A Simple Obfuscation Scheme for Pattern-Matching with Wildcards

Allison Bishop[1,2(✉)], Lucas Kowalczyk[2], Tal Malkin[2], Valerio Pastro[2,3], Mariana Raykova[3], and Kevin Shi[2]

[1] IEX, New York, USA
[2] Columbia University, New York, USA
{allison,luke,tal,valerio,kshi}@cs.columbia.edu
[3] Yale University, New Haven, USA
mariana.raykova@yale.edu

Abstract. We give a simple and efficient method for obfuscating pattern matching with wildcards. In other words, we construct a way to check an input against a secret pattern, which is described in terms of prescribed values interspersed with unconstrained "wildcard" slots. As long as the support of the pattern is sufficiently sparse and the pattern itself is chosen from an appropriate distribution, we prove that a polynomial-time adversary cannot find a matching input, except with negligible probability. We rely upon the generic group heuristic (in a regular group, with no multilinearity). Previous work [9,10,32] provided less efficient constructions based on multilinear maps or LWE.

1 Introduction

The discipline of cryptography is fundamentally about the separation of seemingly intertwined information and abilities: how do we separate the ability the compute a function from the ability to invert a function? How do we separate the ability to encrypt from the ability to decrypt? How do we separate partial knowledge of a key through a side-channel attack from the ability to compromise a cryptographic scheme? The study of cryptographic obfuscation is born from the question: how do we separate the ability to run code from the ability to read code? Since the seminal work of [7] that placed this question firmly on a rigorous theoretical foundation, it has been clear that this kind of separation would be powerful, both inside and outside the typical reach of the discipline of cryptography.

If we can hide secrets inside functioning software, we can protect cryptographic keys, and many of cryptography's disparate and hard won achievements follow as a consequence. We can also protect intellectual property, and the inner workings of critical code like software patches, which in their unprotected form might leak information that could be used to attack remaining vulnerable

V. Pastro—Work done while the author was a postdoc at Columbia University and Yale University.

H. Shacham and A. Boldyreva (Eds.): CRYPTO 2018, LNCS 10993, pp. 731–752, 2018.
https://doi.org/10.1007/978-3-319-96878-0_25

machines. But as with any cryptographic primitive, the suitability of program obfuscation for any particular task depends on three main axes by which we must evaluate proposed constructions: (1) efficiency, (2) the underlying computational and architectural assumptions, and (3) the derived security guarantees.

Two possibilities for (3), defined in [7], are the notion of virtual black box obfuscation (VBB) and the notion of indistinguishability obfuscation (IO). Virtual black box obfuscation is a very powerful and intuitive notion, which requires that anything that can be done by an attacker in possession of the obfuscated code can also be done by a simulator who can only run the software in a "black box", with no access to intermediary values or other properties of the computation between input ingestion and output production. This notion would be suitable for virtually[1] all possible applications of obfuscation, but it is shown in [7] that it is impossible to achieve for general functionalities. The notion of IO requires something weaker, merely that an attacker in possession of two different obfuscations of the *same* functionality cannot tell them apart. In other words, we only enforce indistinguishability for program descriptions that may differ internally but whose external input/output behavior is *identical*.

At the time of its introduction by [7], IO was neither shown to be impossible, nor shown to be particularly useful. Progress instead was made for VBB obfuscation of very basic functionalities, such as point obfuscation [27,31] and hyperplane membership [12], which lie below the reach of the impossibility result for VBB. But following the unprecedented construction of cryptographic multilinear maps in [17], two breakthroughs occurred in quick succession. A first candidate construction for indistinguishability obfuscation of general functions was proposed in [18], and the flexible technique of "punctured programming" was developed for deriving meaningful cryptographic results from the IO security guarantee [30].

Since then, the cryptographic research community has been riding out wave of positive and negative results: increasingly powerful constructions employing idealized models on multilinear maps or new, complex assumptions [2,4–6,18–20,23–26,33], attacks on the underlying multilinear maps [3,13–16,28], and a steady stream of works deriving applications and consequences from various forms of obfuscation(e.g., [1,21,22], and many more).

Our work is focused on the goal of obfuscating a modest but well-motivated functionality, one that does not require the use of multilinear maps, and hence does not inherit the risks of their still volatile security assumptions or the inefficiency that currently comes with using such a general-purpose tool. We consider the problem of *pattern matching with wildcards:* suppose there is an input binary string S of length n, and a pattern specification P also of length n, where for each bit P either dictates a particular bit value, or has a wildcard $*$, indicating that either value is allowed. For example, with $n = 5$, a pattern P would look like: $00 * 11$, and there would be two "matching" input strings S in this case, 00011 and 00111. The function we will obfuscate is the final "yes" or "no" outcome: for each P, we define the associated function $f_P(S)$ that outputs 1 when S matches P and outputs 0 otherwise.

[1] Pun intended.

This kind of functionality might appear, for instance, in a context like software patching. If a pattern P represents a problematic type of user input, say, that needs to be filtered out, we can obfuscate this function f_P to reject bad inputs without unnecessarily revealing P in full and helping attackers learn how to design such bad inputs. If the input length n is reasonably long and the number of matches to the pattern is not too dense in the space of inputs, we can hope that an attacker who queries a polynomial number of input strings will never manage to find a "bad" input that matches the pattern. We find these situations (where the adversary does not have enough information to identify the function being obfuscated) to be the most compelling subset of the standard VBB obfuscation security guarantee (as opposed to the subset involving simulating an adversary that already knows the function being obfuscated). Accordingly, we demonstrate that our construction satisfies a distributional security notion from [9,10,32]: if the pattern P is chosen from a suitable random distribution (and the number of wildcards $w \leq 0.75n$), then a PPT attacker will not be able to distinguish our obfuscation of f_P from an obfuscation of a function that always outputs 0.

Our construction uses only the basic tools of group operations and polynomial interpolation, and so is quite efficient. Our security analysis will be in the generic group model, for a regular cyclic group, with no multilinearity required. It remains an interesting open problem to obtain a security analysis in the standard model, using standard assumptions like DDH, for instance. [29] showed that the easier problem of bounded Hamming distance decoding is at least as hard as the DDH problem. While the result is not applicable to the obfuscation construction, the intermediary problem of finding nontrivial representations of the identity element first described by [11] is potentially applicable.

The functionality of pattern matching with wildcards has been previously obfuscated in [9,10]. These constructions rely on multiplicative encoding schemes that enable multiplication of the encoded values and also zero-testing, i.e. checking whether an encoded value is zero. Unlike multilinear maps, these encoding schemes do not need to have additive properties. This functionality has been realized either through the use of general multilinear maps [9] or through lattice-based encodings relying on a new instance dependent assumption called entropic LWE [10]. A recent work by Wichs and Zirdelis [32] provides an obfuscation construction for a more general high entropy class, called compute-and-compare functions, from LWE. This class includes our pattern matching with wildcards. We view our construction as a simple and highly efficient alternative to such an LWE-based construction, and this is in line with the long tradition of analogous functionalities being achieved in the discrete-logarithm and LWE regimes.

To keep our scheme as intuitive and as efficient as possible, we start from additive basics. Let's first consider a pattern P with no wildcards. In this case, our function f_P is just a point function, since there is only one input string that matches the fully prescriptive pattern. Here we can work over Z_p and choose uniformly random values $a_1, \ldots, a_{n-1} \in Z_p$ and set $a_n = -(a_1 + \cdots + a_{n-1})$. We can choose additional random values $r_1, \ldots, r_n \in Z_p$. Now our obfuscated program can be comprised of $2n$ elements of Z_p, which we will label as $x_{i,b}$ where

$i \in [n]$ and $b \in [0, 1]$. For each input bit position i, if the pattern value P is b, we set $x_{i,b} := a_i$ and $x_{i,1-b} = r_i$. To evaluate the obfuscated program on an input string, the evaluator simply selects the value corresponding to each input bit, and takes the sum modulo p. If it is 0, the output is 1. Otherwise the output is 0. Given these $2n$ values, if an attacker wants to find the pattern P, they are essentially trying to solve the subset sum problem (this is a slight variant since we have this kind of pair structure on the elements, but still the security intuition is the same).

