Cryptanalysis of the New CLT Multilinear Map over the Integers

Jung Hee Cheon[1], Pierre-Alain Fouque[2,3], Changmin Lee[1], Brice Minaud[2] and Hansol Ryu[1]

[1] Seoul National University, Seoul, Korea
[2] Université de Rennes 1, Rennes, France
[3] Institut Universitaire de France, Paris, France

Abstract. Multilinear maps serve as a basis for a wide range of cryptographic applications. The first candidate construction of multilinear maps was proposed by Garg, Gentry, and Halevi in 2013, and soon afterwards, another construction was suggested by Coron, Lepoint, and Tibouchi (CLT13), which works over the integers. However, both of these were found to be insecure in the face of so-called zeroizing attacks, by Hu and Jia, and by Cheon, Han, Lee, Ryu and Stehlé. To improve on CLT13, Coron, Lepoint, and Tibouchi proposed another candidate construction of multilinear maps over the integers at Crypto 2015 (CLT15).
This article presents two polynomial attacks on the CLT15 multilinear map, which share ideas similar to the cryptanalysis of CLT13. Our attacks allow recovery of all secret parameters in time polynomial in the security parameter, and lead to a full break of the CLT15 multilinear map for virtually all applications.

Keywords: Multilinear maps, graded encoding schemes.

1 Introduction

Cryptographic multilinear maps are a powerful and versatile tool to build cryptographic schemes, ranging from one-round multipartite Diffie-Hellman to witness encryption and general program obfuscation. The notion of cryptographic multilinear map was first introduced by Boneh and Silverberg in 2003, as a natural generalization of bilinear maps such as pairings on elliptic curves [BS03]. However it was not until 2013 that the first concrete instantiation over ideal lattices was realized by Garg, Gentry and Halevi [GGH13a], quickly inspiring another construction over the integers by Coron, Lepoint and Tibouchi [CLT13]. Alongside these first instantiations, a breakthrough result by Garg, Gentry, Halevi, Raykova, Sahai and Waters achieved (indistinguishability) obfuscation for all circuits from multilinear maps [GGH+13b]. From that point multilinear maps have garnered considerable interest in the cryptographic community, and a host of other applications have followed.

However this wealth of applications rests on the relatively fragile basis of only three constructions of multilinear maps to date: namely the original construction over ideal lattices [GGH13a], the construction over the integers [CLT13], and another recent construction over lattices [GGH15]. Moreover none of these constructions relies on standard hardness assumptions. In fact all three constructions have since been broken for their more "direct" applications such as one-round multipartite Diffie-Hellman [HJ15, CHL+15, Cor15]. Thus building candidate multilinear maps and assessing their security may be regarded as a challenging work in progress, and research in this area has been very active in recent years.

Following the attack by Cheon, Han, Lee, Ryu and Stehlé (CHLRS attack) on the [CLT13] multilinear map over the integers, several attempts to repair the scheme were published on ePrint, which hinged on hiding encodings of zero in some way; however these attempts were quickly proven insecure [CGH+15]. At Crypto 2015, Coron, Lepoint and Tibouchi set out to repair their scheme by following a different route [CLT15]: they essentially retained the structure of encodings from [CLT13], but added a new type of noise designed to thwart the CHLRS approach. Their construction was thus able to retain the attractive features of the original, namely conceptual simplicity, relative efficiency, and wide range of presumed hard problems on which applications could be built.

1.1 Our contribution

In this paper we propose two polynomial attacks on the new multilinear map over the integers presented by Coron, Lepoint and Tibouchi at Crypto 2015 [CLT15]. These two attacks were originally published independently on ePrint by Cheon, Lee and Ryu [CLR15], and by Minaud and Fouque [MF15]. The present paper is a merge of the two results for publication at Eurocrypt 2016.

The impact of both attacks is the same, and they both use the same starting point ("integer extraction"). The second half of the attacks is where they differ. In a nutshell, the attack by Cheon, Lee and Ryu looks into the exact expression of the value a in the term av_0 appearing in integer extractions. This makes it possible to uncover a matrix product similar to the CHLRS attack on CLT13, albeit a more complex one. As in the CHLRS attack, the secret parameters are then recovered as the eigenvalues of a certain matrix. For this reason we shall call this attack the *eigenvalue* attack.

By contrast the attack by Minaud and Fouque treats the value a in av_0 as a noise, which is removed by first recovering v_0 and taking equations modulo v_0. The secret parameter v_0 is recovered as (a divisor of) the determinant of a CHLRS-type matrix product. For this reason we shall call this attack the *determinant* attack. Once v_0 is recovered, CLT15 essentially collapses to CLT13 and can be broken by the CHLRS attack.

Both of the proposed attacks are polynomial in the security parameter. In addition, in the optimized version of the scheme where an exact multiple of x_0 is provided in the public parameters, the second attack is instant (as no determinant computation is actually required).

Moreover both attacks apply to virtually all possible applications of the CLT15 multilinear map. Indeed, while they do require low-level encodings of zero, these encodings are provided by the ladders given in the public parameters. In this respect CLT15 is weaker than CLT13. A closer look at the impact of our attacks is provided in Section 1.3.

We refer the reader to [MF15] for a third, probabilistic attack with similar properties.

1.2 Overview of the Attacks

We begin by briefly recalling the CLT15 multilinear map (more precisely, graded encoding scheme). The message space is $\mathbb{Z}_{g_1} \times \cdots \times \mathbb{Z}_{g_n}$ for some small primes $g_1, \ldots, g_n$, and $(m_1, \ldots, m_n)$ is encoded at some level $k \leq \kappa$ as:

$$\mathsf{CRT}_{(p_i)}\Big(\frac{r_i g_i + m_i}{z^k}\Big) + a x_0$$

where:

(p_i)	is a sequence of n large primes.
x_0	$= \prod p_i$.
$\mathsf{CRT}_{(p_i)}(x_i)$	is the unique integer in $[0, x_0)$ congruent to x_i modulo p_i.
z	is a fixed secret integer modulo x_0.
r_i	is a small noise.
a	is another noise.

Encodings at the same level can be added together, and the resulting encoding encodes the sum of the messages. Similarly encodings at levels i and j can be multiplied to yield an encoding at level $i + j$ of the coordinate-wise product of the encoded messages. This behavior holds as long as the values $r_i g_i + m_i$ do not go over p_i, *i.e.* reduction modulo p_i does not interfere. In order to prevent the size of encodings from increasing as a result of additions and multiplications, a *ladder* of encodings of zero of increasing size is published at each level. Encodings can then be reduced by subtracting elements of the ladder at the same level.

