Open access • Book Chapter • DOI:10.1007/978-3-662-47989-6_12

Zeroizing Without Low-Level Zeroes: New MMAP Attacks and their Limitations

Jean-Sébastien Coron, Craig Gentry, Shai Halevi, Tancrède Lepoint ...+6 more authors

Institutions: University of Luxembourg, IBM, University of California, Purdue University ...+1 more institutions

Published on: 16 Aug 2015 - International Cryptology Conference

Topics: Multilinear map and mmap

Zeroizing Without Low-Level Zeroes: New MMAP Attacks and Their Limitations[1]

Jean-Sébastien Coron[2] Craig Gentry[3] Shai Halevi[3] Tancrède Lepoint[4]
Hemanta K. Maji[5,6] Eric Miles[6] Mariana Raykova[7] Amit Sahai[6,8]
Mehdi Tibouchi[9]

June 15, 2015

Abstract

We extend the recent zeroizing attacks of Cheon, Han, Lee, Ryu and Stehlé (Eurocrypt'15) on multilinear maps to settings where no encodings of zero below the maximal level are available. Some of the new attacks apply to the CLT13 scheme (resulting in a total break) while others apply to (a variant of) the GGH13 scheme (resulting in a weak-DL attack). We also note the limits of these zeroizing attacks.

Keywords: Cryptanalysis, Hardness Assumptions, Multilinear Maps.

[1]This work subsumes and extends the previous works [13, 8].

[2]University of Luxembourg. jean-sebastien.coron@uni.lu

[3]IBM Research

[4]CryptoExperts. tancrede.lepoint@cryptoexperts.com. This work has been supported in part by the European Union's H2020 Programme under grant agreement number ICT-644209.

[5]Purdue University

[6]University of California, Los Angeles and Center for Encrypted Functionalities. hemanta.maji@gmail.com, enmiles@cs.ucla.edu, amitsahai@gmail.com

[7]SRI International. mariana@cs.columbia.edu. This work has been supported in part from NSF Award 1421102.

[8]Research supported in part from a DARPA/ONR PROCEED award, a DARPA/ARL SAFEWARE award, NSF Frontier Award 1413955, NSF grants 1228984, 1136174, 1118096, and 1065276, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, and an Okawa Foundation Research Grant. This material is based upon work supported by the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0389. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense, the National Science Foundation, or the U.S. Government.

[9]NTT Secure Platform Laboratories. tibouchi.mehdi@lab.ntt.co.jp

Contents

1 Introduction

The GGH13 [10] and CLT13 [7] "approximate multilinear maps" candidates suffer from *zeroizing attacks*, where encodings of zero at levels below the top (zero-test) level can be exploited to recover information that should have been hidden by the encoding scheme. The essence of these attacks is using successful zero tests to obtain equations over the base ring ($\mathbb{Z}$ or $\mathbb{Z}[X]/F(X)$), then solving these equations to get the desired information. First presented in the context of the GGH13 candidate [10], such attacks were recently extended by Cheon et al. [6] also to the CLT13 candidate, where they were shown to be particularly devastating, leading to a total break (when they can be mounted).

As explicitly discussed in [6], however, these attacks seem to depend on the availability of low-level encoding of zeros. This limits the applicability of these attacks, especially since several high-profile applications of multilinear maps (such as for obfuscation [11]) do not reveal such low-level zero encodings.

In this work we show that it is possible to "zeroize without low-level zeroes": that is, we extend the attacks from [6] and apply them against both CLT13 encodings and a matrix variant of GGH13 encodings, even in settings where no low-level encodings of zero are available to the adversary. Our extension to the attacks from [6] to avoid low-level zero encodings was also observed independently and concurrently by Boneh, Wu, and Zimmerman [4]. We further systematize the new attacks and show that they can overcome recent proposals to "immunize" against them [12, 4]. In particular, our attacks also deal with cases where more than one monomial is needed to get a zero, and with modifications of the CLT13 and GGH13 schemes that use matrix-based encodings with the encoded values embedded in the eigenvalues of the matrix. Before describing our zeroizing attacks, we discuss the impact and limitations of these attacks.

1.1 Impact of Our Attacks

Broken Assumptions and Constructions. The most direct consequence of our work is that more hardness assumptions and constructions from the literature are broken. Prior to our work, the attacks of [6] already broke several assumptions and constructions using CLT13 encodings because they provided low-level encodings of zero. Our work extends to new assumptions and constructions, even where no low-level encodings of zero are available. For example, our extensions can be used to break instances of the meta-assumption of Pass et al. [21] (using either GGH13 or CLT13 encodings), even when used without low-level encodings of zero. Furthermore, we show that natural attempts to "immunize" CLT13 or GGH13 encodings by removing low-level encodings of zero [12, 4] fail. In particular, the assumptions used by Gentry et al. [15, 14] are broken, even when "immunized" using the technique from [4]. Perhaps more surprisingly, we also show that simplified variants of certain obfuscation schemes can be broken:

- We show that the GGHRSW branching-program obfuscation procedure from [11], implemented over the CLT13 scheme [7], can be broken when it is applied to branching programs with a very specific "decomposable" structure. See Section 3.4.
- We also show that the simplified circuit obfuscation scheme of Zimmerman [22, Appendix A] and Applebaum-Brakerski [1] can be broken when applied to very simple circuits (e.g., point functions).

Generic multilinear model. Zeroizing attacks provide concrete examples of ways in which GGH13 and CLT13 encodings fail to realize the generic multilinear model. That is, they provide natural examples of information that is learnable from certain combinations of CLT13 and GGH13 encodings, but is not learnable in the generic multilinear model. Such attacks of course do not affect the validity of security proofs within the generic multilinear map model, but they illustrate flaws in the model itself when applied to GGH13 and CLT13 encodings.

However, as discussed in more detail below, zeroizing attacks have their limitations, and thus it is important to explore alternative generic models that give the adversary more power, in the hope that they will provide better approximation to the adversarial capabilities against current multilinear-map candidates. Some early attempts along this line can be found in [20, 1]. In the current work we suggest another direction for alternative generic models that would incorporate information learnable by the adversary using zeroizing attacks; see Appendix A.

Subsequent work. Following our work, Badrinarayanan et al. introduced in [2] a generic model that captures all known zeroizing attacks (including ours) by regarding the creation of any encoding of zero as a complete break of the system, and showed how to securely obfuscate evasive functions in this model.

Coron, Lepoint, and Tibouchi recently proposed in [9] a new version of the CLT13 scheme (denoted CLT15), with a modified zero-testing procedure that seems to avoid zeroizing attacks altogether. No polynomial-time attacks are currently known against this version, hence assumptions and constructions that are broken by [6] and our work when instantiated by CLT13 may still hold when using CLT15.

For GGH13, the graded decisional Diffie-Hellman assumption from [10] was broken by Hu and Jia [18], in the same settings for which the initial zeroizing attack [10] applies (i.e., in the presence of low-level encodings of zero).

1.2 Limitations of Zeroizing attacks

Potent as they are, zeroizing attacks have their limitations. For example, so far we do not have attacks on *any of the* NC^1 *obfuscation candidates in the literature*. Moreover the "dual-input straddling sets" technique of [3] and its variants that are used in several obfuscation schemes [5, 3, 22] appear to be effective in thwarting these attacks. See more details in Section 2.4.

Successful zero tests are necessary. Our work demonstrate that some attacks are possible even if we only have top-level encoded zeros, but crucially all of these attacks depend on *successful zero tests* to get equations over the base ring. Some constructions or assumptions may not provide these top-level zeros, and in that case it is plausible that the GGH13 and CLT13 candidates could even provide semantic security [16] of the encoded values. Even more, as far as we know the standard generic multilinear-map model could provide a good approximation of GGH13 and CLT13 in settings where top-level encoding of zeros are not available.

The equations must be simple. In zeroizing attacks, each successful zero-test provides the adversary one equation over the base ring, and the attack relies on the attacker's ability to solve the resulting system of equations. The successful attacks detailed in our paper (as well as those from [10, 6]) arise in situations where the adversary has *substantial freedom* in creating top-level encodings of zero, and can exploit this freedom to obtain "a simple system of equations" over the base ring that can be solved using linear algebraic techniques.

There are many cases, however, in which the available encodings are constructed such that only

very particular combinations of them yield a top-level encoding of zero, and those combinations do not seem to yield efficiently solvable system of equations. Two such examples, illustrated in Section 2.4, are obfuscation schemes that rely on Barrington's theorem, and schemes that use the "dual-input straddling sets" technique.

We believe that long-term understanding of the security offered by current multilinear map candidates will require tackling long-standing questions about which kinds of systems of nonlinear equations are feasible to solve efficiently, and which are not.

2 Background and Overview

2.1 A Brief Description of the GGH13 and CLT13 Schemes

We begin with a brief description of the GGH13 and CLT13 schemes, omitting many details that are irrelevant for the attacks in question. Both these schemes implement graded encoding schemes where "plaintext elements" are encoded in a way that hides their value but allows to add and multiply them, and also allows to test if a degree-k expression in these values is equal to zero (where k is the "multi-linearity parameter").

2.1.1 The GGH13 scheme

For GGH13 [10], the plaintext space is a quotient ring $R_g = R/gR$ where R is the ring of integers in a number field and $g \in R$ is a "small element" in that ring. The space of encodings is $R_q = R/qR$ where q is a "big integer". An instance of the scheme relies on two secret elements, the generator g itself and a uniformly random denominator $z \in R_q$. A plaintext element (which is a coset $a = \alpha + gR$) is encoded "at level one" as $u = [e/z]_q$ where e is a "small element" in the coset a (i.e., $e = \alpha + gr$ for some $r \in R$). More generally, a level-i encoding of the coset a has the form $u = [e/z^i]_q$ for a small $e \in \alpha + gR$.

Addition/subtraction of encodings at the same level is just addition in R_q, and it results in an encoding of the sum at the same level, so long as the numerators do not wrap around modulo q. Similarly multiplication of elements at levels i, i' is a multiplication in R_q, and as long as the numerators do not wrap around modulo q the result is an encoding of the product at level $i + i'$.

The scheme also includes a "zero-test parameter" in order to enable testing for zero at level k. Noting that a level-k encoding of zero is of the form $u = [gr/z^k]_q$, the zero-test parameter is an element of the form $\mathbf{p}_{\text{zt}} = [hz^k/g]_q$ for a "somewhat small element" $h \in R$. This lets us eliminate the z^k in the denominator and the g in the numerator by computing $[\mathbf{p}_{\text{zt}} \cdot u]_q = h \cdot r$, which is much smaller than q because both h, r are small. If u is an encoding of a non-zero α, however, then multiplying by $\mathbf{p}_{\text{zt}}$ leaves a term of $[h\alpha/g]_q$ which is not small. Testing for zero therefore consists of multiplying by the zero-test parameter modulo q and checking if the result is much smaller than q.

Matrix-GGH13. An unpublished variant of GGH13 (that was meant to protect against zeroizing attacks) uses matrices of native GGH13 encodings, where the encoded value is an eigenvalue of the matrix and the zero-test parameter includes also the corresponding eigenvector. This is essentially the same as the GGHZ countermeasure construction from [12, Sec. 7] (which is described in Section 3.3), except that it uses GGH13 encodings rather than CLT13 encodings.[1]

[1] Our attack from Section 3.3 applies for the most part to this GGH13 variant too, except that in this case we only get a weak-DL attack rather than a complete break; see the full version for details.

2.1.2 The CLT13 Scheme

The CLT13 scheme [7] is similar to above, but it relies on CRT representation modulo a composite integer $x_0 = \prod_{j=1}^{n} p_j$, where the p_j's are "large primes", all of about the same size. We let $\mathsf{CRT}\left(a_1, \ldots, a_t\right)$ denote the unique element $a \in \mathbb{Z}_{x_0}$ that is congruent to a_j modulo p_j for all j. Also we often use the shorthand $\mathsf{CRT}\left(a_j\right)_j$ to denote the same.[2]

The plaintext space in CLT13 consists of vectors $\mathbf{a} \in \mathbb{Z}_{g_1} \times \cdots \times \mathbb{Z}_{g_n}$, where all the g_j's are much smaller than their corresponding p_j's. An instance of the scheme relies on the secrets g_j and p_j (with x_0 public), and on a secret uniformly random denominator $z \in \mathbb{Z}_{x_0}$. Such a vector $\mathbf{a} = (\alpha_1, \ldots, \alpha_n)$ is encoded at level one as $\left[\mathsf{CRT}\left(\alpha_1 + g_1 r_1, \ldots, \alpha_n + g_n r_n\right)/z\right]_{x_0}$, where the r_j's are all small. More generally a level-i encoding of this vector is of the form $\left[\mathsf{CRT}\left(\alpha_j + g_j r_j\right)_j / z^i\right]_{x_0}$.

Addition/subtraction of encodings at the same level is just addition in $\mathbb{Z}_{x_0}$, and it results in an encoding of the sum at the same level, so long as the numerators in the different CRT components do not wrap around modulo their respective p_j's. Similarly multiplication of elements at levels i, i' is a multiplication in $\mathbb{Z}_{x_0}$, and as long as the numerators in the different CRT components do not wrap around modulo their respective p_j's, the result is an encoding at level $i + i'$ of the entry- wise product of the two vectors.

For zero-testing, let us denote $p_j^* = x_0/p_j = \prod_{i \neq j} p_i$, and note the following easy corollary of the Chinese Remainder Theorem:

Proposition 2.1. *For all $a_1, \ldots, a_n \in \mathbb{Z}$, $\mathsf{CRT}\left(p_j^* a_j\right)_j = \sum_{j=1}^{n} p_j^* a_j \pmod{x_0}$.*

Namely when each CRT component j is divisible by p_j^*, then the CRT composition can be computed just by adding all the CRT components modulo x_0.