Now if we want to introduce wildcards, it is clear we cannot simply give out a_i for both values for input bit i, since this will be noticed. The next thing we might try is to choose a random polynomial F of degree n over Z_p whose constant term is 0. Now we can set $x_{i,b} = F(2i + b)$ for positions that match the pattern, including both values of b in a wildcard position i. Our desired functionality can now be evaluated through polynomial interpolation. However, we quickly start to run into attacks based on list-decoding or regular decoding of Reed-Solomon codes, which can enable an attacker to recover the polynomial F once there are enough valid evaluations due to the wild cards.

A key observation at this point is that these decoding-style attacks rely upon non-linear functions of the given values, while the honest evaluation of the intended program needs only linear operations. This allows us to place the values $x_{i,b}$ in the exponent of a group $G = \langle g \rangle$ where discrete-log is difficult, and give out $g^{x_{i,b}}$ instead. This stops the decoding attacks without preventing honest evaluations. In the generic group model, the attacker is essentially limited to linear functions of the given exponents, so we can indeed formalize this intuition and obtain a security proof.

The hardness of noisy polynomial interpolation in the exponent was previously analyzed by [29], who gave a generic group argument concerning the problem of interpolating a polynomial with a slightly different error distribution. Our work follows a similar idea, but the specific wildcard structure we employ for our application creates some subtle differences, so we give a full argument here for completeness. We also provide a more rigorous exposition of the generic group proof argument.

It is an interesting problem to prove security for such a scheme without resorting to a generic group analysis. It seems that we should need a computational assumption like subset sum to assert that even though the group operations allow a discovery of the hidden structure, it is too sparse inside a combinatorially large space of possible input evaluations to be efficiently found. It also seems that we should need a computational assumption like DDH to explain exactly how the group blocks non-linear attacks. However, assumptions like DDH allow us to hide structure that is already non-linear, but requires us to preserve any structure that is linear, since linear structure on any small number of group elements can be discovered by brute force by an attacker. We could try to formulate some new assumption that is a strengthening of the subset sum assumption to the kind of intertwined linear structures that arise from polynomial evaluation, but this doesn't yet seem to yield insight beyond asserting security of the scheme itself.

We would ideally like to see a hybrid argument that combined simple subset-sum like steps with simple DDH-like steps, but designing such a reduction remains an intriguing challenge. Given that LWE-based approaches in the standard model are known, this represents a new test case on the boundary of the analogies we know between DDH-hard groups and the LWE setting. We expect that further study of this disconnect in proof technology between the LWE setting and the DDH setting may yield general insights into the inherent relationships (or lack thereof) between these different mathematical underpinnings.

2 Preliminaries

2.1 The Generic Group Model

We will prove the security of our construction against *generic adversaries*, which interact with group elements via the generic group model as defined in [8]. In this model, an adversary can only interact with the group via oracle calls to its group operation and zero test functionality. Group elements are represented by "handles", which are uniformly random strings long enough that the small probability of collision between handles representing different group elements can be ignored. A generic group operation oracle takes as input two group handles and returns a new handle representing the group element that is the result of the group operation on the two inputs (and is consistent with all handles previously used). Note that such an oracle can be efficiently simulated using a lookup table.

We use $\mathcal{G}$ to denote such a generic group operation oracle that answers adversary calls. $\mathcal{A}^{\mathcal{G}}$ will denote an adversary given access to this oracle and $\mathcal{O}^{\mathcal{G}}$ will denote the set of handles generated by $\mathcal{G}$ corresponding to the group elements in the construction $\mathcal{O}$.

2.2 Distributional Virtual Black-Box Obfuscation in the Generic Group Model

We will use a definition of *distributional virtual black-box (VBB) obfuscation in the generic group model* which is essentially the definition of [9], except using the generic group model instead of the random graded encoding model:

Definition 1 (Distributional VBB Obfuscator). *Let $\mathcal{C} = \{C_n\}_{n\in\mathbb{N}}$ be a family of polynomial-size circuits, where $\mathcal{C}_n$ is a set of boolean circuits operating on inputs of length n, and let $\mathcal{O}$ be a ppt algorithm which takes as input an input length $n \in \mathbb{N}$ and a circuit $C \in \mathcal{C}$ and outputs a boolean circuit $\mathcal{O}(C)$ (not necessarily in $\mathcal{C}$). Let $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ be an ensemble of distribution families $\mathcal{D}_n$ where each $D \in \mathcal{D}_n$ is a distribution over $\mathcal{C}_n$.*

$\mathcal{O}$ is a distributional VBB obfuscator for the distribution class $\mathcal{D}$ over the circuit family $\mathcal{C}$ if it has the following properties:

1. *Functionality-Preserving: For every $n \in \mathbb{N}$, $C \in \mathcal{C}_n$, and $\boldsymbol{x} \in \{0,1\}^n$, with all but $negl(n)$ probability over the coins of $\mathcal{O}$:*

$$(\mathcal{O}(C, 1^n)(\boldsymbol{x}) = C(\boldsymbol{x})$$

2. *Polynomial Slowdown: For every* $n \in \mathbb{N}$ *and* $C \in \mathcal{C}_n$*, the evaluation of* $\mathcal{O}(C, 1^n)$ *can be performed in time* $poly(|C|, n)$.
3. *Distributional Virtual Black-Box in Generic Group Model: For every polynomial (in* n*) time generic adversary* $\mathcal{A}$*, there exists a polynomial time simulator* $\mathcal{S}$*, such that for every* $n \in \mathbb{N}$*, every distribution* $D \in \mathcal{D}_n$ *(a distribution over* $\mathcal{C}_n$*, and every predicate* $P : \mathcal{C}_n \to \{0, 1\}$*:*

$$|\Pr_{C \leftarrow \mathcal{D}_n, \mathcal{G}, \mathcal{O}^{\mathcal{G}}, \mathcal{A}}[\mathcal{A}^{\mathcal{G}}(\mathcal{O}^{\mathcal{G}}(C, 1^n)) = P(C)] - \Pr_{C \leftarrow \mathcal{D}_n, \mathcal{S}}[\mathcal{S}^C(1^{|C|}, 1^n) = P(C)]| = negl(n)$$

Remark 1. As in [9], we remark that a stronger notion of functionality-preserving exists in the literature, where the obfuscated program must agree with $C(\boldsymbol{x})$ on all inputs $\boldsymbol{x}$ *simultaneously*. We use the relaxed requirement that for every input (individually), the obfuscated circuit is correct except for negligible probability. We also note that our construction can be modified to achieve the stronger property by using a group of sufficiently large size (2^{2n}) and the union bound over each of the 2^n inputs.

2.3 Schwartz-Zippel Lemma

A key step in our hybrid proof of security relies on the Schwartz-Zippel Lemma, which we will reproduce here:

Lemma 1. *Let* $\mathbb{Z}_p$ *be a finite field of size* p *and let* $P \in \mathbb{Z}_p[x_1, \ldots, x_n]$ *be a non-zero polynomial of degree* $\leq d$*. Let* $r_1, \ldots, r_n$ *be selected at random independently and uniformly from* $\mathbb{Z}_p$*. Then:* $\Pr[P(r_1, \ldots, r_n) = 0] \leq \frac{d}{p}$.

3 Obfuscating Pattern Matching with Wildcards

The class of functions for pattern matching with wildcards is parametrized by $(n, \boldsymbol{y}, \mathcal{W})$, where $\mathcal{W} \subset [n]$ is an index set and $f_{\boldsymbol{y}} : \{0,1\}^{n-|\mathcal{W}|} \longrightarrow \{0,1\}$ is a point function over $n - |\mathcal{W}|$ input variables that outputs 1 on the single input $\boldsymbol{y} \in \{0,1\}^{n-|\mathcal{W}|}$. The function $\Pi_{\mathcal{W}^c} : \{0,1\}^n \longrightarrow \{0,1\}^{n-|\mathcal{W}|}$ projects a boolean vector of length n onto only the entries not in the index set $\mathcal{W}$. $f_{\boldsymbol{y},\mathcal{W}}$, the function for pattern $\boldsymbol{y}$ with wildcard slots $\mathcal{W}$, is defined to be $f_{\boldsymbol{y},\mathcal{W}}(x) := f_{\boldsymbol{y}}(\Pi_{\mathcal{W}^c}(x))$. Our obfuscation scheme for the class of functions for pattern matching with wildcards is as follows:

Setup(n): sample $a_1, \cdots, a_{n-1} \sim \mathbb{Z}_p$ uniformly at random and construct the fixed polynomial $F(x) := a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$. Let G be a group with generator g of prime order $p > 2^n$.

