark (extension to rings)* We note that our proof directly generalizes to any graded encoding scheme that operates on elements in a ring (as opposed to $\mathbb{F}_p$) as long as a) there exists an efficient method for determining the row-rank of a matrix of this ring, and) the row-rank of a matrix is polynomially bounded by the column-rank. Property a) is needed to test whether we get a linearly independent polynomial (we used Gaussian elimination for the case of $\mathbb{F}_p$) , and property b) is needed to ensure that the maximum number of linearly independent polynomials is polynomially bounded by the number of monomials (for the case of $\mathbb{F}_p$ row-rank equals column-rank, and thus the number of linearly independent polynomials is bounded by the number of monomials).

## 4 Impossibility of Subexponential VBB security

We now consider sub-exponential VBB security and rule out constructions that use $n^{\alpha}$-degree zero-test queries for any $0 < \alpha < 1$.

**Theorem 7** *Assuming the existence of exponentially-secure trapdoor permutations, there exists a family of polynomial-time computable functions $F$ such that for every $0 < \alpha < 1$, every degree-$n^{\alpha}$ ideal graded encoding oracle $M$, a polynomial-size* 0.9-*approximate VBB obfuscator for $F$ does not exist in the $M$ oracle model.*

Recall that, in contrast, Barak *et al*. [BGK+14] show that for every family $F$ of polynomial-time functions, subexponentially-secure VBB obfuscation is possible using $p(n)$-degree ideal graded encodings where $p$ is a polynomial (under appropriate cryptographic hardness assumptions).

We follow the proof of Theorem 5 and prove the following lemma which combined with Theorem 3 proves the theorem.

**Lemma 8** *For every $\alpha < 1$ and degree-$n^\alpha$ ideal graded encoding oracle $M$, if a family of functions $F$ indexed by $k$ has a polynomial-size $\epsilon(n)$-approximate subexponentially-secure VBB obfuscator in the $M$ oracle model, then there exists a subexponential-size $(\epsilon(n) + 1/n)$-approximate subexponentially-secure VBB obfuscator for $F$ in the plain model.*

*Proof.* (Sketch) The construction is identical to the one in the proof of Lemma 6, except that we set $d = n^\alpha$ (instead of it being a constant), where $n = |k|$.

By the same proof, the size of the new (plain-model) obfuscator is polynomial in $\ell(n)^{n^\alpha} = 2^{n^\alpha \log \ell(n)}$, where $\ell(n)$ is a bound on the number of encoding queries made by the original obfuscator. It follows that the size of the obfuscator is subexponential.

Approximate correctness follows as per the proof of Lemma 6. Finally, subexponential VBB simulation follows in exactly the same way as Lemma 6 by appealing to subexponential VBB security of the original VBB obfuscator.

## References


[AB15] Benny Applebaum and Zvika Brakerski. Obfuscating circuits via composite-order graded encoding. In *TCC'2015*, pages 528–556, 2015.

[BGI+01] Boaz Barak, Oded Goldreich, Rusell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *Advances in Cryptology CRYPTO 2001*, pages 1–18. Springer, 2001.

[BGK+14] Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, , and Amit Sahai. Protecting obfuscation against algebraic attacks. In *EUROCRYPT'14*, 2014.

[BP13] Nir Bitansky and Omer Paneth. On the impossibility of approximate obfuscation and applications to resettable cryptography. In *STOC'13*, 2013.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, Virginia, USA, November 3-5, 1993.*, pages 62–73, 1993.

[BR14] Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In *TCC'14*, 2014.

[BS03] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *Contemporary Mathematics*, 324(1):71–90, 2003.

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

[CKP15] Ran Canetti, Yael Tauman Kalai, and Omer Paneth. On obfuscation with random oracles. In *TCC'15*, 2015.

[CV13] Ran Canetti and Vinod Vaikuntanathan. Obfuscating branching programs using black-box pseudo-free groups. Cryptology ePrint Archive, Report 2013/500, 2013. https://eprint.iacr.org/2013/500.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *Advances in Cryptology–EUROCRYPT 2013*, pages 1–17. Springer, 2013.

[GGH+13b] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *Proc. of FOCS 2013*, 2013.

[Had00] Satoshi Hada. Zero-knowledge and code obfuscation. In *Advances in Cryptology–ASIACRYPT 2000*, pages 443–457. Springer, 2000.

[LPS04] Ben Lynn, Manoj Prabhakaran, and Amit Sahai. Positive results and techniques for obfuscation. In *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, pages 20–39, 2004.

[MMN15] Mohammad Mahmoody, Ameer Mohammed, and Soheil Nematihaji. More on impossibility of virtual black-box obfuscation in idealized models. In *To appear in TCC'16*, 2015.

[PST14] Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In *CRYPTO'14*, 2014.

[Rot13] Ron D Rothblum. On the circular security of bit-encryption. In *Theory of Cryptography*, pages 579–598. Springer, 2013.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding*, pages 256–266, 1997.