Cryptanalysis of CLT13 Multilinear Maps with Independent Slots

Jean-Sébastien Coron, Luca Notarnicola

University of Luxembourg
jean-sebastien.coron@uni.lu, luca.notarnicola@uni.lu

Abstract. Many constructions based on multilinear maps require independent slots in the plaintext, so that multiple computations can be performed in parallel over the slots. Such constructions are usually based on CLT13 multilinear maps, since CLT13 inherently provides a composite encoding space, with a plaintext ring $\bigoplus_{i=1}^{n} \mathbb{Z}/g_i\mathbb{Z}$ for small primes g_i's. However, a vulnerability was identified at Crypto 2014 by Gentry, Lewko and Waters, with a lattice-based attack in dimension 2, and the authors have suggested a simple countermeasure. In this paper, we identify an attack based on higher dimension lattice reduction that breaks the author's countermeasure for a wide range of parameters. Combined with the Cheon *et al.* attack from Eurocrypt 2015, this leads to a total break of CLT13 multilinear maps with independent slots. We also show how to apply our attack against various constructions based on composite-order CLT13, such as [FRS17]. Finally, we suggest a set of secure parameters for CLT13 multilinear maps that prevents our attack.

1 Introduction

Multilinear maps. In 2013, Garg, Gentry and Halevi described the first plausible construction of cryptographic multilinear maps based on ideal lattices [GGH13a]. Since then many amazing applications of multilinear maps have been found in cryptography, including program obfuscation [GGH$^+$13b]. Shortly after the publication of GGH13, an analogous construction over the integers was described in [CLT13], based on the DGHV fully homomorphic encryption scheme [DGHV10]. The GGH15 scheme is the third known family of multilinear maps, based on the LWE problem with encoding over matrices [GGH15].

In the last few years, many attacks have appeared against multilinear maps, and the security of multilinear maps is still poorly understood. An important class of attacks against multilinear maps are "zeroizing attacks", which can recover the secret parameters from encodings of zero, using linear algebra. For the non-interactive multipartite Diffie-Hellman key exchange, the zeroizing attack from Cheon *et al.* [CHL$^+$15] recovers all secret parameters from CLT13; the attack can also be extended to encoding variants where encodings of zero are not directly available [CGH$^+$15]. The zeroizing attack from [HJ16] also breaks the Diffie-Hellman key-exchange over GGH13. Finally, the key exchange over GGH15 was also broken in [CLLT16], using an extension of the Cheon *et al.* zeroizing attack.

Even though direct multipartite key exchange protocols are broken for the three known families of multilinear maps, more complex constructions based on multilinear maps are not necessarily broken, in particular indistinguishability obfuscation (iO); namely low-level encodings of zero are generally not available in iO constructions. However the Cheon *et al.* attack against CLT13 was extended in [CGH$^+$15] to matrix branching programs where the input can be partitioned into 3 independent sets. The attack was further extended in [CLLT17] to branching programs without a simple input partition structure, using a tensoring technique. For GGH13 based obfuscation, Miles, Sahai and Zhandry introduced "annihilation attacks" that can break a certain class of matrix branching programs [MSZ16]; the attack was later extended in [CGH17] to break the [GGH$^+$13b] obfuscation under GGH13, using a variant of the input partitioning attack. Finally, Chen, Vaikuntanathan and Wee described in [CVW18] an attack against iO over GGH15, based on computing the rank of a well chosen matrix. In general, the above attacks only apply against branching programs with a simple structure, and breaking more complex constructions (such as dual-input branching programs) is currently infeasible.

Multilinear maps with independent slots. Many constructions based on multilinear maps require independent slots in the plaintext, so that multiple computations can be performed in parallel over the slots when evaluating the multilinear map. For example [GLW14] and [GLSW15] use independent slots to obtain improved security reductions for witness encryption and obfuscation. Multilinear maps with independent slots were also used in the circuit based constructions of [AB15,Zim15], a promising approach for program obfuscation. The construction from [FRS17], which gives a powerful technique for preventing zeroizing attacks against iO, is also based on multilinear maps with independent slots.

The CLT13 multilinear map scheme inherently supports a composite integer encoding space, with a plaintext ring $\mathbb{Z}/G\mathbb{Z} \simeq \bigoplus_{i=1}^{n} \mathbb{Z}/g_i\mathbb{Z}$ for small secret primes g_i's and $G = g_1 \cdots g_n$. For example, in the construction from [FRS17], every branching program works independently modulo each g_i. In that case, the main difference with the original CLT13 is that the attacker can obtain encodings of subring elements which are zero modulo all g_i's except one; for example, in [FRS17] this would be done by carefully choosing the input so that all branching programs would evaluate to zero except one. Whereas in the original CLT13 construction, one never provides encodings of subring elements; instead one uses an "all-or-nothing" approach: either the plaintext element is zero modulo all g_i's, or it is non-zero modulo all g_i's (with high probability).

The attack and countermeasure from [GLW14]. At Crypto 2014, Gentry, Lewko and Waters observed that CLT13 with independent slots leads to a simple lattice attack in dimension 2, which efficiently recovers the (secret) plaintext ring $\bigoplus_{i=1}^{n} \mathbb{Z}/g_i\mathbb{Z}$ [GLW14, Appendix B]. Namely, when using CLT13 with independent slots, the attacker can obtain encodings where all slots are zero modulo g_i except one. For example, for a matrix branching program evaluation as in [FRS17], the result of the program evaluation could have the form:

$$A(x) \equiv \sum_{i=1}^{n} h_i \cdot (r_i + m_i \cdot (g_i^{-1} \bmod p_i)) \cdot \frac{x_0}{p_i} \pmod{x_0}$$

where $m_i = 0$ for all i except $m_j \neq 0$ for some $1 \leq j \leq n$. This implies:

$$g_j \cdot A(x) \equiv h_j(r_j g_j + m_j)\frac{x_0}{p_j} + \sum_{i \neq j} g_j h_i r_i \frac{x_0}{p_i} \pmod{x_0}$$

and therefore $g_j \cdot A(x) \bmod x_0$ is "small". This implies that we can recover g_j (while normally the g_i's are secret in CLT13) using lattice reduction in dimension 2. Moreover, once we know g_j, we can simply multiply the evaluation by g_j to obtain a "small" result, even if the evaluation of the branching program is non-zero modulo g_j; in particular, this cancels the effect of the protection against input partitioning from [FRS17].

The countermeasure considered in [GLW14, Appendix B] is to give many "buddies" to each g_i, so that we do not have a plaintext element which is non-zero modulo a single isolated g_i. Then, either an encoding is 0 modulo g_i and all its prime buddies g_j, or it is (whp) non-zero for all of them. In other words, instead of using individual g_i's to define the plaintext slots, every slot is defined modulo a product of θ prime g_i's, for some $1 \leq \theta < n$. Therefore, we obtain a total of n/θ plaintext slots (instead of n). While the above attack can be extended by multiplying $A(x)$ by the θ corresponding g_i's, for large enough θ the right-hand side of the equation is not "small" anymore and the attack is thwarted.

Our contributions. In this paper we identify an attack based on higher dimension lattice reduction that breaks the countermeasure from [GLW14, Appendix B] for a wide range of parameters, with significant impact on the security of CLT13 multilinear maps with independent slots. More precisely, our contributions are as follows:

1. **Analysis of the attack from [GLW14].** Our first contribution is to provide a theoretical study of the above attack, in order to derive a precise bound on θ as a function of the CLT13 parameters (there was no explicit bound in [GLW14]), where θ is the number of primes g_i's for each plaintext slot. We argue that, when ν denotes the number of bits that can be extracted from zero-testing in CLT13, the 2-dimensional lattice attack requires:
$$\alpha\theta < \frac{\nu}{2} \tag{1}$$
where α is the bit size of the g_i's.

2. **Breaking the countermeasure from [GLW14].** Our main contribution is to extend the 2-dimensional attack to break the countermeasure for larger values of θ. Our attack is based on higher dimension lattice reduction, by using a similar orthogonal lattice attack as in [NS99] for solving the hidden subset sum problem. Our attack uses ℓ encodings c_j where the corresponding plaintexts have only θ non-zero components modulo the g_i's (instead of $\ell = 1$ in the previous attack). Using a lattice attack in dimension $\ell + 1$, we show that our attack requires the approximate condition $\left(1 + \frac{1}{\ell}\right)\alpha\theta < \nu$. Therefore, for moderately large values of ℓ, we get the simpler condition:
$$\alpha\theta < \nu$$
which improves (1) by a factor 2.

   In the same vein, we show how to further improve this condition by considering products of encodings of the form $c_j \cdot d_k$ for $1 \leq j \leq \ell$ and $1 \leq k \leq d$, where as previously the plaintexts of the c_j's have only θ non-zero components modulo the g_i's. In that case, using a variant of the previous lattice attack (this time in dimension $\ell + d$), the bound improves to:
$$\alpha\theta = \mathcal{O}(\nu^2)$$
While the original attack from [GLW14] recovers the secret plaintext ring of CLT13, we additionally recover the plaintext messages m_j for the encodings c_j.

   We provide in Section 4.5 the result of practical experiments. For the original parameters of [CLT13], our attack takes a few seconds for $\theta = 40$, and a few hours for $\theta = 160$, while the original attack from [GLW14] only works for $\theta = 1$. In summary, our attack is more powerful than the attack in [GLW14], as it additionally recovers the plaintext messages, moreover for much larger values of θ. Finally, we suggest a set of secure parameters for CLT13 multilinear maps that prevents our extended attack. For $\lambda = 80$ bits of security, we recommend to take $\theta \geq 1789$.

3. **A total break of CLT13 with independent slots.** We show how to combine our attack with the Cheon *et al.* attack from [CHL$^+$15], in order to recover all secret parameters of CLT13. More precisely, our approach consists in applying the lattice attack to generate intermediate-level encodings of zero; then the Cheon *et al.* attack is applied on these newly-created encodings of zero, to recover all secret parameters.

4. **Application to CLT13-based constructions.** Finally we show how to apply our attack to several schemes based on CLT13 multilinear maps with independent slots, namely the constructions from [GLW14,GLSW15,Zim15] and [FRS17]. In particular, we extend our attack to the case of matrix encodings and matrix product evaluations, as typically used in program obfuscation.

Source code. We provide in Appendix B the source code of our attacks in Sage [S$^+$17].

2 The CLT13 Multilinear Map Scheme

We first recall the CLT13 multilinear map scheme over the integers [CLT13]. For $n \in \mathbb{Z}_{\geq 1}$, the instance generation of CLT13 generates n distinct secret "large" primes $p_1, \ldots, p_n$ of size η bits, and publishes the modulus $x_0 = \prod_{i=1}^{n} p_i$. We let γ denote the bit size of x_0; therefore $\gamma \simeq n \cdot \eta$. One also generates n distinct secret "small" prime numbers $g_1, \ldots, g_n$ of size α bits. The plaintext ring is composite, *i.e.* a plaintext is an element $\boldsymbol{m} = (m_1, \ldots, m_n)$ of the ring $\mathbb{Z}/G\mathbb{Z} \simeq \bigoplus_{i=1}^{n} \mathbb{Z}/g_i\mathbb{Z}$ where $G = \prod_{i=1}^{n} g_i$. Let $\kappa \in \mathbb{Z}_{\geq 1}$ be the multilinearity parameter. For $k \in \{1, \ldots, \kappa\}$, an encoding at level k of the plaintext $\boldsymbol{m}$ is an integer $c \in \mathbb{Z}$ such that

$$c \equiv \frac{r_i g_i + m_i}{z^k} \pmod{p_i} \ , \ \text{for all } 1 \leq i \leq n \tag{2}$$

for "small" random integers r_i of bit size ρ. The random mask $z \in (\mathbb{Z}/x_0\mathbb{Z})^{\times}$ is the same for all encodings. It is clear that two encodings at the same level can be added, and the underlying plaintexts get added in the ring $\mathbb{Z}/G\mathbb{Z}$. Similarly, the product of two encodings at level i and j gives an encoding of the product plaintexts at level $i + j$, as long as the numerators in (2) do not grow too large, *i.e.* they must remain smaller than each p_i.

For an encoding at the last level κ, one defines the following zero-testing procedure. The instance generation publishes the zero-testing parameter p_{zt}, defined by

$$p_{zt} = \sum_{i=1}^{n} h_i z^{\kappa} (g_i^{-1} \bmod p_i) \frac{x_0}{p_i} \bmod x_0 \ , \tag{3}$$

where $h_i \in \mathbb{Z}$ are "small" random integers of size n_h bits. Given an encoding c at the last level κ, we compute the integer:

$$\omega := p_{zt} \cdot c \bmod x_0 \equiv \sum_{i=1}^{n} h_i (r_i + m_i (g_i^{-1} \bmod p_i)) \frac{x_0}{p_i} \pmod{x_0} \tag{4}$$

and we consider that c encodes the zero message if ω is "small" compared to x_0. Namely, if $m_i = 0$ for all i, we obtain $\omega \equiv \sum_{i=1}^{n} h_i r_i \frac{x_0}{p_i} \pmod{x_0}$, and since the integers h_i and r_i are "small", the resulting ω will be "small" compared to x_0.

More precisely, let ρ_f be the maximum bit size of the noise r_i in the encodings. Then the integers $h_i r_i x_0 / p_i$ have size roughly $\gamma - \eta + n_h + \rho_f$, and therefore letting

$$\nu = \eta - n_h - \rho_f \ , \tag{5}$$

the integers $h_i r_i x_0 / p_i$ have size roughly $\gamma - \nu$ bits. Therefore, when $m_i = 0$ for all i, the integer ω has size roughly $\gamma - \nu$ bits; whereas when $m_i \neq 0$ for some i, we expect that ω is of full size modulo x_0, that is γ bits. The parameter ν in (5) corresponds to the number of bits that can be extracted from zero-testing; namely from (4), the ν most significant bits of ω only depend on the plaintext messages m_i, and not on the noise r_i. Note that to get a proper zero-testing procedure, one needs to use a *vector* of n elements p_{zt}; namely with a single p_{zt} there exist encodings c with $m_i \neq 0$ while $p_{zt} \cdot c$ is "small" modulo x_0. In the rest of the paper, for simplicity, we consider a single p_{zt}, as it is usually the case in constructions over CLT13 multilinear maps. We refer to [CLT13, Section 3.1] for the setting of the parameters.

