Survey on Cryptographic Obfuscation

Máté Horváth*

mhorvath@crysys.hu
Laboratory of Cryptography and System Security (CrySyS Lab)†

June 4, 2015‡

Abstract. The recent result of Garg et al. (FOCS 2013) changed the previously pessimistic attitude towards general purpose cryptographic obfuscation. Since their first candidate construction, several authors proposed newer and newer schemes with more persuasive security arguments and better efficiency. At the same time, indistinguishability obfuscation proved its extreme usefulness by becoming the basis of many solutions for long-standing open problems in cryptography e.g. functional or witness encryption and others. In this survey, we give an overview of recent research, focusing on the theoretical results on general purpose obfuscation, particularly, indistinguishability obfuscation.

Keywords: Secure Computation, General Purpose Obfuscation, Indistinguishability Obfuscation, Multilinear Maps, Graded Encodings

1 Introduction

Solving the problem of secure communication was in the focus of cryptography since its beginnings. Today, our communication and its channels are changing drastically. The new opportunities of smart devices, cloud computing, storage outsourcing, data mining and web-based services all highlight that cryptography must tackle far-reaching questions of securing not only communication but also computation itself. The need for new solutions is pressing, especially, as our new era of computing expands the opportunities of not only the confiding users, but also adversaries who are tampering the integrity and privacy of communication, computation and data with more and more sophisticated techniques. In this survey, we attempt to summarize the recent theoretical results of the research community in the field of "encrypting functionalities", which we call obfuscation.

History, Goals and Difficulties. In 1976, Diffie and Hellman [DH76] suggested the use of general-purpose obfuscation to convert private-key cryptosystems to public-key cryptosystems. While the latter was realized - using different methods - soon after, the question of how to implement general-purpose obfuscation remained open until nowadays.

The goal of obfuscation is to hide information, necessarily embedded in a program code, such that the program keeps its original functionality, or in other words, obfuscation aims to make reverse engineering impossible. Learnable programs, i.e. the source code of which can be reconstructed by just executing it on different inputs, are clearly not interesting for obfuscation, although, most of the interesting programs are not learnable and the range of potential applications is extremely wide. The most direct ones are different forms of software protection. For example, any patented algorithm can be hidden even during usage by publishing only its obfuscated form. Secure software patching[1] can be achieved by obfuscating the software patch, such that it will not reveal the bug it intends to fix. In this way, users who update their software later are not exposed to new attacks. Watermarking[2] of software is possible by modifying it such that it identifies the user who bought it, and obfuscation guarantees that the watermark cannot be removed.

These and several other applications implied the widespread practical usage of "code obfuscation", even though the theoretical foundation of the topic was not ready for this. As a result, today heuristical obfuscators

*Any remarks, suggestions or corrections are appreciated. Please contact me via e-mail!

†Budapest University of Technology and Economics

‡See the Update Log about previous versions of this manuscript.

[1] For details, see [BBC+14].

[2] The formal security goals of digital watermarking were formulated first by [HMW07]. For results utilizing obfuscation we refer to [NW15] and [CHV15].

are used for practical purposes, which rely on security by obscurity and try to confuse the code reader[3]. As the security of these methods are largely built upon their secrecy, they cannot serve cryptographic purposes, as it contradicts even with the basic principles of cryptography, such as Kerckhoffs's principle, that says: "a cryptosystem should be secure even if everything about the system, except the key, is public knowledge". Indeed, fulfilling well defined security definitions, obfuscation would also become a new and extremely powerful cryptographic primitive that can be used for solving long-standing open problems in cryptography and this aspect is probably at least as important as the previously mentioned applications. Functional encryption serves as the first example, which was shown to be realizable with access to general purpose obfuscation in the pioneering work of Garg et al. [GGH+13b].

Anyone who ever wanted to understand an unknown program code experienced that finding out how a program works can be really troublesome. Such theoretical results as Rice's Theorem and the hardness of the Halting Problem or Satisfiability also seem to imply that obfuscation is not just possible, but quite natural state of programs.

However, from a different viewpoint[4], its goals seem to be rather ambitious. If we think of a software as the compression of the truth table of a function f, it is straightforward that the ideal encryption of f is a lookup table of its input-output pairs. Unfortunately, this solution would have an exponential size, while we would prefer to preserve the original size (actually the obfuscated version's size should not exceed a polynomial of the original). The difficulty is that in general, compression works by exploiting, and thus, revealing the structure of the function. In this approach, the question is the following:

Can we compress a function without revealing its structure to a computationally bounded adversary?

The formal study of program obfuscation was initiated by Barak et al. in their seminal work [BGI+01]. For the first time, they gave a rigorous definition of an obfuscator's security. Roughly speaking, [BGI+01] requires from an obfuscated version $\mathcal{O}(P)$ of a program P to behave like a "virtual black box" (*VBB*) in the sense that anything one can compute from it, one could also compute from the input-output behaviour of the program. While the *VBB* notion seems to capture the intuitive goal quite naturally, Barak et al. [BGI+01] also showed that we cannot hope to realize it in general. They managed to find a program family that is not-learnable (i.e. it is infeasible to reverse-engineer the source code *given only black-box access* to the program), but it can always be efficiently reverse-engineered *given any program* (including obfuscated ones) computing the same function. This result rules out the possibility of constructing general purpose *VBB* obfuscators, although some still interesting program families might be *VBB* obfuscatable, even if currently we are not able to determine them.

In order to avoid the negative results for *VBB*, Goldwasser and Rothblum [GR07] captured what is the maximum we can hope for when trying to achieve obfuscation. They argue that if any information exposed by $\mathcal{O}(P)$ is also exposed by every other, functionally equivalent program of similar size, then $\mathcal{O}(P)$ is the "best-possible" obfuscation. An obfuscator $\mathcal{O}(P)$ that fulfils this definition possibly expose non black-box information when *VBB* obfuscation is not possible, however it inevitably fulfils the *VBB* definition when it is achievable.

Besides the negative results, [BGI+01] also suggested a weaker definition, that avoids the *VBB* paradigm, and still can lead to making programs "unintelligible" in some rigorous and meaningful way. The definition of indistinguishability obfuscation (*iO*) requires that if two programs of similar size compute the same function, than their obfuscations should be indistinguishable. The main disadvantage of this definition is apparent: it does not express any explicit guarantee that the obfuscated version of a program "hides information". This limitation was overcome when [GR07] proved that, for efficient obfuscators, the definitions of indistinguishability and best-possible obfuscation are equivalent. This means that the indistinguishability definition, which may sometimes be technically easier to handle, can be used with the strong intuitive security guarantee of the "best-possible" definition. While it still remains hard to use the indistinguishability notion (at least compared to the *VBB* definition) it draw the attention of researchers thanks to the recent breakthrough of Garg, Gentry, Halevi, Raykova, Sahai, and Waters [GGH+13b], who proposed the first candidate for general purpose *iO* in 2013.

The first positive results pose a bunch of further questions: How far is it possible to push the limits of the indistinguishability notion? Where is the border between the *VBB* obfuscatable and indistinguishably obfuscatable programs? Can *iO* be proven secure in less and less idealized models, and finally in the standard model? Is it possible to base its security to a hardness assumption that was subject of previous research and widely accepted to be hard? Can we improve the efficiency of obfuscators such, that they become useful for practical purposes? Or the one that is more and more pressing: are we able to bridge the gap between theory and current practices that must handle the problems of securing computation, even in the absence of solid theoretical foundation? In the past years, several research papers aimed to answer these questions, and our work aims to give an overview of the current state of the art.

Way Towards the Breakthrough. The construction of the first candidate general purpose obfuscator by [GGH+13b] is not without antecedents. Its way was prepared by two other important primitives: fully homo-

[3] [DGB14, Chapter 5] provides an extensive review of these techniques.

[4] This viewpoint was mentioned in the inspiring talk of Amit Sahai [Sah14].

morphic encryption (FHE) and cryptographic multilinear maps. FHE, originally called privacy homomorphism, was introduced by Rivest, Adleman and Dertouzos [RAD78], and it aims to evaluate an arbitrary function on the ciphertext, such that the result $\texttt{Eval}(\texttt{Enc}(x))$ is still a valid ciphertext that exactly corresponds to the encryption of the function's evaluation on the original plaintext, i.e. $\texttt{Enc}(\texttt{Eval}(x))$. The first FHE scheme was provided by Gentry [Gen09] in 2009, using ideal lattices. Besides its extreme usefulness, e.g. in outsourced computation, it became a key ingredient of bootstrapping obfuscation (see subsection 3.3), although its efficiency is still an issue in practice[5].

Boneh and Silverberg [BS03] observed first that the generalization of bilinear maps (or pairings) would have far-reaching consequences[6] in cryptography. Ten years later Garg, Gentry and Halevi [GGH13a] proposed the first candidate cryptographic multilinear map, which was followed by other solutions [CLT13, GGH15, YYHK14, CLT15], several attacks [CHL+15, GHMS14, CLT14] and a bunch of applications as obfuscation (for details see subsection 3.1 and 4.2). Current candidates are only so-called approximate multilinear maps, i.e. the number of operations that can be preformed is limited, because of a random "noise" in the representations of encoded elements. Furthermore all current solutions require a trusted setup to create public parameters from secret values, the leakage of which leads to the complete break of the so-called graded encoding scheme (GES), which is directly implied by multilinear maps. These facts and the known attacks draw attention to the still unsolved questions of security and efficiency of this primitive.

Special Purpose Obfuscation. In this survey we focus on general-purpose obfuscation, although here we mention some results for specific functionalities. In fact, one possible way to evade the negative results of [BGI+01] is to restrict the scope of functionalities to be obfuscated. E.g., a quite simple, but still useful function family, called "point-functions" can be used for access control. $f_\alpha(x)$ is a point function if it returns 1 iff $x = \alpha$, and 0 otherwise (note the analogy with password verification). In fact, the obfuscation of this functionality is straightforward with a random oracle $\mathcal{R}$: $\mathcal{R}(\alpha)$ is stored instead of α and the evaluation is done through the comparison of $\mathcal{R}(\alpha)$ and $\mathcal{R}(x)$. Lynn et al. [LPS04] base their obfuscation of "access automaton" on this observation. Wee [Wee05] obfuscated point functions without a random oracle, under general assumptions in the standard model.

To represent the diversity of obfuscated functionalities, we mention some other works: [CRV10] showed how to obfuscate functions that check membership in a hyperplane of constant dimension, [And08] proposed obfuscation for Deterministic Finite Automata, [AW07] for vote mixing, [HRSV11] for re-encryption, [BR14a] for d-CNF formulas, and [BMSZ15] for evasive functions.

White-Box Cryptography and Obfuscation. Traditional (black-box) attack models in cryptography (eg. CPA, CCA) assume the end points of communication to be trusted. On the contrary, white-box cryptography (WBC) deals with protecting cryptographic primitives embedded in a program to which an attacker has white-box access, i.e. the attacker can inspect the code, modify the execution environment or even the code. This scenario especially resembles to the one that obfuscation tries to address, although while obfuscation attempts to hide certain characteristics (e.g. secret keys) of a program independently of an application, WBC specifically focuses on hiding embedded secret keys in software implementations of cryptographic primitives.[7] Generally, WBC is a strict attack model for cryptographic primitives, in which obfuscation might be used. Theoretical foundation for WBC was proposed by Saxena et al. [SWP09], although the security of white-box implementations is still very unclear. Up to now, all of these practical implementations (the first of which was shown by Chow et al. [CEJVO03] for fixed key DES) were broken. For more details on WBC, we refer to [Wys09].

About This Work. To the best of our knowledge, no works attempted to to summarize the state of the art of obfuscation since the breakthrough of Garg et al. [GGH+13b]. While Gentry [Gen14] gave a general introduction to encrypted computation (rather focusing on homomorphic encryption), we would like to introduce the results of the research that was inspired by the first candidate obfuscator by [GGH+13b]. As a first step, we introduce the most important definitional approaches of obfuscation in section 2. Section 3 is dedicated to the construction of [GGH+13b] that serves as a foundation of subsequent works in the topic. Recent attempts towards improving the security and efficiency of obfuscation, are discussed in section 4 and 5 respectively. In a nutshell, we introduce the difficulty of the application of indistinguishability obfuscation in section 6 and conclude the results in section 7.

[5] On the recent development of FHE and obfuscation see the survey of Gentry [Gen14].

[6] The first, straightforward application is a one-round n-way Diffie-Hellman key exchange.

[7] For more details on the relationship of the two notions we refer to [SW08]that tries to formalize a distinction between "useful" and "useless" non-black-box information.

2 Definitional Approaches

Capturing the intuitive goals of obfuscation in formal definitions turns out to be rather tricky. Informally speaking, we expect three properties from an obfuscator $\mathcal{O}$ that takes as input a program (or circuit) P and outputs a new program $\mathcal{O}(P)$. First, $\mathcal{O}(P)$ should preserve the functionality of P. Second, the efficiency of $\mathcal{O}(P)$ should remain comparable to the original program P. And third, we expect that $\mathcal{O}(P)$ is "unintelligible" for anyone, even for those who run the program. While the first two of these are fairly straightforward to formulate, the "unintelligibility" property can be captured in various ways.

The theoretical investigation of obfuscation was initiated by Barak et al. in [BGI+01] where several approaches were proposed. We introduce these and others in this section. From now on, we are going to use Boolean circuits as the model of computation unless stated otherwise.