Construction$(n, \boldsymbol{y}, \mathcal{W})$: the obfuscator outputs $2n$ elements arranged in a $2 \times n$ table of n columns corresponding to the n input variables with two entries each corresponding to the two possible boolean values of each input. For each slot h_{ij} where $(i, j) \in \{0,1\}^n \times \{0,1\}$, if either $i \in \mathcal{W}$ or $y_i = j$, then the obfuscator releases the element $h_{ij} = g^{F(2i+j)}$. Otherwise, the obfuscator releases h_{ij} as a uniformly random element of G.

Evaluation($\boldsymbol{x}$): to evaluate $f_{\boldsymbol{y},\mathcal{W}}(\boldsymbol{x})$, for each $i = 1, \cdots, n$, compute:

$$C_i := \prod_{j \neq i} \frac{-2j - x_j}{2i - x_i - x_j + 2j}$$

choose the elements h_{ix_i}, and compute:

$$T := \prod_{i=0}^{n-1} (h_{ix_i})^{C_i}$$

Output 1 if $T = g^0$ and 0 otherwise.

Functionality-Preserving: The fact that this obfuscation scheme is functionality-preserving follows from the fact that, if $\boldsymbol{x}$ is an accepting input of f ($f(\boldsymbol{x}) = 1$), then the chosen handles form n proper evaluations of the polynomial $F(x)$ on distinct elements. Further, the C_i scalars used in evaluation are Lagrange coefficients, making the evaluation a polynomial interpolation that returns $F(0) = 0$ in this case, causing $T = g^0$ and the evaluation to output 1 (with probability 1).

$$\begin{aligned}\prod_{i=0}^{n-1} (h_{ix_i})^{C_i} &= \prod_{i=0}^{n-1} g^{C_i F(2i+x_i)} \\ &= g^{\sum_{i=0}^{n-1} C_i F(2i+x_i)} \\ &= g^{F(0)} \\ &= g^0\end{aligned}$$

On the other hand, if even one input bit was not accepting (so $f(\boldsymbol{x}) = 0$), then at least one of the h_{ix_i}'s used in interpolation would be a uniformly random group element (not $g^{F(2i+j)}$). Thus, the evaluation product would be a product that includes a uniformly random group element raised to some power, which would result in $T = g^0$ with negligible probability $\frac{1}{p}$.

Polynomial Slowdown: Given a the set of $2n$ group elements, assuming group operations can be performed in $poly(n)$ time, the computation of C_i and T described in the **Evaluation** procedure can be performed in polynomial time.

Distributional Virtual Black-Box: We give a proof of our construction's distributional VBB security in the generic group model in Sect. 4 in Theorem 1.

4 Distributional VBB Security in the Generic Group Model

This section will prove Theorem 1, which establishes the distributional virtual black box security of our construction in the generic group model over the class

of uniform distributions for point functions with wildcards. Our framework for reasoning in the generic group setting draws from [8].

In a generic group proof, there are many closely related but technically distinct kinds of objects that are often conflated. There are the underlying group elements, which can be associated with their exponents in $\mathbb{Z}_p$ relative to the common base. There are the handles that the group oracle associates to these elements. There are formal polynomials which may track known or unknown relationships between group elements. There are subsets of handles which the adversary has previously seen, and other handles whose distribution remains independent of the adversary's view so far. In order to make our proof as rigorous and precise as possible, we will keep explicit track of all of these various objects, and the maps between them.

We define an equivalent security game where an adversary calls two oracles simultaneously, one of whose behavior is already completely known. The purpose of incorporating a known oracle into the security game is to rigorously define when the unknown oracle deviates from expected behavior, and thus, when the adversary has distinguishing power. Given that a low probability failure event does not occur, any algorithm's behavior when interacting with either of these oracles should be identical. The actual calculation of the probability of such a failure event is conceptually simple and done by many previous works for different noise distributions. On the other hand, in order to properly describe the notion of "identical behavior" we introduce some basic technical machinery from category theory.

We establish some notation before proceeding. Let bold letters denote symbolic variables and non-bold letters denote the sampled random values for the corresponding variable. Let $f \in \mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{x}]$ be a fixed polynomial of degree $n-1$ in $\mathbf{x}$ which is linear in each $\mathbf{a}_i$ individually. Let $\mathcal{H}_S$ and $\mathcal{H}_M$ be two identical copies of the same space of strings corresponding to handles in the generic group model.

Since our proof takes place in the generic group model, and our obfuscated program consists of a set of group elements, we will use the notation $\mathcal{G}_S, \mathcal{G}_M, \mathcal{G}_E$ to denote three different ways that an adversary can be supplied with handles representing an obfuscated program and how requests to the generic group operation oracle are answered. $\mathcal{G}_S$ will implement faithful interaction with the true construction in the generic group model. $\mathcal{G}_M$ implements a hybrid setting that we will show is indistinguishable from $\mathcal{G}_S$ to the adversary. Finally, $\mathcal{G}_E$ implements a setting that can be simulated without knowledge of the function drawn from the distribution (and is indistinguishable from $\mathcal{G}_M$).

The high level structure of our proof is pretty typical for a generic group argument. The group oracle $\mathcal{G}_M$ will behave similarly to $\mathcal{G}_S$, but instead of sampling random exponents according to the proscribed polynomial structure, it will work with formal polynomials representing this structure, hence ignoring any spurious relationship arises from a particular choice at the sampling stage. Arguing that $\mathcal{G}_S$ and $\mathcal{G}_M$ are indistinguishable is where we use the Schwartz-Zippel Lemma. An adversary will only receive a different distribution of handles

if it manages to find a spurious relationship while interacting with $\mathcal{G}_S$, which must mean that the sampling happened to choose a root of a non-trivial, low degree formal polynomial. The Schwartz-Zippel Lemma allows us to conclude that this will occur with only negligible probability over the sampling employed by $\mathcal{G}_S$.

To argue that $\mathcal{G}_M$ and $\mathcal{G}_E$ are indistinguishable, we will need to argue that the adversary cannot (except with negligible probability), detect the remaining formal polynomial structure in $\mathcal{G}_M$, since doing so requires referencing many correctly structured elements and avoiding the random elements completely. As long as the wildcards are not too dense, this is an intractable combinatorial problem for the adversary.

Definition 2 ($\mathcal{G}_S$: Oracle *Start*).
First, sample the following uniformly at random:

- $\mathcal{W} = \{i_1, \cdots, i_w\} \subset [n]$
- $y_i \in \{0,1\}$ *for each* $i \notin \mathcal{W}$
- $a_1, \cdots, a_n \in \mathbb{Z}_p$
- *Random embedding* $\Phi_S : G \hookrightarrow \mathcal{H}_S$

For the initial set of handles representing the $2n$ *group elements in the obfuscation of* $f_{\boldsymbol{y},\mathcal{W}}$*, for each entry* $(i,j) \in [n] \times \{0,1\}$*:*

- *If* $i \in \mathcal{W}$ *or* $y_i = j$ *(i.e. the input bit is part of an accepting string), output* $\Phi_S\left(g^{F(a_1,\cdots,a_n,2i+j)}\right)$
- *Otherwise sample a uniformly random exponent* ρ_{ij} *and output* $\Phi_S(g^{\rho_{ij}})$

Given a group operation query on (h_1, h_2)*:*

- *Find* $g_1 = \Phi_S^{-1}(h_1)$ *and* $g_2 = \Phi_S^{-1}(h_2)$*. If either does not exist, ignore the query.*
- *Return* $\Phi_S(g_1 \cdot g_2)$

Note that $\mathcal{G}_S$ faithfully instantiates our construction described in Sect. 3 in the generic group model. We will now describe an alternative oracle implementation that uses symbolic variables instead of group elements to produce the generic group functionality:

Definition 3 ($\mathcal{G}_M$: Oracle *Middle*).
First, sample the following uniformly at random:

- $\mathcal{W} = \{i_1, \cdots, i_w\} \subset [n]$
- $y_i \in \{0,1\}$ *for each* $i \notin \mathcal{W}$
- *Random embedding* $\Phi_M : \mathbb{Z}_p[\boldsymbol{a}_1, \cdots, \boldsymbol{a}_n, \boldsymbol{b}_1, \cdots, \boldsymbol{b}_{n-w}] \hookrightarrow \mathcal{H}_M$*.*

Let $\sigma : \{0,1\}^n \times \{0,1\} \to [n-w]$ *be an arbitrary ordering of the* $(n-w)$ *coordinate pairs* (i,j) *where* $i \notin \mathcal{W}$ *and* $j \neq y_i$*, and which is not defined on the other coordinate pairs.*

For the initial set of handles representing the $2n$ *group elements in the obfuscation of* $f_{\boldsymbol{y},\mathcal{W}}$*, for each entry* $(i,j) \in [n] \times \{0,1\}$*:*

- *If $i \in \mathcal{W}$ or $y_i = j$ (i.e. the input bit is part of an accepting string), output $\Phi_M(F(\boldsymbol{a}_1, \cdots, \boldsymbol{a}_n, 2i + j))$*
- *Otherwise output the label $\Phi_M(\boldsymbol{b}_{\sigma(ij)})$*

Given a group operation query on (h_1, h_2):

- *Find $p_1 = \Phi_M^{-1}(h_1)$ and $p_2 = \Phi_M^{-1}(h_2)$. If either does not exist, ignore the query.*
- *Return $\Phi_M(p_1 + p_2)$*

The two oracles are related by the existence of the following *evaluation map in the exponent*:

$$\begin{aligned} \phi : \mathbb{Z}[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}] &\longrightarrow G \\ F(\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}) &\longmapsto g^{F(a_1, \cdots, a_n, b_1, \cdots, b_{n-w})} \end{aligned}$$

where $b_k = \rho_{\sigma^{-1}(k)}$ are the values of the random exponents sampled by Oracle S for the non-accepting slots. Only the existence of this evaluation map is necessary for the proof, so its dependence on unknown random values is not an issue.

In particular ϕ is a surjective group homomorphism of $(\mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}], +)$ into $(G, \times)$, since it is a composition of an evaluation map with an exponential map, which are both surjective group homomorphisms.

The idea behind defining such an evaluation map is to define the failure event as a substructure of a larger structure which may then be used to formalize when the behavior is identical. In particular, we will see that the failure event corresponds to the kernel of this evaluation map that we just defined.

Simultaneous Oracle Game. Rather than proving that the difference in any adversary's output probabilities when interacting with ($\mathcal{G}_S$ vs. $\mathcal{G}_M$) or ($\mathcal{G}_M$ vs. $\mathcal{G}_E$) is small directly, we will define another security game and exhibit a reduction to the desired statements. In this new security game, the adversary simultaneously queries two oracles for operations on group elements: one oracle $\mathcal{G}_M$ is known and serves as a convenience for formalizing the generic group oracle, and the second $\mathcal{G}_*$ is the unknown that the adversary wishes to identify. We define the game with oracles $(\mathcal{G}_S, \mathcal{G}_M)$ below and note that the game and reduction for oracles $(\mathcal{G}_M, \mathcal{G}_E)$ is symmetric.

Definition 4 (Simultaneous Oracle Game). *An adversary is given access to a pair of oracles $(\mathcal{G}_M, \mathcal{G}_*)$, where $\mathcal{G}_*$ is $\mathcal{G}_M$ with probability $1/2$ and $\mathcal{G}_S$ with probability $1/2$. In each round, the adversary asks the same query to both oracles. The adversary wins the game if he guesses correctly the identity of $\mathcal{G}_*$.*

To make precise the notion of an adversary playing both oracles simultaneously and asking the same queries, the adversary maintains two sets $\mathcal{H}_S^t$ and $\mathcal{H}_M^t$ which are the sets of handles returned by the oracles after t query rounds. The adversary then maintains a function $\Psi : \mathcal{H}_M^t \to \mathcal{H}_S^t$. Initially, the adversary

sets $\Psi(h^b_{ij}) = h^a_{ij}$ for each initial slot location $(i,j) \in \{1,n\} \times \{0,1\}$, where h^a_{ij} is the handle corresponding to the slot (i,j) in oracles S and h^b_{ij} the handle in oracles M. After each query $h^m = \mathcal{G}_M(h^b_1, h^b_2)$ and $h^s = \mathcal{G}_S(\Psi(h^b_1), \Psi(h^b_2))$ the adversary updates the function with the definition $\Psi(h^s) = h^m$.

Lemma 2. *Suppose there exists an algorithm $\mathcal{A}$ such that*

$$\left| Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] - Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1] \right| \geq \delta$$

Then an adversary can win the simultaneous oracle game with probability at least $\frac{1}{2} + \frac{\delta}{2}$ for any pair of oracles $(\mathcal{G}_M, \mathcal{G}_* = \mathcal{G}_M/\mathcal{G}_S)$.

Proof. Let $p = \Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1]$ and $q = \Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1]$. The adversary can estimate these parameters to within a bounded polynomial of the true parameter by simulating each oracle and $\mathcal{A}$'s behavior on each.

Without loss of generality, we can assume that $p \geq q$. Otherwise, we can define p, q to be the inverse quantities $\Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 0], \Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 0]$ respectively.

The adversary will guess $\mathcal{G}_* = \mathcal{G}_M$ if $\mathcal{A}^{\mathcal{G}_*}(\mathcal{O}^{\mathcal{G}_*}) = 1$ and $\mathcal{G}_* = \mathcal{G}_S$ if $\mathcal{A}^{\mathcal{G}_*}(\mathcal{O}^{\mathcal{G}_*}) = 0$. The probability of success is given by

$$\begin{aligned}
\Pr[\mathcal{A}^{\mathcal{G}_*}(\mathcal{O}^{\mathcal{G}_*}) = \mathcal{G}_*] &= \Pr[\mathcal{G}_* = \mathcal{G}_M] \Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] \\
&\quad + \Pr[\mathcal{G}_* = \mathcal{G}_S] \Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 0] \\
&= \frac{1}{2} + \frac{1}{2}(p - q) \\
&\geq \frac{1}{2} + \frac{\delta}{2}
\end{aligned}$$

Indistinguishability between *Start* and *Middle*. The following gives a criteria for overall indistinguishability of the output handle distributions.

Definition 5. *The pair (h^s, h^m) of answers returned by $(\mathcal{G}_S, \mathcal{G}_M)$ after query number t is called identical if it satisfies one of the following:*

1. *$h^s \notin \mathcal{H}^t_S$ and $h^m \notin \mathcal{H}^t_M$*
2. *The oracles return handles $h^s \in \mathcal{H}_S, h^m \in \mathcal{H}_M$ respectively such that $\Psi(h^m) = h^s$*

Note that in case (1), h^s and h^m are both freshly sampled uniformly random strings and their distributions are equal.

Lemma 3. *In the simultaneous oracle game with $\mathcal{G}_* = \mathcal{G}_S$, suppose for every query (h^m_1, h^m_2) to oracle M and corresponding query $(\Psi(h^m_1), \Psi(h^m_2))$ to oracle S, the answers returned are identical. Then for any algorithm $\mathcal{A}$, we have*

$$\Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1] = \Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1]$$

Proof (Proof of Lemma 3). If we had swapped the oracles $\mathcal{G}_S$ and $\mathcal{G}_M$ and the adversary had used Ψ^{-1} instead of Ψ, the answer distributions would have been identical and $\mathcal{A}$ would have to produce the same output distribution.

Remark 2. Note that this argument does not depend on the particular implementations of $\mathcal{G}_S, \mathcal{G}_M$, and therefore the lemma also holds for the pair of oracles $\mathcal{G}_M, \mathcal{G}_E$ (to be defined later in Definition 6).