The power of the multilinear map comes from the zero-testing procedure, which allows users to test whether an encoding at the maximal level κ encodes zero. This is achieved by publishing a so-called zero-testing parameter denoted $\boldsymbol{p}_{zt} \in \mathbb{Z}$, together with a large prime $N \gg x_0$. An encoding at the maximal level κ may be written as:

$$e = \sum (r_i + m_i g_i^{-1} \bmod p_i) u_i + a x_0$$
$$\text{where } u_i \triangleq \left(g_i z^{-\kappa} (p_i^*)^{-1} \bmod p_i\right) p_i^* \quad \text{with } p_i^* = \prod_{j \neq i} p_j.$$

That is, some constants independent of the encoding have been folded with the CRT coefficients into u_i. Now $\boldsymbol{p}_{zt}$ is chosen such that $v_i \triangleq u_i \boldsymbol{p}_{zt} \bmod N$ and

$v_0 \triangleq x_0 \boldsymbol{p}_{zt} \bmod N$ satisfy $|v_i| \ll N$ and $|v_0| \ll N$. In this way, for any encoding e of zero at level κ, since $m_i = 0$, we have:

$$|e\boldsymbol{p}_{zt} \bmod N| = \Big|\sum r_i v_i + a v_0\Big| \ll N$$

provided the noises r_i and a are small enough. Thus, users can test whether e is an encoding of zero at level κ by checking whether $|e\boldsymbol{p}_{zt} \bmod N| \ll N$.

1.2.1 Integer Extraction (ϕ-value). Our attacks proceed in two steps. The first step is shared by both attacks and proceeds as follows. We define the integer extraction procedure $\phi : \mathbb{Z} \to \mathbb{Z}$. In short, ϕ computes $\sum_i r_i v_i + a v_0$ over the integers for any level-κ encoding e (of size up to the largest ladder element). Note that this value is viewed over the integers and not modulo N. If e is "small", then $\phi(e) = e\boldsymbol{p}_{zt} \bmod N$, *i.e.* ϕ matches the computation from the zero-testing procedure.

If e is "large" on the other hand, then e would need to be reduced by the ladder before zero-testing can be applied. However the crucial observation is that ϕ is $\mathbb{Z}$-linear as long as the values $r_i g_i + m_i$ associated with each encoding do not go over p_i. Thus e can be ladder-reduced into e', then $\phi(e') = e'\boldsymbol{p}_{zt} \bmod N$ is known, and $\phi(e)$ can be recovered from $\phi(e')$ by compensating the ladder reduction using $\mathbb{Z}$-linearity.

1.2.2 Eigenvalue Attack. The point of a CHLRS attack can be divided into two parts. The first is that, for a level-κ encoding of zero $e = \sum_{i=1}^{n} \big[\frac{r_i g_i}{z^\kappa} (\frac{x_0}{p_i})^{-1}\big]_{p_i} \frac{x_0}{p_i} + a x_0$,

$$[\boldsymbol{p}_{zt} \cdot e]_{x_0} = \sum_{i=1}^{n} r_i \hat{v}_i,$$

where $\hat{v}_i$ is common to all the encodings in CLT13, holds over the integers. The second point is that the zero-testing value of a product of two encodings is a quadratic form of some values related to each encoding. More precisely, for two encodings $e_1 = \sum_{i=1}^{n} \big[\frac{r_{i1} g_i}{z^t} (\frac{x_0}{p_i})^{-1}\big]_{p_i} \frac{x_0}{p_i} + a_1 x_0$ and $e_2 = \sum_{i=1}^{n} \big[\frac{r_{i2}}{z^{\kappa - t}} (\frac{x_0}{p_i})^{-1}\big]_{p_i} \frac{x_0}{p_i} + a_2 x_0$, the product is $e_1 e_2 \equiv \sum_{i=1}^{n} \big[\frac{r_{i1} r_{i2} g_i}{z^\kappa} (\frac{x_0}{p_i})^{-1}\big]_{p_i} \frac{x_0}{p_i} \bmod x_0$. Therefore, the zero-testing value of $e_1 e_2$ is

$$[\boldsymbol{p}_{zt} \cdot e_1 e_2]_{x_0} = \sum_{i=1}^{n} r_{i1} r_{i2} \hat{v}_i.$$

Let us look at CLT15 in these aspects. For a level-κ encoding of zero $e = \sum_{i=1}^{n} r_i u_{i\kappa} + a x_0$, the zero-testing value of x is written as

$$[\boldsymbol{p}_{zt} \cdot e]_N = \sum_{i=1}^{n} r_i v_i + a v_0,$$

for common v_i's, similar to CLT13. Let e_1 be a level-t encoding of zero, e_2 be a level-$(\kappa - t)$ encoding, and e be a product of e_1 and e_2. Then, these can

be written as $e_1 = \sum_{i=1}^{n} r_{i1}u_{it} + a_1x_0$, $e_2 = \sum_{i=1}^{n} r_{i2}u_{i\kappa-t} + a_2x_0$, and $e = \sum_{i=1}^{n} r_{i1}r_{i2}u_{i\kappa} + ax_0$, for some integers $a, a_1, a_2, r_{i1}, r_{i2}, 1 \le i \le n$, where a is a quadratic form of $a_1, a_2, r_{i1}, r_{i2}, 1 \le i \le n$. Since the size of e is larger than that of x_0, we need to reduce the size of e to perform zero-testing. Let e' be a ladder-reduced encoding of e; then, it is of the form $e' = e - \sum_{j=0}^{M} b_jX_j = \sum_{i=1}^{n}(r_{i1}r_{i2} - \sum_{j=0}^{M} b_js_{ij})u_{i\kappa} + (a - \sum_{j=0}^{M} b_jq_j)x_0$, for some $b_0, \cdots, b_M \in \{0, 1\}$. In this case, the zero-testing value gives

$$\begin{aligned}
[\boldsymbol{p}_{zt} \cdot e']_N &= \Big[\boldsymbol{p}_{zt} \cdot \big(e - \sum_{j=0}^{M} b_jX_j\big)\Big]_N \\
&= \sum_{i=1}^{n} \big(r_{i1}r_{i2} - \sum_{j=0}^{M} b_js_{ij}\big)v_i + \big(a - \sum_{j=0}^{M} b_jq_j\big)v_0 \\
&= \sum_{i=1}^{n} \big(r_{i1}r_{i2}\big)v_i + av_0 - \sum_{j=0}^{M} b_j\big(\sum_{i=1}^{n} s_{ij}v_i + q_jv_0\big).
\end{aligned}$$

Therefore, if one has $\sum_{i=1}^{n} s_{ij}v_i + q_jv_0$ for all j, one can compute $\sum_{i=1}^{n}(r_{i1}r_{i2})v_i + av_0$ and follow a CHLRS attack strategy. For this purpose the integer extraction function ϕ provides exactly what we need.