2.1 Virtual Black-Box Obfuscation

The strongest meaningful theoretical notion of obfuscation security requires from an obfuscated program to behave like a "virtual black box", i.e. anything that can be computed from the obfuscated program (using its source code as well), that could also be computed merely from its input-output behaviour. In this sense, an attacker with access to such an obfuscated program is required to be indistinguishable from a simulator with access merely to the input-output pairs of the program.

Definition 1 (VBB Obfuscation [BGI+01][8]) *An algorithm $\mathcal{O}$, which takes as input a circuit C from a $\mathcal{C}$ circuit family and outputs a new circuit, is said to be a virtual black-box obfuscator for the family $\mathcal{C}$, if it has the following properties:*

- Preserving Functionality: *There exists a negligible function $\mathsf{neg}(n)$, such that for any input length n, for any $C \in \mathcal{C}_n$:*
$$\mathbb{P}[\exists x \in \{0,1\}^n : \mathcal{O}(C)(x) \neq C(x)] \leq \mathsf{neg}(n)$$
The probability is over the coins of $\mathcal{O}$ and the random oracle.
- Polynomial Slowdown: *There exists a polynomial p, such that for every circuit C, $|\mathcal{O}(C)| \leq p(|C|)$.*
- Virtual Black-box: *For any polynomial size circuit adversary $\mathcal{A}$, there exists a polynomial size simulator circuit $\mathcal{S}$ and a negligible function $\mathsf{neg}(n)$ such that for every input length n and every $C \in \mathcal{C}_n$:*
$$|\mathbb{P}[\mathcal{A}(\mathcal{O}(C)) = 1] - \mathbb{P}[\mathcal{S}^C(1^n) = 1]| \leq \mathsf{neg}(n)$$
where the probability is over the coins of the adversary, the simulator and the obfuscator. In the presence of a random oracle, the probability is also taken over the random oracle.

While this definition seems to capture the intuitive goal quite naturally, Barak et al. [BGI+01] showed that we cannot hope to realize it in general. A bit more precisely, they managed to show a program family that is strongly not-learnable, but at the same time, when given as input to any obfuscator, the original program can always be efficiently recovered from the output of the obfuscator[9].

This result rules out the possibility of constructing a general purpose obfuscator according to Definition 1, although other still interesting program families might be obfuscatable even if currently we are not able to determine these families.

Note that the *VBB* definition requires only the existence of the corresponding simulator $\mathcal{S}$ for a given $\mathcal{A}$, but does not say anything about how hard it is to find $\mathcal{S}$. Bitansky et al. [BCC+14] avoids this weakness by requiring the existence of an efficient transformation from an adversary to its corresponding simulator (or equivalently the existence of a universal probabilistic polynomial time (PPT) $\mathcal{S}$ capable of simulating any PPT $\mathcal{A}$). Somewhat counter-intuitively, they also showed that *VBB* with universal simulator is also impossible for function families with super-polynomial pseudo-entropy[10] if a weaker notion of obfuscation, indistinguishability obfuscation (see subsection 2.3) is possible in general.

We also mention a relaxation of *VBB*, namely virtual grey box (*VGB*) obfuscation, defined by Bitansky and Canetti [BC14], that allows the simulator to use unbounded computation time, while still allowing only polynomially many queries to the oracle.

[8] Here we adopt the the formulation of the definition, given by [GR07].

[9] The negative result was strengthened by [GK05] in a setting where the adversary, which is given the obfuscated circuit, may have some additional a priori information (an auxiliary input), that can either depend on the obfuscated functionality or not.

[10] Informally, a function family $\mathcal{F}$ has super-polynomial pseudo-entropy if it is hard to distinguish a function $f \in \mathcal{F}$ from f' that has been randomly modified in some locations.

2.2 Best-Possible Obfuscation

In order to avoid the negative results for *VBB*, Goldwasser and Rothblum [GR07] captured what is the maximum we can hope for when trying to achieve obfuscation. They argue that if any information exposed by $\mathcal{O}(C)$ is also exposed by every other, functionally equivalent circuit of similar size then $\mathcal{O}(C)$ is the best-possible obfuscation.

Definition 2 (Best-Possible Obfuscation [GR07]) *An algorithm $\mathcal{O}$ that takes as input a circuit in $\mathcal{C}$ and outputs a new circuit is said to be a (computational / statistical / perfect) best-possible obfuscator for the family $\mathcal{C}$, if it has the preserving functionality and polynomial slowdown properties as in Definition 1, and also has the following property (instead of the VBB property).*

- Computationally / Statistically / Perfect Best-Possible Obfuscation*: For any polynomial size learner $\mathcal{L}$, there exists a polynomial size simulator $\mathcal{S}$ such that for every large enough input length n, for any circuit pair $C_1, C_2 \in \mathcal{C}_n$ that compute the same function, such that $|C_1| = |C_2|$, the two distributions $\mathcal{L}(\mathcal{O}(C_1))$ and $\mathcal{S}(C_2)$ are (respectively) computationally / statistically / perfectly indistinguishable.*

Although this definition allows $\mathcal{O}(C)$ to leak non black-box information when *VBB* obfuscation is not possible, the best-possible obfuscation is essentially also *VBB* whenever it is achievable, so these definitions coincide in this case.

2.3 Indistinguishability Obfuscation

Besides the negative results, [BGI+01] also suggested two weaker definitions that avoid the *VBB* paradigm, but still can lead to making programs "unintelligible" in some meaningful and precise way. These definitions, namely indistinguishability (iO) and differing-input obfuscation (diO), require that if two circuits of similar size compute the same function, then their obfuscations should be indistinguishable.

Definition 3 ($i\mathcal{O}$ [BGI+01][11]) *An indistinguishability obfuscator is defined in the same way as a VBB obfuscator, except that the VBB property is replaced with the following:*

- Computational / Statistical / Perfect Indistinguishability*: For all large enough input lengths, and for any circuit pair $C_1, C_2 \in \mathcal{C}_n$ that compute the same function, such that $|C_1| = |C_2|$, the two distributions $\mathcal{O}(C_1)$ and $\mathcal{O}(C_2)$ are computationally / statistically / perfectly indistinguishable.*

Barak et al. showed that it is simple to realize inefficient iO:

Example 1 ([BGI+01]) *Let $\mathcal{O}(C)$ be the lexicographically first circuit of size $|C|$ that computes the same function as C.*

From this simple construction we can easily see the biggest hurdle of the iO definition: the lack of an intuitive guarantee that obfuscation hides information. [GR07] overcame this limitation by proving that for efficient obfuscators, the definitions of iO and "best-possible" obfuscation are equivalent. With this, the technically more easily usable iO notion, can be applied with the strong intuitive security guarantee of the "best-possible" definition, which implies that if a functionality is *VBB* obfuscatable, then *any* indistinguishability obfuscator for this functionality is *VBB* secure.

A natural strengthening of the iO notion was formulated recently by Bitansky et al. [BCKP14]. An obfuscator is a *strong* iO (siO) for class $\mathcal{C}$ if $\mathcal{O}(C) \sim \mathcal{O}(C')$ whenever the pair (C, C') is taken from a distribution over $\mathcal{C}$ where, for all x, $C(x) \neq C'(x)$ only with negligible probability. [BCKP14] also proves that siO is in fact equivalent to *VGB* obfuscation.

Later on, we are going to focus on the first iO candidate of Garg et al. [GGH+13b], and review further research that was motivated by this work.

2.4 Differing-Input / Extractability Obfuscation

Informally speaking, differing-input obfuscation (diO) for a class of circuits $\mathcal{C}$ guarantees that if an adversary $\mathcal{A}$ can distinguish between obfuscations $\mathcal{O}(C), \mathcal{O}(C')$ of two circuits $C, C' \in \mathcal{C}$, then $\mathcal{A}$ can efficiently recover (given C and C') a point x on which C and C' differ: i.e., $C(x) \neq C'(x)$. Note that if C and C' are equivalent circuits, then no such input exists, thus requiring obfuscations of the circuits to be indistinguishable (and so diO implies iO).

Intuitively the diO notion seems to be only slightly stronger than iO, however [GGHW14] showed, under the assumption that a specific special-purpose obfuscation exists, that general-purpose diO with general auxiliary input cannot exist. Using this result, [BP14] rules out diO in the absence of auxiliary input. However, on

[11] We adopt the formulation of the definition, given by [GR07].

the positive side, Boyle, Chung and Pass [BCP14] demonstrated that indistinguishability obfuscation directly implies a weak form of the differing-input notion, in which extraction of the input is only required when the pair of circuits differ on only polynomially many inputs. [IPS15] also proposes a new *diO* variant, called public-coin *diO*, that requires the auxiliary input to be a public random string. Unlike in case of standard *diO*, it still remains plausible that existing *iO* candidates also satisfy the public-coin *diO* notion.

Finally, we mention the framework of [BST14], which aims to capture different notions of *iO* and *diO* in a comparable way and also allows to define weaker, but still realizable *diO* notions (by altering the requirements on the auxiliary input).

3 The First Candidate for General Purpose *iO*

After a short introduction on the necessary background, in this section, we give an overview of the first candidate for general purpose *iO* based on Garg et al. [GGH+13b]. The final goal is achieved in two steps. First, they proposed a construction for logarithmic depth circuits[12] in $\mathbf{NC}^1$, and then bootstrapped it with the help of FHE (the reasons for this method are investigated in section 5).

3.1 Preliminaries

Matrix Branching Programs. [GGH+13b] uses "oblivious linear branching programs" as the underlying computational model, so we introduce these in a nutshell.

In this model, a branching program (BP) consists of a sequence of steps (permutation matrices), where in each step, one input bit is examined and one of two permutations is chosen depending on its value. Finally, all these chosen permutations are multiplied, and the output is 1, if the resulting permutation is the identity and 0 otherwise (in general, any pair of permutations π_1, π_2 can be specified to represent the outputs 0 and 1).

Definition 4 (Matrix Branching Program (MBP) [GGH+13b]) *Let $A_0, A_1 \in \{0,1\}^{5\times 5}$ be two distinct permutation matrices. An (A_0, A_1) branching program of length n for ℓ bit inputs is a sequence*

$$BP = (\mathsf{inp}(i), A_{i,0}, A_{i,1})_{i=1}^{n},$$

where $\mathsf{inp}(i) : [n] \to [\ell]$ is the input bit position examined in step i, and the $A_{i,b}$'s are permutation matrices in $\{0,1\}^{5\times 5}$. The function computed by this branching program is

$$f_{BP,A_0,A_1}(x) = \begin{cases} 0 & \text{if } \prod_{i=1}^{n} A_{i,x_{\mathsf{inp}(i)}} = A_0 \\ 1 & \text{if } \prod_{i=1}^{n} A_{i,x_{\mathsf{inp}(i)}} = A_1 \\ \perp & \text{otherwise.} \end{cases}$$

The notation of $\mathsf{inp}(i)$ was naturally extended by [GGH+13b] to a set of steps $S \in [n]$, namely $\mathsf{inp}(S) = \{\mathsf{inp}(i) : i \in S\} \subseteq [\ell]$. Conversely, for bit position $j \in [\ell]$, they denote by I_j the steps in BP that examine the j'th input bit, $I_j = \{i \in [n] : \mathsf{inp}(i) = j\}$, and let $I_J = \cup\{I_j : j \in J\}$ for $J \subseteq [\ell]$.

A BP is said to be input oblivious if its $\mathsf{inp}(\cdot)$ evaluation function only depends on the input length of the circuit, but not on the input values. Without this property, the evaluation function leaks information about BP, however it is not a problem when we want to hide only a single value (e.g. a key) in the program.

The use of matrix branching programs is made possible by the famous theorem of Barrington [Bar86], which guarantees for any depth-d fan-in-2 Boolean circuit C, the existence of an (A_0, A_1) oblivious linear branching program of length at most 4^d that computes the same function as the circuit C (where A_0, A_1 are permutation matrices of 5×5).

Multilinear Maps. An essential tool for creating obfuscators was provided by the recent breakthrough of Garg, Gentry and Halevi [GGH13a] who gave the first candidate[13] for the generalization of bilinear maps, the so-called multilinear maps. These are used as an underlying tool of "multi-linear jigsaw puzzles" in [GGH+13b] and of the more general graded encoding schemes in other constructions. Here we only concentrate on the basic properties which are utilized by the mentioned schemes.

Definition 5 (d-Multilinear Map [BS03][14]) *We say that a map $e : G_1 \times \ldots \times G_d \to G_T$ is a d-multilinear map if it satisfies the following properties:*

1. *$G_1, \ldots, G_d$ and G_T are groups of the same prime order*

[12] $\mathbf{NC}^1$ is the class of polynomial-size circuits with logarithmic depth and bounded fan-in gates.
[13] For more details on different constructions and attacks, see subsection 4.2.
[14] While [BS03] considered the symmetric case only, i.e. when $G_1 = \ldots = G_d$, we will use the asymmetric version.

2. *if* $a_1, \ldots, a_d \in \mathbb{Z}$ *and* $g_1 \in G_1, \ldots g_d \in G_d$, *then*

$$e(g_1^{a_1}, \ldots, g_d^{a_d}) = e(g_1, \ldots, g_d)^{a_1 \cdots a_d}$$

3. *map* e *is non-degenerate in the following sense: if* $g_1 \in G_1, \ldots, g_d \in G_d$ *are generators of* $G_1, \ldots, G_d$ *respectively, then* $e(g_1, \ldots, g_d)$ *is a generator of* G_T.