Thus it suffices to show that

Lemma 4. *Suppose an adversary makes an arbitrary sequence of queries and receives answers*

$$\{h_t^s = \mathcal{G}_S(\Psi(h_{t1}^m), \Psi(h_{t2}^m))\}_{t=1}^{Q}$$
$$\{h_t^m = \mathcal{G}_M(h_{t1}^m, h_{t2}^m)\}_{t=1}^{Q}$$

Then with overall probability at least $1 - \dfrac{(Q+2n)^2}{p}$*, for every* t*,* h_t^s *and* h_t^m *are identical as defined in Definition 5.*

Proof. Initially each set of $2n$ handles given by each oracle are uniformly random strings and hence indistinguishable. The proof is by induction under the following hypothesis:

Suppose the adversary has made t queries so far and has $\mathcal{H}_S^t, \mathcal{H}_M^t$ satisfying the following:

1. For each query made so far, the answer distributions have been identical.
2. For every $h^s \in \mathcal{H}_S^t$, there exists a unique $f \in \mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n]$ such that $\Phi_S \circ \phi(f) = \Phi_M^{-1}(f)$.

We can state this inductive hypothesis this in the following commutative diagram:

$$\begin{array}{ccccc}
\mathbb{Z}_p[\mathbf{a},\mathbf{b}] & \xrightarrow{\Phi_M,\simeq} & \mathrm{Im}(\Phi_M) & \xleftarrow{i_M} & \mathcal{H}_M^t \\
\Big\downarrow{\scriptstyle \phi} & \swarrow^{\exists!} & & & \Big\downarrow{\scriptstyle \Psi,=} \\
G & \xrightarrow[\Phi_S,\simeq]{} & \mathrm{Im}(\Phi_S) & \xleftarrow[i_S]{} & \mathcal{H}_S^t
\end{array}$$

Here $\mathrm{Im}(\Phi_M), \mathrm{Im}(\Phi_S)$ are the relevant handles in the handle spaces. Commutativity of the lower triangle under the unique lift means that for all $h^s \in \mathcal{H}_S^t, \exists! f \in \mathbb{Z}_p[\boldsymbol{x}]$ such that $i_S(h^s) = \Phi_S \circ \phi(f)$. Note that the upper triangle trivially commutes because the unique lift is defined by the composition $\Phi_M \circ i_M \circ \Psi^{-1}$. To ease the notation a little, we'll omit the inclusion maps from here on when it is obvious the handle is in $\mathcal{H}_*^t$.

Now assuming the inductive hypothesis, suppose the $(t+1)$th query is the group operation of $h_1, h_2 \in \mathcal{H}_M^t$ and $\Psi(h_1), \Psi(h_2) \in \mathcal{H}_S^t$. Oracle M will output

the handle $h^m = \Phi_M\left(\Phi_M^{-1}(h_1) + \Phi_M^{-1}(h_2)\right) =: h_1 \cdot h_2$, and Oracle S will output the handle $h^s = \Phi_S\left(\Phi_S^{-1}(\Psi(h_1)) \times \Phi_S^{-1}(\Psi(h_2))\right) =: \Psi(h_1) \cdot \Psi(h_2)$. The $(\cdot)$ notation on handles is justified by the fact that $\mathrm{Im}(\Phi_M) \subset \mathcal{H}_M$ is trivially isomomorphic as a group to $\mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n]$, where its group operation is obtained by pulling back by Φ_M, and likewise for $\mathrm{Im}(\Phi_S) \subset \mathcal{H}_S$.

We have the following two cases:

1. $h^m \in \mathcal{H}_M^t$ (i.e. this handle was seen previously). Then

$$\begin{aligned}\Psi(h_1) \cdot \Psi(h_2) &= (\Phi_S \circ \phi \circ \Phi_M^{-1})(h_1) \cdot (\Phi_S \circ \phi \circ \Phi_M^{-1})(h_2) \\ &= (\Phi_S \circ \phi \circ \Phi_M^{-1})(h_1 \cdot h_2) \\ &= (\Phi_S \circ \phi \circ \Phi_M^{-1})(h^m) \\ &= \Psi(h^m)\end{aligned}$$

   where we use commutativity of the diagram on each factor handle, the homomorphism property of the maps, the definition of oracle M's output, and commutativity of the diagram on the output handle (which we can do since the handle was previously defined).
   Thus the handles in the output pair have the same distribution, and since no new handles are created, the inductive hypothesis trivially remains satisfied.
2. $h^m \notin \mathcal{H}_M^t$ (i.e. this is a new handle).
   (a) If $h^s \notin \mathcal{H}_S^t$ is also a new handle, then the unique lift simply extends to map h^s to $\Phi_M^{-1}(h^m)$, and both $\mathcal{H}_M^t$ and $\mathcal{H}_S^t$ are augmented by one element. The handles in the output pair are new and uniformly distributed, and the inductive hypothesis is satisfied.
   (b) If $h^s \in \mathcal{H}_S^t$, then by the inductive hypothesis, h^s lifts to some $f_s \in \mathbb{Z}_p[\mathbf{x}]$ which maps to some $\tilde{h}^b = \Psi^{-1}(h^s)$. However we also have $f_m = \Phi_M^{-1}(h^m) \neq f_s$, since $h^m \notin \mathcal{H}_M^t$. Thus both f_s and f_m are lifts of h^s which make the diagram commute, so after this query the inductive hypothesis is no longer satisfied for the next query.
   This event only happens if $f_s - f_m \in \ker \phi$ and $f_s - f_m$ is nontrivial. Thus the proof is complete as long as we show this event happens with low probability.

Now consider the following sequential variant of the game. The adversary plays the game using the real Oracle M and his own simulation of Oracle S obtained by outputting a uniformly random string when $\mathcal{G}_M$ does and using the Ψ map when $\mathcal{G}_M$ outputs an existing string. He then plays the exact same sequence to the real Oracle S and compares these answers to the ones produced by the real Oracle M. As long as the bad event does not occur, the sequence of queries asked in this sequential game is identical to the sequence of queries asked playing the real pair of oracles.

Note that the occurrence of the bad event is decided by the initial random sampling of $a_1, \cdots, a_n \in \mathbb{Z}_p$, and thus the bad event either occurs in both the sequential and parallel variants or in neither. So it suffices to just bound the probability of the bad event occurring at any time in the sequential game.

For each pair (f_s, f_m), $f_s - f_m$ is a degree-1 polynomial in n variables over $\mathbb{Z}_p$. Thus the bad event happens with probability at most $\frac{1}{p}$ by Lemma 1, the Schwartz-Zippel lemma. Thus by a union bound, after Q queries of either type, there are at most $(Q+2n)^2$ pairs of symbolic polynomials, so with probability at most $\frac{(Q+2n)^2}{p}$ the two distributions of handles are distinguishable.

We remark that everything in the proof only relied on diagram arguments and did not care about the actual structure of the underlying objects, except for analyzing when $f_s - f_m \in \ker \phi$ occurred. Thus in the proceeding reductions between other oracles, all this automatically follows provided we can define an appropriate evaluation map ϕ, and we only need to analyze the kernel of the corresponding evaluation map.

Lemma 5. *For an adversary $\mathcal{A}$ in the generic group model which makes Q queries to the generic group oracle,*

$$\left| \Pr_{\substack{C \leftarrow \mathcal{D}_n, \\ \mathcal{G}_S, \mathcal{O}, \mathcal{A}}} [\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}(C, 1^n)) = P(C)] - \Pr_{\substack{C \leftarrow \mathcal{D}_n, \\ \mathcal{G}_M, \mathcal{O}, \mathcal{A}}} [\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}(C, 1^n)) = P(C)] \right| \leq \frac{(Q+2n)^2}{2^n}$$

Proof. From Lemma 3 we have that:

$$\Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1] = \Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1]$$

as long as all queries to the generic group oracles are *identical* as defined in Definition 5.

Lemma 4 tells us that the probabilities of all queries not being identical during the simultaneous oracle game between $(\mathcal{G}_S, \mathcal{G}_M)$ is at most $\frac{(Q+2n)^2}{p}$, where Q is the number of the adversary's queries to the generic group oracle and $p > 2^n$ is the order of the group.