3 Basic Attack against CLT13 with Independent Slots

Many constructions based on multilinear maps require independent slots in the plaintext, so that multiple computations can be performed in parallel over the slots when evaluating the multilinear map; see for example [GLW14,GLSW15] and [AB15,Zim15,FRS17]. The CLT13 multilinear maps inherently provide independent slots, as the plaintext ring is $\bigoplus_{i=1}^{n} \mathbb{Z}/g_i\mathbb{Z}$ for small secret primes $g_1, \ldots, g_n$. Therefore we can have independent computations performed over the n plaintext slots modulo g_i; for example, in the construction from [FRS17], every branching program works independently modulo each g_i.

The basic attack from [GLW14]. When using CLT13 with independent slots, the attacker can obtain encodings of plaintext elements where all slots are zero modulo g_i except one. For example, in the [FRS17] construction where each branching program works modulo g_i, the attacker can choose the input so that the resulting evaluation is 0 modulo all g_i's except one, say g_1, without loss of generality. Let c be a level-κ encoding of a plaintext $\boldsymbol{m} = (m_1, \ldots, m_n)$ where $m_i = 0$ for all $2 \leq i \leq n$. From (4) we obtain the following zero-testing evaluation:

$$\omega \equiv h_1 \cdot m_1 \cdot (g_1^{-1} \bmod p_1) \cdot \frac{x_0}{p_1} + \sum_{i=1}^{n} h_i \cdot r_i \cdot \frac{x_0}{p_i} \pmod{x_0}$$

This implies:

$$g_1 \cdot \omega \equiv h_1 \cdot m_1 \cdot \frac{x_0}{p_1} + \sum_{i=1}^{n} g_1 \cdot h_i \cdot r_i \cdot \frac{x_0}{p_i} \pmod{x_0}$$

and therefore $g_1 \cdot \omega \bmod x_0$ is significantly smaller than x_0, as the integers h_i and r_i are "small". This implies that we can recover g_1, and similarly the other g_i's using lattice reduction in dimension 2, while normally the g_i's are secret in CLT13. This eventually recovers the plaintext ring.

The countermeasure from [GLW14]. The following countermeasure was therefore suggested by the authors: instead of using individual g_i's to define the plaintext slots, every slot is defined modulo a product of θ prime g_i's, where $2 \leq \theta < n$. Therefore, a plaintext element cannot be non-zero modulo a single prime g_i; it has to be non-zero modulo at least θ primes g_i's. This gives a total of n/θ plaintext slots (instead of n); for simplicity we assume that θ divides n.

Therefore, the original plaintext ring $R = \mathbb{Z}/g_1\mathbb{Z} \times \cdots \times \mathbb{Z}/g_n\mathbb{Z}$ can be rewritten as $R = \bigoplus_{j=1}^{n/\theta} R_j$, where for all $1 \leq j \leq n/\theta$, the subrings R_j are such that $R_j \simeq \bigoplus_{i=1}^{\theta} \mathbb{Z}/g_{(j-1)\theta+i}\mathbb{Z}$. We can assume that the attacker can obtain encodings of random subring plaintexts in R_j for any $1 \leq j \leq n/\theta$. In that case, the attacker obtains an encoding c of $\boldsymbol{m} = (m_1, \ldots, m_n) \in R$ where $m_i \equiv 0 \pmod{g_i}$ for all $i \in \{1, \ldots, n\} \setminus \{(j-1)\theta + 1, \ldots, j\theta\}$. In that case we will say that $\boldsymbol{m}$ has non-zero support of length θ.

Analysis of the basic attack. In this section we analyze in more details the attack from [GLW14], and we derive an explicit bound on the parameter θ, as a function of the other CLT13 parameters. Given an integer $1 \leq \theta < n$ (the above attack is obtained for $\theta = 1$), we consider a message having non-zero support of length θ; that is, (without loss of generality) of the form $\boldsymbol{m} = (m_1, \ldots, m_n) \in \mathbb{Z}^n$ with $0 \leq m_i < g_i$ such that $m_i = 0$ for $\theta + 1 \leq i \leq n$, *i.e.* we assume that the non-zero support of $\boldsymbol{m}$ is located in the first slot. We consider a top level κ encoding c of $\boldsymbol{m}$, that is:

$$c \equiv \frac{r_i g_i + m_i}{z^\kappa} \pmod{p_i}, \quad 1 \leq i \leq n$$

with integers r_i of bit size ρ_f. From zero-testing, we obtain from (4):

$$\omega \equiv p_{zt} \cdot c \equiv \sum_{i=1}^{\theta} h_i (g_i^{-1} \bmod p_i) m_i \frac{x_0}{p_i} + \sum_{i=1}^{n} h_i r_i \frac{x_0}{p_i} \pmod{x_0}$$

By multiplying out by $g := \prod_{i=1}^{\theta} g_i$ we obtain

$$g\omega \equiv \sum_{i=1}^{\theta} h_i m_i \frac{g}{g_i} \frac{x_0}{p_i} + \sum_{i=1}^{n} g h_i r_i \frac{x_0}{p_i} \pmod{x_0},$$

$$g\omega \equiv U \pmod{x_0} \tag{6}$$

where $U = \sum_{i=1}^{\theta} h_i m_i (g/g_i)(x_0/p_i) + \sum_{i=1}^{n} g h_i r_i (x_0/p_i)$. Since the integers h_i and r_i are "small" in order to ensure correct zero-testing, the integer U is "small" in comparison to x_0. More precisely, the proposition below shows that if $g \cdot U$ is a bit smaller than x_0, then we can recover g and U by lattice reduction in dimension 2.

Proposition 1. *Let $g, \omega, U \in \mathbb{Z}_{\geq 1}$ and $x_0 \in \mathbb{Z}_{\geq 1}$ be such that $g\omega \equiv U \pmod{x_0}$, $\omega \in (\mathbb{Z}/x_0\mathbb{Z})^{\times}$ and $\gcd(U, g) = 1$. Assume that $g \cdot U < x_0/10$. Given ω and x_0 as input, one can recover g and U in polynomial time.*

Proof. Without loss of generality we can assume $g \leq U$, since otherwise we can apply the algorithm with $U\omega^{-1} \equiv g \pmod{x_0}$. Let $B \in \mathbb{Z}_{\geq 1}$ such that $U \leq Bg \leq 2U$. When the bit size of g and U is unknown, such a B can be found by exhaustive search in polynomial time. We consider the lattice $L \subseteq \mathbb{Z}^2$ of vectors (Bx, y) such that $x\omega \equiv y \pmod{x_0}$. From $g\omega \equiv U \pmod{x_0}$ it follows that L contains the vector $\boldsymbol{v} = (Bg, U)$. We show that $\boldsymbol{v}$ is a shortest non-zero vector in L.

By Minkowski's Theorem, we have $\lambda_1(L) \leq \sqrt{2\det(L)}$. From Hadamard's Inequality, with $\det(L) = Bx_0$, we obtain:

$$\lambda_2(L) \geq \frac{\det(L)}{\lambda_1(L)} \geq \frac{\sqrt{\det(L)}}{\sqrt{2}} = \frac{\sqrt{Bx_0}}{\sqrt{2}} > \sqrt{5BgU} \geq \sqrt{5}U.$$

Moreover, we have:

$$\|\boldsymbol{v}\| = ((Bg)^2 + U^2)^{1/2} \leq \sqrt{5}U.$$

This implies that $\|\boldsymbol{v}\| < \lambda_2(L)$ and therefore $\boldsymbol{v}$ is a multiple of a shortest non-zero vector in L: we write $\boldsymbol{v} = k\boldsymbol{u}$ with $\|\boldsymbol{u}\| = \lambda_1(L)$, and $k \in \mathbb{Z}\backslash\{0\}$. Letting $\boldsymbol{u} = (Bu_1, u_2)$, we have $g = ku_1$ and $U = ku_2$. Hence k divides both g and U. Since $\gcd(g, U) = 1$ one has $k = \pm 1$. This shows that $\boldsymbol{v}$ is a shortest non-zero vector of L.

By running Lagrange-Gauss reduction on the matrix of row vectors:

$$\begin{bmatrix} B & \omega \\ 0 & x_0 \end{bmatrix}$$

one obtains in polynomial time a length-ordered basis $(\boldsymbol{b}_1, \boldsymbol{b}_2)$ of L satisfying $\|\boldsymbol{b}_1\| = \lambda_1(L)$ and $\|\boldsymbol{b}_2\| = \lambda_2(L)$, which enables to recover g and U. □

Using the same notations as in Section 2, the integer $g = \prod_{i=1}^{\theta} g_i$ has approximate bit size $\theta \cdot \alpha$, while the integer U has an approximate bit size $\gamma - \eta + n_h + \rho_f + \theta\alpha$. From the condition $g \cdot U < x_0/10$ of Proposition 1, we obtain by dropping the term $\log_2(10)$, the simplified condition

$$\gamma - \eta + n_h + \rho_f + \theta \cdot \alpha + \theta \cdot \alpha < \gamma \ .$$

Writing as previously $\nu = \eta - n_h - \rho_f$ for the number of bits that can be extracted during zero testing, the attack works under the condition:

$$2\alpha\theta < \nu \tag{7}$$

where α is the bit size of the g_i's. In the next section we describe a high-dimensional lattice reduction attack with an improved bound on θ.

4 Our new Attack against CLT13 with Independent Slots

Outline of our new attack. Our new attack improves the bound on θ compared to the attack recalled in Section 3; it also enables to recover the underlying plaintext messages, instead of only the CLT13 plaintext ring. The main difference is that we work with several messages instead of a single one, using high-dimensional lattice reduction instead of dimension 2.

Let $\ell \geq 1$ be an integer. Assume that we have level-κ encodings c_j of plaintext elements $\boldsymbol{m}_j = (m_{j1}, \ldots, m_{jn})$ for $1 \leq j \leq \ell$, where each message has non-zero support of length θ. Without loss of generality, we can assume that $m_{ji} = 0$ for all $\theta + 1 \leq i \leq n$ and all $1 \leq j \leq \ell$. We consider the zero-testing evaluations $\omega_j = p_{zt} \cdot c_j \bmod x_0$ of these encodings, which gives as previously:

$$\omega_j \equiv \sum_{i=1}^{\theta} h_i(r_{ji} + m_{ji}(g_i^{-1} \bmod p_i))\frac{x_0}{p_i} + \sum_{i=\theta+1}^{n} h_i r_{ji} \frac{x_0}{p_i} \pmod{x_0} , \quad 1 \leq j \leq \ell$$

for integers r_{ji}. We can rewrite the above equation as:

$$\omega_j \equiv \sum_{i=1}^{\theta} \alpha_i \cdot m_{ji} + R_j \pmod{x_0} , \quad 1 \leq j \leq \ell$$

for some integers α_i, where for each evaluation ω_j, the integer R_j is "small" modulo x_0. We can see the above equation as an instance of a "noisy" hidden subset sum problem [NS99]. Namely, the weights α_i are hidden as in [NS99], but for each equation we have an additional hidden noisy term R_j. Moreover, the weights $\alpha_i = h_i \cdot (g_i^{-1} \bmod p_i) \cdot x_0/p_i$ have a special structure, instead of being random in [NS99]. Thanks to this special structure, using a variant of the orthogonal lattice approach from [NS99], we can recover the secret product $g = g_1 \cdots g_\theta$ and the plaintext elements m_{ji}, whereas in [NS99] the unknown m_{ji} can only be recovered in relatively small dimension.

4.1 Preliminaries on lattices

Let L be a lattice in $\mathbb{R}^d$ of rank $0 < n \leq d$. We recall that Hadamard's Inequality gives the following upper bound on the determinant of L:

$$\det(L) \leq \prod_{\boldsymbol{b} \in B} \|\boldsymbol{b}\|$$

for every basis B of L. Based on Hadamard's Inequality, we prove the following simple lemma.

Lemma 2. *Let $1 \leq n \leq d$ be integers and let $L \subseteq \mathbb{Z}^d$ be a lattice of rank n. Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n-1} \in L$ be linearly independent. Then for every vector $\boldsymbol{y} \in L$ not in the linear span of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n-1}$, one has $\|\boldsymbol{y}\| \geq \det(L)/\prod_{i=1}^{n-1} \|\boldsymbol{x}_i\|$.*

Proof. Since $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n-1}, \boldsymbol{y} \in L$ are linearly independent, they generate a rank-n sublattice L' of L and hence $\det(L) \leq \det(L')$ as $\det(L)$ divides $\det(L')$. By Hadamard's Inequality, $\det(L) \leq \det(L') \leq \|\boldsymbol{y}\| \cdot \prod_{i=1}^{n-1} \|\boldsymbol{x}_i\|$. The bound follows. □

We recall that the LLL algorithm [LLL82], given an input basis of L, produces a reduced basis of L with respect to the choice of a parameter $\delta \in]1/4, 1[$; we call such a basis *δ-reduced.* More precisely, we will use the following theorem.

Theorem 3. *Let $1 \leq n \leq d$ be integers and let $L \subseteq \mathbb{Z}^d$ be a lattice of rank n. Let $\{\boldsymbol{b}_i : 1 \leq i \leq n\}$ be a basis of L. Let $B \in \mathbb{Z}_{\geq 1}$ be such that $\|\boldsymbol{b}_i\|^2 \leq B$ for $1 \leq i \leq n$. Let $\delta \in]1/4, 1[$. Then the LLL algorithm with reduction parameter δ outputs a δ-reduced basis $\{\boldsymbol{b}'_i : 1 \leq i \leq n\}$ after $\mathcal{O}(n^5 d \log^3 B)$ operations. Moreover, the first vector in such a basis satisfies:*

$$\|\boldsymbol{b}'_1\| \leq c^{(d-1)/2} \|\boldsymbol{x}\|$$

for every non-zero $\boldsymbol{x} \in L$, and where $c = 1/(\delta - 1/4)$.