We always suppose that in groups $G_1, \ldots, G_d$ the discrete logarithm problem is intractable. As it plays an important role in latter obfuscation schemes we mention the case when the groups have composite order. While the [GGH13a] scheme turns out to be insecure, the [CLT13] construction over integers can support this setting as well. We defer the discussion of composite order multilinear maps and their benefits to subsection 4.1.

Graded Encoding Schemes. We attempt to give an intuition about encodings that quite naturally follow from multi-linear maps. According to [GGH13a, GGH+13b, PST14], such generalized or set-based graded encoding schemes enable those who have access to a public parameter pp and encodings $E_S^x = \texttt{Enc}(x, S)$, $E_{S'}^y = \texttt{Enc}(y, S')$ of ring elements x, y under the sets $S, S' \subset [k]$ to efficiently compute an encoding:

- $E_{S \cup S'}^{x \cdot y}$ of $x \cdot y$ under the set $S \cup S'$, as long as $S \cap S' = \emptyset$
- $E_S^{x \pm y}$ of $x \pm y$ under the set S, as long as $S = S'$.

Given just access to the public-parameter pp, generating an encoding to a particular element x may not be efficient; however, it can be efficiently done given access to a secret parameter sp. Additionally, given an encoding E_S^x where the set S is the whole universe $[k]$ - called the "target set" - we can efficiently "zero-test" encodings (check whether $x = 0$). In essence, multi-linear encodings enable computations of certain restricted set of arithmetic circuits (determined by the sets S under which the elements are encoded) and finally determine whether the output of the circuit is zero.

We note that all currently known multilinear maps are "noisy" i.e. the representations of group elements include a random error term. This causes a restriction on the possible number of operations that can be performed, because the error terms increase after adding or multiplying them, so in case of an unlimited number of operations the noise would overwhelm the signal. The existence of "clean" maps is still an open question and a positive answer would entail significant improvement in the efficiency of obfuscation (see subsection 5.1).

3.2 *iO* Candidate for Shallow Circuits

Obfuscating $\mathbf{NC}^1$ circuits is the core of current constructions for obfuscation. In this part we reproduce the first construction from [GGH+13b] with some simplifications and for further details refer to the original work. As suggested above, we assume that the Boolean circuit of log depth that we would like to obfuscate is already transformed to an oblivious linear branching program of length n (that is polynomial in depth-d of the circuit). [GGH+13b] proceed roughly in the following steps:

1. Randomize the matrices of BP with the technique of Kilian [Kil88]: the i'th matrix is randomized by enveloping it with random matrices R_{i-1} and R_i^{-1}. Intuitively it guarantees that the matrix product makes sense only in the right order, given in the BP.

2. "Multiplicative bundling" is applied to prevent arbitrary deviation from the matrix choices, determined by the input bits. More specifically, we avoid an attack that might choose matrices corresponding to bits 0 and 1 as well on steps of BP that consider the same input bit. This is prevented by bundling the steps of BP that correspond to the same input bit by multiplying the matrices with random scalars such that these randomness cancel out only when all matrices were used that belong to a particular input bit.

3. By adding randomized bookends we evade partial evaluation attacks, that occur when only a part of the BP is evaluated on different inputs and the results are compared.

4. Encode the matrices to prevent algebraic attacks. Roughly speaking, we encrypt the BP in a way that allows its homomorphic evaluation. Graded encoding or multi-linear jigsaw puzzles are used in this step, but as a simplification, we view the encoding of the i'th step of BP as an exponentiation in group G_i with the representation of the corresponding matrix in the exponent, i.e. $\texttt{Enc}_{\{i\}}(A_{i,x_{\mathsf{inp}(i)}}) = g_i^{A_{i,x_{\mathsf{inp}(i)}}} \in G_i$. As a result, instead of matrix multiplication, the output is computed by applying the multi-linear map e to all $g_i^{A_{i,x_{\mathsf{inp}(i)}}}$ with respect to the input x. We note that this is the only step of obfuscation that relies on a cryptographic hardness assumption.

More precisely, let $\mathbb{Z}_p$ be the ring of prime order over which we randomize the branching program. A randomized branching program is generated as follows:

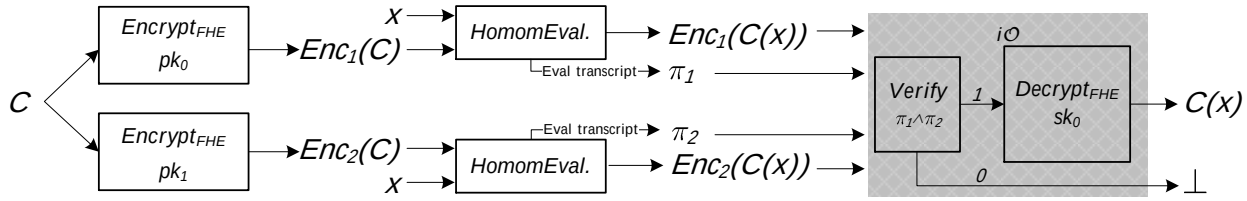

Figure 1: Bootstraping $i\mathcal{O}_{\mathbf{NC}^1}$ to gain $i\mathcal{O}$ for all polynomial size circuits.

- Sample random and independent scalars $\{\alpha_{i,0}, \alpha_{i,1}, \alpha'_{i,0}, \alpha'_{i,1} : i \in [n]\}$ from $\mathbb{Z}_p$, subject to the constraint that $\prod_{i\in I_j} \alpha_{i,0} = \prod_{i\in I_j} \alpha'_{i,0}$ and $\prod_{i\in I_j} \alpha_{i,1} = \prod_{i\in I_j} \alpha'_{i,1}$ for all $j \in [\ell]$.
- For every $i \in [n]$, compute the matrices (where I is identity):

$$D_{i,b} = \alpha_{i,b} A_{i,b}, \;\; D'_{i,b} = \alpha'_{i,b} I \tag{1}$$

- Choose two pairs of random 5-vectors $\mathbf{s}$ and $\mathbf{t}$, and $\mathbf{s}'$ and $\mathbf{t}'$, such that $\langle \mathbf{s}, \mathbf{t} \rangle = \langle \mathbf{s}', \mathbf{t}' \rangle$.
- Sample $2(n+1)$ random full-rank 5×5 matrices over $\mathbb{Z}_p$: $R_0, R_1, \ldots R_n$ and $R'_0, R'_1, \ldots R'_n$ and compute their inverses.
- The randomized branching program over $\mathbb{Z}_p$ is the following:

$$\mathcal{RND}_p(BP) = \left\{ \begin{array}{c} \tilde{\mathbf{s}} = \mathbf{s} R_0^{-1}, \tilde{\mathbf{t}} = R_n \mathbf{t}, \\ \tilde{\mathbf{s}}' = \mathbf{s}' (R'_0)^{-1}, \tilde{\mathbf{t}}' = R'_n \mathbf{t}', \\ \{\tilde{D}_{i,b} = R_{i-1} D_{i,b} R_i^{-1}\}_{i\in[n], b\in\{0,1\}}, \\ \{\tilde{D}'_{i,b} = R'_{i-1} D'_{i,b} (R'_i)^{-1}\}_{i\in[n], b\in\{0,1\}} \end{array} \right\}$$

  which consists of two parallel programs: one embeds the original BP with all the $A_{i,b}$'s and the other embeds a "dummy program" of the same length, consisting only of identity matrices (so it computes the constant function 1). The latest is used for equality test: the original program outputs 1 (on a given input) only when it agrees with the dummy program on that input.

- Encode $\mathcal{RND}_p(BP)$ to get the public output of the obfuscator:

$$\widehat{\mathcal{RND}}_p(BP) = \left\{ \begin{array}{c} \hat{\mathbf{s}} = \mathtt{Enc}_{\{1\}}(\tilde{\mathbf{s}}), \hat{\mathbf{t}} = \mathtt{Enc}_{\{n+2\}}(\tilde{\mathbf{t}}), \\ \hat{\mathbf{s}}' = \mathtt{Enc}_{\{1\}}(\tilde{\mathbf{s}}'), \hat{\mathbf{t}}' = \mathtt{Enc}_{\{n+2\}}(\tilde{\mathbf{t}}'), \\ \{\widehat{D}_{i,b} = \mathtt{Enc}_{\{i+1\}}(\tilde{D}_{i,b})\}_{i\in[n], b\in\{0,1\}}, \\ \{\widehat{D}'_{i,b} = \mathtt{Enc}_{\{i+1\}}(\tilde{D}'_{i,b})\}_{i\in[n], b\in\{0,1\}} \end{array} \right\}$$

  The output of $\widehat{\mathcal{RND}}_p(BP)$ on input x is determined by an equality test in G_T. If the output of the original and dummy BPs are identical, the overall output is 1, otherwise 0.

3.3 Bootstrapping $i\mathcal{O}_{\mathbf{NC}_1}$ for All Circuits

The common property of homomorphic encryption and obfuscation is that both hide information about function evaluation. At the same time, the basic difference is that while the output of homomorphic evaluation is encrypted, obfuscation should give the same result as the computation in clear form. Next, we show how [GGH+13b] uses this relation to obtain iO for all polynomial-size circuits from $i\mathcal{O}_{\mathbf{NC}^1}$, introduced previously.

Now we assume to have access to fully homomorphic encryption (FHE) with $\mathbf{NC}^1$ decryption circuit[15] and also use the fact that any polynomial-time circuit computation can be verified by a low-depth circuit[16]. The key step towards achieving the notion of *indistinguishability* obfuscation via FHE is to utilize a *double* key paradigm.

Let the obfuscation of circuit C (of poly size) be the following: we choose and publish two FHE keys pk_0 and pk_1 together with the encryptions of C under these keys, respectively $\mathtt{Enc}_{pk_0}(C)$ and $\mathtt{Enc}_{pk_1}(C)$. The obfuscation also contains the $i\mathcal{O}_{\mathbf{NC}^1}$ of the supplemented decryption circuit C_{Dec_0} of FHE.

The evaluation of $i\mathcal{O}(C)$ on input x is depicted on Figure 1. First, using the homomorphic evaluation function $\mathtt{Enc}_{pk_0}(C(x))$ and $\mathtt{Enc}_{pk_1}(C(x))$ are computed together with low depth proofs π_0, π_1 that prove that

[15] E.g. [Gen09] or [GSW13] fulfils this requirement.
[16] Find details on this in [GGH+13b, Appendix B].

the previous ciphertexts were computed correctly. Next, both the ciphertexts and proofs are fed to the obfuscated $\mathcal{O}_{\mathbf{NC}^1}(C_{Dec_0})$ circuit, which first verifies that both π_0 and π_1 are valid proofs. In case of positive result, using sk_0, $\mathcal{O}_{\mathbf{NC}^1}(C_{Dec_0})$ decrypts $\mathtt{Enc}_{pk_0}(C(x))$ and outputs $C(x)$, otherwise stops. Note that without the proofs of valid evaluation, the decryption algorithm could be used to get back C.

The resulting method is indeed an indistinguishability obfuscation, because denoting with C_{Dec_1} a functionally equivalent circuit with C_{Dec_0}, which uses sk_1 instead of sk_0 for decryption (but identical otherwise), by definition $i\mathcal{O}_{\mathbf{NC}^1}(C_{Dec_0}) \sim i\mathcal{O}_{\mathbf{NC}^1}(C_{Dec_1})$. As sk_1 is never used in C_{Dec_0}, the semantic security of FHE is maintained, and by alternating between C_{Dec_0} and C_{Dec_1} using the indistinguishability obfuscation property, it can be proved that the obfuscation of any two equivalent circuits C and C' are computationally indistinguishable.

For the formal description and more details about bootstrapping, we refer to [GGH+13b] once again.

4 The Security of Obfuscation

The difference between our knowledge about the security of core obfuscators (for $\mathbf{NC^1}$ circuits) and about the security of bootstrapping these algorithms is as big as the difference in the underlying tools. The security of bootstrapping[17] is based on the security of the core obfuscator, and on the security of the underlying FHE scheme. This latter is already based on strong and well-defined assumptions, such as the learning with errors (LWE) assumption (see [GSW13]) so the security of bootstrapping depends only on the obfuscation of the decryption circuit.

As current constructions of core obfuscators makes critical use of multilinear maps, their security is inherently depends on this other primitive as well. Being as recent as obfuscation candidates, cryptographic multilinear maps and their security has not yet been understood sufficiently, thus the impacts of new attacks [CHL+15, GHMS14, CLT14] and countermeasures [GGHZ14, BWZ14, CLT15] all affect our view on the security of obfuscation. Following the terminology of Badrinarayanan et al. [BMSZ15], we divide our discussion about core obfuscators into two parts: in the first part we introduce results that are persuasive, supposed that the underlying graded encoding scheme resist attacks, called "zeroizing". In the second part, we consider the case when this assumption is not taken for granted anymore.

4.1 Pre-Zeroizing Obfuscation

The argument of the above described iO candidate's security has a significant heuristic component. Garg et al. [GGH+13b] construct a model that they call "generic colored matrix model" to isolate and analyse those attacks that respect the matrix structure and order. Informally speaking, this model considers attacks where the adversary is provided with some matrices, the order of which is specified by assigning left and right colors for each matrix. The adversary is allowed to add matrices with matching colors and multiply those, where the right color of one matches the left color of the other. [GGH+13b] introduces the following assumption and shows that it holds in this model:

Assumption 1 (Equivalent Program Indistinguishability - informal [GGH+13b]) *For two different ways of fixing some inputs to a branching program that result in the same branching program on the remaining, non-fixed inputs, it is infeasible to decide which of the two sets of fixed inputs is used in a given obfuscated program.*

The used model and assumption reflect that the security proof rather gives an intuition of security than a convincing evidence, as the model is fairly restrictive and the assumption, on the one hand, is closely related to the problem, and on the other, not well-established in the sense that its hardness was not a subject of previous research. The subsequent works [BR14b, BGK+14, MSW14, PST14, GLSW14] aimed to improve the construction and its security in various ways. In this section we introduce these results.