By using $(n+1)$ level-t encodings of zero and $(n+1)$ level-$(\kappa - t)$ encodings, we constitute matrix equations that consist only of a product of matrices. As in [CHL+15], we have a matrix, the eigenvalues of which consist of the CRT components of an encoding. From these, we can recover all the secret parameters of the CLT15 scheme. Our attack needs only ladders and two level-0 encodings (which can be provided by ladder elements), and runs in polynomial time.

1.2.3 Determinant Attack. The determinant attack proceeds by first recovering x_0. Once x_0 is known, the original CHLRS attack can be applied by taking all values modulo v_0. We now explain how to recover x_0.

In the optimized variant of the scheme implemented in [CLT15], a small multiple qx_0 of x_0 is given in the public parameters. In that case qx_0 may be regarded as an encoding of zero at level κ, and $\phi(qx_0) = qv_0$. Since this holds over the integers, we can compute $q = \gcd(qx_0, qv_0)$ and then $x_0 = qx_0/q$.

In the general case where no exact multiple of x_0 is given in the public parameters, pick $n+1$ encodings a_i at some level t, and $n+1$ encodings of zero b_i at level $\kappa - t$. Note that ladder elements provide encodings of zero even if the scheme itself does not. Then compute:

$$\omega_{i,j} \triangleq \phi(a_ib_j).$$

If we write $a_i \bmod v_0 = \mathsf{CRT}_{(p_j)}(a_{i,j}/z^t)$ and $b_i \bmod v_0 = \mathsf{CRT}_{(p_j)}(r_{i,j}g_j/z^{\kappa-t})$, then we get:

$$\omega_{i,j} \bmod v_0 = \sum_k a_{i,k}r_{j,k}v_k \bmod v_0.$$

Similar to the CHLRS attack on the CLT13 multilinear map, this equality can be viewed as a matrix product. Indeed, let Ω denote the $(n+1) \times (n+1)$ integer matrix with entries $\omega_{i,j}$, let A denote the $(n+1) \times n$ integer matrix with entries $a_{i,j}$, let R denote the $(n+1) \times n$ integer matrix with entries $r_{i,j}$, and finally let V denote the $n \times n$ diagonal matrix with diagonal entries v_i. If we embed everything into $\mathbb{Z}/v_0\mathbb{Z}$, then we have:

$$\Omega = A \cdot V \cdot R^{\mathrm{T}} \qquad \text{in } \mathbb{Z}/v_0\mathbb{Z}.$$

Since A and R are $(n+1) \times n$ matrices, this implies that Ω is not full-rank when embedded into $\mathbb{Z}/v_0\mathbb{Z}$. As a consequence v_0 divides $\det(\Omega)$. We can repeat this process with different choices of the families (a_i), (b_i) to build another matrix Ω' with the same property. Finally we recover v_0 as $v_0 = \gcd(\det(\Omega), \det(\Omega'))$, and $x_0 = v_0/\boldsymbol{p}_{zt} \bmod N$.

1.3 Impact of the Attacks

Two variants of the CLT15 multilinear map should be considered. Either a small multiple of x_0 is provided in the public parameters. In that case x_0 can be recovered instantly with the determinant attack, and the scheme becomes equivalent to CLT13 in terms of security (cf. Section 4.3.1). In particular it falls victim to the CHLRS attack when low-level encodings of zero are present, but it may still be secure for applications that do not require such encodings, such as obfuscation. However the scheme is strictly less efficient than CLT13 by construction, so there is no point in using CLT15 for those applications.

Otherwise, if no small multiple of x_0 is given out in the public parameters, then ladders of encodings of zero must be provided at levels below the maximal level. Thus we have access to numerous encodings of zero below the maximal level, even if the particular application of multilinear maps under consideration does not require them. As a result both the eigenvalue and the determinant attacks are applicable (cf. Section 4.3.3), and the secret parameters are still recovered in polynomial time, albeit less efficiently than the previous case.

In summary, the optimized version of CLT15 providing a small multiple of x_0 is no more secure than CLT13, and less efficient. On the other hand in the general non-optimized case, the scheme is broken for virtually all possible applications due to encodings of zero provided by the ladder. Thus overall the CLT15 scheme can be considered fully broken.

1.4 Organization of the Paper

For the sake of being self-contained, a presentation of multilinear maps and graded encoding schemes is provided in Appendix A. The CLT15 construction itself is described in Section 3. In Section 3.2 we recall the CHLRS attack on CLT13, as it shares similar ideas with our attacks. Readers already familiar with the CLT15 multilinear map can skip straight to Section 4 where we describe our main attacks.

2 Notation

The symbol $\triangleq$ denotes an equality by definition.
For n an integer, size(n) is the size of n in bits.
For a finite set S, we use $s \leftarrow S$ to denote the operation of uniformly choosing an element s from S.

Modular arithmetic. The group $\mathbb{Z}/n\mathbb{Z}$ of integers modulo n is denoted by $\mathbb{Z}_n$. For $a, b, p \in \mathbb{Z}$, $a \equiv b \bmod p$ or $a \equiv_p b$ means that a is congruent to b modulo p. The notation "mod p" should be understood as having the lowest priority. For instance, the expression $a \cdot b \bmod p$ is equivalent to $(a \cdot b) \bmod p$.

We always view $a \bmod p$ (or $[a]_p$) as an integer in $\mathbb{Z}$. The representative closest to zero is always chosen, positive in case of tie. In other words $-p/2 < a \bmod p \leq p/2$.

Chinese Remainder Theorem. Given n prime numbers (p_i), we define p_i^* as in [Hal15a]:

$$p_i^* = \prod_{j \neq i} p_j.$$

For $(x_1, \ldots, x_n) \in \mathbb{Z}^n$, let $\mathsf{CRT}_{(p_i)}(x_i)$ denote the unique integer in $\mathbb{Z} \cap [0, \prod p_i)$ such that $\mathsf{CRT}_{(p_i)}(x_i) \bmod p_i = x_i \bmod p_i$, as per the Chinese Remainder Theorem.

It is useful to observe that for any $(x_1, \ldots, x_n) \in \mathbb{Z}^n$:

$$\mathsf{CRT}_{(p_i)}(x_i p_i^*) = \sum_i x_i p_i^* \bmod \prod_i p_i. \tag{1}$$

Matrix. For an $n \times n$ square matrix $\boldsymbol{H}$, we use (h_{ij}) to represent a matrix $\boldsymbol{H}$, the (i, j) component of which is h_{ij}. Similarly, for a vector $\boldsymbol{v} \in \mathbb{R}^n$, we define $(\boldsymbol{v})_j$ as the j-th component of $\boldsymbol{v}$. Let $\boldsymbol{H}^T$ be the transpose of $\boldsymbol{H}$ and $\|\boldsymbol{H}\|_\infty$ be the $\max_i \sum_{j=1}^n |h_{ij}|$. We denote by $\mathsf{diag}(d_1, \cdots, d_n)$ the diagonal matrix with diagonal coefficients equal to $d_1, \cdots, d_n$.