4.2 Our first lattice-based attack

Setting. In this section, we describe our first attack based on the hidden subset-sum problem. We consider plaintext elements $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_\ell \in \mathbb{Z}^n$ and write m_{ji} for the i-th entry of the j-th message, where $0 \leq m_{ji} < g_i$ for all $1 \leq i \leq n$ and $1 \leq j \leq \ell$. As previously, we assume that $m_{ji} = 0$ for all $\theta + 1 \leq i \leq n$. We write $\boldsymbol{M}$ for the matrix of row vectors $\boldsymbol{m}_j$ for $1 \leq j \leq \ell$; and we will denote its columns by $\hat{\boldsymbol{m}}_i$ for $1 \leq i \leq n$, that is, $\boldsymbol{M} = \left[\hat{\boldsymbol{m}}_1 \,\middle|\, \cdots \,\middle|\, \hat{\boldsymbol{m}}_n\right] \in \mathrm{Mat}_{\ell \times n}(\mathbb{Z})$. By construction, the vectors $\hat{\boldsymbol{m}}_i$ for $\theta + 1 \leq i \leq n$ are all zero. For $1 \leq j \leq \ell$, we let c_j denote an encoding of $\boldsymbol{m}_j$ at the last level κ:

$$c_j \equiv \frac{r_{ji} g_i + m_{ji}}{z^\kappa} \pmod{p_i} , \quad 1 \leq i \leq n$$

where $r_{ji} \in \mathbb{Z}$ are ρ_f-bit integers. Letting $\boldsymbol{c} = (c_j)_{1 \leq j \leq \ell}$, this gives a vector equation over $\mathbb{Z}^\ell$:

$$\boldsymbol{c} \equiv z^{-\kappa} \left(g_i \boldsymbol{r}_i + \hat{\boldsymbol{m}}_i\right) \pmod{p_i} , \quad 1 \leq i \leq n \tag{8}$$

for $\boldsymbol{r}_i = (r_{ji})_{1 \leq j \leq \ell}$. Let p_{zt} be the zero-testing parameter, as defined in (3). From zero-testing we obtain the following equations:

$$\omega_j \equiv c_j \cdot p_{zt} \equiv \sum_{i=1}^{\theta} h_i m_{ji} (g_i^{-1} \bmod p_i) \frac{x_0}{p_i} + \sum_{i=1}^{n} h_i r_{ji} \frac{x_0}{p_i} \pmod{x_0} , \ 1 \leq j \leq \ell$$

which can be rewritten as $\omega_j \equiv \sum_{i=1}^{\theta} \alpha_i m_{ji} + R_j \pmod{x_0}$ where we use the shorthand notations:

$$\alpha_i := h_i (g_i^{-1} \bmod p_i) \frac{x_0}{p_i} , \ 1 \leq i \leq \theta \tag{9}$$

and $R_j := \sum_{i=1}^{n} h_i r_{ji} \frac{x_0}{p_i}$ for $1 \leq j \leq \ell$. As a vector equation, this reads:

$$\boldsymbol{\omega} \equiv p_{zt} \cdot \boldsymbol{c} \equiv \sum_{i=1}^{\theta} \alpha_i \hat{\boldsymbol{m}}_i + \boldsymbol{R} \pmod{x_0} \tag{10}$$

with $\boldsymbol{\omega} = (\omega_j)_{1 \leq j \leq \ell}$; for $1 \leq i \leq \theta$ the vectors $\hat{\boldsymbol{m}}_i$ are as above and $\boldsymbol{R} = (R_j)_{1 \leq j \leq \ell} = \sum_{i=1}^{n} h_i \frac{x_0}{p_i} \boldsymbol{r}_i$. In the above equation, the components of $\boldsymbol{R}$ have approximate bit size $\rho_R = \gamma - \eta + n_h + \rho_f$. Using as previously $\nu = \eta - n_h - \rho_f$ as the number of bits that can be extracted, we have therefore $\rho_R = \gamma - \nu$.

Equation (10) is similar to an instance of the hidden subset sum problem. Namely, while the vector $\boldsymbol{\omega}$ and the integers x_0 and θ are available to the attacker, the weights α_i and the vectors $\hat{\boldsymbol{m}}_i$ are hidden. As opposed to [NS99], there is an additional "noisy" vector $\boldsymbol{R}$ which is zero in [NS99]. Moreover by (9), the coefficients α_i have a special structure induced by the CLT13 scheme, while in [NS99] these are random coefficients. Therefore, the orthogonal lattice attack as described in [NS99] does not directly apply to Equation (10). Instead, we show a variant attack that recovers the secret CLT13 plaintext ring and the plaintexts $\{\hat{\boldsymbol{m}}_i : 1 \leq i \leq \theta\}$.

The orthogonal lattice attack. We consider the lattice L of vectors $(B\boldsymbol{u}, v) \in \mathbb{Z}^{\ell+1}$ such that $(\boldsymbol{u}, v)$ is orthogonal to $(\boldsymbol{\omega}, 1)$ modulo x_0, where $B \in \mathbb{Z}_{\geq 1}$ is a scaling factor. Since L contains the sublattice $x_0\mathbb{Z}^{\ell+1}$, it has dimension $\ell + 1$. This gives from (10), for every $(B\boldsymbol{u}, v) \in L$:

$$\langle \boldsymbol{u}, \boldsymbol{\omega} \rangle + v \equiv \sum_{i=1}^{\theta} \alpha_i \langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle + \langle \boldsymbol{u}, \boldsymbol{R} \rangle + v \equiv 0 \pmod{x_0}$$

and therefore the vector $(\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_1 \rangle, \ldots, \langle \boldsymbol{u}, \hat{\boldsymbol{m}}_\theta \rangle, \ \langle \boldsymbol{u}, \boldsymbol{R} \rangle + v)$ is orthogonal modulo x_0 to the vector $\boldsymbol{a} = (\alpha_1, \ldots, \alpha_\theta, \ 1)$. To obtain balanced components, we use another scaling factor $C \in \mathbb{Z}_{\geq 1}$ and we consider the vector:

$$\boldsymbol{p}_{\boldsymbol{u},v} := (C\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_1 \rangle, \ldots, C\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_\theta \rangle, \ \langle \boldsymbol{u}, \boldsymbol{R} \rangle + v \,)$$

In the original orthogonal lattice attack from [NS99], if a vector $(B\boldsymbol{u}, v) \in L$ is short enough, then the associated vector $\boldsymbol{p}_{\boldsymbol{u},v} = (C\boldsymbol{x}, y)$ will also be short, and if $(\boldsymbol{x}, y)$ becomes shorter than the shortest non-zero vector orthogonal to $\boldsymbol{a}$ modulo x_0, we must have $\boldsymbol{p}_{\boldsymbol{u},v} = 0$, which implies $\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle = 0$ for all $1 \leq i \leq \theta$. We will see that in our setting, because of the specific structure of the coefficients α_i's, we only get $\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle \equiv 0 \pmod{g_i}$ for all $1 \leq i \leq \theta$. Therefore, by applying lattice reduction to L, we expect to recover the lattice of vectors $\boldsymbol{u}$ which are orthogonal to all $\hat{\boldsymbol{m}}_i$ modulo g_i; in particular, this will reveal the lattice determinant $g = \prod_{i=1}^{\theta} g_i$.

More precisely, we consider the lattice $A^{\perp}$ of vectors $(C\boldsymbol{x}, y) \in \mathbb{Z}^{\theta+1}$, such that $(\boldsymbol{x}, y)$ is orthogonal to $\boldsymbol{a} = (\alpha_1, \ldots, \alpha_\theta,\ 1)$ modulo x_0; therefore $\boldsymbol{p}_{\boldsymbol{u},v} \in A^{\perp}$. The lattice $A^{\perp}$ has dimension $\theta + 1$ and we have $\det(A^{\perp}) = C^{\theta} x_0$. As mentioned previously the coefficients α_i's in the vector $\boldsymbol{a}$ have a particular structure. Namely we have $\alpha_i = (g_i^{-1} \bmod p_i) h_i x_0 / p_i$, and therefore

$$g_i \cdot \alpha_i \equiv h_i \cdot \frac{x_0}{p_i} \pmod{x_0}$$

for all $1 \leq i \leq \theta$. Therefore the lattice $A^{\perp}$ contains the θ linearly independent short vectors $\boldsymbol{q}_i = (0, \ldots, 0, C g_i, 0, \ldots, 0, -s_i)$ where $s_i = h_i \cdot x_0 / p_i$. Using $C := 2^{\rho_R - \alpha}$, we get $\|\boldsymbol{q}_i\| \simeq C \cdot 2^{\alpha}$. From Lemma 2, if $\|\boldsymbol{p}_{\boldsymbol{u},v}\| < \det(A^{\perp}) / \prod_{i=1}^{\theta} \|\boldsymbol{q}_i\|$, then $\boldsymbol{p}_{\boldsymbol{u},v}$ must belong to the linear span generated by the $\boldsymbol{q}_i$'s; since the g_i's are distinct primes, this implies that it must belong to the sublattice generated by the $\boldsymbol{q}_i$'s. In that case, we have:

$$\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle \equiv 0 \pmod{g_i}\ , \quad 1 \leq i \leq \theta \tag{11}$$

From $\det(A^{\perp}) = C^{\theta} \cdot x_0$ and $\|\boldsymbol{q}_i\| \simeq C \cdot 2^{\alpha}$, this happens under the approximate condition:

$$\|\boldsymbol{p}_{\boldsymbol{u},v}\| < 2^{\gamma - \alpha \cdot \theta} \tag{12}$$

Conversely, let $\Lambda^{\perp}$ be the lattice of vectors $\boldsymbol{u} \in \mathbb{Z}^{\ell}$ satisfying (11), namely that are orthogonal to every $\hat{\boldsymbol{m}}_i$ modulo g_i for $1 \leq i \leq \theta$. This is a full-rank lattice of dimension ℓ and determinant $g = \prod_{i=1}^{\theta} g_i$. Therefore, we heuristically expect that the lattice $\Lambda^{\perp}$ contains ℓ linearly independent vectors of norm roughly $(\det \Lambda^{\perp})^{1/\ell} \simeq 2^{\alpha\theta/\ell}$. We show that from any short $\boldsymbol{u} \in \Lambda^{\perp}$, we can generate a vector $(\boldsymbol{u}, v)$ with small v, and orthogonal to $(\boldsymbol{\omega}, 1)$ modulo x_0, and therefore a short vector $(B\boldsymbol{u}, v) \in L$. For this we write $\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle = k_i g_i$ with $k_i \in \mathbb{Z}$, and we have:

$$\begin{aligned} \langle \boldsymbol{u}, \boldsymbol{\omega} \rangle + v &\equiv \sum_{i=1}^{\theta} \alpha_i \langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle + \langle \boldsymbol{u}, \boldsymbol{R} \rangle + v \equiv \sum_{i=1}^{\theta} k_i \cdot g_i \cdot \alpha_i + \langle \boldsymbol{u}, \boldsymbol{R} \rangle + v \pmod{x_0} \\ &\equiv \sum_{i=1}^{\theta} k_i \cdot s_i + \langle \boldsymbol{u}, \boldsymbol{R} \rangle + v \pmod{x_0} \end{aligned}$$

Therefore it suffices to let $v := -\langle \boldsymbol{u}, \boldsymbol{R} \rangle - \sum_{i=1}^{\theta} k_i \cdot s_i$ to obtain $\langle \boldsymbol{u}, \boldsymbol{\omega} \rangle + v \equiv 0 \pmod{x_0}$; the vector $(\boldsymbol{u}, v)$ is then orthogonal to $(\boldsymbol{\omega}, 1)$ modulo x_0, and therefore $(B\boldsymbol{u}, v) \in L$. We obtain $|v| \simeq \|\boldsymbol{u}\| \cdot 2^{\rho_R}$; therefore letting $B := 2^{\rho_R}$, we get $\|(B\boldsymbol{u}, v)\| \simeq 2^{\rho_R} \|\boldsymbol{u}\|$. In summary, the lattice L contains ℓ linearly independent vectors of norm roughly $2^{\rho_R + \alpha\theta/\ell}$.

Therefore, by applying lattice reduction to the lattice L, we expect that the ℓ first vectors $\{(B\boldsymbol{u}_i, v_i) : 1 \leq i \leq \ell\}$ of a reduced basis have norm roughly:

$$\|(B\boldsymbol{u}_i, v_i)\| \simeq 2^{\rho_R + \alpha\theta/\ell} \cdot 2^{\iota(\ell+1)}$$

where $2^{\iota(\ell+1)}$ is the Hermite factor for some positive constant ι depending on the lattice reduction algorithm. With $C = 2^{\rho_R - \alpha}$, we have $\|\boldsymbol{p}_{\boldsymbol{u}_i, v_i}\| \simeq \|(B\boldsymbol{u_i}, v_i)\|$ for all $1 \leq i \leq \ell$. From the condition given by (12), we have that $\boldsymbol{u}_i \in \Lambda^{\perp}$ if $\|\boldsymbol{p}_{\boldsymbol{u}_i, v_i}\| < 2^{\gamma - \alpha \cdot \theta}$; therefore we get the approximate condition:

$$\rho_R + \frac{\alpha\theta}{\ell} + \iota(\ell + 1) < \gamma - \alpha\theta$$

Using $\rho_R = \gamma - \nu$ where ν is the number of bits that can be extracted, this condition becomes

$$\alpha\theta\left(1+\frac{1}{\ell}\right) + \iota(\ell+1) < \nu \; . \tag{13}$$

When the above condition is satisfied, we expect to recover a basis $\{\boldsymbol{u}_i : 1 \le i \le \ell\}$ of the lattice $\Lambda^\perp$; then since $\det(\Lambda^\perp) = g = \prod_{i=1}^{\theta} g_i$, the absolute value of the determinant of the basis matrix reveals g.