4.1.1 Results in Generic Algebraic Models

Brakerski and Rothblum [BR14b] improved the first candidate general purpose obfuscator of [GGH+13b] achieving virtual black-box security in the generic graded encoding scheme model[18] under the Bounded Speedup Hypothesis (BSH) and indistinguishability obfuscation without BSH. Barak et al. [BGK+14] managed to remove this assumption and show that purely algebraic attacks cannot break the security of the candidate obfuscator. This means that the task of achieving security in the plain model is reduced to obtaining strong enough security guarantees on the used instantiation of multilinear maps.

[17] For different bootstrapping methods see subsection 5.2.
[18] It is also called generic multilinear map model.

Generic Model. Security in the generic multilinear or graded encoding model guarantees resistance against a large class of natural algebraic attacks (except zeroizing, if it is possible). Intuitively speaking, the generic multilinear model imagines an exponential-size collection of groups, used by a graded encoding scheme (see subsection 3.1), from some of which the adversary is initially given some collection of elements. However, the only way these elements can be processed is through access to an oracle that performs the operations of the graded encoding scheme and provides the adversary with the resulting values. We recall the security notions of *iO* and *VBB*, now *in the generic multilinear model*, stated in comparable language[19]:

Definition 6 (iO/VBB in the generic multilinear model - informal) Indistinguishability obfuscation / virtual black-box obfuscation *requires (besides the functionality and slowdown requirements) that for every polynomial-time generic adversary, there exists a* computationally unbounded / polynomial-time *simulator, such that for every circuit C, no polynomial-time generic distinguisher can distinguish the output of the adversary, given the obfuscation of C as input, from the output of the simulator given oracle access to C, where the simulator can make an* unbounded / polynomial *number of queries to C.*

As it can be seen, the confine between *iO* and *VBB* turns out to be the efficiency of simulation. [BR14b, BGK+14, MSW14] managed to construct efficient simulators (using different assumptions) and achieve the *VBB* notion in the generic model. At first glance, these results seem to contradict with the negative results of [BGI+01], so before introducing the ideas of these works, we elaborate on their possible interpretations.

First of all, we note that the counterexample of [BGI+01] does not apply to such idealized models as the random oracle or generic multilinear map models[20]. The reason for this is that in these models, the obfuscated circuit does not have a succinct, explicit description, which would be necessary for the [BGI+01] adversary to execute the obfuscated circuit on parts of its own explicit description. As a result, the attack that provides the negative result fails.

We can view generic model *VBB* obfuscators as a criticism of these idealized models, because *VBB* is known to be unrealisable for all circuits in the standard model, thus reflecting that there exist some attacks that are impossible for an idealized adversary, but which can be exploited in reality. This duality pose a particularly interesting question drawn up by Canetti, Kalai and Paneth [CKP15], namely: what are the simplest and minimally-structured abstract models that allow for general-purpose *VBB* obfuscation? To gain a better understanding, [CKP15] propose a different idealised model in which *VBB* obfuscation is still impossible.

Another straightforward implication of generic model results is that if multilinear maps are implemented in a secure hardware, thus realizing the generic model in practice, then *any* program can be protected on that specific hardware, regardless their functionality.

Towards Security in the Generic GES Model. The construction of [BR14b] was inspired by the core obfuscator of [GGH+13b] and the obfuscator for $d-$CNFs in [BR14a]. Here we highlight the main alterations from the construction, described in subsection 3.2 and their consequences.

The first conceptual difference compared with [GGH+13b] (regarding the *iO* perspective) is the way of using randomized generators of groups corresponding to the levels of graded encoding. More precisely each matrix $A_{j,b}$ of the branching program is encoded in group G_j, relative to a *unique* generator of the group, denoted by $g_{j,b}$. As the different generators are used in the same group (depending on the represented input bit), these must also be attached besides the encodings of the matrices. The role of this modification is to allow the simulator to isolate those multilinear expressions (computed by the adversary) that belong to relevant and consistent inputs. This solution leads to a computationally unbounded simulator for obfuscation fulfilling the requirements of the *iO* notion.

However, to achieve an efficient simulator (for proving *VBB* security), further difficulties must be handled. Particularly, an attacker might be able to efficiently find a multilinear expression that corresponds to the evaluation of super-polynomially many consistent inputs at the same time. As a polynomially bounded simulator cannot evaluate the function on super-polynomially many inputs, this would break the obfuscator's security.

To prevent such an attack, [BR14b] build on the "randomizing sub-assignments" technique from [BR14a] to bind the variables together into triples. This is done by adding $\binom{n}{3}$ supplementary groups, denoted G_{bind_T}, where $T \in \binom{[n]}{3}$, i.e. one for each lexicographically ordered triple of input-bit indices. The group G_{bind_T} is associated with the triple of variables $\{i_1, i_2, i_3\}$, and contains 8 pairs of encodings:

$$\left(g_{\text{bind}_T,b_1,b_2,b_3}, \left(g_{\text{bind}_T,b_1,b_2,b_3}^{\beta_{\text{bind}_T,i_1,b_1}\cdot\beta_{\text{bind}_T,i_2,b_2}\cdot\beta_{\text{bind}_T,i_3,b_3}}\right)\right)_{b_1,b_2,b_3\in\{0,1\}^3}$$

where the β's are random constants (just like $\alpha_{i,b}$'s in equation (1)). The bits of each specific input x determine one of these 8 encodings, which must be used in the evaluation. Roughly speaking, now the adversary not only

[19] For the equivalence of Definition 3 and the (informal) *iO* definition here, in the generic model, see the proof of Lemma 2.9 in [BR14b].

[20] For an interesting overview of these idealized models we refer to the work of Koblitz and Menezes [KM15].

needs to consistently choose the value of every single variable, but also to jointly commit to the values of each triple consistently with its choices for the singleton variables. [BR14b] proves that if a polynomial adversary is able to produce an expression that corresponds to a sum of super-polynomially many consistent evaluations, then it can also evaluate a 3-SAT formula on super-polynomially many values simultaneously. However, it would contradict with the BSH assumption, formulated by [BR14a], which is a strengthening of the long-standing exponential time hypothesis[21] (ETH):

Assumption 2 (Bounded Speedup Hypothesis [BR14a][22]) *There exists $\epsilon > 0$ such that, for every subset $\mathcal{X}$ of $\{0,1\}^n$, any circuit C that solves SAT on all inputs in $\mathcal{X}$ must have size at least $|X|^\epsilon$.*

In order to avoid the need for this vague[23] assumption, Barak et al. [BGK+14] proposed a core obfuscator that achieve *VBB* security in the generic graded encoding model *without any further assumptions.*

Generic Security without Further Assumptions. The method of [BGK+14] differs from [GGH+13b, BR14b] in two aspects: the way of handling mixed and partial input attacks and the usage of $\{\alpha_{i,b}\}_{i\in[n],b\in\{0,1\}}$. While in previous works of [GGH+13b, BR14b], the mentioned attacks were circumvented independently from the encoding of the branching program, [BGK+14] embed these countermeasures into the encoding. In order to merge the previously distinct steps, they use set-based graded encodings (described in subsection 3.1) together with specially designed set systems, which they call "straddling set system" (later strengthened by Miles et al. [MSW14]):

Definition 7 (*(Strong)* Straddling Set System - *([MSW14])*,[BGK+14]) *A* (strong) *straddling set system with n entries is a collection of sets $\mathbb{S}_n = \{S_{i,b} : i \in [n], b \in \{0,1\}\}$ over a universe U, such that $\cup_{i\in[n]} S_{i,0} = \cup_{i\in[n]} S_{i,1} = U$, and the following holds:*

- *Collision at universe: If $C, D \subseteq \mathbb{S}_n$ are distinct non-empty collections of disjoint sets such that $\cup_{S\in C} S = \cup_{S\in D} S$, then $\exists b \in \{0,1\}$ such that $C = \{S_{i,b}\}_{i\in[n]}$ and $D = \{S_{i,1-b}\}_{i\in[n]}$.*

(• Strong intersection: For every $i, j \in [n], S_{i,0} \cap S_{j,1} \neq \emptyset$.)

We provide a straightforward example for such set system in order to make the idea clear:

Example 2 ([BGK+14][24]) *Over the universe $U = \{1,2,3,4,5\}$, let $\mathbb{S}_3$ straddling set system be the following:*

$$S_{1,0} = \{1\}, S_{2,0} = \{2,3\}, S_{3,0} = \{4,5\},$$

$$S_{1,1} = \{1,2\}, S_{2,1} = \{3,4\}, S_{3,1} = \{5\}.$$

In other words, there are only two exact covers of the universe in a stradling set system and these are either the "zero sets" i.e. $\cup_{\forall i} S_{i,0}$ or the "one sets" i.e. $\cup_{\forall i} S_{i,1}$. It follows that, if the matrices of a branching program, which correspond to the same input bit $j \in [\ell]$, are encoded in groups under the same straddling set system (denoted by $G_{S^j_{i,b}}$ for some $i \in [n]$ and $b \in \{0,1\}$), then input mixing would involve elements that were encoded in groups under non-disjoint sets. However, in this case, multiplication is not possible in graded encodings.

Enhancing this set system by creating interlocking sets allows [BGK+14] to evade partial evaluation attacks as well. Such attacks would help the adversary to find out whether some steps of the branching program evaluate to the same permutation regardless of the value of other input bits of x. *Dual input branching programs* (diBP) differ from our definition of branching programs (see Definition 4.), in that the permutation matrices A_{i,b_1,b_2} inspect two input bits, $\mathsf{inp}_1(i), \mathsf{inp}_2(i) \in [\ell]$ in step i instead of one. When encoding the steps of a diBP, two input bits have to be considered, thus the i^{th} matrix is encoded in a group under the union of sets: $S^{\mathsf{inp}_1(i)}_{i,b_1} \cup S^{\mathsf{inp}_2(i)}_{i,b_2} := S(i,b_1,b_2)$. This tricky solution can substitute [BR14b]'s binding groups and the BSH assumption, while still preventing the adversary from creating polynomials that combine terms corresponding to a super-polynomial set of different inputs.

The second deviation from [GGH+13b, BR14b] lies in the use of $\alpha \in \mathbb{Z}_p \setminus \{0\}$ variables that randomize the diBP matrices. Indeed, the above idea supersedes the role of these values in enforcing consistency. Even though they still serve an other purpose: to provide "per-input" randomization in polynomial terms, created by the adversary. This role remains necessary in the security proof when simulating zero-testing queries.

[21] ETH states that no sub-exponential time algorithm can resolve the satisfiability of 3-CNF formulas. For more details, see [IP99].

[22] Here we use the formulation of [BGK+14].

[23] Indeed, according to a footnote in [MSW14], which refers to some personal communication, Uriel Feige has shown that the BSH is false, using a SAT-solver based attack.

[24] About the general version of this example see [BGK+14, Appendix A]

The Case of Unrestricted GES. The previously described solutions are all fairly restrictive in terms of the allowed operations, $+, -, \times$ and IsZero. Particularly $\pm$ is only possible when the elements have the same index-set S, $\times$ when the index-sets are disjoint, and IsZero if the index-set is the whole universe U. Miles, Sahai and Weiss [MSW14] refer to this setting as "fully-restricted" GES. However, assuming that operations violating the restrictions reveal no useful information on the encoded elements seems to be too optimistic, especially in the context of recent attacks [CHL+15, GHMS14, BWZ14, CLT14] (for details see subsection 4.2.). [MSW14] investigate the capabilities of obfuscation in less restrictive models, which they call arithmetic setting. This allows the adversary to perform additions across different "levels" of the encoding. In fact the same is allowed by the underlying multilinear maps [GGH13a, CLT13], even if it was not captured in previous GES models. [MSW14] proposes two relaxed models along with different results in them.

First, they extend the results of [BGK+14] (unconditional generic *VBB* security) to the "multiplication-restricted" GES model, in which $\pm$ and IsZero are always allowed, while $\times$ is allowed only with elements under non-intersecting index sets. According to their terminology, this corresponds to the case when the adversary is limited to operations that result in a "valid" multilinear polynomial.

In the "unrestricted" GES model, all arithmetic operations are allowed, even if resulting in invalid polynomials, although these always have to be classified as "non-zero". In this setting, *iO* is proven secure unconditionally, while *VBB* security is proven under a new assumption, which is a parametrized version of the BSH that replaces 3-SAT with the decision version of Max-2-SAT. Interestingly, [MSW14] also reflects that any unconditional proof of *VBB* security in this latter model would entail proving the algebraic analogue of $P \neq NP$.

From the techniques of [MSW14], we only highlight the use of strong straddling set systems (Definition 7), which was followed e.g. in the subsequent work of [BMSZ15], which captures an even broader class of algebraic attacks, including zeroizing (although, not for a general-purpose obfuscator).

4.1.2 Towards Security in the Standard Model

While the previously introduced security arguments were valid in specific generic models, our final objective is to base *iO* security on a reasonable assumption (such as the LWE assumption) via reduction in the standard model, considering arbitrary, computationally bounded adversaries. [GGH+13b] based the security of their construction to Assumption 1 (which holds in their generic model), although it embeds the actual programs to be obfuscated, tying the assumption to the scheme, thus security follows directly, without reduction.