Therefore, the difference $\Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] - \Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1]$ is at most $\frac{(Q+2n)^2}{2^n}$, and so an adversary's advantage in the simultaneous oracle game between $(\mathcal{G}_M, \mathcal{G}_S)$ and $(\mathcal{G}_M, \mathcal{G}_M)$ is:

$$\begin{aligned}
\Pr[\mathcal{A}^{\mathcal{G}_*}(\mathcal{O}^{\mathcal{G}_*}) = \mathcal{G}_*] &= \Pr[\mathcal{G}_* = \mathcal{G}_M] \Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] \\
&\quad + \Pr[\mathcal{G}_* = \mathcal{G}_S] \Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 0] \\
&= \frac{1}{2} + \frac{1}{2}(\Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] - \Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1]) \\
&\leq \frac{1}{2} + \frac{(Q+2n)^2}{2 \cdot 2^n}
\end{aligned}$$

This, plugged into the reduction from Lemma 2, tells us that for all adversaries:

$$\left| Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}) = 1] - Pr[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}) = 1] \right| \leq \frac{(Q+2n)^2}{2^n}$$

Game between *Middle* and *End*

Definition 6 ($\mathcal{G}_E$: Oracle *End*).
First, sample the following uniformly at random:

- *Random embedding* $\Phi_E : \mathbb{Z}_p[\boldsymbol{c}_1, \cdots, \boldsymbol{c}_{2n}] \hookrightarrow \mathcal{H}_E$.

For the initial set of handles representing the $2n$ *group elements in the obfuscation of* $f_{\boldsymbol{y},\mathcal{W}}$*, for each entry* $(i, j) \in [n] \times \{0, 1\}$*:*

- *Output* $\Phi_E(\boldsymbol{c}_{2i+j})$

Given a group operation query on (h_1, h_2)*:*

- *Find* $p_1 = \Phi_E^{-1}(h_1)$ *and* $p_2 = \Phi_E^{-1}(h_2)$*. If either does not exist, ignore the query.*
- *Return* $\Phi_E(p_1 + p_2)$

Oracle M and Oracle E are related by the following *evaluation map* which is defined on the generators of $\mathbb{Z}_p[\mathbf{c}_1, \cdots, \mathbf{c}_{2n}]$ and extended by linearity.

$$\begin{aligned} \phi : \mathbb{Z}_p[\mathbf{c}_1, \cdots, \mathbf{c}_{2n}] &\longrightarrow \mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}] \\ \mathbf{c}_k &\longmapsto \mathbf{b}_{\sigma(\lfloor k/2 \rfloor, k \bmod 2)} \text{ if } \sigma \text{ is defined here} \\ \mathbf{c}_k &\longmapsto F(\mathbf{a}_1, \cdots, \mathbf{a}_n, k) \text{ otherwise} \end{aligned}$$

In other words the monomial c_k is mapped to the same symbolic polynomial that Oracle *Middle* assigned to the slot $(\lfloor k/2 \rfloor, k \mod 2)$, which is either a symbolic variable $\mathbf{b}$ or a symbolic polynomial $F(\mathbf{a}_1, \cdots, \mathbf{a}_n, k)$. Since the $\mathbf{c}_k$'s generate the entire additive group $\mathbb{Z}_p[\mathbf{c}_1, \cdots, \mathbf{c}_{2n}]$, this extends to a group homomorphism of $(\mathbb{Z}_p[\mathbf{c}_1, \cdots, \mathbf{c}_{2n}], +)$ into $(\mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}], +)$.

Lemma 6. *Suppose an adversary makes an arbitrary sequence of queries and receives answers*

$$\begin{aligned} &\{h_t^m = \mathcal{G}_S(\Psi(h_{t1}^e), \Psi(h_{t2}^e))\}_{t=1}^Q \\ &\{h_t^e = \mathcal{G}_M(h_{t1}^e, h_{t2}^e)\}_{t=1}^Q \end{aligned}$$

If $w/n \leq 3/4$*, then with overall probability at least* $1 - \frac{2}{2^{0.0613n}}$ *for every* t*,* h_t^s *and* h_t^m *are identical as defined in Definition 5.*

The proof of this lemma starts with the same setup as the proof of 4. The adversary maintains a function $\Psi : \mathcal{H}_E \to \mathcal{H}_M$ and two sets of handles $\mathcal{H}_E^t, \mathcal{H}_M^t$.

Proof. Inductively, after t queries, assume the following commutative diagram is true:

$$\begin{array}{ccccc} \mathbb{Z}_p[\mathbf{c}_1, \cdots, \cdots, \mathbf{c}_{2n}] & \xrightarrow{\Phi_E, \simeq} & \mathrm{Im}(\Phi_E) & \xleftarrow{i_E} & \mathcal{H}_E^t \\ \Big\downarrow \phi & \swarrow \exists! & & & \Big\downarrow \Psi, = \\ \mathbb{Z}_p[\mathbf{a}_1, \cdots, \mathbf{a}_n, \mathbf{b}_1, \cdots, \mathbf{b}_{n-w}] & \xrightarrow[\Phi_M, \simeq]{} & \mathrm{Im}(\Phi_M) & \xleftarrow[i_M]{} & \mathcal{H}_M^t \end{array}$$

The same diagram chase from the proof of (4) tells us that the next pair of query answers (h^e, h^m) only fails to satisfy the inductive hypothesis if h^m lifts to $f_m \in \mathbb{Z}_p[\mathbf{c}]$ by the inductive hypothesis, but$f_m \neq \Phi_E^{-1}(h^e) =: f_e$, so $f_m - f_e \in \ker \phi$ and $f_m - f_e$ is nontrivial. Necessary but not sufficient conditions for $f_m - f_e$ to be in the kernel of ϕ are:

1. $f_m - f_e$ must have a zero coefficient in front of any $\mathbf{c}_k$ that is defined under the σ map, since each free variable $\mathbf{b}_j$ has a unique preimage.
2. $f_m - f_e$ must have at least $n-1$ nonzero coefficients

As with the proof of (4), we analyze the sequential variant where the adversary plays a sequence of queries to $\mathcal{G}_E$ and then plays the exact same sequence of queries to $\mathcal{G}_M$. After Q queries the adversary has at most $Q + 2n$ symbolic polynomials in $\mathbb{Z}_p[\mathbf{c}]$. For each pair of polynomials f_m, f_e in this set, the variables $\mathbf{c}_k$ are mapped by the initial random sampling of the wildcard slots by Oracle M.

Now suppose the adversary fixes a polynomial containing $n-1$ nonzero coefficients of the $\mathbf{c}_k$'s such that m columns in the original table of $2n$ entries have nonzero coefficients for both entries in the column. This means that the oracle must necessarily choose those m columns to be wildcard slots, since otherwise one of the two entries in the column will not be in the kernel of the ϕ map.

This means that the probability over the initialization of the oracle that these m columns are all chosen to be wildcard slots is $\frac{\binom{n-m}{w-m}}{\binom{n}{w}}$. The remaining $n-1-2m$ columns each must either match the entry chosen by the adversary or be a wildcard slot. There are $(n-1-2m)-(w-m) = (n-1-w)-m$ slots that cannot be wildcard slots and thus have at most probability $1/2$ each of matching the entry chosen by the adversary. Thus the probability that this polynomial is in the kernel of ϕ is

$$\frac{\binom{n-m}{w-m}}{\binom{n}{w}} \left(\frac{1}{2}\right)^{n-1-w-m} \tag{1}$$

An upper bound for this can be computed by maximizing the expression with respect to the adversary's choice of m. If we increment m by 1, the first factor is multiplied by $\frac{w-m}{n-m}$ while the second factor is multiplied by 2. Note that $\frac{w-m}{n-m}$ is monotonically decreasing in m; thus, this quantity is maximized when m is the largest possible integer such that $\frac{w-m}{n-m} > 1/2$ is still true. Note that when $w < n/2$, then the optimal choice is $m = 0$. Assuming $w > n/2$ and solving for this inequality we obtain that $m = 2w - n$. Now the problem also has a physical constraint that $m \leq n/2$ since the adversary can choose at most $n/2$ slots. Thus there are three parameter regimes based on α:

1. $\alpha \leq n/2$: the optimal choice is $m = 0$
2. $n/2 \leq \alpha \leq 3n/4$: the optimal choice is $m = 2w - n$
3. $n > 3n/4$: the optimal choice is $m = n/2$

In case 1, the probability is then clearly bounded by $(1/2)^{n-1-w}$.