3 The CLT15 Multilinear Map and its Cryptanalysis

In order to make our article self-contained, a short introduction to multilinear maps and graded encoding schemes is provided in Appendix A.

3.1 The CLT15 Multilinear Map over the Integers

Shortly after the multilinear map over ideal lattices by Garg, Gentry and Halevi [GGH13a], another construction over the integers was proposed by Coron, Lepoint and Tibouchi [CLT13]. However a devastating attack was published by Cheon, Han, Lee, Ryu and Stehlé at Eurocrypt 2015 (on ePrint in late 2014).

In the wake of this attack, a revised version of their multilinear map over the integers was presented by Coron, Lepoint and Tibouchi at Crypto 2015 [CLT15]. In the remainder of this article, we will refer to the original construction over the integers as CLT13, and to the new version from Crypto 2015 as CLT15.

In this section we recall the CLT15 construction. We omit aspects of the construction that are not relevant to our attack, and refer the reader to [CLT15] for more details. The message space is $R = \mathbb{Z}_{g_1} \times \cdots \times \mathbb{Z}_{g_n}$, for some (relatively small) primes $g_i \in \mathbb{N}$. An encoding of a message $(m_1, \ldots, m_n) \in \mathbb{Z}_{g_1} \times \cdots \times \mathbb{Z}_{g_n}$ at level $k \leq \kappa$ has the following form:

$$e = \mathsf{CRT}_{(p_i)}\left(\frac{r_i g_i + m_i}{z^k} \bmod p_i\right) + a x_0 \tag{2}$$

where:

- The p_i's are n large secret primes.
- The r_i's are random noise such that $|r_i g_i + m_i| \ll p_i$.
- $x_0 = \prod_{i \leq n} p_i$.
- z is a fixed secret integer modulo x_0.
- a is random noise.

The scheme relies on the following parameters:

$$\begin{aligned}
\lambda &: \text{the security parameter.}\\
\kappa &: \text{the multilinearity level.}\\
n &: \text{the number of primes } p_i.\\
\eta &: \text{the bit length of secret primes } p_i.\\
\gamma = n\eta &: \text{the bit length of } x_0.\\
\alpha &: \text{the bit length of the } g_i\text{'s.}\\
\rho &: \text{the bit length of initial } r_i\text{'s.}\\
\beta &: \text{the bit size of matrix } \boldsymbol{H} \text{ used to zero-testing procedure.}
\end{aligned}$$

Addition, negation and multiplication of encodings is exactly addition, negation and multiplication over the integers. Indeed, m_i is recovered from $e \cdot z^k$ as $m_i = (e \cdot z^k \bmod p_i) \bmod g_i$, and as long as $r_i g_i + m_i$ does not go over p_i, addition and multiplication will go through both moduli. Thus we have defined encodings and how to operate on them.

Regarding the sampling procedure from Appendix A.2, for our purpose, it suffices to know that it is realized by publishing a large number of level-0 encodings of random elements. Users can then sample a new random element as a subset sum of published elements. Likewise, the rerandomization procedure is achieved by publishing a large number of encodings of zero at each level, and an element is re-randomized by adding a random subset sum of encodings of zero at the same level. The encoding procedure is realized by publishing a single level-1 encoding y of 1 (by which we mean $(1, \ldots, 1) \in \mathbb{Z}_{g_1} \times \cdots \times \mathbb{Z}_{g_n}$): any encoding can then be promoted to an encoding of the same element at a higher level by

multiplying by y.

Zero-testing in CLT13. We now move on to the crucial zero-testing procedure. This is where CLT13 and CLT15 differ. We begin by briefly recalling the CLT13 approach.

In CLT13, the product x_0 of the p_i's is public. In particular, every encoding can be reduced modulo x_0, and every value below should be regarded as being modulo x_0. Let $p_i^* = \prod_{j \neq i} p_j$. Using (1), define:

$$\boldsymbol{p}_{zt} \triangleq \sum_{i \leq n} \Big(\frac{h_i z^\kappa}{g_i} \bmod p_i \Big) p_i^* = \mathsf{CRT}_{(p_i)} \Big(\frac{h_i z^\kappa}{g_i} p_i^* \bmod p_i \Big) \qquad \bmod x_0.$$

where the h_i's are some relatively small numbers with $|h_i| \ll p_i$. Now take a level-κ encoding of zero:

$$e = \mathsf{CRT}_{(p_i)} \Big(\frac{r_i g_i}{z^\kappa} \bmod p_i \Big) \qquad \bmod x_0.$$

Since multiplication acts coordinate-wise on the CRT components, using (1) again, we have:

$$\omega \triangleq e\boldsymbol{p}_{zt} = \mathsf{CRT}_{(p_i)}(h_i r_i p_i^*) = \sum_i h_i r_i p_i^* \qquad \bmod x_0.$$

Since $p_i^* = x_0/p_i$, as long as we set our parameters so that $|h_i r_i| \ll p_i$, we have $|\omega| \ll x_0$.

Thus the zero-testing procedure is as follows: for a level-κ encoding e, compute $\omega = e\boldsymbol{p}_{zt} \bmod x_0$. Output 1, meaning we expect e to encode zero, iff the ν most significant bits of ω are zero, for an appropriately chosen ν. In [CLT13], multiple $\boldsymbol{p}_{zt}$'s can be defined in order to avoid false positives; we restrict our attention to a single $\boldsymbol{p}_{zt}$.

Zero-testing in CLT15. In CLT13, an encoding at some fixed level is entirely defined by its vector of associated values $c_i = r_i g_i + m_i$. Moreover, addition and multiplication of encodings act coordinate-wise on these values, and the value of the encoding itself is $\mathbb{Z}_{x_0}$-linear as a function of these values. Likewise, ω is $\mathbb{Z}_{x_0}$-linear as a function of the r_i's. This nice structure is an essential part of what makes the devastating attack, so called CHLRS attack [CHL+15] possible. In CLT15, the authors set out to break this structure by introducing a new noise component a.

For this purpose, the public parameters include a new prime number $N \gg x_0$, with size$(N) = \gamma + 2\eta + 1$. Meanwhile x_0 is kept secret, and no longer part of the public parameters. Encodings are thus no longer reduced modulo x_0, and take the general form given in (2), including a new noise value a. Equivalently, we can write an encoding e of (m_i) at level k as:

$$e = \sum_i \left(r_i + m_i (g_i^{-1} \bmod p_i) \right) u_i + a x_0 \tag{3}$$

$$\text{with } u_i \triangleq \left(g_i z^{-k} (p_i^*)^{-1} \bmod p_i \right) p_i^*.$$

That is, we fold the $g_i z^{-k}$ multiplier of r_i with the CRT coefficient into u_i.