Assumption of Semantically-Secure Multilinear Encodings. This problem was first addressed by Pass, Seth and Telang [PST14], who investigated whether the security of *iO* can be reduced to a succinct and general assumption on the underlying graded encodings. They showed an $i\mathcal{O}_{\mathbf{NC^1}}$ construction, based on the existence of *constant-message semantically secure multilinear encodings*. Intuitively, their definition of semantic security requires that the encodings of $\mathbf{m_0}$ and $\mathbf{m_1}$ constant-length vectors of elements (under the sets $\mathbf{S}$) must be indistinguishable in the presence of encodings of "auxiliary" elements $\mathbf{z}$ (under sets $\mathbf{T}$), supposing that $\mathbf{m_0}, \mathbf{m_1}, \mathbf{z}$ are sampled from any "valid" distribution[25].

[PST14] introduced a relaxed notion of *iO*, called *neighbouring-matrix indistinguishability obfuscation* (*nm-iO*), which requires the indistinguishability of only those obfuscated branching programs that are functionally equivalent and differ only in a constant number of matrices. About a simplified variant of [BGK+14]'s scheme, they prove that it satisfies the *nm-iO* notion, based on semantically secure graded encodings, and justify that the considered message distribution is "valid" with the help of the generic security analysis. To achieve the *iO* notion, [PST14] shows a general technique that transforms any *nm-iO* scheme to *iO* without any further assumptions. Roughly speaking, this is done with the help of a merging procedure, that takes two MBPs and a bit b and outputs $\mathsf{Merge}(BP_0, BP_1, b)$, an MBP with doubled width and with so-called "switch" matrices at the beginning and the end of the product. This merged MBP evaluates BP_b (through the dependence of the "switch" matrices on b) and the possible outputs of the merging of BP_0 and BP_1 only differ in a constant number of matrices. Using this tool and *nm-iO*, indistinguishability obfuscation is achieved as follows: first the circuit C_0 (to be obfuscated) is transformed into BP_0, then merged with a dummy branching program that computes the constant 1, and finally the output is the *nm-iO* of this merged branching program as $i\mathcal{O}(C)$. Indistinguishability of two obfuscated, functionally equivalent $BP_0 \sim BP_1$ (corresponding to circuits $C_0 \sim C_1$) is proven through a sequence of hybrid experiments, in each step of which, the dummy BP is replaced matrix by matrix in the following manner:

$$BP_0 = \mathsf{Merge}(BP_0, I, 0) \sim \ldots \sim \mathsf{Merge}(BP_0, BP_1, 0) \sim \mathsf{Merge}(BP_0, BP_1, 1) \sim \ldots$$
$$\ldots \sim \mathsf{Merge}(BP_1, BP_1, 1) \sim \mathsf{Merge}(BP_1, BP_1, 0) \sim \ldots \sim \mathsf{Merge}(BP_1, I, 0) = BP_1.$$

[25] Almost the same assumption was used by [BCKP14] to show an *siO* core obfuscator construction and thus virtual grey-box obfuscation for $\mathbf{NC^1}$ circuits.

The neighbouring-matrix indistinguishability property guarantees that the obfuscation of the above branching programs are indistinguishable in each step of the transition, thus $i\mathcal{O}(C_0) \sim i\mathcal{O}(C_1)$[26].

Falsifiability of Assumptions. Reduction to a particular assumption gives us the intuition of security when we cannot be sure that the assumption does not hold. As a consequence, it is natural to expect that there exists a constructive way to demonstrate that an assumption is false, if this is indeed the case, i.e. the assumption is efficiently falsifiable with the terminology of [Nao03]. The above described assumption of semantic security, however supposed to hold only for $\mathbf{m_0}, \mathbf{m_1}, \mathbf{z}$, which are sampled from a "valid" distribution D. In [PST14], this means that *every* circuit C, enabled by multilinear encodings (see subsection 3.1.) over $(\mathbf{m}_b, \mathbf{z})_{b\in\{0,1\}} \leftarrow D$ must be constant with overwhelming probability. Unfortunately, checking whether a D is valid requires checking all possible circuits, allowed by multilinear map operations, which, in case of encrypting MBPs, would require checking whether BP_0 and BP_1 are functionally equivalent, implying that semantic security is not an efficiently falsifiable assumption. Indeed, it can be viewed as an exponential size class of assumptions: one for each "valid" message distribution.

To overcome this issue, [PST14] use the merge procedure of [BCP14]: for two $\mathbf{NC^1}$ circuits C_0, C_1 (with at most n-bit inputs) and $t \in \{0,1\}^n$, $\widehat{\mathsf{Merge}}(C_0, C_1, t)$ is a circuit that on input x outputs $C_0(x)$ if $x \geq t$ and $C_1(x)$ otherwise, allowing transition from C_0 to C_1 while changing the functionality in each step on at most one input. The notion of neighbouring-input iO (ni-iO) relaxes iO in the style of nm-iO, requiring indistinguishability only for obfuscated versions of $\widehat{\mathsf{Merge}}(C_0, C_1, t)$ and $\widehat{\mathsf{Merge}}(C_0, C_1, t+1)$ if they are functionally equivalent. The equivalence of these circuits can be easily checked (i.e. efficiently falsified) by testing whether $C_0(t) = C_1(t)$. [PST14] shows that their construction satisfies this relaxed notion as well, relying on a *single* (thus efficiently falsifiable) assumption, i.e., semantic security with respect to a *single* distribution over sets and message distributions corresponding of uniformly selected programs of $\widehat{\mathsf{Merge}}(C_0, C_1, t), \widehat{\mathsf{Merge}}(C_0, C_1, t+1)$, where C_0, C_1, t are random. Finally, exponentially-secure ni-iO implies iO, which is shown by a hybrid argument over t through 2^n hybrids (that is why exponential security is necessary).

Multilinear Subgroup Elimination Assumption. The drawbacks of assuming semantic security with respect to distributions over elements that describe obfuscation include that this assumption is not independent from obfuscation, it does not have a simple description and it is not even natural at all. The first steps towards a security reduction to a more desirable assumption were made by Gentry, Lewko, Sahai and Waters [GLSW14] after the nice solution of [GLW14] in the field of witness encryption. Both of these works use an assumption about subgroup elimination in the composite order multilinear setting, which makes no reference to obfuscation or its underlying structure, neither explicitly nor implicitly.

Assumption 3 ((μ,ν)-Multilinear Subgroup Elimination (MSE) [GLW14]) *MSE concerns a μ-linear group G of order $N = a_1 \cdots a_\mu b_1 \cdots b_\nu c$, where the elements of the product are $\mu + \nu + 1$ distinct primes. Generators $g_{a_1}, \ldots, g_{a_\mu}, g_{b_1}, \ldots, g_{b_\nu}$ are given out for each prime order subgroup except for the subgroup of order c: For each $i \in [\mu]$, a group element h_i is also published, which is sampled uniformly at random from the subgroup of order $ca_1 \cdots a_{i-1}a_{i+1} \cdots a_\nu$. The assumption requires the indistinguishability of elements $T, T' \in G$ sampled uniformly at random, T from the subgroup of order $ca_1 \cdots a_\mu$ and T' from the subgroup of order $a_1 \cdots a_\mu$.*

To handle the critical hybrid step of switching the underlying computation between the functionally equivalent programs, [PST14] rely on the connection between the assumption and obfuscation itself. [GLSW14] isolate the critical computation to a single input through the decomposition of obfuscation into 2^n variations, causing a security loss of 2^n. The authors argue that this loss is inherent if the assumption is not automatically false when feeding the reduction with functionally non-equivalent programs, which is necessarily the case of instance-independent assumptions[27].

To obtain iO, [GLSW14] introduces two lower levels of abstractions that they call input-activated obfuscation (iaO) and positional indistinguishability obfuscation (piO). The structure of their framework is the following: they construct an iaO scheme using symmetric, composite order multilinear groups by running in parallel several (somewhat simplified) instances of generic model constructions [GGH+13b, BR14b, BGK+14] in different subgroups, and prove that its security follows from the MSE assumption. As a second step, they show how to build piO from iaO, and finally use this in a simple hybrid argument to obtain standard indistinguishability obfuscation.

The definition of piO relaxes iO in the spirit of ni-iO (but does not mention "merged" BPs): it takes programs P_0, P_1 and a position t to partition the input space. $pi\mathcal{O}(P_0, P_1, t)(x)$ is an obfuscated program evaluating $P_0(x)$ for $x \geq t$ and $P_1(x)$ otherwise. The required security properties are the indistinguishability of the

[26] To simplify our description, we do not mention several difficulties here, for further details see [PST14].

[27] The reduction must confirm the equivalence of programs which takes $O(2^n)$ time, as otherwise the assumption could be efficiently broken using the reduction.

following obfuscations: $pi\mathcal{O}(P_0, P_1, 0) \sim pi\mathcal{O}(P_0, P_0, 0), pi\mathcal{O}(P_0, P_1, 2^n) \sim pi\mathcal{O}(P_1, P_1, 2^n)$ and if $P_0(t) = P_1(t)$ then $pi\mathcal{O}(P_0, P_1, t) \sim pi\mathcal{O}(P_0, P_0, t+1)$. These properties allow similar hybrid steps that we discussed above, leading to standard iO.

Input-activated obfuscation is a rather new notion of obfuscation. It aims to obfuscate a special data structure, that consists of an $n \times \ell \times 2$ matrix M with entries in $\{0,1\}$ and an ordered set of ℓ programs $P_1, \ldots, P_\ell$ associated with the columns of M. The possible combinations of matrices and programs are constrained by the following properties: Each column of M defines a Boolean function $f_j : \{0,1\}^n \to \{0,1\}$ describing on which inputs the j'th column is active. The evaluation of the obfuscated structure on input x must result the same output as $P_j(x)$ for all such $j \in [\ell]$ for which $f_j(x) = 1$. Roughly speaking, iaO requires two types of security guarantees: first, "small" localized changes in M must be indistinguishable when the affected columns are associated with the same programs and the affected entries fulfil some further requirements. Second, a program P_j can be changed to P'_j when it is inactive (the associated $f_j \neq 1$ for all x) or when $f_j = 1$ on a single input x, where $P_j(x) = P'_j(x)$.

The construction of piO from iaO is done by embedding P_0, P_1, t into the data structure of iaO. Matrix M is formed as the concatenation of three pieces: M_L^{t-1} with columns associated to P_0, M_R^t with columns associated to P_1, and a "scratch" column S, initially associated to P_0. The required behaviour of piO is achieved by setting $f_j = 0$ for all Boolean functions corresponding to columns of M_L^{t-1} on inputs $x < t$ and $f_j = 1$ on inputs $x \geq t$. In other words, M_L^{t-1} is activated only on inputs $x \geq t$. Analogously M_R^t is activated only on inputs $x < t$. The necessary security properties of piO are proven using the security guarantees of iaO in a hybrid fashion.

Note that besides providing a reduction to a natural, instance-independent assumption, the above framework does not stick to any particular program description, e.g. branching programs (which are actually used by [GLSW14]), but it could be reused in case of a different (maybe more efficient) form of program description. While it is still unknown how it would be possible to emulate the key features of multilinear maps and obfuscation using only a classic assumption such as the LWE assumption, the technique of [GLSW14] seems to be a suitable starting point for this research.

Before the end of this part, we mention some properties of composite order multilinear maps, which were first used for the purposes of obfuscation by [GLSW14].

Composite Order Multilinear Groups. Similarly to the bilinear case, multilinear maps on composite order groups turn out to be much more powerful, but less efficient at the same time, than the prime order variant. Roughly speaking, this extra power can be originated from the structure of these groups. According to the Chinese remainder theorem, an encoding of an integer in $\mathbb{Z}_N$ for $N = p_1 \cdot p_2 \cdots p_k$, where the elements of the product are distinct (co)primes, is implicitly an encoding in the $\mathbb{Z}_{p_1} \times \cdots \times \mathbb{Z}_{p_k}$ direct product. Following [Zim15], we denote an encoding of $x \in \mathbb{Z}_N$ under the index set S by $[x_1, \ldots, x_k]_S$, where $x \equiv x_i \mod p_i$ for each $i \in [k]$. An encoding corresponds to zero iff it encodes zero in every component (i.e. $\forall i : x_i \equiv 0 \mod p_i$), and similarly, addition and multiplication act componentwise as well. This property allows us to store information in all components, enforcing the performance of operations on each of them. Furthermore, in the absence of the factorization of N, an adversary cannot easily eliminate one component of an encoded value without the elimination of all. Gentry et al. [GLW14, Appendix B] showed how to adapt the [CLT13] construction securely to the composite order setting, in both the symmetric and asymmetric cases.