In case 2, making the substitution $m = 2w - n$ and $w = \alpha n$ where $\alpha \in [0, 1)$ in the expression (1), we obtain

$$\frac{\binom{2(1-\alpha)n}{(1-\alpha)n}}{\binom{n}{\alpha n}} 2^{(3\alpha-2)n} = \frac{[2(1-\alpha)n]!}{[(1-\alpha)n]![(1-\alpha)n]!} \frac{[\alpha n]![(1-\alpha)n]!}{n!} 2^{(3\alpha-2)n}$$

$$= \frac{[2(1-\alpha)n]![\alpha n]!}{[(1-\alpha)n]!n!} 2^{(3\alpha-2)n}$$

Recall that for all integers k the following is true by Sterling's formula:

$$\sqrt{2\pi}\sqrt{k}\left(\frac{k}{e}\right)^k \leq k! \leq e\sqrt{k}\left(\frac{k}{e}\right)^k$$

We can absorb the factors of $\sqrt{2\pi}$ and e in front into a small constant term less than 2. Note that since each factorial is a constant multiple of n, then the $\sqrt{k}$ term also yields a constant term, so we only need to compute the $(k/e)^k$ terms. This gives

$$\frac{[2(1-\alpha)n/e]^{2(1-\alpha)n}[\alpha n/e]^{\alpha n}}{[(1-\alpha)n/e]^{(1-\alpha)n}[n/e]^n} 2^{(3\alpha-2)n} = \left(\frac{[2(1-\alpha)n/e]^{2(1-\alpha)}[\alpha n/e]^{\alpha}}{[(1-\alpha)n/e]^{(1-\alpha)}[n/e]^1} 2^{(3\alpha-2)}\right)^n$$

We just need to show that the base is a constant bounded away from 1. Collecting terms in this, we obtain

$$\begin{aligned}
&2^{2(1-\alpha)}[1-\alpha]^{2(1-\alpha)}[n/e]^{2(1-\alpha)}\alpha^{\alpha}[n/e]^{\alpha}[1-\alpha]^{-(1-\alpha)}[n/e]^{-(1-\alpha)}[n/e]^{-1}2^{3\alpha-2} \\
&= [n/e]^{2(1-\alpha)n-(1-\alpha)+\alpha-1}[1-\alpha]^{2(1-\alpha)-(1-\alpha)}\alpha^{\alpha}2^{2(1-\alpha)+3\alpha-2} \\
&= (1-\alpha)^{1-\alpha}\alpha^{\alpha}2^{\alpha}
\end{aligned}$$

Taking $\log_2$ we obtain $(1-\alpha)\log_2(1-\alpha)+\alpha\log_2\alpha+\alpha \leq -0.0613$ when $\alpha \leq 3/4$, so the probability of success is bounded by $\frac{2}{2^{0.0613n}}$.

Finally in case 3, substituting $m = n/2$ in the expression (1) gives

$$\frac{\binom{n/2}{(\alpha-1/2)n}}{\binom{n}{\alpha n}} 2^{(\alpha-1/2)n} = \frac{[n/2]!}{[(1-\alpha)n]![(\alpha-1/2)n]!} \frac{[\alpha n]![(1-\alpha)n]!}{n!} 2^{(\alpha-1/2)n}$$

$$= \frac{[n/2]![\alpha n]!}{n![(\alpha-1/2)n]!} 2^{(\alpha-1/2)n}$$

Applying the Sterling approximation, we obtain

$$\frac{[n/e]^{n/2}2^{-n/2}[\alpha n/e]^{\alpha n}}{[n/e]^{n}[(\alpha-1/2)n/e]^{(\alpha-1/2)n}}2^{(\alpha-1/2)n} = \left(\frac{[n/e]^{1/2}2^{-1/2}[\alpha n/e]^{\alpha}}{[n/e]^{1}[(\alpha-1/2)n/e]^{(\alpha-1/2)}}2^{(\alpha-1/2)}\right)^{n}$$

The base of the exponent is

$$[n/e]^{1/2+\alpha-1-(\alpha-1/2)}[\alpha]^{\alpha}[\alpha-1/2]^{(1/2-\alpha)}2^{\alpha-1} = \alpha^{\alpha}(\alpha-1/2)^{1/2-\alpha}2^{\alpha-1}$$

Again taking $\log_2$ we obtain the condition $(1/2-\alpha)\log_2(\alpha-1/2)+\alpha\log\alpha+\alpha-1<0$, which is satisfied when $\alpha<0.774$. This does not give much of an improvement over the previous constraint of $\alpha \leq 3/4$, so we state our final result just in that regime.

Apply a union bound of this probability over all $(Q+2n)^2$ pairs of symbolic polynomials to get the statement in the theorem.

Lemma 7. *For an adversary $\mathcal{A}$ in the generic group model which makes Q queries to the generic group oracle,*

$$\left|\Pr_{\substack{C\leftarrow\mathcal{D}_n,\\ \mathcal{G}_M,\mathcal{O},\mathcal{A}}}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}(C,1^n))=P(C)] - \Pr_{\substack{C\leftarrow\mathcal{D}_n,\\ \mathcal{G}_E\mathcal{O},\mathcal{A}}}[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E}(C,1^n))=P(C)]\right| \leq \frac{1}{2^{0.0613n}}$$

Proof. Uses Lemmas 3 (recalling that the statement also holds for the pair $\mathcal{G}_M, \mathcal{G}_E$) and 6 plugged into the reduction from Lemma 2.

From Lemma 3 (recalling that the statement also holds for the pair $\mathcal{G}_M, \mathcal{G}_S$) we have that:

$$\Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M})=1] = \Pr[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E})=1]$$

as long as all queries to the generic group oracles are *identical* as defined in Definition 5.

Lemma 4 tells us that the probabilities of all queries not being identical during the simultaneous oracle game between $(\mathcal{G}_M, \mathcal{G}_E)$ is at most $\frac{2}{2^{0.0613n}}$.

Therefore, the difference $\Pr[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E})=1] - \Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M})=1]$ is at most $\frac{2}{2^{0.0613n}}$, and so an adversary's advantage in the simultaneous oracle game between $(\mathcal{G}_M, \mathcal{G}_E)$ and $(\mathcal{G}_M, \mathcal{G}_M)$ is:

$$\begin{aligned}
\Pr[\mathcal{A}^{\mathcal{G}_*}(\mathcal{O}^{\mathcal{G}_*})=\mathcal{G}_*] &= \Pr[\mathcal{G}_*=\mathcal{G}_E]\Pr[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E})=1] \\
&\quad + \Pr[\mathcal{G}_*=\mathcal{G}_M]\Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M})=0] \\
&= \frac{1}{2} + \frac{1}{2}(\Pr[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E})=1] - \Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M})=1]) \\
&\leq \frac{1}{2} + \frac{2}{2^{0.0613n}}
\end{aligned}$$

This, plugged into the reduction from Lemma 2, tells us that for all adversaries:

$$\left|Pr[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M})=1] - Pr[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E})=1]\right| \leq \frac{1}{2^{0.0613n}}$$

Theorem 1. *The obfuscator for pattern matching with wildcards defined in Sect. 3 satisfies distributional VBB security for the ensemble of uniform distributions over $\{0,1\}^n$.*

Proof. For any adversary $\mathcal{A}$ in the Distributional VBB game (in the generic group model), consider the following Simulator $\mathcal{S}$ which simply runs $\mathcal{A}$ on input produced by and interacted with like in Oracle *End* and outputs the same. Note that none of the behavior in Oracle *End* is dependent on the actual function $f_{\boldsymbol{y},\mathcal{W}}$ obfuscated. Therefore a simulator with no access to the function $f_{\boldsymbol{y},\mathcal{W}}$ drawn from the distribution is able to simulate $\mathcal{A}$ as described.