The zero-test parameter in CLT13 is $\mathbf{p}_{\mathrm{zt}} = [z^k \cdot \mathsf{CRT}\left(p_j^* h_j g_j^{-1}\right)_j]_{x_0}$ for small elements $h_j \ll p_j$, where g_j^{-1} is computed modulo p_j. Multiplying this zero-test parameter by a level-k encoding of zero, that has the form $u = \left[\mathsf{CRT}\left(g_j r_j\right)_j / z^k\right]_{x_0}$, yields

$$\left[\mathbf{p}_{\mathrm{zt}} \cdot u\right]_{x_0} = \mathsf{CRT}\left(p_j^* h_j r_j\right)_j = \sum_j p_j^* h_j r_j.$$

Since $h_j r_j \ll p_j$ for all j, then $p_j^* h_j r_j = (x_0/p_j) h_j r_j \ll x_0$, and also the sum is much smaller than x_0. Testing for zero therefore consists of multiplying by the zero-test parameter modulo x_0 and checking if the result is much smaller than x_0.

2.1.3 Common Properties

The GGH13 and CLT13 schemes share a very similar structure; here we summarize the common features that are used in the attacks:

- Each encoding is "associated" with the vector of small integers in the numerator. For GGH13 this is a 1-vector consisting of a single algebraic integer,[3] and for CLT13 this is a vector of n integers in $\mathbb{Z}$. Below we write informally $u \sim (a_1, \ldots, a_n)$ to denote the fact that the encoding u is associated with the vector of a_i's. Roughly speaking, the goal of the attacks is to recover the vector $(a_j)_j$ from the encoding u. Recovering this vector (even if not in full) is usually considered a break of the scheme.

[2] We do not assume that the a_j's are smaller than their corresponding p_j's.

[3] The matrix-GGH13 variant has vectors in the numerator rather than a single algebraic integer.

- An encoding of zero is associated with a vector divisible by the g_j's, namely $u \sim (g_j r_j)_j$ for some r_j's.
- Addition and multiplication of encodings acts entry-wise on the vector of integers in the numerator. Importantly, the addition and multiplication of these vectors is done *over the integers, with no modular reduction.* This is because a wrap-around in these operations is an error condition, and so the parameters are always set to ensure that it does not happen.
- If $u \sim (g_j r_j)_j$ is an encoding of zero at the top level, then applying the zero-test to u returns the integer $w = \sum_j r_j \rho_j$, where the r_j's are the multipliers from the numerator vector and the ρ_j's are system parameters independent of u.

  In other words, applying the zero-test to an encoding of zero yields the inner-product of the associated vector (sans the g_j's) with a fixed secret vector. (In GGH13 this is the 1-vector (h), in CLT13 the vector is $(p_j^* h_j)_j$.) Importantly, here too the inner product is over the integers, with no modular reduction.

2.2 Overview of Existing Attacks

The GGH13 Zeroizing Attack. The following "zeroizing" attack on the GGH13 scheme was described in [10]. It gets as input a level-t encoding of zero $u_0 \sim (gr)$ and many other level-$(k-t)$ encodings $u_m \sim (a_m)$. Multiplying u_0 by any of the u_m's yields a top-level encoding of zero $u_0 u_m \sim (gra_m)$, and applying the zero-test yields the algebraic integer $w_m = hra_m$. Note that this almost recovers the numerators a_m's; indeed we have them up to the common factor $h' = hr$.

If we also knew the ideal $I_g = gR$ that defines the plaintext space, then being able to recover the numerator up to a constant is enough to break many hardness assumptions. For example, given an encoded matrix we could compute its determinant (mod I_g) up to a constant, which would tell us whether or not the encoded matrix has full rank.

Even when I_g is not explicitly given, Garg et al. described in [10] how it can be recovered in certain cases using GCD computations. Roughly, we can use GCD to identify and remove the common factor h', thereby getting the a_m's themselves, except that these are all algebraic integers so we only have GCD in terms of their ideals. Recovering the ideal $I_a = aR$ is not always useful, e.g., if I_a and I_g are co-prime then knowing I_a does not tell us anything about our plaintext coset $a + I_g$. However if some of the u_i's are themselves encoding of zero, namely $a_i = gr_i$, then given enough ideals $I_{a_i} = gr_i R$ we could again use GCD calculations to recover the ideal I_g itself, and then use that knowledge to attack the non-zero encodings among the u_i's. This attack was called in [10] a "weak discrete-log attack". Recently, this attack was used by Hu and Jia [18] as a component in a new attack that breaks the key-exchange protocol from [10].

We note that the GGH13 zeroizing attack does not work against CLT13 encodings, since rather than a simple product we now have an inner product $w_m = \sum_j a_{m,j} \rho_j$, and we cannot use this to compute GCDs. (For the same reason, this attack does not work against the matrix-GGH13 variant.)

The CHLRS Zeroizing Attack. Cheon, Han, Lee, Ryu and Stehlé recently described in [6] a major upgrade of the GGH13 zeroizing attack, which can be used to completely break CLT13-based schemes in some cases, recovering the factorization of x_0 and all secret information. To mount the CHLRS zeroizing attack we need three sets of encoded inputs, which we denote by

$\mathcal{A} = \{A_i : i = 1, \ldots, n\}$, $\mathcal{B} = \{B_0, B_1\}$, and $\mathcal{C} = \{C_j : j = 1, \ldots, n\}$ (with n the dimension of the numerator vectors). The A's are all random encoding of zeros, the B's are the target of the attack, and the C's are just helper encodings of random vectors. The levels of these encodings are such that multiplying $A_i \cdot B_\sigma \cdot C_j$ yields a top-level encoding of zero for any i, σ, j. Below we denote the numerator vectors associated with these encodings by

$$A_i \sim (g_1 r_{i,1}, \ldots, g_n r_{i,n}),\ B_\sigma \sim (b_{\sigma,1}, \ldots, b_{\sigma,n}),\ \text{and}\ C_j \sim (c_{j,1}, \ldots, c_{j,n}).$$

Multiplying $A_i \cdot B_\sigma \cdot C_j$ yields a top-level encoding of zero, associated with the vector $A_i \cdot B_\sigma \cdot C_j \sim (g_1 r_{i,1} b_{\sigma,1} c_{j,1}, \ldots, g_n r_{i,n} b_{\sigma,n} c_{j,n})$. Applying the zero-test we get a four-wise inner product, yielding the integer $w_\sigma[i,j] = \sum_{k=1}^{n} \rho_k r_{i,k} b_{\sigma,k} c_{j,k}$. We can write this four-wise inner product in matrix form as

$$w_\sigma[i,j] = (r_{i,1}\ \ldots\ r_{i,n}) \times \begin{pmatrix} \rho_1 b_{\sigma,1} & & \\ & \ddots & \\ & & \rho_n b_{\sigma,n} \end{pmatrix} \times \begin{bmatrix} c_{j,1} \\ \vdots \\ c_{j,n} \end{bmatrix},$$

and denote the vector on the left by $\mathbf{a}_i$, the matrix in the middle by B'_σ, and the vector on the right by $\mathbf{c}_j$. For a fixed σ, let i, j range over $1, \ldots, n$. This yields an $n \times n$ matrix of integers $W_\sigma = [w_\sigma[i,j]]_{i,j} = A' \times B'_\sigma \times C'$, where A' has the $\mathbf{a}_i$'s for rows and C' has the $\mathbf{c}_j$'s for columns. Since the $r_{i,k}$'s, $b_{\sigma,k}$'s, $c_{j,k}$'s and ρ_k's are all random (small) quantities, then with high probability the matrices are all invertible (over the rationals). Having computed the matrices W_σ, the attacker now sets

$$W = W_0 \times W_1^{-1} = (A' B'_0 C') \times (A' B'_1, C')^{-1} = A' \times (B'_0 \times {B'_1}^{-1}) \times {A'}^{-1}.$$

Observe now that $B^* = B'_0 \times {B'_1}^{-1}$ is a diagonal matrix with $b_{0,j}/b_{1,j}$ on the diagonal, and thus the eigenvalues of B^* are all the ratios $b_{0,j}/b_{1,j}$. And since W and B^* are similar matrices, then also the eigenvalues of W are the $b_{0,j}/b_{1,j}$'s. Hence once it computes W, the attacker can find its eigenvalues (over the rationals) and obtain all the ratios $b_{0,j}/b_{1,j}$.

These ratios may be enough by themselves to break some hardness assumptions, but for CLT13 it is possible to use them to factor x_0, thereby getting a complete break. Specifically, since each ratio is rational it can be written as $u/v = b_{0,j}/b_{1,j}$ with u, v co-prime integers. Recalling now that B_0, B_1 are two encodings at the same level (say, level t) with numerator vectors $(b_{0,1}, \ldots, b_{0,n})$ and $(b_{1,1}, \ldots, b_{1,n})$, respectively, we get that

$$uB_1 - vB_0 = [\mathsf{CRT}\,(ub_{1,1} - vb_{0,1}, \ldots, ub_{1,n} - vb_{0,n})\,/z^t]_{x_0}.$$

This means that the j'th CRT component is $ub_{1,j} - vb_{0,j} = 0$, and with high probability the others are not, so we get $GCD(x_0, uB_1 - vB_0) = p_j$.

2.3 Extending the CHLRS Attack

In the current work we describe several extensions to attacks of Cheon et al. from [6]; below we describe these extensions briefly.

2.3.1 GGH13 vs. CLT13

We can also apply these zeroizing attacks to a matrix variant of GGH13, not just to CLT13 encodings, resulting in a "weak discrete-log" attack. This is described in Section 4.

2.3.2 Orthogonal encodings

We also note that these attacks do not actually require low-level encoding of zeros. Indeed all we need is that for every i, σ, j, the product $A_i B_\sigma C_j$ is a top-level encoding of zero, so we could have the A's with zeros in a few CRT components, the B's with zeros in some other components, and the C's with zeros in all the CRT components not covered by the A's and B's. This observation was also made concurrently by Boneh et al. [4].

2.3.3 More than one monomial

The attack also extends to a setting where more than a single monomial is needed to get a zero. For example, consider the case where we have not three but six sets of encodings. Similar to before we have $\mathcal{A} = \{A_i : i = 1, \ldots, 2n\}$, $\mathcal{B} = \{B_0, B_1\}$, and $\mathcal{C} = \{C_j : j = 1, \ldots, 2n\}$, but now we also have $\tilde{\mathcal{A}} = \{\tilde{A}_i : i = 1, \ldots, 2n\}$, $\tilde{\mathcal{B}} = \{\tilde{B}_0, \tilde{B}_1\}$, and $\tilde{\mathcal{C}} = \{\tilde{C}_j : j = 1, \ldots, 2n\}$. (Note that the indices i, j now range over $[1, 2n]$, not $[1, n]$). The new attack requires that $A_i B_\sigma C_j + \tilde{A}_i \tilde{B}_\sigma \tilde{C}_j$ is a top-level encoding of zero for every i, σ, j. We denote the numerator vectors associated with these encodings by

$$\begin{array}{lll} A_i \sim (a_{i,1}, \ldots, a_{i,n}), & B_\sigma \sim (b_{\sigma,1}, \ldots, b_{\sigma,n}), & C_j \sim (c_{j,1}, \ldots, c_{j,n}), \\ \tilde{A}_i \sim (\tilde{a}_{i,1}, \ldots, \tilde{a}_{i,n}), & \tilde{B}_\sigma \sim (\tilde{b}_{\sigma,1}, \ldots, \tilde{b}_{\sigma,n}), & \tilde{C}_j \sim (\tilde{c}_{j,1}, \ldots, \tilde{c}_{j,n}). \end{array}$$

We can think of the pairs $(A_i, \tilde{A}_i)$, $(B_\sigma, \tilde{B}_\sigma)$, $(C_j, \tilde{C}_j)$ as encodings that are associated with numerator vectors of twice the dimension, and the CHLRS attack can be applied to these new "double encodings". The only difference (other than the larger dimension) is that we can no longer associate the division-by-g_i with any single vector. Instead, applying the zero-test to $A_i B_\sigma C_j + \tilde{A}_i \tilde{B}_\sigma \tilde{C}_j$ yields a four-wise inner product *divided by the g_i's*, which we can write in matrix form:

$$w_\sigma[i,j] = (a_{i,1}\ \tilde{a}_{i,1}\ \ldots\ a_{i,n}\ \tilde{a}_{i,n}) \times \begin{pmatrix} \frac{\rho_1 b_{\sigma,1}}{g_1} & & & & \\ & \frac{\rho_1 \tilde{b}_{\sigma,1}}{g_1} & & & \\ & & \ddots & & \\ & & & \frac{\rho_n b_{\sigma,n}}{g_n} & \\ & & & & \frac{\rho_n \tilde{b}_{\sigma,n}}{g_n} \end{pmatrix} \times \begin{bmatrix} c_{j,1} \\ \tilde{c}_{j,1} \\ \vdots \\ c_{j,n} \\ \tilde{c}_{j,n} \end{bmatrix}.$$

Importantly, even though we have division by g_i's, this equation holds over the rationals, without modular reduction. The attack itself proceeds just as before, and the g_i^{-1} factors conveniently fall off when we compute $B'_0 \times {B'_1}^{-1}$. This extension can be used to break the "immunized" CLT13 variant from [4].

2.3.4 Using Cayley-Hamilton

In response to the CHLRS attacks, Garg et al. described in [12, Sec. 7] a variant of the CLT13 encoding that uses matrices for encoding, rather than single $\mathbb{Z}_{x_0}$ elements (see description in Section 3.3 below).

The attacks above apply also to this variant for the most part, but the resulting matrices B'_0, B'_1 are no longer diagonal. Instead they are block-diagonal with the block dimension corresponding to the dimension of the encoding matrices, and different blocks corresponding to different CRT

components (i.e. $B_\sigma \bmod p_j$). The eigenvalues of $B'_0 \times {B'_1}^{-1}$ in this case need not be rational numbers anymore, they can be arbitrary complex numbers, and so the final step in the CHLRS attack cannot be applied.

However the characteristic polynomial of $B^* = B'_0 \times {B'_1}^{-1}$ is still the product of the characteristic polynomials of the blocks. We can factor the characteristic polynomial of B^* to find the block characteristic polynomials, and then apply these block polynomials to the matrix $M = B_1 \times B_0^{-1}$. Applying a block polynomial to M zeros out the corresponding CRT component (by the Cayley-Hamilton theorem), but not the others (whp), and we can then compute the GCD of x_0 and any matrix element to recover the prime corresponding to the zeroed CRT component. Note this assumes that the block polynomials are irreducible over $\mathbb{Q}$ (which indeed holds for [12, Sec. 7]), so that they can be efficiently found by factoring B^*'s characteristic polynomial.