4.2 Post-Zeroizing Obfuscation

Being the most important underlying cryptographic primitive of general purpose obfuscation, graded encoding (and thus multilinear mapping) plays a key role in the security of obfuscation. In this, different plaintext elements are encoded at different levels, and at the top level, it can be tested whether an element encodes zero or not. Unfortunately, the available candidates for multilinear maps [GGH13a, CLT13] are not yet clear-out, although, for a while, it seemed plausible that the different security arguments for obfuscation are not affected by the attacks on graded encodings. The reason for this is that the so-called "zeroizing" attacks rely on low-level encodings of zero and none of the proposed obfuscator candidates require such low-level encodings of zero to be given to the adversary. However, the recent works of Gentry et al. [GHMS14], Boneh et al. [BWZ14] and Coron et al. [CLT14] revealed that zeroizing attacks are possible when top-level encodings of zero can be created, even if no low-level encodings of zero are available for the adversary. This is an alarming result, because although there is still no known attack on current obfuscators [BR14b, BGK+14, AGIS14, Zim15, MSW14], this type of attack is not captured by any of the current security arguments. The generic proofs seemed to be plausible as in related cases, like in the simpler generic bilinear group model, the bilinear structure allowed to separate attacks based on the utilized properties, e.g. specific properties of the instantiations of groups from the properties of the actual construction. The possibility of zeroizing revealed that, in the multilinear case, the analogous generic model fails to rule out real attacks, which are related to the instantiation of the multilinear mapping. Similarly, the security proofs in the standard model [PST14, GLSW14] both reduce to assumptions that are known to be false due to the new zeroizing attacks.

What is the perspective of obfuscation then? We see three different directions that have been emerging to solve the issue.

Improvement of Graded Encodings. The first is straightforward: immunize the candidate multilinear maps against such attacks, allowing the use of previous assumptions.

Garg, Gentry, and Halevi [GGH13a] introduced the zeroizing attack and observed that the decision linear (DLIN) and subgroup membership assumptions are not satisfied in their own construction based on ideal lattices, although the graded computational and decisional Diffie-Hellman assumptions remain plausible. Later, in [GHMS14, Section 4] it was shown that a weak-discrete-logarithm attack is still possible even after some countermeasures by [GGH+13b], that prevents the adversary to obtain low-level encodings of zero. A variant of the [GGH13a] scheme by Chunsheng [Chu15] tries to eliminate the encoding of zero in order to avoid the attack.

Cheon et al. [CHL+15] demonstrated that zeroizing attacks are devastating in case of the [CLT13] scheme over integers, and lead to the recovery of all secret parameters. Concurrent proposals for fixing the [CLT13] scheme were suggested, soon after publishing the [CHL+15] attack, in [GGHZ14, Section 7] and [BWZ14]. Both of these countermeasures attempt to remove the multiplicative structure obtained during the zero-testing procedure in Cheon et al.'s attack. As an answer, Coron et al. [CLT14] showed that both fixes are insecure against a variant of the zeroizing attack that still recover the factorization of the public modulus (thus all secret parameters) in polynomial time. In a subsequent work, Coron, Lepoint and Tibouchi [CLT15] described a new candidate multilinear map over the integers that again seems to prevent the Cheon et al. attack by making the zero-testing element non-linear in the encoding.[28]

A third variant of cryptographic multilinear maps by Gentry et al. [GGH15] is defined with respect to a directed acyclic graph[29]. This approach received less attention than the previous ones and its security is only guaranteed by withstanding the attack attempts that the authors tried to break it.

We mention that multilinear maps are also constructed using indistinguishability obfuscation by Yamakawa et al. [YYHK14] through self-bilinear maps, although this kind of approaches are less interesting when the goal is the realization of iO itself.

Refinement of Generic Models. The second possible direction that can lead to plausible security arguments for obfuscation is building models that capture the power that zeroizing offers to an attacker. The need for this kind of refined generic model was first posed in [GHMS14]. The graded encoding scheme's zero testing function was previously modelled as a Boolean function and real vulnerabilities remained hidden as successful zero-tests in candidate constructions [GGH13a, CLT13] return a full ring element and it can be exploited. Badrinarayanan et al. [BMSZ15] considered the most restrictive generic model, in which the creation of an encoding of zero, at any level, is enough for the adversary to "win". While this approach inherently rules out the zeroizing attack together with its possible future extensions, current constructions in the generic graded encoding model [BR14b, BGK+14, AGIS14, Zim15, MSW14] clearly allow more: they do not prohibit the adversary to construct top-level encodings of zero that may be unrelated to any honest function evaluation. In their restricted model, [BMSZ15] examines so-called evasive functions, for which it is hard to find a 0 output. They manage to achieve that a top-level encoding of zero can be created if and only if the output is zero, so for this easy special case of evasive functions, they manage to avoid top-level zeros, thus prove security in their new model.

We note that the main technical barrier is a key element of former constructions: Kilian's statistical simulation. Each previous security argument first somehow isolated the adversary's view of the obfuscation to a single input, after which Kilian's theorem [Kil88] provided the assurance that this view only encoded information about the output of the computation within the iterated product of randomized elements of the branching program, and nothing more. However, Kilian's statistical simulation theorem does not guarantee that an encoding of zero will not be obtained, regardless what the circuit's output is on some input, because it only allows simulation. To get around this barrier, [BMSZ15] replaces Kilian's theorem and uses a matrix randomization that is equivalent to the one used when applying Kilian's randomization. Furthermore, they introduce an analysis of polynomials over graded encodings.

In order to prove security of the obfuscation of more general functions in this restrictive model, it is essential to gain better understanding of the information that is provided by the zero-testing procedures. Especially, as it is inevitable to handle zero outputs when they can occur.

Basing Obfuscation on Different Primitives. Secure obfuscation can also be achieved through completely avoiding graded encodings and basing obfuscation on strong cryptographic foundations. Although the substi-

[28] In the same work Coron et al. also propose a candidate fix of the [GGH13a] scheme, without a detailed security analysis.

[29] More precisely the encoded values are associated with paths in the graph, and it is only possible to add encoding relative to the same paths, or to multiply encodings relative to connected paths.

tution of such an extremely useful tool might be as hard as making graded encoding secure, there are already some attempts in this direction.

Recently, several works have made progress in this direction utilizing public-key functional encryption (FE) to construct iO. In an FE scheme, functional secret keys FSK_f are generated for some function f by the owner of the master secret key (MSK). From an encryption of input x, $f(x)$ can be computed using FSK_f, but everything else about x remains hidden.

Goldwasser et al.[GGG+14] showed that the indistinguishability notion of multi-input secret-key FE[30] unconditionally implies indistinguishability obfuscation.

Bitansky and Vaikuntanathan [BV15] achieved iO from sub-exponentially secure, public key FE with succinct ciphertexts. The intuition behind their work somewhat resembles to the bootstrapping idea based on fully homomorphic encryption. Let us start the interpretation from private-key functional encryption. According to [BS15], in this setting, it is always possible to guarantee function hiding property by taking advantage of message hiding. It means that the obfuscation of any circuit can be reduced to the obfuscation of the encryption function. Indeed, a function-hiding FE gives the desired output of obfuscation, provided that the input is first transformed into FE ciphertext and this encryption procedure is already hidden by obfuscation. For this latter problem, [BV15] shows a recursive method[31] to reduce iO of an n bit input FE encryption algorithm to the iO of an $n-1$ bit input FE encryption. At the end of the recursion, only a circuit with a single input bit needs to be obfuscated, which can be done simply by returning the output, in the style of a truth table. [BV15] builds the iO construction on public-key FE and enforces function-hiding with similar techniques to those used by [BS15] in the private key setting.

In a concurrent and independent work, Ananth and Jain [AJ15] achieved similar results with a technique that they call "arity amplification" in secret-key miFE (which is also a recursive method in essence). The common drawback of the three mentioned iO constructions [GGG+14, BV15, AJ15] is that - to the best of our knowledge - none of the specific FE schemes that they require has been realised yet. In fact, the only known way to obtain the necessary FE schemes with the needed ciphertext compactness properties is based on iO itself (e.g. FE by [GGH+13b]), which implies the equivalence of the two primitives.

Canetti and Vaikuntanathan [CV13] outlined a different approach and showed how to *VBB* obfuscate branching programs using a highly idealized black-box group model over pseudo-free groups.[32] Unfortunately, in this case, we end up with the previous obstacle: there are no known candidates for concretely instantiating these groups.

Even if current solutions without graded encodings are lacking concrete realization of the needed primitives, these approaches open important frontiers in achieving secure indistinguishability obfuscation.

5 The Problem of Efficiency

In order to be applicable in practice, the slowdown caused by obfuscation must be kept as small as possible. To give an idea about the performance of the previously described schemes, we draw attention to the first implementation of obfuscation by Apon et al. [AHKM15]. The most complex function that they obfuscated was a 16-bit point function containing 15 AND gates. The process took 9 hours and resulted in an obfuscated program of 31.1 GB size, the evaluation of which on a single input takes around 3.3 hours on a machine with 32 cores and 244 GB RAM. This results supersedes any further attempt to describe the distance between theory and practice even in case of such simple functionalities as the mentioned one[33]. In this section, we investigate the sources of inefficiencies in previous constructions and introduce the current approaches towards practically usable methods.

As the construction of obfuscators are built in two distinct steps, their efficiency also have to be examined according to these. First, we concentrate on core obfuscators (for circuits in $\mathbf{NC^1}$), then turn our attention to methods that enable the obfuscation of any polynomial sized circuits.

5.1 Efficiency of Core Obfuscators

Challenges. The reason why we need to bootstrap our core obfuscators is that applying any of these methods directly to circuits requires an exponentially growing overhead, depending on the circuit depth. This is rooted on

[30] As the name suggests, miFE allows functional keys to correspond to multi-input functions which can be evaluated on tuples of ciphertexts.

[31] This recursive application of FE leads to the constraint that the ciphertext size must be polynomial in the input size.

[32] Informally, a black-box group is an algebraic group adjoined with a random representation of group elements. The group in this case is assumed to be a pseudo-free group i.e. a group that is indistinguishable from a free group by any PPT adversary. A free group is an infinite group defined by a set of generators $A = \{a_1, \dots a_n\}$. The elements of this group are all the words that use the symbols in A, along with their inverses, that are also treated as formal symbols.

[33] Bernstein et al. [BHLN15] showed several techniques to speed up the evaluation, and in fact they managed to broke the "point obfuscation challenge" (announced at the Crypto 2014 rump session [AHKM15]) in just 19 minutes using a cluster of 21 PCs.

two facts: (1) The necessary levels of graded encodings grow exponentially with the depth d of the circuit being obfuscated, which is problematic as the complexity of currently used multilinear maps [GGH13a, CLT13] grows polynomially with the level of multilinearity. Furthermore, these "noisy" maps are limited to a predetermined number of operations on the encoded elements before the noise overwhelms the signal, inhibiting the zero-testing procedure. (2) The very first step of each previously introduced constructions [GGH+13b, BR14b, BGK+14, MSW14, PST14, GLSW14] is to convert the circuit into a matrix branching program (MBP), using Barrington's theorem [Bar86]. This guarantees that an arbitrary fan-in-2, depth-d Boolean circuit can be transformed into a matrix branching program of length $n \leq 4^d$, causing an exponential overhead again.

These constraints imply that circuits might not be the best approach to represent a program which we would like to obfuscate efficiently.

The Burden of Barrington and Kilian Theorems. When we examine the previously introduced constructions with an eye looking for squandering steps, the first thing that we observe is the overhead of Barrington's [Bar86] transformation of circuits to MBP. Indeed, this conversion takes place before the concrete steps of obfuscation, thus, it is natural to ask whether it is possible to minimize the MBP size before obfuscating it? The question is crucial, especially as the length of MBP also determines the necessary levels of multilinearity in the used graded encoding scheme. A straightforward attempt is to "balance" the depth of the formula or circuit before Barrington's transformation. The best we can achieve in this direction is depth $d = 1.82 \log s$ following [PM76] (where s is the length of the Boolean formula), although it still means a bound of $s^{3.64}$ on the length of BP.

To eliminate this inefficiency, Ananth et al. [AGIS14] proposed the first solution, without needing to invoke Barrington's theorem. They considered Boolean formulas as the representation of programs, which are fan-out-1 Boolean circuits. [AGIS14] introduces the notion of "relaxed matrix branching program", that replaces permutation matrices by general *full-rank* matrices over a finite field and zero-tests some fixed entry in the matrix product in order to get the output. The advantage of this relaxed notion is twofold: Boolean formulas can be transformed into relaxed MBP without Barrington's method and it is still sufficient to adapt the [BGK+14] construction and security argument.

To do such transformation, Giel [Gie01] showed a technique which guarantees that a formula of any complete basis can be converted into a relaxed matrix branching program of size $O(s^{1+\epsilon})$ where the width of each matrix is a constant depending only on ϵ, and $\epsilon > 0$ can be any constant. For the same conversion, [AGIS14] shows a different method in two steps: considering formulas of size s with AND, OR, and NOT gates, they first convert them into a layered graph-based branching program satisfying certain technical conditions, then turn this into a relaxed MBP of length $O(s)$ (consisting of $O(s)$ dimension matrices). The idea here is that the evaluation of a Boolean formula is traced back to testing whether two specific vertices are connected in the related directed graph. This latter graph connectivity problem however can be formulated as matrix multiplication, that leads us to relaxed MBPs. The only thing left is to make the relaxed MBP input oblivious, i.e., make the inp evaluation function independent of the formula (as it is in other cases) which has a multiplicative overhead of ℓ.[34] At the end, we get that the necessary levels of multilinearity is $O(\ell s + \ell^2)$ and the total size of obfuscation is $O(\ell s^3 + \ell^2)$ elements. This is better than the results after balancing the formula, but we must think of that the upper bound for s is $s \leq 2^d$, in case of gates with fan-in-2, causing that both of the previous values remain exponential in d.