$\mathcal{S}$ then perfectly simulates the behavior of $\mathcal{A}$ interacting with oracle $\mathcal{O}_E$:

$$\Pr_{C\leftarrow\mathcal{D}_n,\mathcal{S}}[\mathcal{S}^C(1^{|C|},1^n)=P(C)] = \Pr_{C\leftarrow\mathcal{D}_n,\mathcal{G}_E,\mathcal{O},\mathcal{A}}[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E}(C,1^n))=P(C)]$$

From Lemma 7, we have that the difference in output probabilities between $\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E})$ and $\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M})$ in the distributional VBB game in the generic group model is at most $\frac{1}{2^{0.0613n}}$:

$$|\Pr_{\substack{C\leftarrow\mathcal{D}_n,\\ \mathcal{G}_E,\mathcal{O},\mathcal{A}}}[\mathcal{A}^{\mathcal{G}_E}(\mathcal{O}^{\mathcal{G}_E}(C,1^n))=P(C)] - \Pr_{\substack{C\leftarrow\mathcal{D}_n,\\ \mathcal{G}_M,\mathcal{O},\mathcal{A}}}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}(C,1^n))=P(C)]| \leq \frac{1}{2^{0.0613n}}$$

From Lemma 5, we have that the difference in output probabilities between $\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M})$ and $\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S})$ in the distributional VBB game in the generic group model is at most $\dfrac{(Q+2n)^2}{2^n}$:

$$|\Pr_{\substack{C\leftarrow\mathcal{D}_n,\\ \mathcal{G}_M,\mathcal{O},\mathcal{A}}}[\mathcal{A}^{\mathcal{G}_M}(\mathcal{O}^{\mathcal{G}_M}(C,1^n))=P(C)] - \Pr_{\substack{C\leftarrow\mathcal{D}_n,\\ \mathcal{G}_S,\mathcal{O},\mathcal{A}}}[\mathcal{A}^{\mathcal{G}_S}(\mathcal{O}^{\mathcal{G}_S}(C,1^n))=P(C)]| \leq \frac{(Q+2n)^2}{2^n}$$

Now, recall that $\mathcal{G}_S$ faithfully instantiates $\mathcal{O}$ in the generic group model. Therefore, using the triangle inquality we have:

$$|\Pr_{\substack{C\leftarrow\mathcal{D}_n,\\ \mathcal{G},\mathcal{O},\mathcal{A}}}[\mathcal{A}^{\mathcal{G}}(\mathcal{O}^{\mathcal{G}}(C,1^n))=P(C)] - \Pr_{\substack{C\leftarrow\mathcal{D}_n,\\ \mathcal{S}}}[\mathcal{S}^C(1^{|C|},1^n)=P(C)]| \leq \frac{(Q+2n)^2}{2^n} + \frac{1}{2^{0.0613n}}$$

which is a negligible function of n since the number of an adversary's generic group queries Q is a polynomial function of n, and so $\mathcal{O}$ satisfies distributional VBB security in the generic group model.

Acknowledgments. The second, third, fourth, and fifth authors are supported in part by the Defense Advanced Research Project Agency (DARPA) and Army Research Office (ARO) under Contract W911NF-15-C-0236.

The second and third authors are supported in part by NSF grants CNS-1445424 and CCF1423306, and the Leona M. & Harry B. Helmsley Charitable Trust.

The fourth and fifth authors are supported in part by NSF grants CNS-1633282, 1562888, 1565208, and DARPA SafeWare W911NF-16-1-0389.

The first and second authors are supported in part by NSF grant CNS-1552932.

The second author is supported in part by an NSF Graduate Research Fellowship DGE-16-44869.

Any opinions, findings and conclusions or recommendations expressed are those of the authors and do not necessarily reflect the views of the Defense Advanced Research Projects Agency, Army Research Office, the National Science Foundation, or the U.S. Government.

The authors wish to thank Cong Zhang for discussions in preliminary stages of this work.

References

1. Ananth, P., Jain, A., Sahai, A.: Patchable indistinguishability obfuscation: $i\mathcal{O}$ for evolving software. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part III. LNCS, vol. 10212, pp. 127–155. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56617-7_5
2. Ananth, P., Sahai, A.: Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part I. LNCS, vol. 10210, pp. 152–181. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56620-7_6
3. Apon, D., Döttling, N., Garg, S., Mukherjee, P.: Cryptanalysis of indistinguishability obfuscations of circuits over GGH13. In: 44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, Warsaw, Poland, 10–14 July 2017, pp. 38:1–38:16 (2017)
4. Applebaum, B., Brakerski, Z.: Obfuscating circuits via composite-order graded encoding. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 528–556. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_21
5. Badrinarayanan, S., Miles, E., Sahai, A., Zhandry, M.: Post-zeroizing obfuscation: new mathematical tools, and the case of evasive circuits. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 764–791. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_27
6. Barak, B., Garg, S., Kalai, Y.T., Paneth, O., Sahai, A.: Protecting obfuscation against algebraic attacks. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 221–238. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_13
7. Barak, B., et al.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_1
8. Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical identity based encryption with constant size ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_26
9. Brakerski, Z., Rothblum, G.N.: Obfuscating conjunctions. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 416–434. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_24
10. Brakerski, Z., Vaikuntanathan, V., Wee, H., Wichs, D.: Obfuscating conjunctions under entropic ring LWE. In: Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, Cambridge, MA, USA, 14–16 January 2016, pp. 147–156 (2016)
11. Brands, S.: Untraceable off-line cash in wallet with observers. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 302–318. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_26

12. Canetti, R., Rothblum, G.N., Varia, M.: Obfuscation of hyperplane membership. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 72–89. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11799-2_5
13. Chen, Y., Gentry, C., Halevi, S.: Cryptanalyses of candidate branching program obfuscators. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part III. LNCS, vol. 10212, pp. 278–307. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56617-7_10
14. Cheon, J.H., Han, K., Lee, C., Ryu, H., Stehlé, D.: Cryptanalysis of the multilinear map over the integers. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 3–12. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46800-5_1
15. Coron, J.-S., et al.: Zeroizing without low-level zeroes: new MMAP attacks and their limitations. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015, Part I. LNCS, vol. 9215, pp. 247–266. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_12
16. Coron, J.-S., Lee, M.S., Lepoint, T., Tibouchi, M.: Zeroizing attacks on indistinguishability obfuscation over CLT13. In: Fehr, S. (ed.) PKC 2017, Part I. LNCS, vol. 10174, pp. 41–58. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54365-8_3
17. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 1–17. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38348-9_1
18. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: FOCS (2013)
19. Garg, S., Miles, E., Mukherjee, P., Sahai, A., Srinivasan, A., Zhandry, M.: Secure obfuscation in a weak multilinear map model. In: Hirt, M., Smith, A. (eds.) TCC 2016. LNCS, vol. 9986, pp. 241–268. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_10
20. Gentry, C., Lewko, A.B., Sahai, A., Waters, B.: Indistinguishability obfuscation from the multilinear subgroup elimination assumption. In: IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17–20 October 2015, pp. 151–170 (2015)
21. Goyal, R., Koppula, V., Waters, B.: Lockable obfuscation. In: 58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, 15–17 October 2017, pp. 612–621 (2017)
22. Hofheinz, D., Jager, T., Khurana, D., Sahai, A., Waters, B., Zhandry, M.: How to generate and use universal samplers. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part II. LNCS, vol. 10032, pp. 715–744. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53890-6_24
23. Lin, H.: Indistinguishability obfuscation from constant-degree graded encoding schemes. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016, Part I. LNCS, vol. 9665, pp. 28–57. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49890-3_2
24. Lin, H.: Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 599–629. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_20

25. Lin, H., Tessaro, S.: Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 630–660. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_21
26. Lin, H., Vaikuntanathan, V.: Indistinguishability obfuscation from DDH-like assumptions on constant-degree graded encodings. In: IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, Hyatt Regency, New Brunswick, New Jersey, USA, 9–11 October 2016, pp. 11–20 (2016)
27. Lynn, B., Prabhakaran, M., Sahai, A.: Positive results and techniques for obfuscation. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 20–39. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_2
28. Miles, E., Sahai, A., Zhandry, M.: Annihilation attacks for multilinear maps: cryptanalysis of indistinguishability obfuscation over GGH13. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part II. LNCS, vol. 9815, pp. 629–658. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_22
29. Peikert, C.: On error correction in the exponent. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 167–183. Springer, Heidelberg (2006). https://doi.org/10.1007/11681878_9
30. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: STOC (2014)
31. Wee, H.: On obfuscating point functions. In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, 22–24 May 2005, pp. 523–532 (2005)
32. Wichs, D., Zirdelis, G.: Obfuscating compute-and-compare programs under LWE. IACR Cryptology ePrint Archive 2017:276 (2017)
33. Zimmerman, J.: How to obfuscate programs directly. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part II. LNCS, vol. 9057, pp. 439–467. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46803-6_15