The zero-testing parameter $\boldsymbol{p}_{zt}$ is now defined modulo N in such a way that:

$$\begin{array}{lll} & v_0 \triangleq x_0 \boldsymbol{p}_{zt} \bmod N & \forall i, v_i \triangleq u_i \boldsymbol{p}_{zt} \bmod N \\ \text{satisfy:} & |v_0| \ll N & |v_i| \ll N \end{array} \tag{4}$$

To give an idea of the sizes involved, $\text{size}(v_0) \approx \gamma$ and $\text{size}(v_i) \approx \gamma + \eta$ for $i > 0$. We refer the reader to [CLT15] for how to build such a $\boldsymbol{p}_{zt}$. The point is that if e is an encoding of zero at level κ, then we have:

$$\omega = e\boldsymbol{p}_{zt} \bmod N = \sum r_i v_i + a v_0 \bmod N.$$

In order for this quantity to be smaller than N, the size of a must be somehow controlled. Conversely as long as a is small enough and the noise satisfies $|r_i| \ll p_i$ then $|\omega| \ll N$. We state the useful lemma for an exact zero-testing, the so-called the zero-testing lemma, more precisely.

Lemma 1 (Zero-testing lemma). *Let e be a level-κ encoding of zero with $e = \sum_{i=1}^{n} r_i u_i + a x_0, (r_1, \cdots, r_n, a \in \mathbb{Z})$. Then,*

$$[e\boldsymbol{p}_{zt}]_N = \sum_{i=1}^{n} r_i v_i + a v_0,$$

holds over the integers, if $|a| < 2^{2\eta - \beta - \log_2 n - 1}$ and $|r_i| < 2^{\eta - \beta - \log_2 n - 6}$ for $1 \leq i \leq n$.

Proof. By the construction of the zero-testing element, we have $e\boldsymbol{p}_{zt} \equiv \sum_{i=1}^{n} r_i v_i + a v_0 \bmod N$. It is sufficient to show that the right hand side is smaller than $N/2$. For $1 \leq i \leq n$,

$$v_i \equiv \sum_{j=1}^{n} h_j \alpha_j p_j^{-1} u_i \equiv h_i \beta_i + \sum_{j \neq i} h_j \alpha_j \left[\frac{g_i}{z^\kappa} \left(\frac{x_0}{p_i} \right)^{-1} \right]_{p_i} \frac{x_0}{p_i p_j} \bmod N,$$

and therefore, $|v_i| < 2^{\gamma + \eta + \beta + 4}$ for $1 \leq i \leq n$. Moreover, $v_0 = \sum_{j=1}^{n} h_j \alpha_j \frac{x_0}{p_j}$ and $|v_0| < n 2^{\gamma + \beta - 1}$. □

Thus the size of a must be controlled. The term $a x_0$ will be dominant in (3) in terms of size, so decreasing a is the same as decreasing the size of the encoding as a whole. The scheme requires a way to achieve this without altering the encoded value (and without publishing x_0).

For this purpose, inspired by [VDGHV10], a *ladder* $(X_i^{(k)})_{0 \leq i \leq \gamma'}$ of encodings of zero of increasing size is published for each level $k \leq \kappa$, where $\gamma' = \gamma + \lfloor \log_2 \ell \rfloor$. The size of an encoding e at level k can then be reduced without altering the encoded value by recursively subtracting from e the largest ladder element smaller than e, until e is smaller than $X_0^{(\kappa)}$. More precisely we can choose $X_0^{(\kappa)}$

small enough that the previous zero-testing procedure goes through, and then choose $X_{\gamma'}^{(\kappa)}$ twice the size of $X_0^{(\kappa)}$, so that the product of any two encodings smaller than $X_0^{(\kappa)}$ can be reduced to an encoding smaller than $X_0^{(\kappa)}$. After each addition and multiplication, the size of the resulting encoding is reduced via the ladder.

In the end, the zero-testing procedure is very similar to CLT13: given a (ladder-reduced) level-κ encoding e, compute $\omega = e\boldsymbol{p}_{zt} \bmod N$. Then output 1, meaning we expect e to encode zero, iff the ν high-order bits of ω are zero.

Extraction. The extraction procedure simply outputs the ν high-order bits of ω, computed as above. For both CLT13 and CLT15, it can be checked that they only depend on the m_i's (as opposed to the noises a and the r_i's).

3.2 CHLRS Attack on CLT13

In this section we provide a short description of CHLRS attack on CLT13 [CHL+15], as elements of this attack appear in our own. We actually present (a close variant of) the slightly simpler version in [CGH+15].

Assume we have access to a level-0 encoding a of some random value, n level-1 encodings (b_i) of zero, and a level-1 encoding y of 1. This is the case for one-round multi-party Diffie-Hellman (see previous section). Let $a_i = a \bmod p_i$, *i.e.* a_i is the i-th value "$r_i g_i + m_i$" associated with a. For $i \leq n$, define $r_{i,j} = b_i z / g_j \bmod p_j$, *i.e.* $r_{i,j}$ is the j-th value "r_j" associated with b_i (recall that b_i is an encoding of zero, so $m_j = 0$). Finally let $y_k = yz \bmod p_k$.

Now compute:

$$\begin{aligned} e_{i,j} &= a \cdot b_i \cdot b_j \cdot y^{\kappa-2} \bmod x_0 & \omega_{i,j} &= e_{i,j}\boldsymbol{p}_{zt} \bmod x_0 \\ e'_{i,j} &= \quad b_i \cdot b_j \cdot y^{\kappa-2} \bmod x_0 & \omega'_{i,j} &= e'_{i,j}\boldsymbol{p}_{zt} \bmod x_0 \end{aligned}$$

Note that:

$$\begin{aligned} \omega_{i,j} &= \sum_k \Big(a_k \frac{r_{i,k} g_k}{z} \frac{r_{j,k} g_k}{z} \frac{y_k^{\kappa-2}}{z^{\kappa-2}} \frac{h_k z^\kappa}{g_k} \bmod p_k \Big) p_k^* \\ &= \sum_k a_k r_{i,k} r_{j,k} c_k \qquad \text{with } c_k = g_k y_k^{\kappa-2} h_k p_k^*. \end{aligned} \tag{5}$$

Crucially, in the second line, the modulo p_k disappears and the equation holds over the integers, because $e_{i,j}$ is a valid encoding of zero, so the correctness of the scheme requires $|e_{i,j} z^\kappa / g_k \bmod p_k| \ll p_k$.

Equation (5) may be seen as a matrix multiplication. Indeed, define Ω, resp. Ω', as the $n \times n$ matrix with entries $\omega_{i,j}$, resp. $\omega'_{i,j}$, and likewise R with entries $r_{i,j}$. Moreover let A, resp. C, be the diagonal matrix with diagonal entries a_i, resp. c_i. Then (5) may be rewritten:

$$\begin{aligned} \Omega &= R \cdot A \cdot C \cdot R^{\mathrm{T}} \\ \Omega' &= R \cdot C \cdot R^{\mathrm{T}} \\ \Omega \cdot (\Omega')^{-1} &= R \cdot A \cdot R^{-1}. \end{aligned}$$

Here matrices are viewed over $\mathbb{Q}$ for inversion (they are invertible whp).