The actual procedure that we use differs slightly, in order to handle an unpublished generalization of [12, Sec. 7] in which the encoding matrices themselves are constructed to be block-diagonal, say with block dimension d. With this change B^* is still block-diagonal, but the block dimension is now larger by a factor of d, and each polynomial that we want to apply to M is the product of d factors of B^*'s characteristic polynomial. We do not know of a way to efficiently partition these factors into the correct sets of size d. Instead, we remove one irreducible factor from B^*'s characteristic polynomial, and apply the resulting polynomial to M. This has the effect of zeroing out all CRT components *except* the one corresponding to the removed factor, so computing the GCD with x_0 recovers the product of all but one of the primes, and dividing x_0 by this recovers an individual prime. Cycling over all irreducible factors, we recover all of the primes.

2.4 Attack Limitations

As sketched in the introduction, zeroizing attacks have their limitations, in that they require zeros and moreover need the equations that yield these zeros to be "simple." Two scenarios that seem outside the scope of these attacks due to "non-simple" equations are discussed next.

Obfuscation using Barrington's theorem. Consider the obfuscation schemes in the literature that obfuscate matrix-based branching programs (BP) resulting from Barrington's theorem [11, 5, 3, 21]. These schemes are designed so that the only way to get a top-level zero encoding is using the prescribed routines for evaluating the obfuscated circuit on various inputs, so we only need to examine the type of expressions that arise from such evaluation.

Recall that a matrix-based BP has a sequence of steps, each specified by two matrices and controlled by an input bit. On a given input, we choose one of the two matrices in each step (based on the corresponding input bit), then multiply all of the selected matrices in order to get the result. In the BPs that are generated by Barrington's theorem, each input bit controls several steps that are spaced far apart, and so changing the value of that bit changes the selection of all these matrices. This makes it hard to apply our attacks in this setting, since these attacks require a multilinear setting where we can get many different zeros by changing just a single variable in every monomial. Therefore, even though we do get equations over the base ring from top-level zeros in this scheme, these equations appear to be correlated in a highly non-linear manner, foiling our attempts to glean useful information from them. (Rather than the individual matrices, one can consider for each bit position j two tensor products of matrices corresponding to this bit position, one tensor of the matrices for value 0 and the other for the value 1. Then we can get a multilinear equations in these tensor vectors, but we do not see how to use them to mount a successful attack.)

We contrast this situation with the attack that we describe in Section 3.4, that breaks obfuscation of very simple branching programs which are "separable" in the sense that different subsets of the input bits control different consecutive intervals of steps, thus giving us the simple system of equations that we need.

Binding variables. The CHLRS attacks and our extensions rely on the ability to partition the variables into groups ($\mathcal{A}, \mathcal{B}, \mathcal{C}$ above), so that we can independently choose variables from the different groups and every such choice yields a top-level zero. Several schemes in the literature use explicit binding variables to make it hard to partition the encodings into independent sets. For example, the obfuscation schemes of Barak et al. [3] and Zimmerman [22] use "dual-input straddling sets" to create a "high connectivity" interlocking set of encodings.

These schemes contain, for each pair i, j of input bits, four encoded variables $U_{i,j,0,0}, U_{i,j,0,1}, U_{i,j,1,0}$, and $U_{i,j,1,1}$, such that obtaining a top-level encoding of zero requires multiplying $U_{i,j,*,*}$'s that are consistent with some n-bit input x (i.e., it requires computing some expression $\cdot \prod_{i,j} U_{i,j,x_i,x_j}$). This structure seems to foil attempts of separating the variables into independent sets, since changing any input bit creates a cascading effect. To illustrate the difficulty of applying the attack in this setting, we describe in Appendix B. a relatively simple source-group hardness assumption involving such binding variables, which we do not know how to break even though we are given many low-level CLT13 encodings of zero. We mention that in some simple cases one can use GCDs to remove the binding variables and mount the same attack; this is described in Appendix B.1. However we see no way of extending that attack to the binding variables as they are used in the literature.

3 A Unified Attack against CLT13-Based Schemes

Below we present a general attack on CLT13-based schemes that combines all the ideas from Section 2.3, and show how this attack can be used against:

- The CLT13 scheme with orthogonal encodings (details omitted), and the proposed modifications by Boneh et al. [4] and Garg et al. [12, Sec. 7] (that were suggested in response to the CHLRS attacks);
- Obfuscations of branching programs with specific structure using the iO procedure of Garg et al. [11]; and
- Obfuscations of simple circuits (such as point functions) when the "simplified iO variant" of Zimmerman [22, App. A] and Applebaum and Brakerski [1].

Central to our general attack is the notion of a "good attack set," which roughly plays the role of the sets $\mathcal{A}, \mathcal{B}, \mathcal{C}$ from Section 2 (together with the zero-test parameter). To define this notion formally, fix an instance of CLT13 with n secret primes $p_1, \ldots, p_n$ and modulus $x_0 := \prod_i p_i$. An *attack set* (of dimension d) consists of three sets of matrices $\mathcal{A}, \mathcal{B}, \mathcal{C} \subset \mathbb{Z}_{x_0}^{d \times d}$, of sizes $|\mathcal{A}| = |\mathcal{C}| = nd$ and $|\mathcal{B}| = 2$, and two vectors $s \in \mathbb{Z}_{x_0}^{1 \times d}$ and $t \in \mathbb{Z}_{x_0}^{d \times 1}$. These sets are constructed from the available public parameters and encodings of a given scheme, in such a way that for every choice of $(A_i, B_\sigma, C_j) \in \mathcal{A} \times \mathcal{B} \times \mathcal{C}$, the value

$$W_\sigma[i,j] := s \times A_i \times B_\sigma \times C_j \times t \ \in \ \mathbb{Z}_{x_0}$$

Input: $\mathcal{A} = \{A_i\}_i,\ \mathcal{B} = \{B_\sigma\}_\sigma,\ \mathcal{C} = \{C_j\}_j,\ s,\ t$

1. Compute $(nd) \times (nd)$ matrices W_0, W_1 as $W_\sigma[i,j] := [s \times A_i \times B_\sigma \times C_j \times t]_{x_0}$.
2. Compute $W := W_0 \times W_1^{-1}$ over $\mathbb{Q}$, and $M := B_0 \times B_1^{-1} \pmod{x_0}$.
3. Compute W's characteristic polynomial $f := \mathsf{charPoly}(W)$ over $\mathbb{Q}$, and factor it into monic irreducible factors over $\mathbb{Q}$ as $f = f_1 f_2 \cdots f_m$.
4. For all $k \in \{1, \ldots, m\}$ define $F_k := f/f_k = \prod_{i \neq k} f_i \in \mathbb{Q}[X]$, let d_k be the common denominator of the coefficients of F_k, and set $G_k := F_k \cdot d_k$.
5. Evaluate the G_k's at the matrix M mod x_0, $M_k := [G_k(M)]_{x_0}\ \forall k \leqslant m$.
6. Compute $S := \{GCD(M_k[i,j], x_0) \mid i,j \in [nd]; k \in [m]\}$, and return $\{x_0/q \mid q \in S\}$.

Figure 1: Our general attack on CLT13-based schemes

is a zero-tested top-level encoding of 0. (The CHLRS attack can be thought as a special case where all the "matrices" are of dimension $d = 1$, and we have $s = 1$ and $t = \mathbf{p}_{\mathrm{zt}}$.) Given such an attack set, the attack proceeds as in Figure 1, where we denote by $[z]_p$ the reduction of z modulo p into the interval $[-p/2, p/2)$, and this notation extends entry-wise to vectors and matrices.

3.1 Sufficient conditions for the attack to succeed

Next we state and prove sufficient conditions on the attack set that ensures that the attack in Figure 1 succeeds. Specifically, we would like to show that each M_k in step 5 must be zero modulo all the primes except one, and hence any non-zero entry in it yields a nontrivial factor of x_0 (i.e. the product of those primes).

Referring to the intuition from Section 2.3.4, the matrix $W = A \times B^* \times A^{-1}$ is similar to a block-diagonal matrix B^* that has one block for each CRT component. Specifically, the jth block of B^* is $B_j^* = [B_0]_{p_j} \times ([B_1]_{p_j})^{-1}$ (inverse over $\mathbb{Q}$). The characteristic polynomial of W is then the product of the characteristic polynomials of all the blocks. For simplicity, assume the block polynomials are the irreducible factors f_i from Figure 1. Then each F_k is thus the product of all block polynomials except the kth, and by the Cayley-Hamilton theorem we have that $F_k(B_j^*) = 0$ (and therefore also $G_k(B_j^*) = 0$) for all blocks $j \neq k$. But $G_k(B_j^*) = 0$ over $\mathbb{Q}$ implies that also $G_k(B_0 \times B_1^{-1}) = 0 \pmod{p_j}$, so $G_k(M)$ is zero modulo all primes $j \neq k$. The only thing left to ensure is that for the last prime p_k we get $G_k(M) \neq 0 \pmod{p_k}$, which is the essence of our sufficient condition. The actual condition in Definition 1 below is slightly more complex, to account for the case when the block polynomials are reducible over $\mathbb{Q}$.

Definition 1. *Fix an attack set $(\mathcal{A}, \mathcal{B}, \mathcal{C}, s, t)$. Let B_0, B_1, M, W be the matrices from Figure 1, and let $g_j := \mathsf{charPoly}\left([B_0]_{p_j} \times [B_1]_{p_j}^{-1}\right)$ over $\mathbb{Q}$. We say that $(\mathcal{A}, \mathcal{B}, \mathcal{C}, s, t)$ is* good *if:*

1. *$f := \mathsf{charPoly}(W) = \prod_{j \leqslant n} g_j$;*
2. *B_1 is non-singular modulo x_0;*
3. *The common denominators d_k from step 4 are all co-prime with x_0;*

4. *For any $j \leqslant n$ and any divisor f_k of g_j of degree $\geqslant 1$ (possibly $f_k = g_j$), denoting $G_k = d_k \cdot f/f_k$ as in step 4, we have $G_k(M) \neq 0 \pmod{p_j}$.*

Theorem 1. *For any good attack set $(\mathcal{A}, \mathcal{B}, \mathcal{C}, s, t)$, the algorithm in Figure 1 recovers the secret primes $p_1, \ldots, p_n$.*

To prove Theorem 1 we use the following lemma:

Lemma 3.1. *Let $p > 1$ and $u_1, \ldots, u_t$, $v_1, \ldots, v_t$ be integers, s.t. the v_i's are invertible mod p, and denote $w_i = [u_i \cdot v_i^{-1}]_p$. If g is a multivariate integer polynomial such that $g(\frac{u_1}{v_1}, \ldots, \frac{u_t}{v_t}) = 0$ over $\mathbb{Q}$, then $g(w_1, \ldots, w_t) = 0 \pmod{p}$.*

Proof. It is enough to prove it for a linear g, since we can replace any non-linear term $\prod_{i \in I} (\frac{u_i}{v_i})^{e_i}$ (for some $I \subset [t]$ and e_i's) by new variables $u' = \prod_{i \in I} u_i^{e_i}$, $v' = \prod_{i \in I} v_i^{e_i}$, and $w' = [\prod_{i \in I} w_i^{e_i}]_p = [u' \cdot v'^{-1}]_p$, and then prove the same statement on the resulting new polynomial.

Now denote $V = \prod_i v_i$ and for each i denote $v_i^* = V/v_i = \prod_{j \neq i} v_j$. For a linear g we can write $\sum_i g_i \cdot \frac{u_i}{v_i} = 0$ over $\mathbb{Q}$, so also $\sum g_i u_i v_i^* = V \cdot \sum_i g_i \cdot \frac{u_i}{v_i} = 0$, and in particular $\sum g_i u_i v_i^* = 0 \pmod{p}$. Finally, since V is invertible modulo p we get

$$\sum_i g_i w_i = \sum_i g_i u_i v_i^{-1} = V^{-1} \cdot \sum_i g_i u_i v_i^* = 0 \pmod{p}.$$

□

Proof of Theorem 1. For all i denote $B_i^* = [B_0]_{p_i} \times [B_1]_{p_i}^{-1}$ over $\mathbb{Q}$ and $\hat{B}_i = [B_0]_{p_i} \times [B_1]_{p_i}^{-1}$ over $\mathbb{Z}_{p_i}$. Let $t_i := det([B_1]_{p_i})$ (over $\mathbb{Q}$), and since B_1 is non-singular modulo x_0 then in particular $t_i \neq 0 \pmod{p_i}$. We can therefore write $B_i^* = \tilde{B}_i/t_i$ for an integer matrix $\tilde{B}_i$, and clearly we also have $\hat{B}_i = \tilde{B}_i \cdot t^{-1} \pmod{p_i}$.

Denote the characteristic polynomial of B_i^* over $\mathbb{Q}$ by $g_i := \mathsf{charPoly}(B_i^*)$. By the first condition in Definition 1 we have $f := \mathsf{charPoly}(W) = \prod_{j \leqslant n} g_j$. Note, however, that the g_j's are not necessarily irreducible, so there isn't necessarily a 1-1 correspondence between the g_j's and the irreducible factors f_k of f.

Fix an index $j \leqslant n$ and we show that for some k it holds that $G_k(M) \neq 0 \pmod{p_j}$ but $G_k(M) = 0 \pmod{p_i}$ for all $i \neq j$. Clearly this g_j is divisible by at least one f_k (which has degree $\geqslant 1$), so the last condition of Definition 1 implies that $G_k(M) = d_k \cdot F_k(M) \neq 0 \pmod{p_j}$. It remains to show that for all the other primes p_i, $i \neq j$, we have $G_k(M) = 0 \pmod{p_i}$.