We observe that from the above bound the parameter ℓ can be kept relatively small (say $\ell \simeq 10$), as a larger ℓ would not significantly improve the bound; this implies that the lattice dimension $\ell+1$ on which LLL is applied can be kept relatively small. Moreover for LLL, experiments show that $2^\iota \simeq 1.021$ so that ι is approximately 0.03, and therefore for such small ℓ the term $\iota \cdot (\ell+1)$ is negligible. Thus we can use the simpler approximate bound for our attack:

$$\alpha\theta < \nu \tag{14}$$

which gives a factor 2 improvement compared to the previous bound given by (7). In the next section we will see how to get a much more significant improvement with $\alpha\theta = \mathcal{O}(\nu^2)$.

A proven variant. The above algorithm is heuristic only. Below we describe a proven variant that can recover a vector $\boldsymbol{u}$ such that $\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i\rangle \equiv 0 \pmod{g_i}$ for all $1 \le i \le \theta$, using the LLL reduction algorithm. Although we only recover a single vector $\boldsymbol{u}$ instead of a lattice basis, this will be enough when combined with the Cheon *et al.* attack to recover all secret parameters of CLT13 (see Section 5). We provide the proof in Appendix A.

Proposition 4. *Let $\ell, \theta \in \mathbb{Z}_{\ge 1}$, $x_0 \in \mathbb{Z}_{\ge 1}$ and let $g_i \in \mathbb{Z}_{\ge 2}$ be distinct α-bit prime numbers for $1 \le i \le \theta$ and some $\alpha \in \mathbb{Z}_{\ge 1}$. For $1 \le i \le \theta$, let $\alpha_i \in \mathbb{Z}$ such that $g_i \cdot \alpha_i \equiv s_i \pmod{x_0}$, for $s_i \in \mathbb{Z}$ satisfying $|s_i| \le 2^{\rho_R}$, for some $\rho_R \in \mathbb{Z}_{\ge 1}$. For $1 \le i \le \theta$, let $\hat{\boldsymbol{m}}_i \in \mathbb{Z}^\ell$ be vectors with entries in $[0, g_i[\cap\mathbb{Z}$ for all i, and let $\boldsymbol{R} \in \mathbb{Z}^\ell$ such that $\|\boldsymbol{R}\|_\infty \le 2^{\rho_R}$. Let $\boldsymbol{\omega} \in \mathbb{Z}^\ell$ such that $\boldsymbol{\omega} \equiv \sum_{i=1}^{\theta} \alpha_i \hat{\boldsymbol{m}}_i + \boldsymbol{R} \pmod{x_0}$. Assume that*

$$\alpha\theta\left(1+\frac{1}{\ell}\right) + \frac{\ell+\theta}{2} + \log_2(\ell\sqrt{\ell+1}\cdot\theta) + 4 < \log_2(x_0) - \rho_R \; . \tag{15}$$

Given the integers $\ell, \theta, \rho_R, x_0$ and the vector $\boldsymbol{\omega}$, one can recover in polynomial time a vector $\boldsymbol{u} \in \mathbb{Z}^\ell$ such that $\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i\rangle \equiv 0 \pmod{g_i}$ for all $1 \le i \le \theta$, satisfying $\|\boldsymbol{u}\| \le 2^{\ell/2}\sqrt{\ell(\ell+1)}(\prod_{i=1}^{\theta} g_i)^{1/\ell}$.

We remark that by replacing $\log_2(x_0) - \rho_R$ by $\gamma - \rho_R = \nu$, we recover, up to additional logarithmic terms, the approximate bound established in (13).

4.3 Extended Orthogonal Lattice Attack

In this section we describe an extended attack that significantly improves the bound on θ established in (14). Let $\ell, d \ge 1$ be integers. As previously, we assume that we have encodings c_j of plaintext elements $\boldsymbol{m}_j = (m_{j1}, \ldots, m_{jn})$ for $1 \le j \le \ell$, where only the first θ components of each $\boldsymbol{m}_j$ are non-zero, that is $m_{ji} = 0$ for $\theta + 1 \le i \le n$. However, we assume that these encodings are at level $\kappa - 1$, and that we also have an additional set of d level-1 encodings $\{c'_k : 1 \le k \le d\}$ of plaintext elements $\boldsymbol{x}_k = (x_{k1}, \ldots, x_{kn})$ for $1 \le k \le d$. We can therefore obtain the following zero-testing evaluations:

$$\omega_{jk} \equiv c_j \cdot c'_k \cdot p_{zt} \equiv \sum_{i=1}^{\theta} h_i m_{ji} x_{ki} (g_i^{-1} \bmod p_i)\frac{x_0}{p_i} + \sum_{i=1}^{n} h_i r_{jki}\frac{x_0}{p_i} \pmod{x_0}$$

for some integers r_{jki}. As previously, we can rewrite this equation as:

$$\omega_{jk} \equiv \sum_{i=1}^{\theta} \alpha_{ik} m_{ji} + R_{jk} \pmod{x_0}$$

where we let

$$\alpha_{ik} = h_i x_{ki} (g_i^{-1} \bmod p_i) \frac{x_0}{p_i} \ , \quad 1 \le i \le \theta,\ 1 \le k \le d$$

and $R_{jk} = \sum_{i=1}^{n} h_i r_{jki} x_0 / p_i$ for all $1 \le j \le \ell$ and $1 \le k \le d$. As before, we denote by $\hat{\boldsymbol{m}}_i \in \mathbb{Z}^\ell$ the vector with components m_{ji} for $1 \le j \le \ell$, and similarly $\boldsymbol{\omega}_k$ and $\boldsymbol{R}_k$ the corresponding vectors in $\mathbb{Z}^\ell$. Therefore the previous equation can be rewritten as:

$$\boldsymbol{\omega}_k \equiv \sum_{i=1}^{\theta} \alpha_{ik} \hat{\boldsymbol{m}}_i + \boldsymbol{R}_k \pmod{x_0} \tag{16}$$

The difference with Equation (10) from our first lattice attack is that the vectors $\hat{\boldsymbol{m}}_i$ satisfy d equations for $1 \le k \le d$, instead of a single equation. With more constraints on the vectors $\hat{\boldsymbol{m}}_i$, we can therefore break the countermeasure from [GLW14] for much higher values of θ.

As previously, for a scaling factor $B \in \mathbb{Z}_{\ge 1}$, we consider the lattice L of vectors $(B\boldsymbol{u}, \boldsymbol{v}) \in \mathbb{Z}^{\ell+d}$ such that $(\boldsymbol{u}, \boldsymbol{v})$ is orthogonal to the d vectors $\{(\boldsymbol{\omega}_k, \boldsymbol{e}_k) : 1 \le k \le d\}$ modulo x_0, where $\boldsymbol{e}_k \in \mathbb{Z}^d$ are the unit vectors for $1 \le k \le d$. This gives for all $1 \le k \le d$ and all $(B\boldsymbol{u}, \boldsymbol{v}) \in L$:

$$\langle \boldsymbol{u}, \boldsymbol{\omega}_k \rangle + v_k \equiv \sum_{i=1}^{\theta} \alpha_{ik} \langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle + \langle \boldsymbol{u}, \boldsymbol{R}_k \rangle + v_k \equiv 0 \pmod{x_0}$$

and therefore the vector $(\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_1 \rangle, \ldots, \langle \boldsymbol{u}, \hat{\boldsymbol{m}}_\theta \rangle,\ \langle \boldsymbol{u}, \boldsymbol{R}_1 \rangle + v_1, \ldots, \langle \boldsymbol{u}, \boldsymbol{R}_d \rangle + v_d)$ is orthogonal modulo x_0 to the d vectors $\boldsymbol{a}_k = (\alpha_{1k}, \ldots, \alpha_{\theta k}, \boldsymbol{e}_k)$, where as previously $\boldsymbol{e}_k$ are the unit vectors for $1 \le k \le d$. Again, using a scaling factor $C \in \mathbb{Z}_{\ge 1}$, we let

$$\boldsymbol{p}_{\boldsymbol{u},\boldsymbol{v}} = (C\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_1 \rangle, \ldots, C\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_\theta \rangle,\ \langle \boldsymbol{u}, \boldsymbol{R}_1 \rangle + v_1, \ldots, \langle \boldsymbol{u}, \boldsymbol{R}_d \rangle + v_d)\,,$$

where $\boldsymbol{v} = (v_1, \ldots, v_d)$. As previously, we consider the lattice $\Lambda^\perp$ of vectors $(C\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{Z}^{\theta+d}$ orthogonal to the d vectors $\boldsymbol{a}_k$ modulo x_0; therefore $\boldsymbol{p}_{\boldsymbol{u},\boldsymbol{v}} \in \Lambda^\perp$. The lattice $\Lambda^\perp$ has dimension $\theta + d$ and determinant $C^\theta x_0^d$. As previously the coefficients α_{ik} in the vectors $\boldsymbol{a}_k$ have a special structure, since they satisfy the congruence relations

$$g_i \cdot \alpha_{ik} \equiv h_i \cdot x_{ik} \cdot \frac{x_0}{p_i} \pmod{x_0}$$

for all $1 \le i \le \theta$ and $1 \le k \le d$. Therefore letting $s_{ik} = h_i \cdot x_{ik} \cdot x_0 / p_i$, the lattice $\Lambda^\perp$ contains the θ short vectors $\boldsymbol{q}_i = (0, \ldots, 0, Cg_i, 0, \ldots, 0, -s_{i1}, \ldots, -s_{id})$ for $1 \le i \le \theta$. Using $C = 2^{\rho_R - \alpha}$, we get as previously $\|\boldsymbol{q}_i\| \simeq C \cdot 2^\alpha$. We expect a reduced basis of $\Lambda^\perp$ to have the first θ vectors with approximately the same norm as the $\boldsymbol{q}_i$, and to have the last d vectors with norm U satisfying $(C \cdot 2^\alpha)^\theta \cdot U^d \simeq \det(\Lambda^\perp)$. Using $\det(\Lambda^\perp) = C^\theta x_0^d$, this gives $U \simeq x_0 / 2^{\alpha\theta/d}$. This implies that, heuristically, if $\|\boldsymbol{p}_{\boldsymbol{u},\boldsymbol{v}}\| < U$, then $\boldsymbol{p}_{\boldsymbol{u},\boldsymbol{v}}$ must belong to the sublattice generated by the θ vectors $\{\boldsymbol{q}_i : 1 \le i \le \theta\}$. As previously, in that case we have that for all $1 \le i \le \theta$:

$$\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle \equiv 0 \pmod{g_i}\ . \tag{17}$$

Using the same reasoning as previously, we consider the lattice $\Lambda^\perp$ of vectors $\boldsymbol{u} \in \mathbb{Z}^\ell$ satisfying (17). Since $\Lambda^\perp$ heuristically contains ℓ linearly independent vectors of norm roughly $(\det \Lambda^\perp)^{1/\ell} \simeq 2^{\alpha\theta/\ell}$, the lattice L contains ℓ linearly independent vectors of norm roughly $2^{\rho_R + \alpha\theta/\ell}$. Therefore, by applying lattice reduction to the lattice L, we expect that the ℓ first vectors $\{(B\boldsymbol{u}_i, \boldsymbol{v}_i) : 1 \le i \le \ell\}$ of the basis have norm roughly:

$$\|(B\boldsymbol{u}_i, v_i)\| \simeq 2^{\rho_R + \alpha\theta/\ell} \cdot 2^{\iota(\ell+d)}$$

where $2^{\iota(\ell+d)}$ is the Hermite factor. With $B = 2^{\rho_R}$ and $C = 2^{\rho_R - \alpha}$, we have $\|\boldsymbol{p}_{\boldsymbol{u}_i,\boldsymbol{v}_i}\| \simeq \|(B\boldsymbol{u_i}, v_i)\|$. From the condition $\|\boldsymbol{p}_{\boldsymbol{u}_i,\boldsymbol{v}_i}\| < U$, we get the condition:

$$\rho_R + \frac{\alpha\theta}{\ell} + \iota(\ell + d) < \gamma - \frac{\alpha\theta}{d}$$

which gives using $\rho_R = \gamma - \nu$:

$$\alpha\theta \cdot \left(\frac{1}{\ell} + \frac{1}{d}\right) + \iota(\ell + d) < \nu \tag{18}$$

Remark that with $d = 1$ the previous bound gives Equation (13). Since (18) is symmetric in ℓ and d, the optimum is to take $\ell = d$. This gives the bound:

$$\frac{2\alpha\theta}{\ell} + 2\iota\ell < \nu \tag{19}$$

In particular, it follows that the attack requires $\ell > 2\alpha\theta/\nu$, and we must have:

$$\iota < \frac{\nu^2}{4\alpha\theta}$$

Heuristically achieving a Hermite factor of $2^{\iota 2\ell}$ requires $2^{\Omega(1/\iota)}$ using BKZ reduction with block-size $\beta = \omega(1/\iota)$ [HPS11]. The attack has therefore complexity $2^{\Omega(\alpha\theta/\nu^2)}$; heuristically the attack has polynomial-time complexity under the condition:

$$\alpha\theta = \mathcal{O}(\nu^2)$$

which significantly improves our previous bound given by (14). Conversely, one expects that the attack is prevented under the condition:

$$\theta = \omega\left(\frac{\nu^2}{\alpha}\log\lambda\right) \tag{20}$$

In Section 4.5 we provide concrete parameters for CLT13 multilinear maps with independent slots. We will see that Condition (20) requires a much higher value for θ than the condition $2\theta\alpha \ge \nu$ for preventing the [GLW14] attack. Namely for $\lambda = 80$ bits of security, the bound $2\theta\alpha \ge \nu$ already holds for $\theta = 2$, while a concrete variant of Condition (20) requires $\theta \ge 1789$.