Once $\Omega \cdot (\Omega')^{-1}$ has been computed, the (diagonal) entries of A can be recovered as its eigenvalues. In practice this can be achieved by computing the characteristic polynomial, and all computations can be performed modulo some prime p larger than the a_i's (which are size 2ρ).

Thus we recover the a_i's, and by definition $a_i = a \bmod p_i$, so p_i can be recovered as $p_i = \gcd(a - a_i, x_0)$. From there it is trivial to recover all other secret parameters of the scheme.

4 Main Attack

4.1 Integer Extraction (ϕ-value)

Integer extraction essentially removes the extra noise induced by ladder reductions when performing computations on encodings. In addition, as we shall see in Section 4.3.2, this step is enough to recover x_0 when an exact multiple is known, as is the case in the optimized variant proposed and implemented in [CLT15].

In the remainder we say that an encoding at level k is small iff it is less than $X_0^{(k)}$ in absolute value. In particular, any ladder-reduced encoding is small.

Now, we describe our idea of attack. For a level-κ encoding of zero $e = \sum_{i=1}^{n} r_i u_i + a x_0$ of arbitrary size, if one can compute the integer value $\sum_{i=1}^{n} r_i v_i + a v_0$, which is not reduced modulus N, then a CHLRS attack can be applied similarly. Hence, we define the function ϕ such that it represents such a value and examine how to obtain the function values for a level-κ encoding of zero of arbitrary size.

When the size of e is small, by the zero-testing lemma, $[\boldsymbol{p}_{zt} \cdot e]_N$ gives the integer value $\sum_{i=1}^{n} r_i v_i + a v_0$. However, if the size of e is large, the zero-testing lemma does not hold and one cannot compute the integer value directly. To reach the goal, we use the ladder $X_j^{(\kappa)} = \sum_{i=1}^{n} r_{ij}^{(\kappa)} u_i + a_j^{(\kappa)}$. Let e be a level-κ encoding of zero. Then, we can compute the size-reduced encoding e' using the ladder and obtain the quantity (for short, we define γ' as $\gamma + \lfloor \log_2 \ell \rfloor$.)

$$
\begin{aligned}
[\boldsymbol{p}_{zt} \cdot e']_N &= \Big[\boldsymbol{p}_{zt} \cdot \Big(e - \sum_{j=0}^{\gamma'} b_j X_j^{(\kappa)}\Big)\Big]_N \\
&= \sum_{i=1}^{n} \Big(r_i - \sum_{j=0}^{\gamma'} b_j r_{ij}^{(\kappa)}\Big) v_i + \Big(a - \sum_{j=0}^{\gamma'} b_j a_j^{(\kappa)}\Big) v_0 \\
&= \sum_{i=1}^{n} r_i v_i + a v_0 - \sum_{j=0}^{\gamma'} b_j \Big(\sum_{i=1}^{n} r_{ij}^{(\kappa)} v_i + a_j^{(\kappa)} v_0\Big).
\end{aligned}
$$

Therefore, if one can compute $\sum_{i=1}^{n} r_{ij}^{(\kappa)} v_i + a_j^{(\kappa)} v_0$ from $X_j^{(\kappa)}$, one can easily obtain $\sum_{i=1}^{n} r_i v_i + a v_0$.

To compute $\sum_{i=1}^{n} r_{ij}^{(\kappa)} v_i + a_j^{(\kappa)} v_0$ for all $j \in \{0, \cdots, \gamma + \lfloor \log_2 \ell \rfloor\}$, we use an induction on j. When $j = 0$, $[\boldsymbol{p}_{zt} \cdot X_0^{(\kappa)}]_N$ gives $\sum_{i=1}^{n} r_{i0}^{(\kappa)} v_i + a_0^{(\kappa)} v_0$, by the zero-testing lemma. Suppose we have $\sum_{i=1}^{n} r_{ij}^{(\kappa)} v_i + a_j^{(\kappa)} v_0$ for $j \in \{0, \cdots, t-1\}$; then, $[\boldsymbol{p}_{zt} \cdot X_t]_N = \sum_{i=1}^{n} r_{it}^{(\kappa)} v_i + a_t^{(\kappa)} v_0 - \sum_{j=0}^{t-1} b_j (\sum_{i=1}^{n} r_{ij}^{(\kappa)} v_i + a_j^{(\kappa)} v_0)$ for computable $b_i \in \{0, 1\}$, where X_t is a size-reduced encoding of $X_t^{(\kappa)}$ using $\{X_0^{(\kappa)}, \cdots, X_{t-1}^{(\kappa)}\}$. Since we know the latter terms, we can also compute $\sum_{i=1}^{n} r_{it}^{(\kappa)} v_i + a_t^{(\kappa)} v_0$. This idea can be extended to any level ladder.

Now, we give a precise description of function ϕ.

$$\begin{aligned} \phi : \mathbb{Z} &\to \mathbb{Z} \\ e &\mapsto \sum_{i=1}^{n} \left[e \cdot \frac{z^\kappa}{g_i} \right]_{p_i} v_i + \frac{x - \sum_{i=1}^{n} [e \cdot \frac{z^\kappa}{g_i}]_{p_i} u_i}{x_0} v_0, \end{aligned}$$

where $v_i = [\boldsymbol{p}_{zt} \cdot u_i]_N (1 \le i \le n)$ and $v_0 = [\boldsymbol{p}_{zt} \cdot x_0]_N$. Note that ϕ is defined over the integers, and not modulo N. Indeed the v_i's are seen as integers: recall from Section 2 that throughout this paper x mod N denotes an integer in $\mathbb{Z} \cap (-N/2, N/2]$.

Proposition 1. *Let e be an integer such that $e \equiv \frac{r_i \cdot g_i}{z^\kappa}$ mod p_i for $1 \le i \le n$. If $|r_i| < p_i/2$ for each i, then x can be uniquely expressed as $\sum_{i=1}^{n} r_i u_i + a x_0$ for some integer a, and $\phi(e) = \sum_{i=1}^{n} r_i v_i + a v_0$.*

Proof. We can see that $e \equiv \sum_{i=1}^{n} r_i u_i \bmod p_j$ for each j and thus there exists an integer a such that $e = \sum_{i=1}^{n} r_i u_i + a x_0$. For uniqueness, suppose e can be written as $e = \sum_{i=1}^{n} r'_i u_i + a' x_0$ for integers $r'_1, \cdots, r'_n, a'$ with $|r'_i| < p_i/2$. Then, $e \equiv r'_i [\frac{g_i}{z^\kappa} (\frac{x_0}{p_i})^{-1}]_{p_i} \equiv \frac{r'_i g_i}{z^\kappa} \bmod p_i$, which implies $r_i \equiv r'_i \bmod p_i$. Since $|r_i - r'_i| < p_i$, we have $r'_i = r_i$ for each i and therefore $a' = a$, which proves the uniqueness. □

The point is that if e is a small encoding of zero at level κ, then $\phi(e) = e\boldsymbol{p}_{zt}$ mod N. In that case $\phi(e)$ matches the extraction in the sense of the ext procedure of Appendix A.2 (more precisely ext returns the high-order bits of $\phi(e)$).