Clearly F_k is divisible by g_i for every $i \neq j$, so the Cayley-Hamilton theorem implies that $F_k(B_i^*) = 0$ (over $\mathbb{Q}$) for all $i \neq j$, and therefore also $G_k(B_i^*) = 0$. Viewing $G_k(B_i^*)$ as a collection of multivariate polynomials over the elements of B_i^*, and using the facts that $B_i^* = \tilde{B}_i/t_i$ and $\hat{B}_i = \tilde{B}_i \cdot t^{-1} \pmod{p_i}$, we can apply Lemma 3.1 to conclude that also $G_k(\hat{B}_i) = 0 \pmod{p_i}$. And since $M = \hat{B}_i \pmod{p_i}$ then also $G_k(M) = 0 \pmod{p_i}$, as needed.

We have shown that $M_k := G_k(M)$ satisfies $M_k \neq 0 \pmod{p_j}$ but $M_k = 0 \pmod{p_i}$ for all $i \neq j$, so there exists an entry $z = M_k[a, b]$ such that $z \neq 0 \pmod{p_j}$ but $z = 0 \pmod{p_i}$ for all $i \neq j$. Thus $GCD(z, x_0) = \prod_{i \neq j} p_i$, and $x_0/GCD(z, x_0) = p_j$. □

Below we construct good attack sets for some schemes in the literature. We will repeatedly use the fact that for a CLT13 encoding u associated with numerator vector $u \sim (r_i g_i + m_i)_i$, the randomization vector $(r_i)_{i \in [n]}$ is nearly uniform for each encoding. Specifically we have the following, which is proved in [4, Lemma 5.7].

Lemma 3.2 ([4]). *There exists a prime $q = 2^{\Omega(n)}$ which is determined by the CLT13 system parameters such that, for each encoding, the distribution on $(r_i \bmod q)_{i\in[n]}$ is $\mathsf{negl}(n)$-close to the uniform distribution on $\mathbb{Z}_q^n$.*

3.2 Attacking the Boneh-Wu-Zimmerman "Immunized" Variant

Boneh, Wu and Zimmerman [4] proposed an "immunization" of the CLT13 scheme, which more generally applies to any composite-order asymmetric graded encoding scheme. At a high level, the idea is to add two additional slots to the underlying scheme, and use these to maintain functionality while preventing the possibility of having two encodings that multiply to 0 at the top level.

For purposes of illustration, a BWZ-encoding of plaintext m is given by two CLT13-encodings $[a, a']$: a encodes (m, α, β_1) at level z_1 and a' encodes $(\beta_2, \alpha, \beta_3)$ at level z_2, where α and the β_i are independent and uniform in the appropriate range. Two additional CLT13-encodings are given out for zero testing: t_L encodes $(1, 1, 0)$ at level z_2, and t_R encodes $(0, 1, 0)$ at level z_1. The top encoding level is $z_1 z_2$. One can add and multiply BWZ-encodings component-wise, and given a BWZ-encoding $[a, a']$ one can check if it is zero by computing the top-level CLT13 encoding $t_L \cdot a - t_R \cdot a'$. With high probability this will be a CLT13-encoding of $(0, 0, 0)$ iff $[a, a']$ was a BWZ-encoding of 0. (Note that [4] can handle a general asymmetric level structure; we chose a simple one here for clarity.)

The motivation for this scheme is that it will never give out two CLT13-encodings a and b such that ab is a top-level CLT13-encoding of 0. However, $(t_L \cdot a - t_R \cdot a') \cdot \mathbf{p}_{\mathrm{zt}}$ is still a $2(n + 2)$ linear form in the CRT components of a and a', and we can extend the Cheon et al. attack to this setting.

Attack Sets. We construct the three sets of matrices $\mathcal{A}, \mathcal{B}, \mathcal{C}$ for the attack on the scheme as follows. We start with three sets of BWZ-encodings, which can be obtained from the $\mathsf{ZMM.Encode}$ function of [4, Construction 3.1]:

- $\{[\hat{a}_i, \hat{a}'_i] : i = 1, \ldots, 2n'\}$: $\hat{a}_i$ at level z_a and $\hat{a}'_i$ at level z'_a.
- $\{[\hat{b}_\sigma, \hat{b}'_\sigma], : \sigma = 0, 1\}$: $\hat{b}_\sigma$ at level z_b and $\hat{b}'_\sigma$ at level z'_b.
- $\{[\hat{c}_j, \hat{c}'_j] : j = 1, \ldots, 2n'\}$: $\hat{c}_j$ at level z_c and $\hat{c}'_j$ at level z'_c.

Here $n' := n + 2$ is the number of slots in each component of a BWZ-encoding (i.e. n is the number of slots in the "un-immunized" scheme.) The scheme also gives out two other encodings: $\hat{t}_L$ at level $z'_a z'_b z'_c z_T$ and $\hat{t}_R$ at level $z_a z_b z_c z_T$, where the top CLT13 encoding level is $z := z_a z'_a z_b z'_b z_c z'_c z_T$. These encodings have the property that for all i, σ, j: $(\hat{a}_i \hat{b}_\sigma \hat{c}_j, \hat{a}'_i \hat{b}'_\sigma \hat{c}'_j)$ is a top-level BWZ encoding of 0, i.e. $\hat{t}_L \hat{a}_i \hat{b}_\sigma \hat{c}_j - \hat{t}_R \hat{a}'_i \hat{b}'_\sigma \hat{c}'_j$ is a top-level CLT13-encoding of 0.

Then we set

$$\mathcal{A} = \left\{ A_i = \begin{bmatrix} \hat{a}_i & 0 \\ 0 & \hat{a}'_i \end{bmatrix} \right\} \qquad \mathcal{B} = \left\{ B_\sigma = \begin{bmatrix} \hat{b}_\sigma & 0 \\ 0 & \hat{b}'_\sigma \end{bmatrix} \right\} \qquad \mathcal{C} = \left\{ C_j = \begin{bmatrix} \hat{c}_j & 0 \\ 0 & \hat{c}'_j \end{bmatrix} \right\}.$$

Finally we set the vectors $s = [\hat{t}_L, 1]$ and $t = \mathbf{p}_{\mathrm{zt}} \cdot [1, -\hat{t}_R]^T$, where

$$\mathbf{p}_{\mathrm{zt}} = \left[z \cdot \mathsf{CRT}\left(p_k^* h_k g_k^{-1} \bmod p_k \right)_k \right]_{x_0}$$

is the zero-testing parameter of the underlying CLT13 scheme (as explained in Section 2.1.2).

Set Properties. Next we show that $(\mathcal{A}, \mathcal{B}, \mathcal{C}, s, t)$ form a good attack set according to Definition 1. First, we observe that

$$W_\sigma[i,j] = s \times A_i \times B_\sigma \times C_j \times t = \left([\hat{a}_i \hat{b}_\sigma \hat{c}_j, \hat{a}'_i \hat{b}'_\sigma \hat{c}'_j] \cdot [\hat{t}_L, -\hat{t}_R] \right) \mathbf{p}_{\mathrm{zt}}$$

and thus each $W_\sigma[i,j]$ is a zero-tested CLT13-encoding of 0.

Following the intuition from Section 2.3.3, we can write

$$W_\sigma[i,j] = \underbrace{\left(a_{i,1}\ a'_{i,1}\ \ldots\ a_{i,n'}\ a'_{i,n'}\right)}_{\tilde{a}_i} \times \underbrace{\begin{pmatrix} \frac{\rho_1 b_{\sigma,1}}{g_1} & & & & \\ & \frac{\rho'_1 b'_{\sigma,1}}{g_1} & & & \\ & & \ddots & & \\ & & & \frac{\rho_{n'} b_{\sigma,n'}}{g_{n'}} & \\ & & & & \frac{\rho'_{n'} b'_{\sigma,n'}}{g_{n'}} \end{pmatrix}}_{\tilde{B}_\sigma} \times \underbrace{\begin{bmatrix} c_{j,1} \\ c'_{j,1} \\ \vdots \\ c_{j,n'} \\ c'_{j,n'} \end{bmatrix}}_{\tilde{c}_j}$$

where the g_i's are set by the CLT13 scheme and the remaining values are set as follows.

Each $a_{i,k}$ ($k \in [n']$) is the numerator of the kth CRT component of $\hat{a}_i$. That is, $a_{i,k} = g_k r_{a,i,k} + \alpha_{i,k}$ for independent randomness $r_{a,i,k}$, where $\hat{a}_i$ encodes plaintext $(\alpha_{i,k})_{k\in[n']}$. Similarly $a'_{i,k}$, $b_{\sigma,k}$, $b'_{\sigma,k}$ $c_{j,k}$ and $c'_{j,k}$ are the kth numerators of their respective encodings. Finally the ρ_k's contain the remaining "system parameters", which here include the CRT components of $\hat{t}_L$ and $\hat{t}_R$. Namely, $\rho_k = p^*_k h_k t_{L,k}$ and $\rho'_k = -p^*_k h_k t_{R,k}$, where $t_{L,k}$ and $t_{R,k}$ are the kth numerators of $\hat{t}_L$ and $\hat{t}_R$ respectively. Note that this equation holds over $\mathbb{Q}$, without reducing mod x_0, because the encodings are chosen so that $a_{i,k} b_{\sigma,k} c_{j,k} t_{L,k} - a'_{i,k} b'_{\sigma,k} c'_{j,k} t_{R,k}$ is a small multiple of g_k for every i, j, k, σ.

Now, for $\sigma \in \{0,1\}$, we can write $W_\sigma = \tilde{A} \times \tilde{B}_\sigma \times \tilde{C}_j$ where $\tilde{A}$ has ith row $\tilde{a}_i$ and $\tilde{C}$ has jth column $\tilde{c}_j$. Each W_σ is invertible over $\mathbb{Q}$ with high probability, as follows. Clearly $\tilde{B}_\sigma$ is invertible. To see that $\tilde{A}$ is invertible, consider its determinant as a non-trivial degree-n polynomial in the randomizers $r_{a,i,k}$. Since these are $\mathsf{negl}(n)$-close to uniform over a suitably large field by Lemma 3.2, $\tilde{A}$'s determinant will be non-zero over $\mathbb{Q}$ with high probability by the Schwartz-Zippel lemma. The same argument shows $\tilde{C}$ is invertible with high probability. Thus we can write $W := W_0 \times W_1^{-1} = \tilde{A} \times \tilde{B}_0 \times \tilde{B}_1^{-1} \times \tilde{A}^{-1}$.

To prove the first property of Definition 1, we observe that

$$\begin{aligned} \mathsf{charPoly}(W) = \mathsf{charPoly}\left(\tilde{B}_0 \times \tilde{B}_1^{-1}\right) &= \prod_{k\in[n']} \left(\lambda - \frac{b_{0,k}}{b_{1,k}}\right)\left(\lambda - \frac{b'_{0,k}}{b'_{1,k}}\right) \\ &= \prod_{k\in[n']} \mathsf{charPoly}\left((B_0 \bmod p_k) \times (B_1 \bmod p_k)^{-1}\right). \end{aligned}$$

where λ is the characteristic polynomial's indeterminate.

To prove the second property of Definition 1, we consider each $det([B_1]_{p_k})$ as a polynomial over the randomizers $r_{b,1,k}$. By Lemma 3.2 and the Schwartz-Zippel lemma, we have $det([B_1]_{p_k}) \neq 0 \pmod{p_k}$ with probability $1 - \mathsf{negl}(n)$. Then B_1 is non-singular mod p_k for all $k \in [n]$ simultaneously with probability $1 - \mathsf{negl}(n)$, and thus it is non-singular also mod x_0.

To prove the third property of Definition 1, we observe that any value d_k in step 4 of Figure 1 is either $b_{1,k}$ or $b'_{1,k}$ for some $k \in [n']$, and these are all co-prime to x_0.

Proving the fourth property of Definition 1 reduces to showing that, for every $k \in [n']$, both $(b_{1,k}\lambda - b_{0,k})$ and $(b'_{1,k}\lambda - b'_{0,k})$ are non-zero modulo p_k when setting $\lambda = B_0 \times B_1^{-1} (\text{mod } x_0)$. This again holds with high probability over the CLT randomizers.

We remark that this attack can be extended to break the assumption of Gentry et al. [14] even when their encodings are transformed using the above technique from [4]. In general, for this attack on [4], we only require encodings that "BWZ-multiply" to 0 at the "BWZ-zero-testing-level"; thus any set of encodings susceptible to the "basic" attack will be susceptible to the attack here after applying the [4] transformation.

3.3 Attacking the Garg-Gentry-Halevi-Zhandry Countermeasure

Garg, Gentry, Halevi, and Zhandry proposed in [12, Sec. 7] a variant of the CLT13 scheme, that was designed to resist the CHLRS attack. This variant uses matrices of native CLT13 encodings, where the encoded value is an eigenvalue of the matrix and the zero-test parameter includes also the corresponding eigenvector. The CHLRS attack from [6] indeed does not apply to this variant, but below we show that this variant still gives rise to a good attack set, and thus our new attack from Figure 1 recovers the secret primes.

The GGHZ variant relies on the same parameters as CLT13, namely we choose $(\{g_i\}_i, \{p_i\}_i, \mathbf{p}_{\text{zt}}, \{z_i\})$ (with $x_0 := \prod_i p_i$ and top level corresponding to denominator $z^* = \prod z_i$). Let $d := 2\kappa + 1$, and choose a secret matrix $T \in \mathbb{Z}_{x_0}^{d \times d}$ uniformly. An encoding of a plaintext value c at some level is given by $C \in \mathbb{Z}_{x_0}^{d \times d}$, where[4]

$$C := T \times \underbrace{\begin{bmatrix} \hat{\$} & \hat{0} & \dots & \hat{0} \\ \hat{0} & \hat{\$} & \dots & \hat{0} \\ \vdots & & & \vdots \\ \hat{0} & \hat{0} & \dots & \hat{c} \end{bmatrix}}_{C^*} \times T^{-1} \mod x_0.$$

Each $\hat{\$}$ in C^* is a "native CLT13 encoding" of an independent random value at the given level, each $\hat{0}$ is an independent native encoding of 0, and $\hat{c}$ is a native encoding of c. For zero-testing, two dimension-d vectors s, t are provided:

$$\begin{array}{lll} s := & [\hat{\$} \dots \hat{\$}\ \hat{0} \dots \hat{0}\ \hat{\$}] \times T^{-1} & \mod x_0 \\ t := \mathbf{p}_{\text{zt}} \cdot T \times & [\hat{0} \dots \hat{0}\ \hat{\$} \dots \hat{\$}\ \hat{\$}]^T & \mod x_0 \end{array}$$

where $\hat{0}$ and $\hat{\$}$ are CLT13 native "level-zero" encodings (i.e, corresponding to denominator 1). Then a GGHZ-encoding C as above at the top level level can be zero tested by computing $s \times C \times t = (\hat{\$} \cdot \hat{c} + \hat{0}) \cdot \mathbf{p}_{\text{zt}} \pmod{x_0}$ and checking for smallness.