In their manuscript, Sahai and Zhandry [SZ14][35] take an other step forward to reduce the size and evaluation time of obfuscation. In [AGIS14], the dimension of each matrices in the BP is $O(s)$ (i.e. it is the order of the formula size) in order to have full rank, which is essential for the use of Kilian's randomization theorem [Kil88] during the obfuscation. Full-rank matrices are invertible and [SZ14, BMSZ15] interpret this in the context of branching programs as information about the actual state (represented by the matrix) cannot be forgotten. This state information, carried by the matrices, blows up the MPB width and size, so the authors ask the natural question: is this overhead of sometime useless information necessary or is it possible to directly obfuscate programs that can "forget" (i.e. the matrices in their MBP representation are not full-rank)? The question is answered positively, by eliminating the use of Kilian's theorem[36]: [SZ14, BMSZ15] show how to obfuscate low-rank matrices achieving $\frac{1}{4} s \log_2 s^2$ elements, as the output of the obfuscator. We note that, for the first time, this method allows to output *multiple bits* without running different BPs for each output bit.

Direct Obfuscation of Circuits. Instead of improving MBPs before obfuscation, in concurrent and independent works, Zimmerman [Zim15], Applebaum and Brakerski [AB15] proposed a radically new approach: to obfuscate circuits directly. We follow [Zim15] and briefly introduce the technique used there. For simplicity, here we consider only keyed circuit families, although the results extend to all arithmetic circuits (after making them input-oblivious). Indeed, most cryptographic applications use keyed functions, which can be obfuscated more

[34] This additional overhead can be avoided when input-obliviousness is not a requirement e.g. in case of keyed functions, when the goal is to hide the key.

[35] Latter this work was extended [BMSZ15] and with the used techniques security was improved as well, see subsection 4.2.

[36] See subsection 4.2 on this.

efficiently, as there is no need to hide the structure, only the key. The evaluation of $\mathcal{O}(C(\cdot, y))$, with embedded key $y \in \{0,1\}^m$ on input $x \in \{0,1\}^\ell$ is done by performing the operations of C on encoded versions of x and y. First of all, the adversary must be prevented from evaluating an incorrect expression $C' \not\equiv C$ and acquire a bit of the secret key y. This is achieved with the help of composite-order asymmetric multilinear maps[37] which allows to check the computations of the adversary. More precisely the encodings are elements of the composite-order $\mathbb{Z}_N$, where $N = N_{\text{ev}}N_{\text{chk}}$, thus $\mathbb{Z}_N = \mathbb{Z}_{N_{\text{ev}}} \times \mathbb{Z}_{N_{\text{chk}}}$. Intuitively $\mathbb{Z}_{N_{\text{chk}}}$ serves as a "checksum" for the adversary's computation, the use of which is enforced by the fact that operations in $\mathbb{Z}_N$ act componentwise in $\mathbb{Z}_{N_{\text{ev}}}$ and $\mathbb{Z}_{N_{\text{chk}}}$. Following our previous notation, $[x_i, \alpha_i]_S$ and $[y_i, \beta_i]_S$ are used during the evaluation of a circuit $C'(x_1, \ldots, x_\ell, y_1, \ldots, y_m)$, where $\alpha_1, \ldots, \alpha_\ell, \beta_1, \ldots, \beta_m$ are uniformly random values. At the end, zero testing is possible only after subtracting a precomputed value $C(\alpha_1, \ldots, \alpha_\ell, \beta_1, \ldots, \beta_m)$ from the $\mathbb{Z}_{N_{\text{chk}}}$ component of the result. In this way, the computation is going to result zero only if the adversary honestly evaluated C, otherwise the received component in $\mathbb{Z}_{N_{\text{chk}}}$ and thus in $\mathbb{Z}_N$ are non-zero (with overwhelming probability).

In the absence of BPs, to enforce the consistency of input bits, [Zim15] needs new, although in spirit similar techniques to previous solutions. Each input bit $(\hat{x}_{1,0}, \hat{x}_{1,1}, \hat{x}_{2,0}, \ldots)$ is encoded at its own singleton index set $(X_{1,0}, X_{1,1}, X_{2,0}, \ldots)$, so the adversary can evaluate expressions of his choice, and the associated index sets will track the degree of the expression in each variable. Although, in order to reach top level encodings that can be zero-tested, the adversary is forced to incorporate "interlocking" elements into the created monomials. The index set of these $\hat{z}_{i,b}$ elements contain $X_{i,1-b}^{\deg(x_i)}$ for each bit choice $b \in \{0,1\}$ (where $\deg(x_i)$ is the degree of the variable x_i in the actual circuit C). These index sets prevent the adversary from making any input-inconsistent choices within a given monomial. To prove security, Zimmerman uses a similar generic model as [BGK+14] and straddling set systems to achieve the *VBB* notion. We note that this construction allows a straightforward extension to obfuscate circuits with *multi-bit output* without a significant increase of operations [Zim14, Remark 3.18].

The avoidance of the transformation from an arithmetic circuit to MBP results in considerable improvement in terms of obfuscation size and evaluation time, which are - for the first time - not exponential in the circuit depth d and require only $O(d^2s^2 + \ell^2)$ elements and multilinear map operations. Obfuscating keyed functions (which are in the center of interest in case of most cryptographic applications) an even better result with $O(m + \ell^2)$ elements and $O(s + \ell^2)$ operations is achieved, where m denotes the key size. We remark that this latter result is enough for the purpose of achieving *general purpose* obfuscation for all polynomial size circuit, as the FHE decryption algorithm that we need to obfuscate for this, is a keyed function.

After the elimination of Barrington's theorem, we could ask whether it is still indispensable to bootstrap the algorithm or not. Unfortunately, the necessary degree of multilinearity is still $O(2^d\ell + \ell^2)$, hence the "noise" growth during multilinear map operations does not allow the obfuscation of circuits outside $\mathbf{NC^1}$ using currently known multilinear map constructions. As [Zim15] points out: finding "clean" (and secure) multilinear maps is not only a technicality, but one of the most fundamental open problems in cryptography.

Concurrently and independently, [AB15] also achieved direct obfuscation via composite order symmetric graded encodings. While this work only achieves the iO notion in a generic model, the authors also consider a more robust model in which the adversary is allowed to zero-test elements even below the top level, and with some efficiency loss, they prove iO security in this model as well.

5.2 Inefficient Bootstrapping

As reflected above, in the absence of "clean" multilinear maps, the use of core obfuscators remains limited even after significant improvements of efficiency. To overcome this obstacle bootstrapping of $\mathbf{NC^1}$ obfuscators is inevitable. The idea of using FHE for the goals of obfuscation appeared first in [DMMQN12] by Döttling et al., although in a different context. [GGH+13b] and others also used it for bootstrapping $i\mathcal{O}_{\mathbf{NC^1}}$, but from a practical point of view, we have to note that the existence of FHE is still a strong public-key assumption and for most applications it is still not efficient enough. Therefore, it is natural to ask whether it can be relaxed or not?

The question was first considered by Applebaum [App14] who found a positive answer and showed that bootstrapping can be based on a "Minicrypt"[38] type assumption. He showed that if $\mathbf{TC}^0$ can be *VBB* obfuscated (in some idealized model), then every polynomial-size circuit family can be *VBB* obfuscated as well, using randomized encoding (RE) and a pseudo random function (PRF), the complexity of which are in $\mathbf{TC}^0$ and which are inherently far more efficient than FHE[39].

While this result applies for *VBB* obfuscation, Canetti et al. [CLTV15] and Bitansky et al.[BGT14] proved an analogous statement for indistinguishability obfuscation, assuming the existence of a sub-exponentially hard

[37] For our description on the composite-order setting see subsection 4.1.2.

[38] According to Impagliazzo's terminology [Imp95], in Minicrypt one-way functions exist, but public-key encryption is not possible

[39] $\mathbf{TC}^0 \subseteq \mathbf{NC}^1$ is the class of all Boolean circuits with constant depth and polynomial size, containing only unbounded-fan in AND gates, OR gates, and majority gates.

$i\mathcal{O}_{\mathbf{NC}^1}$ and a subexponential, indistinguishable puncturable[40] PRF in $\mathbf{NC}^1$.

6 About the Application of Indistinguishability Obfuscation

In this work, we do not aim to sum up all the fields where iO found applications. Undoubtedly, it turned out to be extremely useful since the first candidate of [GGH+13b], even if its application involves difficulties. Here, we only attempt to reflect the source of these obstacles.

Unlike the black-box definition of obfuscation, indistinguishability obfuscation and, in the efficient case, equivalent best-possible obfuscation do not quantify or qualify the information that is hidden by obfuscation, if anything is hidden (see example 1). Clearly, if the obfuscated circuit is already in an obvious canonical form, then indistinguishability obfuscation does not need to hide anything. In order to make use of the definition, it is necessary to construct circuits with the same functionality that inherently have multiple equivalent forms. We have seen an example for such a design principle in the bootstrapping of $i\mathcal{O}_{\mathbf{NC}^1}$ (see subsection 3.3.): the two key paradigm allows us to construct circuits that achieve the same functionality, but use different keys. Another idea is the "punctured programming" approach of Sahai and Waters [SW14], where a key element of the circuit should be removed, such that this alters the functionality only in a single "point". With randomization, it is possible to move the place of puncturing to a location that is accessed by the program only with negligible probability. This method allows us to construct an alternative program by removing some information from the original program, without changing the functionality in practice. In this way, we can give a characterization of the information that the indistinguishability obfuscation of a program hides.

The applications of iO include, but are not limited to public-key encryption [SW14], functional encryption [GGH+13b], witness encryption [GGSW13], deniable encryption [SW14], multi-party computation [GGHR14], replacing random oracles [HSW14], separation results [GKMZ14] and more.

7 Conclusion

In this work, we reviewed current candidate obfuscators, focusing on the utilized techniques, achieved security guarantees, and the efficiency of the schemes. As it turned out, the underlying *graded encoding scheme* serves as the main source of opportunities and the main bottleneck at the same time, both in terms of security and efficiency of the obfuscation schemes. Security concerns about multilinear maps endanger our trust in current security proofs as the most recent attacks undermine our previous view on security. Finding convincing solution to these concerns is pressing either by eliminating the flaws in current graded encodings or modifying our models and assumptions that we use to investigate the security of obfuscation. For instance, security reduction to a standard assumption could resolve the question, although currently, we do not know how to connect such assumptions with graded encodings. The third possible solution, namely basing obfuscation on a different primitive, would also be desirable as it would mean a fundamentally new approach towards obfuscation.

From the viewpoint of efficiency, the weak point is the same. As Zimmerman [Zim15] showed, using "clean" multilinear maps, all circuits could be obfuscated directly, without the need for bootstrapping. On our way towards practically usable obfuscation, we might also have to change some of our current expectations, such as our longing after *general-purpose* solution. Indeed, the obfuscation of keyed functions has significant advantage in performance and what is more, in most of the cases, we would obfuscate these functionalities (let's think of e.g. the decryption circuit of homomorphic encryption). On the basis of this observation, we think that in the future, optimizing obfuscation for different purposes (based on observations on general purpose constructions) can be the main source of efficiency gains.

Besides the clear open questions of obfuscation, another important direction of future work would be to integrate the results of efforts made in different directions. The works of [GLSW14] and [Zim15] are for example orthogonal, in the sense that the starting point of both is the construction of [BGK+14], but they improve different aspects of it. While [GLSW14] manages to reduce security on an instance-independent assumption, it does not deal with efficiency, and vica versa, [Zim15] achieves direct obfuscation of circuits, but it sticks to the generic security argument. However, it would be challenging to construct schemes, which do not focus on one specific feature, but try to achieve the best we can hope for.

Acknowledgements.

I would like to thank the support of CrySyS Lab and the help of Levente Buttyán and Ágnes Kiss, who have read and commented the drafts of this survey. I am also grateful for the remark of Ryo Nishimaki on software watermarking.

[40] For details on puncturable PRFs see [SW14]

References

[AB15] Benny Applebaum and Zvika Brakerski. Obfuscating Circuits via Composite-Order Graded Encoding. In Yevgeniy Dodis and JesperBuus Nielsen, editors, *Theory of Cryptography*, volume 9015 of *Lecture Notes in Computer Science*, pages 528–556. Springer Berlin Heidelberg, 2015.

[AGIS14] Prabhanjan Ananth, Divya Gupta, Yuval Ishai, and Amit Sahai. Optimizing Obfuscation: Avoiding Barrington's Theorem. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 646–658. ACM, 2014.

[AHKM15] Daniel Apon, Yan Huang, Jonathan Katz, and Alex J. Malozemoff. Implementing cryptographic program obfuscation. Cryptology ePrint Archive, Report 2014/779, February 10. 2015. Crypto 2014 rump session.

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. Cryptology ePrint Archive, Report 2015/173, February 27. 2015. http://eprint.iacr.org/.

[And08] W Erik Anderson. On the secure obfuscation of deterministic finite automata. *IACR Cryptology ePrint Archive*, 2008:184, 2008.

[App14] Benny Applebaum. Bootstrapping obfuscators via fast pseudorandom functions. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, pages 162–172, 2014.

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

[Bar86] David A Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in nc 1. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 1–5. ACM, 1986.

[BBC+14] Boaz Barak, Nir Bitansky, Ran Canetti, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Obfuscation for evasive functions. In *Theory of Cryptography*, pages 26–51. Springer, 2014.

[BC14] Nir Bitansky and Ran Canetti. On strong simulation and composable point obfuscation. *Journal of Cryptology*, 27(2):317–357, 2014.