Analogy of the attacks. In summary, we remark that our extended attacks share similarities with the 2-dimensional attack from Section 3. For $\ell, d \in \mathbb{Z}_{\ge 1}$ our extended lattice attack works by reducing the $(\ell + d)$-dimensional lattice

$$L_{(\ell,d)} = \{(B\boldsymbol{u}, \boldsymbol{v}) \in \mathbb{Z}^\ell \times \mathbb{Z}^d : \langle(\boldsymbol{u}, \boldsymbol{v}), (\boldsymbol{\omega}_k, \boldsymbol{e}_k)\rangle \equiv 0 \pmod{x_0}, 1 \le k \le d\} ,$$

where $B \in \mathbb{Z}_{\ge 1}$ is fixed. With this notation, the three attacks work by reducing the lattices $L_{(1,1)}$, $L_{(\ell,1)}$ and $L_{(\ell,d)}$, respectively. Note that $L_{(1,1)}$ is the lattice $\{(Bu, v) \in \mathbb{Z}^2 : u\omega + v \equiv 0 \pmod{x_0}\}$. For the extended attacks, the $\ell \times \ell$ top-left submatrix of a reduced basis of $L_{(\ell,d)}$ (divided by B) has determinant $\pm g$. Note that this coincides with the 2-dimensional case $\ell = d = 1$: the first entry (divided by B) of the first vector in a reduced basis equals $\pm g$ (*i.e.* a "1×1 submatrix" of determinant $\pm g$). As such, our higher-dimensional attacks are consistent generalizations of the 2-dimensional attack.

4.4 Recovering the plaintext elements

We show that our attack not only reveals the secret CLT13 plaintext ring but also the secret plaintext elements $\{\hat{\boldsymbol{m}}_i : 1 \le i \le \theta\}$. Namely, the orthogonal lattice attack not only recovers $g = \prod_{i=1}^{\theta} g_i$, but also constructs a matrix $\boldsymbol{U}$ of rows $\{\boldsymbol{u}_j : 1 \le j \le \ell\}$ orthogonal to the vectors $\{\hat{\boldsymbol{m}}_i : 1 \le i \le \theta\}$ modulo g_i and we can use this matrix $\boldsymbol{U}$ to recover the plaintext elements.

More precisely, we show that for each $1 \le i \le \theta$, we can recover the one-dimensional linear space generated by $\hat{\boldsymbol{m}}_i$ modulo g_i. The first step is to factor $g = \prod_{i=1}^{\theta} g_i$ to recover the primes g_i's; this is feasible if the g_i's are small enough.[1] Since we have a basis of the vectors $\boldsymbol{u}$ with $\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle \equiv 0 \pmod{g_i}$ for all $1 \le i \le \theta$, it suffices to compute the $\mathbb{Z}/g_i\mathbb{Z}$-kernel of the $\ell \times \ell$ basis matrix $\boldsymbol{U}$ of this lattice; assuming that $\hat{\boldsymbol{m}}_i \not\equiv 0 \pmod{g_i}$, we have that $\ker(\boldsymbol{U})$ over $\mathbb{Z}/g_i\mathbb{Z}$ has dimension 1 and therefore we recover a non-trivial multiple $\lambda_i \hat{\boldsymbol{m}}_i$ of the original messages $\hat{\boldsymbol{m}}_i$ modulo g_i, for $1 \le i \le \theta$. With the ECM [Len87] the factorization of $g = \prod_{i=1}^{\theta} g_i$ can be computed in time $\exp(c\sqrt{\alpha \ln \alpha})$ for some positive constant c and where α is the bit size of the g_i's, which gives a sub-exponential time attack.

Alternatively, to avoid the factorization of g, we can compute the integer right kernel of the matrix $\boldsymbol{U}_g = [\boldsymbol{U} \mid g\boldsymbol{I}_\ell]$, where $\boldsymbol{I}_\ell$ denotes the identity matrix in dimension ℓ. The following proposition shows that we can recover in polynomial time a non-trivial multiple of the vector $\hat{\boldsymbol{m}}$ such that $\hat{\boldsymbol{m}} \equiv \hat{\boldsymbol{m}}_i \pmod{g_i}$ for all $1 \le i \le \theta$.

Proposition 5. *Let $\ell, \theta \in \mathbb{Z}_{\ge 1}$. Let $g_1, \ldots, g_\theta$ be distinct prime numbers. For $1 \le i \le \theta$, let $\hat{\boldsymbol{m}}_i \in \mathbb{Z}^\ell \cap [0, g_i[^\ell$ be vectors such that $\hat{\boldsymbol{m}}_i \not\equiv 0 \pmod{g_i}$ for every $1 \le i \le \theta$. Let $\{\boldsymbol{u}_j : 1 \le j \le \ell\}$ be a basis of the lattice of vectors $\boldsymbol{u} \in \mathbb{Z}^\ell$ such that $\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle \equiv 0 \pmod{g_i}$ for all $1 \le i \le \theta$. Then, given $g = \prod_{i=1}^{\theta} g_i$ and the vectors $\{\boldsymbol{u}_j : 1 \le j \le \ell\}$, one can recover in polynomial time a vector $\lambda \cdot \hat{\boldsymbol{m}} \in \mathbb{Z}^\ell \cap [0, g[^\ell$ with $\gcd(\lambda, g) = 1$ such that $\hat{\boldsymbol{m}} \equiv \hat{\boldsymbol{m}}_i \pmod{g_i}$ for all $1 \le i \le \theta$.*

Proof. Let $\boldsymbol{U}$ denote the $\ell \times \ell$ matrix with rows $\boldsymbol{u}_j$ and let $\boldsymbol{U}_g = \boldsymbol{U} \bmod g \in \mathrm{Mat}_{\ell\times\ell}(\mathbb{Z}/g\mathbb{Z})$ denote the matrix obtained by reduction. Since $\langle \boldsymbol{u}_j, \hat{\boldsymbol{m}}_i \rangle \equiv 0 \pmod{g_i}$ for all i, j, the vectors $\hat{\boldsymbol{m}}_i$ are in the $(\mathbb{Z}/g_i\mathbb{Z})$-kernel of $\boldsymbol{U}$ for each i and since $\hat{\boldsymbol{m}}_i \not\equiv 0 \pmod{g_i}$, each kernel has dimension 1 over $\mathbb{Z}/g_i\mathbb{Z}$. From the Chinese Remainder Theorem, it follows that $\ker(\boldsymbol{U}_g)$ is a free $\mathbb{Z}/g\mathbb{Z}$-module of rank 1. Namely, there exists a unique vector $\hat{\boldsymbol{m}} \in \mathbb{Z}^\ell \cap [0, g[^\ell$ satisfying $\hat{\boldsymbol{m}} \equiv \hat{\boldsymbol{m}}_i \pmod{g_i}$ for all i. Then $\langle \boldsymbol{u}_j, \hat{\boldsymbol{m}}_i \rangle \equiv 0 \pmod{g_i}$ for all i, j if and only if $\langle \boldsymbol{u}_j, \hat{\boldsymbol{m}} \rangle \equiv 0 \pmod{g}$ for all j. In particular, there exists $\boldsymbol{k} \in \mathbb{Z}^\ell$ such that $(\hat{\boldsymbol{m}}, \boldsymbol{k})$ belongs to the $\mathbb{Z}$-kernel of the matrix $[\boldsymbol{U} \mid g\boldsymbol{I}_\ell]$. The integer kernel of this matrix can be computed in polynomial time from g and $\boldsymbol{U}$ and the left $\ell \times \ell$ submatrix of the Hermite normal form of the basis of the $\mathbb{Z}$-kernel gives in the first row a vector $\lambda\hat{\boldsymbol{m}}$ with $\lambda \in (\mathbb{Z}/g\mathbb{Z})^\times$. □

4.5 Concrete parameters and practical experiments

Concrete parameters. We provide concrete parameters for CLT13 multilinear maps with independent slots, for various values of the security parameter λ. We start from the same concrete parameters as provided in [CLT13]; we assume that the encoding noise is set so that the number of extracted bits is $\nu = 2\lambda + 12$; we take $\alpha = \lambda$. We then provide the minimum value of θ that ensures the same level of security against lattice attacks; see Table 1. As in [CLT13], the goal is to ensure that the best attack takes at least 2^λ clock cycles.

While in Table 1 the number of independent slots $n_{\mathsf{slots}} = \lfloor n/\theta \rfloor$ appears to be relatively small, we show in Section 6.3 that when dealing with matrices of encodings (as in matrix branching programs), the number of independent slots can be multiplied by a factor δ, where δ is the matrix dimension. Moreover it is always possible to increase the number of slots $n_{\mathsf{slots}} = \lfloor n/\theta \rfloor$ by increasing the value of n.

[1] For the concrete parameters provided in [CLT13], the g_i's are 80-bit primes; therefore the factorization is straightforward.

Instantiation	λ	n	η	$\gamma = n \cdot \eta$	ν	θ	n_{slots}
Small	52	1080	1981	$2.1 \cdot 10^6$	116	540	2
Medium	62	2364	2055	$4.9 \cdot 10^6$	136	1182	2
Large	72	8250	2261	$18.7 \cdot 10^6$	156	1472	5
Extra	80	26115	2438	$63.7 \cdot 10^6$	172	1789	14

Table 1. Concrete parameters for CLT13 multilinear maps with independent slots, for security parameter λ.

Practical experiments. We have run our extended attack from Section 4.3 with the "Extra" parameters of CLT13 from Table 1, for increasing values of θ. Note that for such parameters the original attack from [GLW14] only applies for $\theta = 1$. To improve efficiency we give as input to LLL a truncated matrix basis, where we keep only the ν most significant bits. Table 2 shows that our attack works in practice for much larger values of θ than the original attack from [GLW14]. We provide in Appendix B the source code in Sage [S$^+$17].

	θ	α	ν	$\ell = d$	lat. dim.	running time
Basic attack [GLW14]	1	80	172	1	2	ε
Extended attack (Section 4.3)	2	80	172	2	4	ε
Extended attack (Section 4.3)	40	80	172	39	78	10 s
Extended attack (Section 4.3)	100	80	172	100	200	11 min
Extended attack (Section 4.3)	160	80	172	163	326	11 hours

Table 2. Running time of our LLL-based attack, as a function of the parameter θ, for the “Extra” parameters of CLT13. The lattice dimension is $\ell + d = 2\ell$.

5 Application to the Cheon *et al.* Attack

In 2015, Cheon *et al.* published in [CHL$^+$15] a polynomial time attack against CLT13 resulting in a total break of the multipartite Diffie-Hellman key exchange protocol. The attack relies on the availability of low-level encodings of zero. In this section, we show how to adapt the Cheon *et al.* attack to the setting of CLT13 with independent slots: we assume that no encodings of zero are available to the attacker (otherwise the Cheon *et al.* attack would apply immediately), but as previously the attacker can obtain low-level encodings where only θ components of the plaintext are non-zero. In particular, this contributes to a cryptanalysis of CLT13 multilinear maps where no encodings of zero are available beforehand; this was considered as an open problem in [CLR15, Section 4].

5.1 The original Cheon *et al.* attack with encodings of zero

We first recall the basic Cheon *et al.* attack against CLT13. For simplicity, we take $\kappa = 3$; the attack is easily extended to $\kappa > 3$. Consider a set $\mathcal{A} = \{a_j : 1 \le j \le n\}$ of encodings of zero at level one, a pair $\mathcal{B} = \{b_0, b_1\}$ of encodings at level one, and a set $\mathcal{C} = \{c_k : 1 \le k \le n\}$ of encodings at level one. We write $a_j \equiv a_{ji}/z \pmod{p_i}$, $b_t \equiv b_{ti}/z \pmod{p_i}$, $c_k \equiv c_{ki}/z \pmod{p_i}$, with integers $a_{ji} \equiv 0 \pmod{g_i}$, for all $1 \le j, i, k \le n$ and $t \in \{0, 1\}$. We obtain the zero-testing evaluations:

$$\omega_{jk}^{(t)} = a_j b_t c_k p_{zt} \bmod x_0 = \sum_{i=1}^{n} h_i (g_i^{-1} \bmod p_i) a_{ji} b_{ti} c_{ki} \frac{x_0}{p_i}$$

where the equality holds over $\mathbb{Z}$ because the products $a_j b_t c_k$ are level-3 encodings of 0. This can be written in matrix form as

$$\omega_{jk}^{(t)} = \begin{bmatrix} a_{j1} \cdots a_{jn} \end{bmatrix} \begin{bmatrix} b_{t1} p_{zt,1} & & \\ & \ddots & \\ & & b_{tn} p_{zt,n} \end{bmatrix} \begin{bmatrix} c_{k1} \\ \vdots \\ c_{kn} \end{bmatrix} .$$

where $p_{zt,i} = h_i(g_i^{-1} \bmod p_i) x_0 / p_i$ for all $1 \leq i \leq n$. Writing out the matrices $\boldsymbol{W}_t = (\omega_{jk}^{(t)})_{1 \leq j,k \leq n}$ for $t \in \{0,1\}$, one obtains the integer matrix equalities $\boldsymbol{W}_t = \boldsymbol{A} \varDelta_t \boldsymbol{C}$ for $t \in \{0,1\}$, where the rows of $\boldsymbol{A}$ are the vectors $(a_{j1}, \cdots, a_{jn})_j$, the columns of $\boldsymbol{C}$ are the vectors $(c_{k1}, \cdots, c_{kn})_k$, and $\varDelta_t$ is the diagonal matrix $\mathrm{diag}(b_{t1} p_{zt,1}, \ldots, b_{tn} p_{zt,n})$.

Provided that at least one of $\boldsymbol{W}_0, \boldsymbol{W}_1$ is invertible over $\mathbb{Q}$ (say $\boldsymbol{W}_1$), one then evaluates over $\mathbb{Q}$ the matrix product:

$$\boldsymbol{W}_0 \cdot \boldsymbol{W}_1^{-1} = \boldsymbol{A}(\varDelta_0 \varDelta_1^{-1}) \boldsymbol{A}^{-1}$$

The attacker can thus compute the eigenvalues of $\boldsymbol{W}_0 \boldsymbol{W}_1^{-1}$, by factoring the characteristic polynomial (over $\mathbb{Q}$). By similarity of these matrices, these eigenvalues coincide with those of $\varDelta_0 \varDelta_1^{-1} = \mathrm{diag}(b_{01}/b_{11}, \ldots, b_{0n}/b_{1n})$, which are $\{b_{0i}/b_{1i} : 1 \leq i \leq n\}$. These ratios are now enough to factor x_0. Namely, writing the quotients $b_{0i}/b_{1i} = x_i/y_i$ for coprime integers x_i, y_i and using that $b_t \equiv b_{ti}/z \pmod{p_i}$, we obtain:

$$x_i b_1 - y_i b_0 \equiv (x_i b_{1i} - y_i b_{0i})/z \equiv 0 \pmod{p_i}$$

and therefore $\gcd(x_i b_1 - y_i b_0, x_0) = p_i$ with good probability. In summary, the Cheon *et al.* attack recovers all secret p_i's in polynomial time given the low-level encodings of zero $\{a_j : 1 \leq j \leq n\}$.