Attack set. The matrix sets $\mathcal{A}, \mathcal{B}, \mathcal{C}$ consist directly of GGHZ-encodings, since these are already in matrix form. Specifically, we assume that $[1, \kappa]$ is partitioned into three intervals $I_A = [1, k_A], I_B = [k_A + 1, k_B], I_C = [k_B + 1, \kappa]$, such that we have GGHZ-encodings

[4] The attack applies also when one uses many matrices $T_0, T_0^{-1}, \dots, T_\kappa, T_\kappa^{-1}$ (rather than just T, T^{-1}), so multiplication can only be performed in a specific order, as described in [12].

- $\mathcal{A} = \left\{ A_i = T \times A_i^* \times T^{-1} : A_i \text{ encoded at level } I_A \right\}_{i \in [nd]}$
- $\mathcal{B} = \left\{ B_\sigma = T \times B_\sigma^* \times T^{-1} : B_\sigma \text{ encoded at level } I_B \right\}_{\sigma \in \{0,1\}}$
- $\mathcal{C} = \left\{ C_k = T \times C_k^* \times T^{-1} : C_k \text{ encoded at level } I_C \right\}_{k \in [nd]}$

where $A_i \times B_\sigma \times C_k$ is a GGHZ-encoding of 0 for all $i, k \in [nd]$ and $\sigma \in \{0,1\}$. The vectors s and t are the zero testing vectors from the GGHZ scheme.

Attack set properties. We prove that $(\mathcal{A}, \mathcal{B}, \mathcal{C}, s, t)$ form a good attack set according to Definition 1. We write

$$\begin{aligned} W_\sigma[i,k] &= s \times A_i \times B_\sigma \times C_k \times t \\ &= s \times T \times A_i^* \times B_\sigma^* \times C_k^* \times T^{-1} \times t \;=\; \mathbf{a}^i \times B_\sigma^* \times \mathbf{c}^k \end{aligned}$$

where $\mathbf{a}^i := s' \times A_i^*$ and $\mathbf{c}^k := C_k^* \times t'$ are dimension-d vectors. The above equality holds over the integers, not only modulo x_0, since all the variables in the final right-hand-side are small compared to x_0.

We denote $\mathbf{a}_\ell^i := \mathbf{a}^i \bmod p_\ell$ and $\mathbf{c}_\ell^k := \mathbf{c}^k \bmod p_\ell$ for $i \in [nd], \ell \in [n]$. Now we can write $W_\sigma = \tilde{A} \times \tilde{B}_\sigma \times \tilde{C}$, where $\tilde{A}$ is an $nd \times n^2 d$ matrix, $\tilde{C}$ is an $n^2 d \times nd$ matrix, and $\tilde{B}_\sigma$ is a $n^2 d \times n^2 d$ block-diagonal matrix, defined as follows.

$$\tilde{A} = \begin{bmatrix} \mathbf{a}_1^1 & \mathbf{a}_2^1 & \cdots & \mathbf{a}_n^1 \\ \mathbf{a}_1^2 & \mathbf{a}_2^2 & \cdots & \mathbf{a}_n^2 \\ \vdots & \vdots & & \vdots \\ \mathbf{a}_1^{nd} & \mathbf{a}_2^{nd} & \cdots & \mathbf{a}_n^{nd} \end{bmatrix} \quad \tilde{C} = \begin{bmatrix} (\mathbf{c}_1^1)^T & (\mathbf{c}_1^2)^T & \cdots & (\mathbf{c}_1^{nd})^T \\ (\mathbf{c}_2^1)^T & (\mathbf{c}_2^2)^T & \cdots & (\mathbf{c}_2^{nd})^T \\ \vdots & \vdots & & \vdots \\ (\mathbf{c}_n^1)^T & (\mathbf{c}_n^2)^T & \cdots & (\mathbf{c}_n^{nd})^T \end{bmatrix}$$

$$\tilde{B}_\sigma = \begin{bmatrix} B_\sigma^* \bmod p_1 & 0 & & 0 \\ 0 & B_\sigma^* \bmod p_2 & & 0 \\ & & \ddots & \\ 0 & 0 & & B_\sigma^* \bmod p_n \end{bmatrix}$$

Using Lemma 3.2 and the Schwartz-Zippel lemma, it can be shown that with high probability over the randomness in the CLT13 encodings, $\tilde{A}$, $\tilde{C}$, and each B_σ^* have full rank nd. Under this condition each W_σ has rank nd and is thus invertible, so we can write $W = W_0 \times W_1^{-1} = \tilde{A} \times \tilde{B}_0 \times \tilde{B}_1^{-1} \times \tilde{A}^{-1}$, where $\tilde{A}^{-1}$ denotes the right inverse of the (non-square, full-rank) matrix $\tilde{A}$. Then we have

$$\begin{aligned} \mathsf{charPoly}(W) = \mathsf{charPoly}\left(\tilde{B}_0 \times \tilde{B}_1^{-1}\right) &= \prod_{i=1}^{n} \mathsf{charPoly}\left([B_0^*]_{p_i} \times [B_1^*]_{p_i}^{-1}\right) \\ &= \prod_{i=1}^{n} \mathsf{charPoly}\left([B_0]_{p_i} \times [B_1]_{p_i}^{-1}\right) \end{aligned}$$

so the first property of Definition 1 holds. The second property of Definition 1 follows similarly to the proof for the BWZ attack (Section 3.2). We were not able to prove that the last two properties

in Definition 1 hold, but we verified them experimentally by running the attack on several random instances and checking that they indeed hold in all of them. For the fourth property, we can prove that it holds under the following natural conjecture:

Conjecture 1. *For each $i \in [n]$, with high probability over the randomness in the CLT13 encodings,* $\mathsf{charPoly}\left([B_0^*]_{p_i} \times [B_1^*]_{p_i}^{-1}\right)$ *is irreducible over* $\mathbb{Q}$.

We make two remarks about this conjecture. First, we have verified it experimentally. Second, a work of Kuba [19] shows that among the degree-n univariate integer polynomials whose coefficients are bounded in absolute value by an integer t, the polynomials that are reducible over $\mathbb{Q}$ make up a roughly $1/t$ fraction. In particular, a random polynomial with r-bit coefficients is irreducible over $\mathbb{Q}$ with probability roughly $1 - 2^{-r}$. Thus provided that $\mathsf{charPoly}\left([B_0^*]_{p_i} \times [B_1^*]_{p_i}^{-1}\right)$ is well-distributed among polynomials with an appropriate coefficient bound, Conjecture 1 should hold. We note that the relationship between a random polynomial and the characteristic polynomial of a random matrix has been explored by Hansen and Schmutz [17]. However, their results do not seem directly applicable here because they study polynomials over a finite field $\mathbb{F}$, and a uniform degree-n polynomial is irreducible over $\mathbb{F}$ only with probability $\approx 1/n$.

Assuming Conjecture 1, the fourth property of Definition 1 reduces to showing that for every prime factor p_j of x_0, $\left(\prod_{i \neq j} d_i f_i\right)(M) \neq 0 \pmod{p_j}$ where d_i, f_i, and M are as in Fig. 1. Choose all values in the CLT13 encodings except for the random values in the jth slot of the encodings in B_0, and call the unchosen values R. With high probability over this choice, each entry of M is a non-trivial linear polynomial in R, and $\left(\prod_{i \neq j} d_i f_i\right)$ is a non-trivial degree-$(n-1)$ polynomial in M. Thus each entry of $\left(\prod_{i \neq j} d_i f_i\right)(M)$ is a non-trivial degree-$(n-1)$ polynomial in R, and is non-zero modulo p_i with high probability by Lemma 3.2 and the Schwartz-Zippel lemma.

3.4 Attacking GGHRSW Obfuscation for Simple Branching Programs

We observe that our unified attack can be applied also to the candidate obfuscation construction of Garg et al. [11] when instantiated with the CLT13 multilinear maps and applied to branching programs with specific "partitionable" structure that we define below. We stress that applying Barrington's theorem to a circuits *does not have the required structure*, so as far as we know, the iO candidate from [11] for NC^1 circuits remains plausible.

The GGHRSW Obfuscation Candidate for Branching Programs

Recall that the obfuscator of Garg et al. [11] consists of encoded, randomized versions of two BPs; one is the BP that we want to obfuscate and the other is a "dummy BP" consisting of only identity matrices (and hence computing the all-one function). Even though neither program computes a zero, they are constructed such that their difference on accepting computations yields an encoding of zero, which can be recognized by zero testing. The core construction from [11] works with oblivious branching programs. An oblivious branching program of length L over ℓ input variables is defined as follows

$$BP = \{(\mathsf{inp}(i), A_{i,0}, A_{i,1}) : i \in [L], \mathsf{inp}(i) \in [\ell], A_{i,b} \in \{0,1\}^{w \times w}\},$$

where the $A_{i\sigma}$'s are invertible matrices and $\mathsf{inp}(i)$ is the input bit position examined in step i. The function computed by this branching program is defined (using some fixed matrix $A_0 \neq I$) as

$$f_{BP,A,I} = \begin{cases} 0 & \text{if } \prod_{i=1}^{L} A_{i,x_{\mathsf{inp}i}} = A_0 \\ 1 & \text{if } \prod_{i=1}^{L} A_{i,x_{\mathsf{inp}i}} = I \\ \mathsf{undef} & \text{otherwise.} \end{cases}$$

Let $\mathbb{Z}_p$ be a ring that we use for randomization, and for each input bit j denote by $I_j := \{i \in [L] : \mathsf{inp}(i) = j\}$ the set of steps where the branching program examine the j'th input bit. The GGHRSW construction, on input an L-step branching program BP over ℓ input bits, proceeds as follows:

1. Sample random and independent scalars $\{\alpha_{i,0}, \alpha_{i,1}, \alpha'_{i,0}, \alpha'_{i,1} \in \mathbb{Z}_p : i \in [L]\}$, subject to the constraint that for any input bit $j \in [\ell]$, we have $\prod_{i \in I_j} \alpha_{i,0} = \prod_{i \in I_j} \alpha'_{i,0}$ and $\prod_{i \in I_j} \alpha_{i,1} = \prod_{i \in I_j} \alpha'_{i,1}$.

2. Let $m = 2L + w$. For every $i \in [n]$, choose two block-diagonal $m \times m$ matrices $D_{i,0}, D_{i,1}$ where the diagonal entries $1, \ldots, 2L$ are chosen at random (\$) and the bottom-right $w \times w$ are the scaled $A_{j,b}$'s. Also choose two more $m \times m$ matrices $D'_{i,0}, D'_{i,1}$ where the diagonal entries $1, \ldots, 2L$ are random and the bottom-right $w \times w$ are the scaled identity:

$$D_{i,b} \sim \begin{pmatrix} \$ & & & \\ & \ddots & & \\ & & \$ & \\ & & & \alpha_{i,b} A_{i,b} \end{pmatrix}, \quad D'_{i,b} \sim \begin{pmatrix} \$ & & & \\ & \ddots & & \\ & & \$ & \\ & & & \alpha'_{i,b} I \end{pmatrix}, b \in \{0,1\}.$$

3. Choose vectors $\mathbf{s}$ and $\mathbf{t}$, and $\mathbf{s}'$ and $\mathbf{t}'$ of dimension $m = 2L + w$ as follows:

$$\begin{aligned} \mathbf{s} &\sim (0 \ldots 0 \ \$ \ldots \$ \quad \mathbf{s}^*) & \mathbf{t} &\sim (\$ \ldots \$ \ \ 0 \ldots 0 \quad \mathbf{t}^*)^T \\ \mathbf{s}' &\sim (0 \ldots 0 \ \$ \ldots \$ \quad \mathbf{s}'^*) & \mathbf{t}' &\sim (\$ \ldots \$ \ \ 0 \ldots 0 \quad \mathbf{t}'^*)^T \end{aligned}$$

   Here $\mathbf{s}^*, \mathbf{t}^*, \mathbf{s}'^*, \mathbf{t}'^* \in \mathbb{Z}_p^w$ are uniform up to $\langle \mathbf{s}^*, \mathbf{t}^* \rangle = \langle \mathbf{s}'^*, \mathbf{t}'^* \rangle$, and $0 \ldots 0$ and $\$ \ldots \$$ are length-L vectors of zeros and uniform elements of $\mathbb{Z}_p$, respectively.

4. Sample $2(L+1)$ uniform full-rank matrices $R_0, \ldots, R_L, R'_0, \ldots, R'_L \in \mathbb{Z}_p^{m \times m}$.

5. The randomized branching program over $\mathbb{Z}_p$ is the following:

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

6. Finally, encode the randomized program using an $(L+2)$-level asymmetric multilinear map scheme. Here we use the CLT13 scheme, choosing $x_0 = \prod_{i=1}^{n} p_i$, for equal-size primes p_i, $g = \mathsf{CRT}(g_i)$ for small $g_i \ll p_i$'s, random denominators $z_0, z_1, \ldots, z_{L+1} \in \mathbb{Z}_{x_0}$ with $z^* = [\prod_i z_i]_{x_0}$, and an element h with mid-size CRT components, used for the zero-testing parameter $\mathbf{p}_{\mathrm{zt}} = [h z^* g^{-1}]_{x_0}$.