[BCC+14] Nir Bitansky, Ran Canetti, Henry Cohn, Shafi Goldwasser, Yael Tauman Kalai, Omer Paneth, and Alon Rosen. The impossibility of obfuscation with auxiliary input or a universal simulator. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II*, pages 71–89, 2014.

[BCKP14] Nir Bitansky, Ran Canetti, Yael Tauman Kalai, and Omer Paneth. On virtual grey box obfuscation for general circuits. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II*, pages 108–125, 2014.

[BCP14] Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In *Theory of Cryptography*, pages 52–73. Springer, 2014.

[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 *Advances in Cryptology–EUROCRYPT 2014*, pages 221–238. Springer, 2014.

[BGT14] Nir Bitansky, Sanjam Garg, and Sidharth Telang. Succinct randomized encodings and their applications. Cryptology ePrint Archive, Report 2014/771, September 30. 2014. http://eprint.iacr.org/.

[BHLN15] Daniel J. Bernstein, Andreas Hülsing, Tanja Lange, and Ruben Niederhagen. Bad directions in cryptographic hash functions. Cryptology ePrint Archive, Report 2015/151, February 23. 2015. http://obviouscation.cr.yp.to/.

[BMSZ15] Saikrishna Badrinarayanan, Eric Miles, Amit Sahai, and Mark Zhandry. Post-zeroizing obfuscation: The case of evasive circuits. Cryptology ePrint Archive, Report 2015/167, March 11. 2015. http://eprint.iacr.org/.

[BP14] Elette Boyle and Rafael Pass. Limits of extractability assumptions with distributional auxiliary input. Cryptology ePrint Archive, Report 2013/703, October 21. 2014. http://eprint.iacr.org/.

[BR14a] Zvika Brakerski and Guy N Rothblum. Black-box obfuscation for d-CNFs. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 235–250. ACM, 2014.

[BR14b] Zvika Brakerski and Guy N Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In *Theory of Cryptography*, pages 1–25. Springer, 2014.

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

[BS15] Zvika Brakerski and Gil Segev. Function-private functional encryption in the private-key setting. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 306–324, 2015.

[BST14] Mihir Bellare, Igors Stepanovs, and Stefano Tessaro. Poly-many hardcore bits for any one-way function and a framework for differing-inputs obfuscation. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, pages 102–121, 2014.

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability Obfuscation from Functional Encryption. Cryptology ePrint Archive, Report 2015/163, February 26. 2015. http://eprint.iacr.org/.

[BWZ14] Dan Boneh, David J. Wu, and Joe Zimmerman. Immunizing multilinear maps against zeroizing attacks. Cryptology ePrint Archive, Report 2014/930, November 13. 2014. https://eprint.iacr.org/2014/930.

[CEJVO03] Stanley Chow, Phil Eisen, Harold Johnson, and Paul C Van Oorschot. A white-box des implementation for drm applications. In *Digital Rights Management*, pages 1–15. Springer, 2003.

[CHL+15] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehle. Cryptanalysis of the multilinear map over the integers. Cryptology ePrint Archive, Report 2014/906, February 24. 2015. http://eprint.iacr.org/.

[Chu15] Gu Chunsheng. Multilinear maps using ideal lattices without encodings of zero. Cryptology ePrint Archive, Report 2015/023, January 11. 2015. http://eprint.iacr.org/.

[CHV15] Aloni Cohen, Justin Holmgren, and Vinod Vaikuntanathan. Publicly verifiable software watermarking. Cryptology ePrint Archive, Report 2015/373, April 22 2015. http://eprint.iacr.org/.

[CKP15] Ran Canetti, Yael Tauman Kalai, and Omer Paneth. On obfuscation with random oracles. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 456–467, 2015.

[CLT13] Jean-Sébastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In *Advances in Cryptology–CRYPTO 2013*, pages 476–493. Springer, 2013.

[CLT14] Jean-Sebastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Cryptanalysis of two candidate fixes of multilinear maps over the integers. Cryptology ePrint Archive, Report 2014/975, November 30. 2014. http://eprint.iacr.org/.

[CLT15] Jean-Sebastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. New multilinear maps over the integers. Cryptology ePrint Archive, Report 2015/162, February 25. 2015. http://eprint.iacr.org/.

[CLTV15] Ran Canetti, Huijia Lin, Stefano Tessaro, and Vinod Vaikuntanathan. Obfuscation of probabilistic circuits and applications. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 468–497, 2015.

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

[CV13] Ran Canetti and Vinod Vaikuntanathan. Obfuscating branching programs using black-box pseudo-free groups. Cryptology ePrint Archive, Report 2013/500, August 14. 2013. http://eprint.iacr.org/.

[DGB14] Bruce Dang, Alexandre Gazet, and Elias Bachaalany. *Practical Reverse Engineering: X86, X64, ARM, Windows Kernel, Reversing Tools, and Obfuscation*. John Wiley & Sons, 2014.

[DH76] Whitfield Diffie and Martin E Hellman. Multiuser cryptographic techniques. In *Proceedings of the June 7-10, 1976, national computer conference and exposition*, pages 109–112. ACM, 1976.

[DMMQN12] Nico Döttling, Thilo Mie, Jörn Müller-Quade, and Tobias Nilges. Basing obfuscation on simple tamper-proof hardware assumptions. Cryptology ePrint Archive, Report 2011/675, January 16. 2012. http://eprint.iacr.org/.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, volume 9, pages 169–178, 2009.

[Gen14] Craig Gentry. Computing on the edge of chaos: Structure and randomness in encrypted computation. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:106, 2014.

[GGG+14] Shafi Goldwasser, S Dov Gordon, Vipul Goyal, Abhishek Jain, Jonathan Katz, Feng-Hao Liu, Amit Sahai, Elaine Shi, and Hong-Sheng Zhou. Multi-input functional encryption. In *Advances in Cryptology–EUROCRYPT 2014*, pages 578–602. Springer, 2014.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *Eurocrypt*, volume 7881, 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. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 40–49. IEEE, 2013.

[GGH15] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 498–527, 2015.

[GGHR14] Sanjam Garg, Craig Gentry, Shai Halevi, and Mariana Raykova. Two-round secure mpc from indistinguishability obfuscation. In *Theory of Cryptography*, pages 74–94. Springer, 2014.

[GGHW14] Sanjam Garg, Craig Gentry, Shai Halevi, and Daniel Wichs. On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, pages 518–535, 2014.

[GGHZ14] Sanjam Garg, Craig Gentry, Shai Halevi, and Mark Zhandry. Fully secure functional encryption without obfuscation. Cryptology ePrint Archive, Report 2014/666, November 12. 2014. http://eprint.iacr.org/.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 467–476. ACM, 2013.

[GHMS14] Craig Gentry, Shai Halevi, Hemanta K. Maji, and Amit Sahai. Zeroizing without zeroes: Cryptanalyzing multilinear maps without encodings of zero. Cryptology ePrint Archive, Report 2014/929, November 12. 2014. http://eprint.iacr.org/.

[Gie01] Oliver Giel. Branching Program Size Is Almost Linear in Formula Size . *Journal of Computer and System Sciences*, 63(2):222 – 235, 2001.

[GK05] Shafi Goldwasser and Yael Tauman Kalai. On the impossibility of obfuscation with auxiliary input. In *Foundations of Computer Science, 2005. FOCS 2005. 46th Annual IEEE Symposium on*, pages 553–562. IEEE, 2005.

[GKMZ14] Matthew D. Green, Jonathan Katz, Alex J. Malozemoff, and Hong-Sheng Zhou. A unified approach to idealized model separations via indistinguishability obfuscation. Cryptology ePrint Archive, Report 2014/863, October 27. 2014. http://eprint.iacr.org/.

[GLSW14] Craig Gentry, Allison B Lewko, Amit Sahai, and Brent Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. *IACR Cryptology ePrint Archive*, 2014:309, November 7. 2014.

[GLW14] Craig Gentry, Allison B. Lewko, and Brent Waters. Witness encryption from instance independent assumptions. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, pages 426–443, 2014.

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

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Advances in Cryptology-CRYPTO 2013*, pages 75–92. Springer, 2013.

[HMW07] Nicholas Hopper, David Molnar, and David Wagner. From weak to strong watermarking. In SalilP. Vadhan, editor, *Theory of Cryptography*, volume 4392 of *Lecture Notes in Computer Science*, pages 362–382. Springer Berlin Heidelberg, 2007.

[HRSV11] Susan Hohenberger, Guy N Rothblum, Abhi Shelat, and Vinod Vaikuntanathan. Securely obfuscating re-encryption. *Journal of cryptology*, 24(4):694–719, 2011.

[HSW14] Susan Hohenberger, Amit Sahai, and Brent Waters. Replacing a random oracle: Full domain hash from indistinguishability obfuscation. In *Advances in Cryptology–EUROCRYPT 2014*, pages 201–220. Springer, 2014.

[Imp95] Russell Impagliazzo. A personal view of average-case complexity. In *Structure in Complexity Theory Conference, 1995., Proceedings of Tenth Annual IEEE*, pages 134–147. IEEE, 1995.

[IP99] Russell Impagliazzo and Ramamohan Paturi. Complexity of k-SAT. In *Computational Complexity, 1999. Proceedings. Fourteenth Annual IEEE Conference on*, pages 237–240. IEEE, 1999.

[IPS15] Yuval Ishai, Omkant Pandey, and Amit Sahai. Public-Coin Differing-Inputs Obfuscation and Its Applications. In Yevgeniy Dodis and JesperBuus Nielsen, editors, *Theory of Cryptography*, volume 9015 of *Lecture Notes in Computer Science*, pages 668–697. Springer Berlin Heidelberg, 2015.

[Kil88] Joe Kilian. Founding crytpography on oblivious transfer. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, pages 20–31, New York, NY, USA, 1988. ACM.

[KM15] Neal Koblitz and Alfred Menezes. The Random Oracle Model: A Twenty-Year Retrospective. Cryptology ePrint Archive, Report 2015/140, February 25. 2015. http://eprint.iacr.org/.

[LPS04] Benjamin Lynn, Manoj Prabhakaran, and Amit Sahai. Positive results and techniques for obfuscation. In *Advances in Cryptology-EUROCRYPT 2004*, pages 20–39. Springer, 2004.

[MSW14] Eric Miles, Amit Sahai, and Mor Weiss. Protecting obfuscation against arithmetic attacks. Cryptology ePrint Archive, Report 2014/878, March 2. 2014. http://eprint.iacr.org/.

[Nao03] Moni Naor. On cryptographic assumptions and challenges. In *Advances in Cryptology-CRYPTO 2003*, pages 96–109. Springer, 2003.

[NW15] Ryo Nishimaki and Daniel Wichs. Watermarking cryptographic programs against arbitrary removal strategies. Cryptology ePrint Archive, Report 2015/344, April 20. 2015. http://eprint.iacr.org/.

[PM76] Franco P. Preparata and David E. Muller. Efficient parallel evaluation of boolean expressions. *IEEE Transactions on Computers*, 25(5):548–549, 1976.

[PST14] Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In *Advances in Cryptology - CRYPTO 2014*, pages 500–517. Springer, 2014.

[RAD78] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

[Sah14] Amit Sahai. How to encrypt a functionality (talk). Quantum Games and Protocols Workshop (Simons Institute, UC Berkeley), February 25. 2014. http://simons.berkeley.edu/talks/amit-sahai-2014-02-25.

[SW08] Amitabh Saxena and Brecht Wyseur. On white-box cryptography and obfuscation. *arXiv preprint arXiv:0805.4648*, 2008.

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: Deniable encryption, and more. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, STOC '14, pages 475–484, New York, NY, USA, 2014. ACM.

[SWP09] Amitabh Saxena, Brecht Wyseur, and Bart Preneel. Towards security notions for white-box cryptography. In *Information Security*, pages 49–58. Springer, 2009.

[SZ14] Amit Sahai and Mark Zhandry. Obfuscating low-rank matrix branching programs. Cryptology ePrint Archive, Report 2014/773, September 30. 2014. http://eprint.iacr.org/.

[Wee05] Hoeteck Wee. On obfuscating point functions. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 523–532. ACM, 2005.

[Wys09] Brecht Wyseur. *White-Box Cryptography*. PhD thesis, Katholieke Universiteit Leuven, 2009.

[YYHK14] Takashi Yamakawa, Shota Yamada, Goichiro Hanaoka, and Noboru Kunihiro. Self-bilinear Map on Unknown Order Groups from Indistinguishability Obfuscation and Its Applications. In *Advances in Cryptology-CRYPTO 2014*, pages 90–107. Springer, 2014.

[Zim14] Joe Zimmerman. How to obfuscate programs directly. Cryptology ePrint Archive, Report 2014/776, October 1. 2014. http://eprint.iacr.org/.

[Zim15] Joe Zimmerman. How to Obfuscate Programs Directly. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015*, volume 9057 of *Lecture Notes in Computer Science*, pages 439–467. Springer Berlin Heidelberg, 2015.

Update Log

- Original upload to ePrint Archive: April 24, 2015
- Update 1 (June 4, 2015):
  - Refinement and extension of references (with [Sah14, BBC+14, HMW07, NW15, CHV15])
  - Mistakes corrected:
    - in paragraph Generic Security without Further Assumptions: the mentioned attacks were circumvented not after, but independently from the encoding of the BP.
    - in step 4 of the description of the first iO candidate: "$B_{i,x_{\mathsf{inp}(i)}}$" was a mistake (replaced with $g_i^{A_{i,x_{\mathsf{inp}(i)}}}$)
    - in definition 7 the indices are corrected
  - Extension with Example 2
  - Corrected typos