5.2 Adaptation of the Cheon *et al.* attack

We now show how to adapt the Cheon *et al.* attack when no encodings of zero are available, but the attacker can obtain low-level encodings where only θ components of the underlying plaintexts are non-zero. The attack is divided in two steps: first the attacker generates encodings of zero using the orthogonal lattice attack from Section 4, and then applies the original Cheon *et al.* attack to reveal the primes p_i.

We consider the following setting with $\kappa = 4$. Let $\ell \geq 1$; we consider a set $\mathcal{Y} = \{y_j : 1 \leq j \leq \ell\}$ of level-one encodings of messages $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_\ell$ where only the first θ components of each $\boldsymbol{m}_j$ are non-zero. Moreover, we consider as in the previous section three sets $\mathcal{A} = \{a_j : 1 \leq j \leq n\}$, $\mathcal{B} = \{b_0, b_1\}$ and $\mathcal{C} = \{c_k : 1 \leq k \leq n\}$ of level-one encodings of non-zero messages.

First step: orthogonal lattice attack. We show that the orthogonal lattice attack from Section 4.2 can compute a short vector $\boldsymbol{u} \in \mathbb{Z}^\ell$ such that $y' = \langle \boldsymbol{u}, \boldsymbol{y} \rangle$ is a level-1 encoding of zero, where $\boldsymbol{y} = (y_1, \ldots, y_\ell)$. We write for all $1 \leq j \leq \ell$:

$$y_j \equiv \frac{r_{ji} \cdot g_i + m_{ji}}{z} \pmod{p_i} , \quad 1 \leq i \leq n,$$

with the usual CLT13 notations, where $m_{ji} = 0$ for $\theta + 1 \leq i \leq n$. Note that our orthogonal lattice attack from Section 4.2 uses level-κ encodings; therefore it can be applied on level-κ encodings of the form:

$$e_j = y_j \cdot a_1 \cdot b_0 \cdot c_1 \bmod x_0$$

for level-one encodings a_1, b_0, c_1; we obtain:

$$e_j \equiv \frac{r'_{ji} \cdot g_i + m_{ji} \cdot x_i}{z^\kappa} \pmod{p_i} , \quad 1 \leq i \leq n$$

for some $r'_{ji} \in \mathbb{Z}$ and where x_i is the i-th component of the plaintext corresponding to the encoding $a_1 \cdot b_0 \cdot c_1$. Clearly, since the messages $\{\boldsymbol{m}_j : 1 \leq j \leq \ell\}$ have non-zero support of length θ, the messages $\{(m_{ji} \cdot x_i)_{1 \leq i \leq n} : 1 \leq j \leq \ell\}$ have non-zero support of length at least θ. Therefore, applying the orthogonal lattice attack from Section 4.2 on the encodings e_j (*i.e.* on the vector $\boldsymbol{\omega} = p_{zt} \cdot (e_j)_{1 \leq j \leq \ell} \bmod x_0$), we obtain a vector $\boldsymbol{u} \in \mathbb{Z}^\ell$ such that $\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \cdot x_i \rangle \equiv 0 \pmod{g_i}$ for all $1 \leq i \leq \theta$, where the $\hat{\boldsymbol{m}}_i$'s are the vectors $(m_{1i}, \ldots, m_{\ell i})$ for $1 \leq i \leq \theta$. Provided that $x_i \not\equiv 0 \pmod{g_i}$, this implies $\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle \equiv 0 \pmod{g_i}$ for all $1 \leq i \leq \theta$. Therefore, for all $1 \leq i \leq n$, we can write $\sum_{j=1}^{\ell} u_j m_{ji} = k_i g_i$ for integers k_i (and $k_i = 0$ for $\theta + 1 \leq i \leq n$). This gives:

$$y' = \sum_{j=1}^{\ell} u_j y_j \equiv g_i \left(\sum_{j=1}^{\ell} u_j r_{ji} + k_i \right) \cdot z^{-1} \pmod{p_i}, \quad 1 \leq i \leq n$$

and therefore y' is a level-1 encoding of zero, moreover with small noise since the vector $\boldsymbol{u}$ is short. Note that we only need a single vector $\boldsymbol{u}$; therefore the first step of the attack is proven by Proposition 4.

Second step: Cheon *et al.* attack. The second step consists in applying the Cheon *et al.* attack with the three sets $\mathcal{A}' = \{y' \cdot a_j : 1 \leq j \leq n\}$, $\mathcal{B} = \{b_0, b_1\}$ and $\mathcal{C} = \{c_k : 1 \leq k \leq n\}$. Since y' is an encoding of zero, all encodings in $\mathcal{A}'$ are encodings of zero, and we can apply the Cheon *et al.* attack on the three sets $\mathcal{A}'$, $\mathcal{B}$ and $\mathcal{C}$ to recover all secret primes p_i.

Since the orthogonal lattice attack more generally provides a set of ℓ vectors $\boldsymbol{u}_j \in \mathbb{Z}^\ell$ (instead of a single $\boldsymbol{u}$; and all satisfying $\langle \boldsymbol{u}_j, \hat{\boldsymbol{m}}_i \rangle \equiv 0 \pmod{g_i}$ for all i), a variant of the above attack with $\kappa = 3$ consists in starting from a set $\mathcal{A} = \{a_j : 1 \leq j \leq n\}$ of $\ell = n$ encodings where only the first θ components of the underlying plaintexts are non-zero, and then generating a set $\mathcal{A}' = \{\langle \boldsymbol{u}_j, \boldsymbol{a} \rangle : 1 \leq j \leq n\}$ of encodings of zero, with the vector of encodings $\boldsymbol{a} = (a_1, \ldots, a_n)$. One can then apply the Cheon *et al.* attack as previously on the three sets $\mathcal{A}'$, $\mathcal{B}$ and $\mathcal{C}$.

Note that the first step of the attack above (*i.e.* the generation of encodings of zero) uses the orthogonal lattice attack from Section 4.2 with the bound $\alpha\theta < \nu$. The attack from Section 4.3 is easily adapted to reach the improved bound $\alpha\theta = \mathcal{O}(\nu^2)$. In this case the attacker can obtain $\ell \cdot d$ level-two encodings of zero given by $\{\langle \boldsymbol{u}_j, \boldsymbol{c}_k \rangle : 1 \leq j \leq \ell, 1 \leq k \leq d\}$ where $\boldsymbol{c}_k$ is the vector of encodings $(c_j \cdot c'_k)_{1 \leq j \leq \ell}$ with the encodings $c_j \cdot c'_k$ considered in Section 4.3.

6 Cryptanalysis of constructions based on CLT13 with independent slots

In this section we show that our orthogonal lattice attack from Section 4 can be applied to various constructions over CLT13 multilinear maps with independent slots.

6.1 The multilinear subgroup elimination assumption from [GLW14,GLSW15]

The multilinear subgroup elimination assumption is used in [GLW14] for witness encryption and in [GLSW15] for constructing program obfuscation, based on a single assumption, independent of the particular circuit to be obfuscated. The multilinear subgroup elimination assumption is stated for a generic model of composite-order multilinear maps. Below, we show that our attacks break this assumption over CLT13 composite-order multilinear maps. We recall the definition from [GLSW15].

Definition 6 ((μ, ν)-multilinear subgroup elimination assumption [GLSW15]). *Let G be a group of order $N = a_1 \cdots a_\mu b_1 \cdots b_\nu c$ where $a_1, \ldots, a_\mu, b_1, \ldots, b_\nu, c$ are $\mu + \nu + 1$ distinct primes. We give out generators $x_{a_1}, \ldots, x_{a_\mu}, x_{b_1}, \ldots, x_{b_\nu}$ for each prime order subgroup except for the subgroup of order c. For each $1 \leq i \leq \mu$, we also give out a group element h_i sampled uniformly at random from the subgroup of order $c a_1 \cdots a_{i-1} a_{i+1} \cdots a_\mu$. The challenge term is a group element $T \in G$ that is either sampled uniformly at random from the subgroup of order $c a_1 \cdots a_\mu$ or uniformly at random from the subgroup of order $a_1 \cdots a_\mu$. The task is to distinguish between these two distributions of T.*

For simplicity, we consider the assumption with $\mu = 1$ and $\nu = 0$; the generalization of our attack to any (μ, ν) is straightforward. Therefore G is a group of order $a_1 c$. The challenge $T \in G$ is either generated at random from the subgroup of order $a_1 c$, or from the subgroup of order a_1. In the context of a CLT13 instantiation, we assume that $a_1 = \prod_{i=1}^{\theta} g_i$ and $c = \prod_{i=\theta+1}^{n} g_i$. In that case, a_1 and c are not primes, but the assumption can still be considered for non-prime a_i's, b_i's and c. The encoding T is then either generated from a random plaintext $m \in \bigoplus_{i=1}^{n} \mathbb{Z}/g_i\mathbb{Z}$, or from a random plaintext with only the θ first components non-zero, that is $m \equiv 0 \pmod{g_i}$ for $\theta + 1 \leq i \leq n$. It is easy to see that our attacks from Section 4.2 and Section 4.3 apply in this setting. Namely when only the first θ components of the plaintext m corresponding to the challenge T are non-zero, our attacks recover the product $a_1 = \prod_{i=1}^{\theta} g_i$, whereas the attacks will fail when m is a random plaintext. Therefore the challenge T is easily distinguished unless θ is large enough; more precisely, θ must satisfy the bound given by (20) to prevent the attack.

6.2 The Zimmerman circuit obfuscation scheme

At Eurocrypt 2015, Zimmerman described a technique to obfuscate programs without matrix branching programs, based on composite-order multilinear maps [Zim15]. A plaintext m belongs to $\mathbb{Z}/N\mathbb{Z}$ for a composite modulus $N = N_{\mathsf{ev}} \cdot N_{\mathsf{chk}}$, and the ring $\mathbb{Z}/N\mathbb{Z}$ is viewed as a direct product of an "evaluation" ring $\mathbb{Z}/N_{\mathsf{ev}}\mathbb{Z}$ to evaluate the circuit, and of a "checksum" ring $\mathbb{Z}/N_{\mathsf{chk}}\mathbb{Z}$ to prevent the adversary from evaluating a different circuit; those two evaluations are performed in parallel. Using the CLT13 notations from Section 2, one can let $N_{\mathsf{ev}} = \prod_{i=1}^{\theta} g_i$ and $N_{\mathsf{chk}} = \prod_{i=\theta+1}^{n} g_i$. In that case, the parameter θ must satisfy the bound given by (20) to prevent our lattice attack.

6.3 The FRS17 construction for preventing input partitioning attacks

At Asiacrypt 2017, Fernando, Rasmussen and Sahai described three constructions of "stamping functions" for preventing input-partitioning attacks on matrix branching programs [FRS17]. Their third construction is based on permutation hash functions and is instantiated over CLT13 multilinear maps with independent slots. More precisely, the permutation hash function is written as a matrix branching program, and multiple such permutation hash functions h_i are evaluated in parallel along with the main matrix branching program; this is to ensure that only inputs of the form $x \| h(x)$ can be evaluated, where $h(x) = h_1(x) \| \cdots \| h_t(x)$, which prevents input partitioning attacks.

Matrix branching programs. We first recall the construction of [GGH+13b] to obfuscate matrix branching programs. A matrix branching program BP of length n_p on ℓ-bit inputs $x \in \{0,1\}^\ell$ is evaluated by computing:

$$C(x) = \boldsymbol{b}_0 \cdot \prod_{i=1}^{n_p} \boldsymbol{B}_{i,x_{\mathsf{inp}(i)}} \cdot \boldsymbol{b}_{n_p+1} \tag{21}$$

where $\{\boldsymbol{B}_{i,b} : 1 \leq i \leq n_p, b \in \{0,1\}\}$ are $2n_p$ square matrices and $\boldsymbol{b}_0$ and $\boldsymbol{b}_{n_p+1}$ are bookend vectors; then $\mathsf{BP}(x) = 0$ if $C(x) = 0$, and $\mathsf{BP}(x) = 1$ otherwise. The integer $\mathsf{inp}(i) \in \{1, \ldots, \ell\}$ indicates which bit of x is read at step i of the product matrix computation. The matrices $\boldsymbol{B}_{i,b}$ are first randomized by choosing $n_p + 1$ random invertible matrices $\{\boldsymbol{R}_i : 0 \leq i \leq n_p\}$ and letting $\tilde{\boldsymbol{B}}_{i,b} = \boldsymbol{R}_{i-1}\boldsymbol{B}_{i,b}\boldsymbol{R}_i^{-1}$ for $1 \leq i \leq n_p$, with also $\tilde{\boldsymbol{b}}_0 = \boldsymbol{b}_0\boldsymbol{R}_0^{-1}$ and $\tilde{\boldsymbol{b}}_{n_p+1} = \boldsymbol{R}_{n_p}\boldsymbol{b}_{n_p+1}$. We obtain a randomized matrix branching program with the same result since the randomization matrices $\boldsymbol{R}_i$ cancel each other:

$$C(x) = \tilde{\boldsymbol{b}}_0 \cdot \prod_{i=1}^{n_p} \tilde{\boldsymbol{B}}_{i,x_{\mathsf{inp}(i)}} \cdot \tilde{\boldsymbol{b}}_{n_p+1}.$$

The entries of the matrices $\tilde{\boldsymbol{B}}_{i,b}$ are then independently encoded, as well as the bookend vectors $\tilde{\boldsymbol{b}}_0$ and $\tilde{\boldsymbol{b}}_{n_p}$. We obtain the matrices and vectors $\hat{\boldsymbol{B}}_{i,b} = \mathsf{Encode}_{\{i+1\}}(\tilde{\boldsymbol{B}}_{i,b})$, $\hat{\boldsymbol{b}}_0 = \mathsf{Encode}_{\{1\}}(\tilde{\boldsymbol{b}}_0)$ and

$\hat{\boldsymbol{b}}_{n_p+1} = \mathsf{Encode}_{\{n_p+2\}}(\tilde{\boldsymbol{b}}_{n_p+1})$. Here $\mathsf{Encode}_{\{i\}}(\cdot)$ denotes an encoding relative to the singleton i. The matrix branching program from (21) can then be evaluated over the encoded matrices:

$$\hat{C}(x) = \hat{\boldsymbol{b}}_0 \cdot \prod_{i=1}^{n_p} \hat{\boldsymbol{B}}_{i,x_{\mathsf{inp}(i)}} \cdot \hat{\boldsymbol{b}}_{n_p+1} \tag{22}$$

The resulting $\hat{C}(x)$ is then a last-level encoding that can be zero-tested to check if $C(x) = 0$, which reveals the output of the branching program $\mathsf{BP}(x)$, without revealing the matrices $\boldsymbol{B}_{i,b}$.