However we want to compute $\phi(e)$ even when e is larger. For this purpose, the crucial point is that ϕ is actually $\mathbb{Z}$-linear as long as for all encodings involved, the associated r_i's do not go over $p_i/2$, *i.e.* reduction modulo p_i does not interfere. More formally:

Proposition 2. *Let $e_1, \cdots, e_m$ be level-κ encodings of zero such that $e_j \equiv \frac{r_{ij} g_i}{z^\kappa}$ mod p_i and $|r_{ij}| < p_i/2$ for all $1 \le i \le n$, $1 \le j \le m$. Then, the equality*

$$\phi(\sum_{j=1}^{m} e_j) = \sum_{j=1}^{m} \phi(e_j),$$

holds if $\left| \sum_{j=1}^{m} r_{ij} \right| < \frac{p_i}{2}$, for all $1 \le i \le n$.

Proof. From Proposition 1, each e_j can be uniquely written as $e_j = \sum_{i=1}^{n} r_{ij}u_i + a_j x_0$ for some integer a_j, and $\phi(e_j) = \sum_{i=1}^{n} r_{ij}v_i + a_j v_0$. Then,

$$\sum_{j=1}^{m} \phi(e_j) = \sum_{i=1}^{n} \Big(\sum_{j=1}^{m} r_{ij}\Big) \cdot v_i + \Big(\sum_{j=1}^{m} a_j\Big) \cdot v_0$$
$$= \phi\Big(\Big(\sum_{j=1}^{m} r_{ij}\Big) \cdot u_i + \Big(\sum_{j=1}^{m} a_j\Big) \cdot x_0\Big) = \phi\Big(\sum_{j=1}^{m} e_j\Big),$$

where the source of the second equality is Proposition 1, since $\big|\sum_{j=1}^{m} r_{ij}\big| < p_i/2$. □

An important remark is that the conditions on the r_{ij}'s above are also required for the correctness of the scheme to hold. In other words, as long as we perform valid computations from the point of view of the multilinear map (*i.e.* there is no reduction of the r_{ij}'s modulo p_i, and correctness holds), then the $\mathbb{Z}$-linearity of ϕ also holds.

4.2 Eigenvalue Attack

Our strategy to attack CLT15 is similar to that in [CHL+15]. The goal is to construct a matrix equation over $\mathbb{Q}$ by computing the ϕ values of several products of level-0, 1, and $(\kappa - 1)$ encodings, fixed on level-0 encoding. We proceed using the following three steps.

(Step 1) Compute the ϕ-value of level-κ ladder
(Step 2) Compute the ϕ-value of level-κ encodings of large size
(Step 3) Construct matrix equations over $\mathbb{Q}$.

Using the matrix equations in **Step 3**, we have a matrix, the eigenvalues of which are residue modulo p_i of level-0 encoding. From this, we deduce a secret modulus p_i.

4.2.1 Computing the ϕ-value of $X_j^{(\kappa)}$

To apply the zero-testing lemma to a level-κ encoding of zero $e = \sum_{i=1}^{n} r_i u_i + a x_0$, the size of r_i and a has to be bounded by some fixed values. By the parameter setting, η is larger than the maximum bit size of the noise r_i of a level-κ encoding obtained from the multiplication of lower level encodings. Hence, we need to reduce the size of e so that a satisfies the zero-testing lemma.

Let us consider a ladder of level-κ encodings of zero $\{X_j^{(\kappa)}\}$. This is provided to reduce the size of encodings to that of $2x_0$. More precisely, given a level-κ encoding of zero e of size smaller than $2^{2\gamma+\lfloor \log_2 \ell \rfloor}$, one can compute $e' = e - \sum_{j=0}^{\gamma'} b_j X_j^{(\kappa)}$ for $\gamma' = \gamma + \lfloor \log_2 \ell \rfloor$, which is an encoding of the same plaintext; its size is smaller than $X_0^{(\kappa)}$. As noted in [CLT15], the sizes of $X_j^{(\kappa)}$ are increasing

and differ by only one bit, and therefore, $b_j \in \{0,1\}$, which implies the noise grows additively. We can reduce a to an integer much smaller than $2^{2\eta-\beta-1}/n$ so that the zero-testing lemma can be applied. We denote such e' as $[e]_{\boldsymbol{X}^{(\kappa)}}$. More generally, we use the notation

$$[e]_{\boldsymbol{X}^{(t)}} := [\cdots[[e]_{X^{(t)}_{\gamma'}}]_{X^{(t)}_{\gamma'-1}}\cdots]_{X^{(t)}_0} \quad \text{for } \boldsymbol{X}^{(t)} = (X^{(t)}_0, X^{(t)}_1, \ldots, X^{(t)}_{\gamma'}), 1 \le t \le \kappa.$$

Note that, if e satisfies the condition in Lemma 1, i.e., it is an encoding of zero of small size, then $\phi(e)$ is exactly the same as $[\boldsymbol{p}_{zt} \cdot e]_N$. However, if the size of e is large, it is congruent only to $[\boldsymbol{p}_{zt} \cdot e]_N$ modulo N. Now, we show how to compute the integer value $\phi(e)$ for an encoding e of zero, although e does not satisfy the condition in Lemma 1.