Choose random small vectors $\mathbf{r_s}\ \mathbf{r'_s}\ \mathbf{r_t}\ \mathbf{r'_t}$, and random small matrices $U_{i,b}$ and $U'_{i,b}$, and publish the zero-testing parameter $\mathbf{p}_{\mathrm{zt}}$ and the obfuscation

$$\mathcal{O}(BP) \;=\; \left\{ \begin{array}{l} \hat{\mathbf{s}} = [z_0^{-1}(\tilde{\mathbf{s}} + g\mathbf{r_s})]_{x_0}, \qquad \hat{\mathbf{t}} = [z_{L+1}^{-1}(\tilde{\mathbf{t}} + g\mathbf{r_t})]_{x_0}, \\ \left\{\hat{D}_{i,b} = [z_i^{-1}(\tilde{D}_{i,b} + gU_{i,b})]_{x_0}\right\}_{i\in[L],b\in\{0,1\}}, \\ \hat{\mathbf{s}}' = [z_0^{-1}(\tilde{\mathbf{s}}' + g\mathbf{r'_s})]_{x_0}, \qquad \hat{\mathbf{t}}' = [z_{L+1}^{-1}(\tilde{\mathbf{t}}' + g\mathbf{r'_t})]_{x_0}, \\ \left\{\hat{D}'_{i,b} = [z_i^{-1}(\tilde{D}'_{i,b} + gU'_{i,b})]_{x_0}\right\}_{i\in[L],b\in\{0,1\}} \end{array} \right\}.$$

To evaluate $\mathcal{O}(BP)(x)$, compute $y = \tilde{s}\left(\prod_{i=1}^{L}\tilde{D}_{i,x_{\mathsf{inp}(i)}}\right)\tilde{t} - \tilde{s}'\left(\prod_{i=1}^{L}\tilde{D}'_{i,x_{\mathsf{inp}(i)}}\right)\tilde{t}'$, and output 1 if y encodes 0 (as determined by $\mathbf{p}_{\mathrm{zt}}$).

Attack

Our attack is applicable to branching programs with the following structure: there exists a partition of the input bits $[\ell] = X_1 \cup X_2 \cup X_3$ and the branching program steps $[L] = A \cup B \cup C$ such that A, B and C consist of consecutive steps in the branching program and $\mathsf{inp}(i) \in X_1\ \forall i \in A$, $\mathsf{inp}(i) \in X_2\ \forall i \in B$ and $\mathsf{inp}(i) \in X_3\ \forall i \in C$. We consider a branching program BP of length L and input length ℓ, computing the constant-1 function, that can be written as $BP(x) = A(x_1)\circ B(x_2)\circ C(x_3)$, where $A(x_1)$, $B(x_2)$, and $C(x_3)$ are branching programs over positions in the sets A, B, and C depending on inputs x_1, x_2, and x_3, respectively. We are given the obfuscation:

$$\mathcal{O}(BP) = \left(\mathbf{p}_{\mathrm{zt}}, \hat{\mathbf{s}}, \hat{\mathbf{t}}, \hat{\mathbf{s}}', \hat{\mathbf{t}}', \{\hat{D}_{i,b}, \hat{D}'_{i,b}\}_{i\in[L],b\in\{0,1\}}\right).$$

Attack Sets. We construct the sets $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ as follows. Let $A(x) = \prod_{i\in A} D_{i,x_{\mathsf{inp}(i)}}$, $A'(x) = \prod_{i\in A} D'_{i,x_{\mathsf{inp}(i)}}$. We define similarly $B(x), B'(x)$ and $C(x), C'(x)$. We note that using $\mathcal{O}$ we can compute $R_0A(x)R_{|A|}^{-1} = \prod_{i\in A}\tilde{D}_{i,x_{\mathsf{inp}(i)}}$ and $R_0A'(x)R_{|A|}^{-1} = \prod_{i\in A}\tilde{D}'_{i,x_{\mathsf{inp}(i)}}$, and so on. Let $\alpha_1, \ldots, \alpha_{mn} \in \{0,1\}^{|X_1|}$ be any set of distinct strings, and similarly for $\beta_0, \beta_1 \in \{0,1\}^{|X_2|}$ and $\gamma_1, \ldots, \gamma_{mn} \in \{0,1\}^{|X_3|}$. We set $s = (\tilde{s}, \tilde{s}')$ and $t = (\tilde{t}, -\tilde{t}')\mathbf{p}_{\mathrm{zt}}$, and define

$$\mathcal{A} = \left\{\tilde{A}_i = \begin{bmatrix} R_0A(\alpha_i)R_{|A|}^{-1} & 0 \\ 0 & R_0A'(\alpha_i)R_{|A|}^{-1} \end{bmatrix}\right\}_{i\in[(2L+w)n]}$$

$$\mathcal{B} = \left\{\tilde{B}_\sigma = \begin{bmatrix} R_{|A|}B(\beta_\sigma)R_{|A\cup B|}^{-1} & 0 \\ 0 & R_{|A|}B'(\beta_\sigma)R_{|A\cup B|}^{-1} \end{bmatrix}\right\}_{\sigma\in\{0,1\}}$$

$$\mathcal{C} = \left\{\tilde{C}_k = \begin{bmatrix} R_{|A\cup B|}C(\gamma_k)R_L^{-1} & 0 \\ 0 & R_{|A\cup B|}C'(\gamma_k)R_L^{-1} \end{bmatrix}\right\}_{k\in[(2L+w)n]}.$$

Set Properties. We consider the values

$$W_0[i,k] = s \times \tilde{A}_i \times \tilde{B}_0 \times \tilde{C}_k \times t = (s \times A_i \times B_0 \times C_k \times t - s' \times A'_i \times B'_0 \times C'_k \times t')\mathbf{p}_{\mathrm{zt}}.$$

Since $W_0[i,k]$ is a zero-tested encoding of zero by the definition of the obfuscated branching programs, the above equality holds not only $\bmod x_0$ but also over the integers. W_1 is constructed analogously.

The rest of the attack proceeds in the same manner as the attack on GGHZ encodings from Section 3.3. Let $\mathbf{a}_i = (s \times A_i, s' \times A'_i)$ for $i \in [(2m+w)n]$, $\mathbf{c}_k = (C_k \times t \times \mathbf{p}_{\mathrm{zt}}, -C'_k \times t' \times \mathbf{p}_{\mathrm{zt}})$ for $k \in [(2m+w)n]$ and $X_0 = \begin{bmatrix} B_0 & 0 \\ 0 & B'_0 \end{bmatrix}$. We set the matrix $\hat{A}$ to have i-th row that is concatenations of the vectors $\mathbf{a}_i \bmod p_j$ for $j \in [n]$, the matrix $\hat{C}$ to have i-th column that is concatenation of $\mathbf{c}_i^T \bmod p_j$ for $j \in [n]$, and the matrix $\hat{B}_0$ to be a diagonal matrix with diagonal consisting of $X_0 \bmod p_j$ for $j \in [n]$. Then we have that $W_0 = \hat{A} \times \hat{B}_0 \times \hat{C}$. We compute analogously $W_1 = \hat{A} \times \hat{B}_1 \times \hat{C}$. We use these matrices as in the attack on GGHZ encodings to break the underlying CLT13 encodings.

3.5 Attacking Recent Circuit-Obfuscation Schemes

Recently, Zimmerman [22] and Applebaum and Brakerski [1] proposed very similar obfuscation schemes that handle circuits directly, without needing to transform them first into branching programs. This scheme operates with MMAP schemes that have composite-order plaintext space, it was proved secure in a generic MMAPs model, and at the time it was suggested to realize this scheme over the CLT13 graded encoding scheme [7]. Although the attacks of Cheon et al. serve as a powerful demonstration that the CLT13 scheme is susceptible to a lot more attacks than those available in the generic model, it was still not known how to apply these attacks to break the schemes from [22, 1].

Each of the works [22, 1] contain a "simple scheme" (which is essentially the same in both works) and then a more complicated variant (which differs between the two papers). Below we show that the common simple scheme from [22, Appendix A] and [1] can be broken using an attack similar to the one from 3.2, when instantiated with simple enough circuits. For example, when that scheme is used to obfuscate point functions, it is possible to recover the secret point from the obfuscated circuit.

The Circuit-Obfuscation Scheme

We begin with an overview of the scheme from [22, Appendix A]. We omit some details which are not relevant to the attacks.

The construction uses a "universal circuit" $C(y,x) = C_y(x)$, $C : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}$. The starting point of this construction is the following simple scheme: obfuscating a circuit C_y is done by encoding the bits of y and x using a graded encoding scheme, providing for the bits of y only encoding of the correct bit value (0 or 1) and for the bits of x the encoding of both 0 and 1. The user can then compute $C(y,x)$ on the encoded bits and check if the answer is zero using the zero-test. Of course this scheme is insecure, and so several modifications are introduced:

- The scheme uses a composite order multi-linear maps scheme that allows each encoding to have two slots relative to two different factors N_{ev}, $\mathsf{N}_{\mathsf{chk}}$ of the plaintext space. One slot encodes the bits as before (call it the *signal slot* and is defined by the value modulo N_{ev}, while the other slot encodes random elements (call it the *control slot*) and is defined by the value modulo $\mathsf{N}_{\mathsf{chk}}$. An encoding of a message m such that $m = m_1 \bmod \mathsf{N}_{\mathsf{ev}}$ and $= m_2 \bmod \mathsf{N}_{\mathsf{chk}}$ with respect to a set S is denoted as $[m_1, m_2]_S$.

For each bit position i in x, we provide two encodings $\tilde{x}_{i,0} = [0, \alpha_i]_{X_{i,0}}$ and $\tilde{x}_{i,1} = [1, \alpha_i]_{X_{i,1}}$ with the same value α_i in the control slot and values zero and one in the signal slot. Similarly for each bit position j in y we encode a random element β_j in the control slot together with the value y_i in the signal slot: $\tilde{y}_i = [y_i, \beta_i]_Y$. The encodings of the bits belonging to the function description y use the same set Y, while the encoding of each input bit b at each position i of x uses a different set $X_{i,b}$.

The obfuscation includes the above encodings $\tilde{y}_i$ and $\tilde{x}_j$ together with an encoding of

$$\tilde{C}^* = [0, C^*]_{Y^{\mathsf{deg}(y)} \prod_{i\in[n]} (X_{i,0}X_{i,1})^{\mathsf{deg}(x_i)} Z_i},$$

where C^* is computed as $C(\alpha_1, \ldots, \alpha_k, \beta_1, \ldots, \beta_m)$, $\mathsf{deg}(y)$ is the total degree of C in all of the variables $y_1, \ldots, y_m$ and $\mathsf{deg}(x_i)$ is the total degree of the C in the variable x_i. The sets Z_i are new sets, which are used in the straddling encoding sets as we explain in the next paragraph.

To evaluate the program on any specific input x, we choose the appropriate encodings $\tilde{x}_{i,x_i}$ together with the encodings $\tilde{y}_i$ and use them to evaluate the circuit C and obtain $\tilde{C} = [C_y(x), C^*]_{Y^{\mathsf{deg}(y)} \prod_{i\in[k]} (X_{i,x_i})^{\mathsf{deg}(x_i)}}$. In order apply the zero test we need to remove the value in the control slot of the encoding. For this purpose we will subtract the encoding $\tilde{C}^*$. However, since $\tilde{C}$ and $\tilde{C}^*$ are encoded at different levels, the obfuscation provides the following additional encodings:

$$\tilde{z}_{i,b} = [\delta_{i,b}, \gamma_{i,b}]_{X_{i,1-b}^{\mathsf{deg}(x_i)} Z_i W_i} \quad \tilde{w}_{i,b} = [0, \gamma_{i,b}]_{W_i}.$$

These encodings extend the straddling sets to enforce guarantees that each input bit is used consistently throughout the circuit. More specifically $\tilde{C}$ can be completed to the zero-testing level using the $\tilde{z}_{i,b}$ encodings if and only if it is encoded at a level that has component either $X_{i,0}^{\mathsf{deg}(x_i)}$ or $X_{i,1}^{\mathsf{deg}(x_i)}$.

Using the above encodings we can compute the following encoding at zero-testing level:

$$z = \left(\tilde{C} \prod_{i\in[n]} \tilde{z}_{i,x_i} - \tilde{C}^* \prod_{i\in[n]} \tilde{w}_{i,x_i} \right).$$

We can apply the zero test on z, which encodes 0 if and only if $C_y(x) = 0$.

- Since the straddling sets for the encodings for the input bits above use different sets $X_{i,b}$ for each input position and value, we need one more modification of the scheme to enable addition operations in the evaluated circuit. Since in the underlying graded encoding scheme we can add only encodings associated with the same level set, in order to enable addition of the values encoded in the signal slot in $\tilde{x}_{i,b} = [b, \alpha_i]_{X_{i,b}}$ and $\tilde{x}_{j,b'} = [b', \alpha_j]_{X_{j,b'}}$ the obfuscation provides also the following encodings: $\tilde{u}_{i,b} = [1, 1]_{X_{i,b}}$ and $\tilde{u}_{j,b'} = [1, 1]_{X_{j,b'}}$. The encoding of the sum of the signal slot values and the auxiliary encoding used for further additions are computed as

$$\tilde{x}_{i,b}\tilde{u}_{j,b'} + \tilde{x}_{j,b'}\tilde{u}_{i,b} = [b + b', \alpha_i + \alpha_j]_{X_{i,b}X_{j,b'}} \qquad \tilde{u}_{i,b}\tilde{u}_{j,b'} = [1, 1]_{X_{i,b}X_{j,b'}}.$$

Attack Sets

For our attack we will consider a circuit C that has only multiplication gates and odd number n of inputs. Our attack can be extended to "universal" circuits for point functions of the form $C(y,x) = C_y(x) = \chi_{x\oplus y=0}$. This circuit is described with a point y, it gets as input a point x, compares the respective bits of x and y and then computes an AND of the results. We can apply the attack that we describe next using the encoded bits $b_i = x_i \oplus y_i$ in the role of the input bits.

We split the k bits of the input x into three sets X_1, X_2 and X_3 where X_1 contains the first $(k-1)/2$ bits, X_2 is the middle bit and X_3 are the last $(k-1)/2$ bits of the input.