Parallel matrix branching programs: new attack bound. As explained in [FRS17], multiple branching programs can be evaluated in parallel with composite order multilinear maps; with the countermeasure from [GLW14] over CLT13, each branching program is then evaluated modulo a product of θ of the primes g_i's. Consider the result of a matrix branching program as in (21), with matrices of dimension $\delta \in \mathbb{Z}_{\geq 1}$. In the following we derive a new bound for θ, as a function of the matrix dimension δ.

Let $\ell, d \geq 1$ be integers. For a set of input messages x_{jk} we first rewrite (21) as:

$$y_{jk} = C(x_{jk}) \equiv \boldsymbol{m}_j \cdot \boldsymbol{v}_k^T \pmod{G} \tag{23}$$

for $1 \leq j \leq \ell$ and $1 \leq k \leq d$, where $G = \prod_{i=1}^{n} g_i$. Assume that all matrix branching programs evaluate to zero except one; in that case, without loss of generality we obtain $y_{jk} \equiv 0 \pmod{g_i}$ for all $\theta+1 \leq i \leq n$ and all j, k. This implies that we obtain the following zero-testing evaluations (with the usual notations):

$$\omega_{jk} \equiv \sum_{i=1}^{\theta} h_i (y_{jk} \bmod g_i)(g_i^{-1} \bmod p_i)\frac{x_0}{p_i} + \sum_{i=1}^{n} h_i r_{jki} \frac{x_0}{p_i} \pmod{x_0} \tag{24}$$

We rewrite (23) as $y_{jk} = \sum_{a=1}^{\delta} m_{ja} v_{ka}$. Letting $m_{jai} = m_{ja} \bmod g_i$ and $v_{kai} = v_{ka} \bmod g_i$ for all $1 \leq i \leq \theta$, we obtain $y_{jk} \equiv \sum_{a=1}^{\delta} m_{jai} v_{kai} \pmod{g_i}$ and therefore:

$$\omega_{jk} \equiv \sum_{i=1}^{\theta}\sum_{a=1}^{\delta} h_i m_{jai} v_{kai} (g_i^{-1} \bmod p_i)\frac{x_0}{p_i} + \sum_{i=1}^{n} h_i r_{jki}\frac{x_0}{p_i} \pmod{x_0}$$

Letting $\alpha_{iak} = h_i v_{kai}(g_i^{-1} \bmod p_i)x_0/p_i$ and $R_{jk} = \sum\limits_{i=1}^{n} h_i r_{jki} x_0/p_i$, this becomes

$$\omega_{jk} \equiv \sum_{i=1}^{\theta}\sum_{a=1}^{\delta} \alpha_{iak} \cdot m_{jai} + R_{jk} \pmod{x_0} \ ,$$

which gives, using the same vector notation as in Section 4:

$$\boldsymbol{\omega}_k \equiv \sum_{i=1}^{\theta}\sum_{a=1}^{\delta} \alpha_{iak} \hat{\boldsymbol{m}}_{ai} + \boldsymbol{R}_k \pmod{x_0} \ , \ 1 \leq k \leq d \tag{25}$$

where the vectors $\boldsymbol{\omega}_k$, $\hat{\boldsymbol{m}}_{ai}$ and $\boldsymbol{R}_k$ have dimension ℓ, for all $1 \leq k \leq d$ and $1 \leq a \leq \delta$.

We see that Equation (25) is similar to Equation (16) from Section 4.3; namely we obtain a noisy hidden subset-sum problem with dimension $\delta \cdot \theta$ instead of θ. Moreover, as in Section 4.3, the hidden vectors $\hat{\boldsymbol{m}}_{ai}$ must satisfy d equations (for $1 \leq k \leq d$). We can therefore replace θ by $\theta\delta$ in the bounds from Section 4.3 and the extended orthogonal lattice attack applies and reveals $g = \prod_{i=1}^{\theta} g_i$.

In particular, the attack has heuristic complexity $2^{\Omega(\alpha\theta\delta/\nu^2)}$ instead of $2^{\Omega(\alpha\theta/\nu^2)}$ and is prevented under the new condition:

$$\theta\delta = \omega\left(\frac{\nu^2}{\alpha}\log\lambda\right)$$

Therefore, compared to the condition given by (20), the parameter θ can be divided by the matrix dimension δ. This implies that the number of independent slots $n_{\mathsf{slots}} = n/\theta$ can be multiplied by δ. For example, with a matrix dimension $\delta = 10$ and the concrete parameters from Section 4.5, we can use $n_{\mathsf{slots}} = 140$ instead of $n_{\mathsf{slots}} = 14$.

Remark 7. We note that the attack from Section 4.2 still applies independently of the matrix dimension δ; namely it can be applied on ℓ of the evaluations ω_{jk} from (24), without taking into account the particular structure of the messages y_{jk} from (23). Therefore we must still ensure $\alpha\theta \geq \nu$ to prevent the attack from Section 4.2. We note that for the CLT13 parameters provided in Table 1, this constraint is always satisfied.

Application to the FRS17 construction. The [FRS17] scheme constructs a modified matrix branching program BP' that receives as input $u\|v_1 \ldots v_t$ and checks whether $v_i = h_i(u)$ for all $1 \leq i \leq t$, where the h_i's are permutation hash functions; in that case, BP' returns $\mathsf{BP}(u)$ where BP is the original branching program; otherwise, it returns some non-zero value. It is easy to generate an input $u\|v_1 \ldots v_t$ such that $\mathsf{BP}(u) = 0$ and $v_i = h_i(u)$ for all $1 \leq i \leq t$ except for some $i = i^\star$. This corresponds to the setting considered in the previous section, where only one of the $t+1$ parallel matrix branching program will evaluate to a non-zero value; this provides evaluations $y_{jk} = \mathsf{BP}'(x_{jk})$ where $y_{jk} \equiv 0 \pmod{g_i}$ for all $\theta + 1 \leq i \leq n$ and the above attack can recover the secret plaintext ring $\bigoplus_{i=1}^{n} \mathbb{Z}/g_i\mathbb{Z}$ of CLT13. The FRS17 construction should therefore be instantiated with the two above constraints on the parameter θ.

References

AB15. Benny Applebaum and Zvika Brakerski. *Obfuscating Circuits via Composite-Order Graded Encoding*, pages 528–556. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

CGH$^+$15. Jean-Sébastien Coron, Craig Gentry, Shai Halevi, Tancrède Lepoint, Hemanta K. Maji, Eric Miles, Mariana Raykova, Amit Sahai, and Mehdi Tibouchi. Zeroizing without low-level zeroes: New MMAP attacks and their limitations. In *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 247–266. Springer, 2015.

CGH17. Yilei Chen, Craig Gentry, and Shai Halevi. Cryptanalyses of candidate branching program obfuscators. In *Advances in Cryptology - EUROCRYPT 2017 - Proceedings, Part III*, pages 278–307, 2017.

CHL$^+$15. Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 3–12. Springer, 2015.

CLLT16. Jean-Sébastien Coron, Moon Sung Lee, Tancrède Lepoint, and Mehdi Tibouchi. Cryptanalysis of GGH15 multilinear maps. In *Advances in Cryptology - CRYPTO 2016 - Proceedings, Part II*, pages 607–628, 2016.

CLLT17. Jean-Sébastien Coron, Moon Sung Lee, Tancrède Lepoint, and Mehdi Tibouchi. Zeroizing attacks on indistinguishability obfuscation over CLT13. In *Public-Key Cryptography - PKC 2017 - Proceedings, Part I*, pages 41–58, 2017.

CLR15. Jung Hee Cheon, Changmin Lee, and Hansol Ryu. Cryptanalysis of the new CLT multilinear maps. *IACR Cryptology ePrint Archive*, 2015:934, 2015.

CLT13. Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In Ran Canetti and Juan A. Garay, editors, *CRYPTO*, volume 8042 of *LNCS*, pages 476–493. Springer, 2013.

CVW18. Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. GGH15 beyond permutation branching programs: Proofs, attacks, and candidates. In *Advances in Cryptology - CRYPTO 2018 - Proceedings, Part II*, pages 577–607, 2018.

DGHV10. Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 24–43. Springer, 2010.

FRS17. Rex Fernando, Peter M. R. Rasmussen, and Amit Sahai. Preventing CLT attacks on obfuscation with linear overhead. In *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part III*, pages 242–271, 2017.

GGH13a. Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT*, volume 7881 of *LNCS*, pages 1–17. Springer, 2013.

GGH$^+$13b. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, pages 40–49. IEEE Computer Society, 2013.

GGH15. Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC*, volume 9015 of *LNCS*, pages 498–527, 2015.

GLSW15. Craig Gentry, Allison Bishop Lewko, Amit Sahai, and Brent Waters. Indistinguishability obfuscation from the multilinear subgroup elimination assumption. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 151–170, 2015.

GLW14. Craig Gentry, Allison B. Lewko, and Brent Waters. Witness encryption from instance independent assumptions. In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, pages 426–443, 2014. https://eprint.iacr.org/2014/273.

HJ16. Yupu Hu and Huiwen Jia. Cryptanalysis of GGH map. In *Advances in Cryptology - EUROCRYPT 2016 - Proceedings, Part I*, pages 537–565, 2016.

HPS11. Guillaume Hanrot, Xavier Pujol, and Damien Stehlé. Analyzing blockwise lattice algorithms using dynamical systems. In *Proceedings of the 31st Annual Conference on Advances in Cryptology*, CRYPTO'11, pages 447–464, Berlin, Heidelberg, 2011. Springer-Verlag.

Len87. H. W. Lenstra, Jr. Factoring integers with elliptic curves. *Ann. of Math. (2)*, 126(3):649–673, 1987.

LLL82. A. K. Lenstra, H. W. Lenstra, and L. Lovasz. Factoring polynomials with rational coefficients, 1982.

MSZ16. Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. In *Advances in Cryptology - CRYPTO 2016 - Proceedings, Part II*, pages 629–658, 2016.

NS99. Phong Nguyen and Jacques Stern. The hardness of the hidden subset sum problem and its cryptographic implications, 1999.

S$^+$17. W. A. Stein et al. *Sage Mathematics Software (Version 8.0)*. The Sage Development Team, 2017. `http://www.sagemath.org`.

Zim15. Joe Zimmerman. How to obfuscate programs directly. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II*, pages 439–467, 2015.

A Proof of Proposition 4

Let $\boldsymbol{a} = (\alpha_1, \ldots, \alpha_\theta, 1) \in \mathbb{Z}^{\theta+1}$. We let $C = 2^{\rho_R - \alpha + 1}$ and consider the lattice $A^\perp$ of vectors $(C\boldsymbol{x}, y) \in \mathbb{Z}^\theta \times \mathbb{Z}$ such that $(\boldsymbol{x}, y)$ is orthogonal to $\boldsymbol{a}$ modulo x_0. Further, we let $B = \theta 2^{\rho_R+2}$ and let $L \subseteq \mathbb{Z}^{\ell+1}$ denote the lattice of vectors $(B\boldsymbol{u}, v) \in \mathbb{Z}^\ell \times \mathbb{Z}$ such that the vector $(\boldsymbol{u}, v)$ is orthogonal to the vector $(\boldsymbol{\omega}, 1)$ modulo x_0.

Let $\Lambda^\perp$ be the lattice of vectors $\boldsymbol{u} \in \mathbb{Z}^\ell$ such that $\langle \boldsymbol{u}, \hat{\boldsymbol{m}}_i \rangle \equiv 0 \pmod{g_i}$ for all $1 \le i \le \theta$. We denote by $\boldsymbol{u}_0$ a shortest non-zero vector of $\Lambda^\perp$. We write $\langle \boldsymbol{u}_0, \hat{\boldsymbol{m}}_i \rangle = k_i g_i$ with $k_i \in \mathbb{Z}$. To $\boldsymbol{u}_0$ we thus associate the vector $F(\boldsymbol{u}_0) = (B\boldsymbol{u}_0, -\sum_{i=1}^\theta k_i s_i - \langle \boldsymbol{u}_0, \boldsymbol{R} \rangle)$. From the definition of $\boldsymbol{\omega}$ and the congruence relations $g_i \alpha_i \equiv s_i \pmod{x_0}$, we have that $(\boldsymbol{u}_0, -\sum_{i=1}^\theta k_i s_i - \langle \boldsymbol{u}_0, \boldsymbol{R} \rangle)$ is orthogonal to $(\boldsymbol{\omega}, 1)$ modulo x_0, and therefore $F(\boldsymbol{u}_0) \in L$.

Letting $g = \prod_{i=1}^\theta g_i$, we now show that $F(\boldsymbol{u}_0)$ has square norm upper bounded by

$$\|F(\boldsymbol{u}_0)\|^2 \le (\ell+1) B^2 \|\boldsymbol{u}_0\|^2 \le \ell(\ell+1) B^2 g^{2/\ell} \,. \tag{26}$$

Indeed, we write $\|F(\boldsymbol{u}_0)\|^2 \le B^2 \|\boldsymbol{u}_0\|^2 + (\sum_{i=1}^\theta |k_i s_i| + \|\boldsymbol{u}_0\| \|\boldsymbol{R}\|)^2$. From $\|\hat{\boldsymbol{m}}_i\| \le \sqrt{\ell} 2^\alpha$, we obtain $2^{\alpha-1}|k_i| \le |k_i| g_i \le \|\boldsymbol{u}_0\| \|\hat{\boldsymbol{m}}_i\| \le \sqrt{\ell} 2^\alpha \|\boldsymbol{u}_0\|$; *i.e.* $|k_i| \le 2\sqrt{\ell}\|\boldsymbol{u}_0\|$ for all i. Combined with $\|\boldsymbol{R}\| \le \sqrt{\ell}\|\boldsymbol{R}\|_\infty \le \sqrt{\ell} 2^{\rho_R}$, this gives

$$\sum_{i=1}^{\theta} |k_i s_i| + \|\boldsymbol{u}_0\| \|\boldsymbol{R}\| \le \sqrt{\ell}\|\boldsymbol{u}_0\| \cdot (2^{\rho_R+1}\theta + 2^{\rho_R}) \le \sqrt{\ell}\|\boldsymbol{u}_0\| (2 \cdot 2^{\rho_R+1}\theta) = \sqrt{\ell} B \|\boldsymbol{u}_0\|$$

Therefore, $\|F(\boldsymbol{u}_0)\|^2 \leq B^2\|\boldsymbol{u}_0\|^2 + \ell B^2\|\boldsymbol{u}_0\|^2 = (\ell+1)B^2\|\boldsymbol{u}_0\|^2$. Now, since $\boldsymbol{u}_0$ has length $\lambda_1(\Lambda^\perp)$, it follows from Minkowski's Theorem that $\|\boldsymbol{u}_0\| \leq \sqrt{\ell} g^{1/\ell}$ where $g = \det(\Lambda^\perp)$, and (26) easily follows.

Let $\boldsymbol{x}_1 = (B\boldsymbol{u}_1, v_1)$ be the first vector in a (3/4)-reduced basis of the lattice L, obtained from LLL. By Theorem 3, it satisfies $\|\boldsymbol{x}_1\| \leq 2^{\ell/2}\|F(\boldsymbol{u}_0)\|$, that is, combined with (26), $\|\boldsymbol{x}_1\| \leq 2^{\ell/2}\sqrt{\ell(\ell+1)}Bg^{1/\ell}$. In particular, we obtain the bounds

$$\|\boldsymbol{u}_1\| \leq 2^{\ell/2}\sqrt{\ell(\ell+1)} \cdot g^{1/\ell} \tag{27}$$

$$|v_1| \leq 2^{\ell/2}B\sqrt{\ell(\ell+1)} \cdot g^{1/\ell}. \tag{28}$$

For simplicity we write $K = 2^{\ell/2}\sqrt{\ell(\ell+1)}g^{1/\ell}$. Now, to the vector $\boldsymbol{x}_1 \in L$, we associate, for C as above, the vector $f(\boldsymbol{x}_1) = (C\langle \boldsymbol{u}_1, \hat{\boldsymbol{m}}_1\rangle, \ldots, C\langle \boldsymbol{u}_1, \hat{\boldsymbol{m}}_\theta\rangle, \langle \boldsymbol{u}_1, \boldsymbol{R}\rangle + v_1) \in A^\perp$. Because $(B\boldsymbol{u}_1, v_1) \in L$, it is a direct check that $f(\boldsymbol{x}_1) \in A^\perp$. Its square norm is upper bounded by

$$\|f(\boldsymbol{x}_1)\|^2 \leq C^2 \sum_{i=1}^{\theta} \|\boldsymbol{u}_1\|^2\|\hat{\boldsymbol{m}}_i\|^2 + (\|\boldsymbol{u}_1\|\|\boldsymbol{R}\| + v_1)^2.$$

Using once again that $\|\hat{\boldsymbol{m}}_i\| \leq 2^\alpha\sqrt{\ell}$ and $\|\boldsymbol{R}\| \leq 2^{\rho_R}\sqrt{\ell}$, and combining with (27) and (28), we obtain

$$\begin{aligned}\|f(\boldsymbol{x}_1)\|^2 &\leq C^2K^2 \cdot \theta\ell 2^{2\alpha} + (K\sqrt{\ell}2^{\rho_R} + KB)^2 \leq C^2K^2 \cdot \theta\ell 2^{2\alpha} + (2K\sqrt{\ell}B)^2 \\ &= K^2\ell(C^2\theta 2^{2\alpha} + 4B^2)\end{aligned}$$

so that, using $C^2\theta 2^{2\alpha} \leq B^2 = 16\theta^2 2^{2\rho_R}$, this gives

$$\|f(\boldsymbol{x}_1)\| \leq 4\sqrt{5} \cdot \sqrt{\ell} \cdot \theta \cdot K \cdot 2^{\rho_R}. \tag{29}$$

We now consider the vectors $\{\boldsymbol{q}_i : 1 \leq i \leq \theta\}$ defined by $\boldsymbol{q}_i = (0, \ldots 0, Cg_i, 0, \ldots, 0, -s_i) \in \mathbb{Z}^{\theta+1}$. They are linearly independent; moreover, from the congruence relations $g_i\alpha_i \equiv s_i \pmod{x_0}$ for $1 \leq i \leq \theta$ we deduce that for all i, $\langle \boldsymbol{q}_i, \boldsymbol{a}\rangle \equiv 0 \pmod{x_0}$; *i.e.* $\boldsymbol{q}_i \in A^\perp$. Further, as $|s_i| \leq 2^{\rho_R}$, their norm is upper bounded by $\|\boldsymbol{q}_i\|^2 \leq C^2g_i^2 + 2^{2\rho_R} \leq C^2g_i^2 + Cg_i^2 \leq 2C^2g_i^2$ because $Cg_i \geq 2^{\rho_R-\alpha+1} \cdot 2^{\alpha-1} = 2^{\rho_R}$. Consequently,

$$\prod_{i=1}^{\theta} \|\boldsymbol{q}_i\| \leq 2^{\theta/2}C^\theta \prod_{i=1}^{\theta} g_i = 2^{\theta/2}C^\theta g. \tag{30}$$

Now, (15) together with $g \leq 2^{\alpha\theta}$, implies $(1 + 1/\ell)\log_2(g) + (\ell+\theta)/2 + \log_2(4\sqrt{5}\sqrt{\ell+1}\theta\ell) < \log_2(x_0) - \rho_R$ and, by raising to the power of 2, we obtain $g^{1+1/\ell} \cdot 2^{\ell/2} \cdot 2^{\theta/2} \cdot 4\sqrt{5}\sqrt{\ell+1}\theta\ell < x_0/2^{\rho_R}$. This is equivalent to

$$g^{1/\ell} \cdot 2^{\ell/2} \cdot 2^{\rho_R} \cdot 4\sqrt{5}\sqrt{\ell+1} \cdot \theta\ell < \frac{C^\theta x_0}{C^\theta 2^{\theta/2}g}\ . \tag{31}$$

The left hand side is lower bounded by $\|f(\boldsymbol{x}_1)\|$ by (29), and the right hand side is upper bounded by $\det(A^\perp)/\prod_{i=1}^{\theta}\|\boldsymbol{q}_i\|$, by (30) together with $\det(A^\perp) = C^\theta x_0$. Therefore (31) implies $\|f(\boldsymbol{x}_1)\| < \det(A^\perp)/\prod_{i=1}^{\theta}\|\boldsymbol{q}_i\|$. It follows from Lemma 2 that $f(\boldsymbol{x}_1)$ is in the linear span generated by the vectors $\{\boldsymbol{q}_i : 1 \leq i \leq \theta\}$. Since g_i are prime numbers for $1 \leq i \leq \theta$, we conclude that $f(\boldsymbol{x}_1)$ is in the sublattice generated by the vectors $\{\boldsymbol{q}_i : 1 \leq i \leq \theta\}$. Consequently, for all $1 \leq i \leq \theta$, one has $\langle \boldsymbol{u}_1, \hat{\boldsymbol{m}}_i\rangle \equiv 0 \pmod{g_i}$.

The rows $\{\boldsymbol{b}_j : 1 \leq j \leq \ell+1\}$ of the matrix

$$\begin{bmatrix} B\boldsymbol{I}_\ell & -\boldsymbol{\omega}^T \\ 0 & x_0 \end{bmatrix},$$

where $\boldsymbol{I}_\ell$ denotes the $\ell \times \ell$ identity matrix, form a $\mathbb{Z}$-basis of L. Hence, by running LLL on this matrix with $\delta = 3/4$, we obtain a vector $\boldsymbol{x}_1$ of which the first ℓ entries, divided by B, produce a vector $\boldsymbol{u} = \boldsymbol{u}_1$ satisfying $\langle \boldsymbol{u}_1, \hat{\boldsymbol{m}}_i\rangle \equiv 0 \pmod{g_i}$ for all i. By Theorem 3, the algorithm terminates in polynomial time. □

B Source code of the lattice attacks

```
n=4
eta=60
alpha=20
nh=10
rho=20
theta=2
gam=n*eta
ell=4
d=3

def test():
  p=[random_prime(2^eta,False,2^(eta-1)) for i in range(n)]
  x0=prod(p)
  g=[random_prime(2^alpha,False,2^(alpha-1)) for i in range(n)]
  invg=[inverse_mod(g[i],p[i]) for i in range(n)]
  h=[ZZ.random_element(2^(nh-1),2^nh) for i in range(n)]

  m=[ZZ.random_element(2^alpha) for i in range(theta)]+
    [0 for i in range(n-theta)]
  r=[ZZ.random_element(2^rho) for i in range(n)]
  w=(sum([h[i]*m[i]*invg[i]*x0/p[i] for i in range(theta)])+
     sum([h[i]*r[i]*x0/p[i] for i in range(n)])) % x0

  print "x0=",x0
  print "p=",p
  print "g=",g
  print "m=",m
  print "w=",w
  pg=prod([g[i] for i in range(theta)])
  print "pg=",pg

  print "\nBasic␣attack:␣we␣should␣have␣nh+rho+2*theta*alpha<eta"
  print "nh+rho+2*theta*alpha=%d,␣eta=%d" % (nh+rho+2*theta*alpha,eta)

  rhoR=gam-eta+nh+rho
  B=2^rhoR
  M=Matrix([[B,w],
            [0,x0]])
  ML=M.LLL()
  print "rec␣pg=",abs(ML[0,0]/B),abs(ML[0,0]/B)==pg

  # extended attack
  m=Matrix([[ZZ.random_element(2^alpha) for i in range(theta)]+
            [0 for i in range(n-theta)]
            for j in range(ell)])
  r=Matrix([[ZZ.random_element(2^rho) for i in range(n)]
           for j in range(ell)])

  w=[(sum([h[i]*m[j,i]*invg[i]*x0/p[i] for i in range(theta)])+
             sum([h[i]*r[j,i]*x0/p[i] for i in range(n)])) % x0
            for j in range(ell)]

  M=Matrix(ZZ,ell+1,ell+1)
  for i in range(ell):
    M[i,i]=B
    M[i,-1]=w[i]
  M[-1,-1]=x0
  ML=M.LLL()
  MLg=ML[:ell,:ell]/B
  print "\nExtended␣attack:␣"
  print "we␣should␣have␣theta*alpha*(1+1/ell)+nh+rho<eta"
  print "theta*alpha*(1+1/ell)+nh+rho=",N(theta*alpha*(1+1/ell)+nh+rho),
  print "eta=",eta
  print "rec␣pg=",abs(MLg.det()),abs(MLg.det())==pg

  # with multiple vectors
  x=Matrix([[ZZ.random_element(2^alpha) for i in range(theta)]
           for k in range(d)])
  w=Matrix([[(sum([h[i]*m[j,i]*x[k,i]*invg[i]*x0/p[i]
             for i in range(theta)])+
             sum([h[i]*r[j,i]*x0/p[i] for i in range(n)])) % x0
            for j in range(ell)] for k in range(d)])
  M=Matrix(ZZ,ell+d,ell+d)
  for i in range(ell):
    M[i,i]=B
    for k in range(d):
      M[i,ell+k]=w[k,i]
  for i in range(ell,ell+d):
    M[i,i]=x0

  ML=M.LLL()
  MLg=Matrix(ZZ,ML[:ell,:ell]/B)
  print "\nWith␣multiple␣vectors:␣"
  print "we␣should␣have␣theta*alpha*(1/d+1/ell)+nh+rho<eta"
  print "theta*alpha*(1/d+1/ell)+nh+rho=",
  print N(theta*alpha*(1/d+1/ell)+nh+rho),"eta=",eta
  rpg=abs(MLg.det())
  print "rec␣pg=",rpg,rpg==pg

  print "With␣factoring:"
  print "␣␣Normalized␣messages:"
  for i in range(theta):
    mg=Matrix(Integers(g[i]),m[:,i]).T
    print "␣",mg/mg[0,0]

  print "␣␣Recovered␣messages:"
  for i in range(theta):
    MLgi=MLg.change_ring(Integers(g[i]))
    print "␣",MLgi.right_kernel().matrix()

  print "Without␣factoring:"
  print "␣␣Normalized␣message:"
  v=Matrix(Integers(pg),[[crt([m[j,i] for i in range(theta)],
                         [g[i] for i in range(theta)])
                          for j in range(ell)]])
  print "␣",v[0]/v[0,0]

  print "␣␣Recovered␣message:"
  MLext=Matrix(ZZ,ell,2*ell)
  MLext[:ell,:ell]=MLg
  for i in range(ell):
    MLext[i,ell+i]=pg
  print "␣",MLext.right_kernel().matrix()[0][:ell]
```