We construct the sets $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ as follows:

- $\mathcal{A} = \left\{\prod_{i\in X_1}(\tilde{x}_{i,x_i}\tilde{z}_{i,x_i}) \ : \ \text{at least one of the values } \{\tilde{x}_{i,x_i}\}_{i\in X_1} \text{ encodes } 0\right\}$,
- $\mathcal{B} = \left\{b_{x_{(k-1)/2+1}} = \tilde{x}_{(k-1)/2+1,x_{(k-1)/2+1}} \cdot \tilde{z}_{(k-1)/2+1,x_{(k-1)/2+1}} \ : \ x_{(k-1)/2+1} \in \{0,1\}\right\}$,
- $\mathcal{C} = \left\{\prod_{i\in X_3}(\tilde{x}_{i,x_i}\tilde{z}_{i,x_i}) \ : \ X_3 \in \{0,1\}^{(k-1)/2}\right\}$.

We have $s = 1$ and $t = \mathbf{p}_{\mathrm{zt}}$. Also $M = b_0 \times b_1^{-1}$.

Set Properties. Let $\mathcal{A} = \{a_i\}_{i\in[n+1]}$, $\mathcal{B} = \{b_0, b_1\}$ and $\mathcal{C} = \{c_k\}_{k\in[n+1]}$. Let $u = \tilde{C}^* \prod_{i\in[n]} \tilde{w}_{i,x_i}$. We consider

$$W_0[j,k] = s \times a_j \times b_0 \times c_k \times t - u \times t = g^{-1}h \cdot (u_{x_1} \cdot u_{x_2} \cdot u_{x_3} - u),$$

where $g^{-1}h, u_{x_1}, u_{x_2}, u_{x_3}$ and u have the following CRT components:

$$\begin{array}{rcllll}
g^{-1}h & = & (g_1^{-1}h_1, & g_2^{-1}h_2, & g_3^{-1}h_3, & \ldots, \ g_n^{-1}h_n) \\
u_{x_1} & = & (g_1\rho_{x_1,1}, & g_2\rho_{x_1,2} + r_{x_1}, & g_3\rho_{x_1,3}, & \ldots, \ g_t\rho_{x_1,n}) \\
u_{x_2} & = & (\tau_{x_2,1}, & g_2\tau_{x_2,2} + r_{x_2}, & \tau_{x_2,3}, & \ldots, \ \tau_{x_2,n}) \\
u_{x_3} & = & (\sigma_{x_3,1}, & g_2\sigma_{x_3,2} + r_{x_3}, & \sigma_{x_3,3}, & \ldots, \ \sigma_{x_3,n}) \\
u & = & (g_1\psi_1, & g_2\psi_2 + r^*, & g_3\psi_3, & \ldots, \ g_n\psi_n).
\end{array}$$

The above CRT component correspond to CLT13 encoding of the value in the signal slot except the the values in the second CRT component which is the control slot. Since u_{x_1} encodes 0 in its signal slot, all its CRT components are 0 mod p_i except the component in the second slot.

Let denote $\rho'_{x_1,2} = g_2\rho_{x_1,2} + r_{x_1}$, $\tau'_{y,2} = g_2\tau_{x_2,2} + r_{x_2}$, $\sigma'_{z,2} = g_2\sigma_{x_3,2} + r_{x_3}$, and $\psi'_2 = g_2\psi_2 + r^*$. Then, $W_0[j,k]$ equals

$$g^{-1}h\cdot\big(g_1(\rho_{x_1,1}\tau_{x_2,1}\sigma_{x_3,1} - \psi_1),\ (\rho'_{x_1,2}\tau'_{x_2,2}\sigma'_{x_3,2} - \psi'_2),\ g_3(\rho_{x_1,3}\tau_{x_2,3}\sigma_{x_3,3} - \psi_3),\ \ldots,\ g_n(\rho_{x_1,n}\tau_{x_2,n}\sigma_{x_3,n} - \psi_n)\big),$$

which is a zero-tested encoding of 0 since by construction $\psi'_2 - \rho'_{x_1,2}\tau'_{x_2,2}\sigma'_{x_3,2} \pmod{g_2}$. The above equality holds not only $\bmod x_0$ but also over the integers since g^{-1} cancels out and all other variables are much smaller than x_0.

We can write the last equality in matrix notations as

$$W_0[j,k] = g^{-1} \cdot \overbrace{\begin{pmatrix} g_1\rho_{x,1} & \rho'_{x_1,2} & g_3\rho_{x_1,3} & \cdots & g_n\rho_{x_1,n} & 1 \end{pmatrix}}^{\mathbf{a}_{x_1}} \times \underbrace{\begin{pmatrix} h_1\tau_{x_2,1} & & & & & \\ & h_2\tau'_{x_2,2} & & & & \\ & & h_3\tau_{x_2,3} & & & \\ & & & \ddots & & \\ & & & & h_n\tau_{x_2,n} & \\ & & & & & -\psi^* \end{pmatrix}}_{B_{x_2}} \times \underbrace{\begin{pmatrix} \sigma_{x_3,1} \\ \sigma'_{x_3,2} \\ \sigma_{x_3,3} \\ \vdots \\ \sigma_{x_3,n} \\ 1 \end{pmatrix}}_{\mathbf{c}_{x_3}}$$

where $\psi^* = \psi'_2 h_2 + \sum_{i \neq 2} g_i \psi_i h_i$, which depends only on u, g and h and is independent of the values in $\mathcal{B}$.

We construct matrix A using $n+1$ vectors $\mathbf{a}_{x_1}$ obtained from the set $\mathcal{A}$ and matrix C using $n+1$ vectors $\mathbf{c}_{x_3}$ vectors obtained from $\mathcal{C}$ as columns. We thus get the two matrices

$$W_0 = g^{-1} \cdot A \times B_0 \times C \quad \text{and} \quad W_1 = g^{-1} \cdot A \times B_1 \times C,$$

The matrices W_j are invertible with high probability over the encodings' randomness by the same argument as in the attack on BWZ (Section 3.2). Hence $W = W_0 \times W_1^{-1} = \tilde{A} \times \tilde{B}_0 \times \tilde{B}_1^{-1} \times \tilde{A}^{-1}$, and

$$\begin{aligned} \mathsf{charPoly}(W) = \mathsf{charPoly}\left(\tilde{B}_0 \times \tilde{B}_1^{-1}\right) &= \prod_{i=1}^{n} \left(\mathsf{charPoly}(b_0 \times (b_1)^{-1} \bmod p_i)\right) \\ &= \prod_{i=1}^{n} \left(\mathsf{charPoly}(M \bmod p_i)\right), \end{aligned}$$

which is the first property for our attack sets.

The second through fourth properties of Definition 1 follow via essentially the same argument as in the attack on BWZ (Section 3.2).

4 A Weak-DL Attack on Matrix-GGH13

Given the GGH13 "weak discrete logarithm" attacks [10], a matrix variant of GGH13 can be constructed with the aim of avoiding public GGH13 encodings of zero in the public parameters (fashioned after the "Multilinear Jigsaw Puzzle" techniques in [11]). This variant replaces the native GGH13 encoding of the values of interest by encoding matrices related to these values (e.g. have the desired value as an eigenvalue). In Section 3.3, we have shown that a similar GGHZ countermeasure for CLT can be broken. In this section we show that the matrix GGH13 is also susceptible to a new "weak-discrete-logarithm" attack.

As in GGH13, we have plaintext space $R_g = R/gR$ where R is the ring of integers in some number field and $g \in R$, and ciphertext space $R_q = R/qR$ for some integer q. Below we describe the attack in the "asymmetric" setting where we encode elements relative to subsets of $[k] = \{1, \ldots, k\}$,

and use $[k]$ itself as our zero-test level. The secrets associated with an instance of the scheme include the plaintext-space element g and k random denominators $z_i \in R_q$.

Trying to protect against "zeroizing attacks", one could consider the following solution: we choose a random small vector $\mathbf{s}^* \in R^n$ and invertible matrix $T \in R_q^{n\times n}$ (for some n), and then encode $\alpha \in R_g$ relative to the set $S \subset [k]$ by choosing a random small matrix A^* such that $\mathbf{s}^* \times A^* = \alpha\mathbf{s}^* \pmod{gR}$, and publishing the matrix

$$A^{\{S\}} = \left[T \times A^* \times T^{-1}/\prod_{i\in S} z_i\right]_q .$$

For the purpose of zero-test we choose another mid-size random vector $\mathbf{t}^* \in R^n$ and publish the two vectors $\mathbf{s} = [g^{-1}\mathbf{s}^* \times T^{-1}]_q$ and $\mathbf{t} = [\prod_i z_i \cdot T \times \mathbf{t}^*]_q$. (The h element from GGH13 is implicitly defined as $h = \langle \mathbf{t}^*, \mathbf{s}^*\rangle$, which is indeed a mid-size element.) It is easy to see that this provides the functionality of a graded-encoding scheme, where a level-$[k]$ encoding $A^{[k]}$ can be tested for zero by checking that

$$\left\| \left[\mathbf{s} \times A^{[k]} \times \mathbf{t}\right]_q \right\| \ll q.$$

On the other hand, it seems hard to obtain a native GGH13-encoding of zero even if we are given matrices $A^{\{i\}}$ that encode zero, so we could naively hope that the zeroizing attacks from [10] do not apply. Unfortunately, we show that this is not the case.

4.1 The Updated Weak-DL Attack.

As in the original "weak-discrete-logarithm" attack from [10], we use encodings of "double zeros" in order to recover the ideal $I_g = gR$ and then use the knowledge of that ideal together with "single zeros" to attack other encodings. We use below the same "attack set" terminology as in our CLT attacks.

Attack Sets. As in all other attacks in the current work, we are given three sets of encodings, at levels corresponding to a partition of $[k]$ so that they product is the top level. Let $S_1, S_2, S_3 \subset [k]$ be disjoint, and their union is $[k]$. Specifically assume that we are given the following encoding.

- Many level-S_1 encodings of zero

$$\mathcal{A}^{S_1} = \left\{ A_j^{S_1} = \left[T \times A_j^* \times T^{-1}/\prod_{i\in S_1} z_i\right]_q : \mathbf{s}^* A_j^* = \mathbf{0} \pmod{gR} \right\}_j ;$$

- Many level-S_2 encodings of zero and the two "target encodings" at level S_2 (encoding two random scalars $\delta_0, \delta_1 \in R$)

$$\mathcal{Z}^{S_2} = \left\{ Z_j^{S_2} = \left[T \times Z_j^* \times T^{-1}/\prod_{i\in S_2} z_i\right]_q : \mathbf{s}^* Z_j^* = \mathbf{0} \pmod{gR} \right\}_j ,$$

$$\mathcal{B}^{S_2} = \left\{ B_\sigma^{S_2} = \left[T \times B_\sigma^* \times T^{-1}/\prod_{i\in S_2} z_i\right]_q \right\}_{\sigma=0,1} ,$$

such that $\mathbf{s}^* B_\sigma^* = \delta_\sigma \mathbf{s}^* \pmod{gR}$ for $\sigma = 0, 1$, where the δ's are small.

- Many level-S_3 encodings of nonzero elements

$$\mathcal{C}^{S_3} \;=\; \left\{ C_j^{S_3} = \left[T \times C_j^* \times T^{-1} / \prod_{i \in S_3} z_i \right]_q : \mathbf{s}^* C_j^* = \varepsilon_j \mathbf{s}^* \pmod{gR} \right\}_j .$$

  for small scalars $\varepsilon_j \in R$, $\varepsilon_j \notin gR$.

Given these encodings, we show how to recover the ratio $\delta_0/\delta_1 \pmod{gR}$. The set $\mathcal{Z}^{S_2}$ is only used to recover (a representation of) the ideal $I_g = gR$. Below we first assume that we know I_g and show how to recover the ratio $\delta_0/\delta_1 \pmod{gR}$, and later explain how I_g can be recovered using $\mathcal{Z}^{S_2}$.

Observe that since the $A_j^{S_1}$'s are encodings of zero then they cancel the term g^{-1} in the vector $\mathbf{s}$. Namely there exist small vectors $\mathbf{u}_j \in R^n$ such that $\mathbf{s}^* A_j^* = g \cdot \mathbf{u}_j$, and therefore $\left[\prod_{i \in S_1} z_i \cdot \mathbf{s} \times A_j^{S_1} \times T\right]_q = \mathbf{s}^* A_j^* / g = \mathbf{u}_j$. Let us also denote below $\mathbf{v}_j = \left[\prod_{i \in S_3} z_i \cdot T \times C_j^S \times \mathbf{t}\right]_q$ and notice $\mathbf{v}_j = C_j^{S,*} \mathbf{t}^*$ (equality in R) and $\mathbf{v}_j$ is the small. Then for all i, σ, j we get

$$W_\sigma[i,j] := \left[\mathbf{s} \times A_i^{S_1} \times B_\sigma^{S_2} \times C_j^{S_3} \times \mathbf{t}\right]_q \;=\; \mathbf{u}_i \times B_\sigma^* \times \mathbf{v}_j \text{ (equalities over } R).$$

Denote by $U, V \in R^{n \times n}$ the matrices with $\mathbf{u}_j$'s as rows and $\mathbf{v}_j$'s as columns, respectively, and also define the two $n \times n$ matrices $W_\sigma = W_\sigma[i,j]_{i,j}$. Then by definition we have $W_\sigma = U \times B_\sigma^* \times V$ for $\sigma = 0, 1$.

Assuming that we know the plaintext space ideal I_g and that W_1 is invertible modulo I_g, we can proceed similarly to the CLT attacks by computing the eigenvalues of $W = W_0 \times W_1^{-1}$, but this time *modulo* I_g, and noting that the ratio δ_0/δ_1 must be one of these eigenvalues. (If W_1 is not invertible modulo I_g, we can still recover the same by finding the roots of the polynomial $p(\lambda) = \lambda W_1 - W_0$.) This implies that as long as we know the ideal I_g (and can find roots of polynomials modulo I_g), then similarly to the original "weak-discrete-log" attack from [10] we can recover any encoded value δ at level S_2 up to a multiplicative factor of $1/\delta_1$.

If we are not explicitly given the ideal I_g, then we can use the set $\mathcal{Z}^{S_2}$ to recover it. Similarly to above we define

$$X_\ell[i,j] := \left[\mathbf{s} \times A_i^{S_1} \times Z_\ell^{S_2} \times C_j^{S_3} \times \mathbf{t}\right]_q \;=\; \mathbf{u}_i \times Z_\ell^* \times \mathbf{v}_j$$

and then we have $X_\ell = X_\ell[i,j]_{i,j} = U \times Z_\ell^* \times V$, with equalities over R

Computing the determinant of all the matrices W_σ and X_ℓ, we observe that they are all divisible by $det(UV)$, and with good probability this is the only common factor. We can therefore take the GCD (over the ideals) to recover and eliminate the factor $det(UV)$, thereby getting the ideals that are generated by all the elements $t_\ell := det(Z_\ell^*)$. Recalling that all the Z_ℓ^*'s are singular modulo I_g (since $\mathbf{s}^* \times Z_\ell^* \in gR$), we have that g divides all the t_ℓ's, and again w.h.p. this is the only factor that divides them all. We therefore can use GCD again to recover the ideal I_g from all the ideals $I_{t_\ell} = t_\ell R$.

5 Conclusions

In this work we extended the recent CHLRS zeroizing attacks to many new settings, and also illustrated some of the limitations of this attack technique. The underlying message of recent attacks is that for current multilinear-map candidates, successful zero-tests give the adversary equations over the base ring (i.e. the integers or the ring of integers in a number field). Understanding the security of these candidates therefore hinges on a better understanding of which types of systems of nonlinear equations can be solved efficiently.

References

[1] B. Applebaum and Z. Brakerski. Obfuscating circuits via composite-order graded encoding. In Y. Dodis and J. B. Nielsen, editors, *Theory of Cryptography - TCC'15, Part II*, volume 9015 of *Lecture Notes in Computer Science*, pages 528–556. Springer, 2015. `http://eprint.iacr.org/2015/025`.

[2] S. Badrinarayanan, E. Miles, A. Sahai, and M. Zhandry. Post-zeroizing obfuscation: The case of evasive circuits. Cryptology ePrint Archive, Report 2015/167, 2015. `http://eprint.iacr.org/`.

[3] B. Barak, S. Garg, Y. T. Kalai, O. Paneth, and A. Sahai. Protecting obfuscation against algebraic attacks. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 221–238. Springer, 2014.

[4] D. Boneh, D. J. Wu, and J. Zimmerman. Immunizing multilinear maps against zeroizing attacks. Cryptology ePrint Archive, Report 2014/930, 2014. `http://eprint.iacr.org/`.

[5] Z. Brakerski and G. N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In Y. Lindell, editor, *Theory of Cryptography - TCC 2014*, volume 8349 of *Lecture Notes in Computer Science*, pages 1–25. Springer, 2014.

[6] J. H. Cheon, K. Han, C. Lee, H. Ryu, and D. Stehlé. Cryptanalysis of the multilinear map over the integers. In E. Oswald and M. Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 3–12. Springer, 2015. `http://eprint.iacr.org/2014/906`.

[7] J. Coron, T. Lepoint, and M. Tibouchi. Practical multilinear maps over the integers. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013*, volume 8042 of *Lecture Notes in Computer Science*, pages 476–493. Springer, 2013.

[8] J. Coron, T. Lepoint, and M. Tibouchi. Cryptanalysis of two candidate fixes of multilinear maps over the integers. Cryptology ePrint Archive, Report 2014/975, 2014. `http://eprint.iacr.org/`.

[9] J. Coron, T. Lepoint, and M. Tibouchi. New multilinear maps over the integers. In R. Gennaro and M. Robshaw, editors, *CRYPTO 2015*, Lecture Notes in Computer Science. Springer, 2015. To appear.

[10] S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In T. Johansson and P. Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2013.

[11] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS 2013*, pages 40–49. IEEE Computer Society, 2013.

[12] S. Garg, C. Gentry, S. Halevi, and M. Zhandry. Fully secure functional encryption without obfuscation. Cryptology ePrint Archive, Report 2014/666, 2014. `http://eprint.iacr.org/`.

[13] C. Gentry, S. Halevi, H. K. Maji, and A. Sahai. Zeroizing without zeroes: Cryptanalyzing multilinear maps without encodings of zero. Cryptology ePrint Archive, Report 2014/929, 2014. `http://eprint.iacr.org/`.

[14] C. Gentry, A. B. Lewko, A. Sahai, and B. Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. *IACR Cryptology ePrint Archive*, 2014:309, 2014.

[15] C. Gentry, A. B. Lewko, and B. Waters. Witness encryption from instance independent assumptions. In J. A. Garay and R. Gennaro, editors, *CRYPTO 2014*, volume 8616 of *Lecture Notes in Computer Science*, pages 426–443. Springer, 2014.

[16] S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[17] J. C. Hansen and E. Schmutz. How random is the characteristic polynomial of a random matrix? *Math. Proc. Camb. Phi. Soc.*, 114:507–515, 1993.

[18] Y. Hu and H. Jia. Cryptanalysis of GGH map. Cryptology ePrint Archive, Report 2015/301, 2015. `http://eprint.iacr.org/`.

[19] G. Kuba. On the distribution of reducible polynomials. *Math. Slovaca*, 59(3):349–356, 2009.

[20] E. Miles, A. Sahai, and M. Weiss. Protecting obfuscation against arithmetic attacks. *IACR Cryptology ePrint Archive*, 2014:878, 2014.

[21] R. Pass, K. Seth, and S. Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In *CRYPTO 2014*, pages 500–517, 2014.

[22] J. Zimmerman. How to obfuscate programs directly. In E. Oswald and M. Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 439–467. Springer, 2015. `http://eprint.iacr.org/2014/776`.

A A Refined Generic Model

The zeroising attacks from [10, 6] and the current work point to the inadequacy of the generic graded-encoding model as used in recent work. Indeed these attacks are highly algebraic and yet

they are not captured by that generic model. The main difference is that in the generic graded-encoding model the zero-test returns just a 0/1 bit, whereas in the GGH13/CLT13 schemes this test returns a full ring element.

We therefore propose to augment this generic model as follows: In addition to the standard interfaces in the graded-encoding model (with some plaintext space $R' = R/gR$), we will now also have a black-box-field over the ring R, except that we cannot directly obtain handles to this black-box field. Instead, the zero-test would serve as a translation device, letting us move things from the graded-encoding oracle to the black-box-field oracle.

In more detail, we would have the usual oracles to sample/encode elements in the graded-encoding scheme and to add and multiply them with the usual semantics. However, with each encoded element the graded-encoding oracle will also associate a "random element of R" from the appropriate coset. Namely, with each encoded value α the oracle will also have an associated $r_\alpha \in R$ and the intended semantics is that we use $\alpha + g \cdot r_\alpha$ to represent the coset $\alpha + gR$. The oracle keeps track of the representatives via the addition and multiplication operations of the graded-encoding scheme, by adding and multiplying the representatives in the ring R.

Then, if the zero-test is called on an encoding of zero with representative $g \cdot r \in R$, then in addition to the bit 1 the oracle will also give us a handle to the element $r \in R$ in the black-box field. Namely, if we call it on an element which is divisible by g then it will divide by g and move it to the black-box field.

B Hardness Assumption with Binding Variables

Below we describe a relatively simple source-group hardness assumption involving "binding variables", similar to thte ones used, e.g., in [3, 22]. In this assumption we are given many low-level CLT13 encodings of zero, and yet we do not know how to break it. We stress that this assumption is not meant to be useful for any crypto applications, its goal is just to highlight the difficulty that such binding variables pose when trying to mount a zeroizing attack.

This assumption is staged in the "asymmetric multilinear setting" where the levels correspond to integer vectors rather than to simple integers. (This setting makes it more convenient to enforce the need to multiply by the binding variables.) It would be interesting either to show how to extend the attacks so as to break this hardness assumption, or to provide some evidence that this line of attacks is incapable of breaking it.

The setting in this hardness assumption consists of $4\binom{n}{2} + 2n + 1$ encoded variables: One is the target encoding, which is either a zero encoding $U^* \sim (g_i r_i^*, \ldots, g_n r_n^*)$ or an encoding of a random vector $U^* \sim (r_1^*, \ldots, r_n^*)$. There are $4\binom{n}{2}$ binding variables $U_{i,j,\sigma_1,\sigma_2}$ and $2n$ filler variables $V_{i,\sigma}$, all of them encoding zeros. That is, $U_{i,j,\sigma_1,\sigma_2} \sim (g_1 r^1_{i,j,\sigma_1,\sigma_2}, \ldots, g_n r^n_{i,j,\sigma_1,\sigma_2})$ and $V_{i,\sigma} \sim (g_1 r^1_{i,\sigma}, \ldots, g_n r^n_{i,\sigma})$.

The levels of these encodings are designed to ensure that the only way to get an encoding of zero at the top level is multiply together the target encoding U^* and other variables that are consistent with some n-bit string x. Namely, the n filler encodings V_{i,x_i}, $i = 1, \ldots, n$, and the $\binom{n}{2}$ biding encodings U_{i,j,x_i,x_j}, $1 \leqslant i < j \leqslant n$. The level structure of the scheme consists of integer vectors of dimension $1 + 2n + \binom{n}{2}$, as described below. In this description we let e_i denote the i'th unit vector, with indexing beginning at zero.

- The target variable U^* is encoded relative to level $e_0 = (1, 0, \ldots, 0)$.

- The filler variable $V_{i,0}$ is encoded relative to level $(n-1)\cdot e_i$ (i.e. $n-1$ in position i and 0 elsewhere), and the filler variable $V_{i,1}$ is encoded relative to level $(n-1)\cdot e_{i+n}$.
- Let k be the index of the pair (i,j). Then the levels of all four binding variable $U_{i,j,*,*}$'s have 1 in position k and also 1's in two of the four positions $i, j, n+i, n+j$:
  - $U_{i,j,0,0}$ is encoded at level $e_k + e_{n+i} + e_{n+j}$;
  - $U_{i,j,1,0}$ is encoded at level $e_k + e_i + e_{n+j}$;
  - $U_{i,j,0,1}$ is encoded at level $e_k + e_{n+i} + e_j$; and
  - $U_{i,j,1,1}$ is encoded at level $e_k + e_i + e_j$.
- The top level (where we perform zero-test) corresponds to the vector

$$e_0 + (n-1)(\sum_{k=1}^{2n} e_k) + \sum_{k=1}^{\binom{n}{2}} e_{2n+k} = (1, \underbrace{n-1,\ldots,n-1}_{2n \text{ entries}}, \underbrace{1,\ldots,1}_{\binom{n}{2} \text{ entries}}).$$

Recall that multiplying encodings add their levels (as vectors over the integers). So we want to show that the only way to hit the target level is by adding levels of the input encodings in a manner consistent with some n-bit string x. Clearly we need to take one from every four $U_{i,j,*,*}$'s to get all the ones in the last $\binom{n}{2}$ positions. These $\binom{n}{2}$ vectors contributes a "weight" of $2\binom{n}{2} = n(n-1)$ in positions $1,\ldots,n$, so we need to add n more fillers to make up the missing "weight" of $n(n-1)$ in these positions.

But we cannot add $V_{i,0}$ if for some j we used $u_{i,j,1,*}$, and similarly cannot add $V_{i,1}$ if for some j we used $u_{i,j,0,*}$ (lest we overshoot the $n-1$ target in positions i or $n+i$). Hence it must be the case that for every i we used exactly one of $V_{i,0}$, $V_{i,1}$, and for all i,j if we used V_{i,σ_1} and V_{j,σ_2} then we also use $U_{i,j,\sigma_1,\sigma_2}$.

B.1 Attacking Simplified Binding Variables

The reason that binding variables make it hard to mount zeroizing attacks is that for every input $t = xyz$ that leads to zero, we no longer get the nice separated expression $w_t = \mathbf{a}_x \times B_y \times \mathbf{c}_z$ as before, but rather it is multiplied by these variables. Namely rather than the four-wise inner-product that we use in the attacks in the work, we now have many more vectors that come from the binding variables, and we cannot partition these vectors in a nice way.

Below we show, however, that in a simplified setting where these vectors are replaced by individual integers, we can use GCD computations to remove them and then mount the standard zeroizing attacks. Assume that our zero-tested top-level zeros are given by the following expression:

$$w_t = \mathbf{a}_x \times B_y \times \mathbf{c}_z \cdot \prod_{i<j} u_{i,j,t_i,t_j}. \tag{1}$$

Let X, Y, Z denote the bit positions belonging to the x, y, z parts of the input, respectively, and consider a set of inputs that vary x, z and keep y fixed, $t^k = x^k y z^k$. Since y is fixed then is means that for $i, j \in Y$ we always get the same two bit values for t_i^k, t_j^k (for all k). But for $(i,j) \in X \times Y$ and for $(i,j) \in Y \times Z$ we can have two different values for the pair (t_i^k, t_j^k) for different k's, and

for $(i,j) \in X \times X$, $(i,j) \in X \times Z$, and $(i,j) \in Z \times Z$ we can have all four values. Assume that our collection of inputs is such that for all i,j we have as much variability in the values (t_i^k, t_j^k) as possible (subject to y being fixed).

Also assume for simplicity that these binding variable are chosen to that for every i,j, the four variables u_{i,j,b_1,b_2} are pairwise co-prime. Then all the w_{t^k}'s have a factor of $U_y = \prod_{i,j \in Y} u_{i,j,y_i,y_j}$, but they are very unlikely to have any other common factors. Taking the GCD of all these w_{t^k}'s can therefore give us U_y.

Assuming that we can have such sets of inputs for every desirable partition of the input bits, we would be able to compute the product U_Y for every subset $Y \in [n]$. In particular we can compute it for every pair $\{i,j\} \subset [n]$, thereby recovering all the variables u_{i,j,b_1,b_2}. Then we can go back to Eqn. (1) and divide out these variables, thus recovering our simpler form and then running the attacks from above.