First, we adapt the size reduction process to a level-κ ladder itself. We can compute binary b_{ij} for each i, j, satisfying

$$\begin{aligned}
[X^{(\kappa)}_0]_{\boldsymbol{X}^{(\kappa)}} &= X^{(\kappa)}_0 \\
[X^{(\kappa)}_1]_{\boldsymbol{X}^{(\kappa)}} &= X^{(\kappa)}_1 - b_{10} \cdot X^{(\kappa)}_0 \\
[X^{(\kappa)}_2]_{\boldsymbol{X}^{(\kappa)}} &= X^{(\kappa)}_2 - \sum_{k=0}^{1} b_{2k} \cdot X^{(\kappa)}_k \\
&\vdots \\
[X^{(\kappa)}_j]_{\boldsymbol{X}^{(\kappa)}} &= X^{(\kappa)}_j - \sum_{k=0}^{j-1} b_{jk} \cdot X^{(\kappa)}_k.
\end{aligned}$$

Each $[X^{(\kappa)}_j]_{\boldsymbol{X}^{(\kappa)}}$ is an encoding of zero at level κ and therefore can be written as $[X^{(\kappa)}_j]_{\boldsymbol{X}^{(\kappa)}} = \sum_{i=1}^{n} r'_{ij} u_i + a'_j x_0$ for some integers r'_{ij} and a'_j. Moreover, its bit size is at most γ and therefore a'_j is small enough to satisfy the condition in Lemma 1. Therefore,

$$\phi([X^{(\kappa)}_j]_{\boldsymbol{X}^{(\kappa)}}) = [\boldsymbol{p}_{zt} \cdot [X^{(\kappa)}_j]_{\boldsymbol{X}^{(\kappa)}}]_N = \sum_{i=1}^{n} r'_{ij} v_i + a'_j v_0.$$

If we write $X^{(\kappa)}_j = \sum_{i=1}^{n} r_{ij} u_i + a_j x_0$ for some integer $r_{1j}, \ldots, r_{nj}, a_j$, we have $r'_{ij} = r_{ij} - \sum_{k=0}^{j-1} b_{jk} r_{ik}$ for each i and $a'_j = a_j - \sum_{k=0}^{j-1} b_{jk} a_k$, since all the coefficients of u_i are sufficiently smaller than p_i for each i. Therefore,

$$\sum_{i=1}^{n} r'_{ij} v_i + a'_j v_0 = \sum_{i=1}^{n} r_{ij} v_i + a_j v_0 - \sum_{k=0}^{j-1} b_{jk} \Big(\sum_{i=1}^{n} r_{ik} v_i + a_k v_0 \Big)$$

holds over the integers. Hence, we have the following inductive equations for $0 \le j \le \gamma'$.

$$\phi(X^{(\kappa)}_j) = \Big[\boldsymbol{p}_{zt} \cdot [X^{(\kappa)}_j]_{\boldsymbol{X}^{(\kappa)}} \Big]_N + \sum_{k=0}^{j-1} b_{jk} \cdot \phi\Big(X^{(\kappa)}_k\Big),$$

which gives all $\phi(X_0^{(\kappa)}), \phi(X_1^{(\kappa)}), \ldots, \phi(X_{\gamma'}^{(\kappa)})$, inductively. The computation consists of $(\gamma'+1)$ zero-testing and $O(\gamma^2)$-times comparisons and subtractions of $(\gamma+\gamma')$-bit integers, and therefore, the total computation cost is $\widetilde{O}(\gamma^2)$ by using fast Fourier transform. Hence, we obtain the following lemma.

Lemma 2. *Given the public parameters of the CLT15 scheme, one can compute*

$$\phi(X_j^{(\kappa)}) = \left[\boldsymbol{p}_{zt} \cdot [X_j^{(\kappa)}]_{\boldsymbol{X}^{(\kappa)}}\right]_N + \sum_{k=0}^{j-1} b_{jk} \cdot \phi\left(X_k^{(\kappa)}\right)$$

in $\widetilde{O}(\gamma^2)$ bit computations.

4.2.2 Computing the ϕ-value of Level-κ Encodings of Large Size

Using the ϕ values of the κ-level ladder, we can compute the ϕ value of any κ-level encoding of zero, the bit size of which is between γ and $\gamma+\gamma'$.

Lemma 3. *Let e be a level-κ encoding of zero, $e = \mathsf{CRT}_{(p_i)}\left(\frac{r_i g_i}{z^\kappa}\right) + qx_0 = \sum_{i=1}^n r_i u_i + ax_0$ for some integer $r_1, \ldots, r_n, a$ satisfying $|r_i| < 2^{\eta-\beta-\log_2 n-7}$ for each i and $|a| < 2^{\gamma'}$. Given the public parameters of the CLT15 scheme, one can compute the value $\phi(e) = \sum_{i=1}^n r_i v_i + a v_0$ in $\widetilde{O}(\gamma^2)$ bit computations.*

Proof. Let e be a level-κ encoding of zero satisfying the above conditions. As in Section 4.2.1, we can find binary b_j's satisfying $[e]_{\boldsymbol{X}^{(\kappa)}} = e - \sum_{j=0}^{\gamma'} b_j \cdot X_j^{(\kappa)}$. Then, we have

$$\phi(e) = \phi([e]_{\boldsymbol{X}^{(\kappa)}}) + \sum_{j=0}^{\gamma'} b_j \cdot \phi(X_j^{(\kappa)}).$$

Since $[e]_{\boldsymbol{X}^{(\kappa)}}$ is a κ-level encoding of zero of at most γ-bit and the size of noise is bounded by $(\eta-\beta-\log_2 n-6)$-bit, we can compute the value $\phi([e]_{\boldsymbol{X}^{(\kappa)}})$ via the zero-testing procedure. Finally, the ϕ values of the κ-level ladder and $\phi([e]_{\boldsymbol{X}^{(\kappa)}})$ give the value $\phi(e)$. The source of the complexity is Lemma 2. □

We apply Lemma 3 to obtain the ϕ value of a κ-level encoding of zero that is a product of two encodings of $(\gamma+\gamma')$-bit size.

Lemma 4. *Let X be a level-1 encoding and Y a level-$(\kappa-1)$ encoding of zero of bit size at most $\gamma+\gamma'$. Then, one can compute $\phi(XY)$ in $\widetilde{O}(\gamma^3)$ bit computations.*

Proof. We apply Lemma 3 to a product of two γ-bit encodings. From $[X_1^{(1)}]_{\boldsymbol{X}^{(1)}} = X_1^{(1)} - b \cdot X_0^{(1)}$ for some $b \in \{0,1\}$, we find $\phi(X_1^{(1)} \cdot X_0^{(\kappa-1)}) = \phi([X_1^{(1)}]_{\boldsymbol{X}^{(1)}} \cdot X_0^{(\kappa-1)}) + b \cdot \phi(X_0^{(1)} \cdot X_0^{(\kappa-1)})$, since $[X_1^{(1)}]_{\boldsymbol{X}^{(1)}}$ is γ-bit. Thus, we can obtain inductively all $\phi(X_j^{(1)} \cdot X_k^{(\kappa-1)})$ for each j,k from $\phi(X_{l_j}^{(1)} \cdot X_{l_k}^{(\kappa-1)})$, $0 \le l_j \le j, 0 \le l_k \le k, (l_j, l_k) \neq (j,k)$.

Let $[X]_{\boldsymbol{X}^{(1)}} = X - \sum_{j=0}^{\gamma'} b_j \cdot X_j^{(1)}$ and $[Y]_{\boldsymbol{X}^{(\kappa-1)}} = Y - \sum_{j=0}^{\gamma'} b'_j \cdot X_j^{(\kappa-1)}$. Then,

$$[X]_{\boldsymbol{X}^{(1)}} \cdot [Y]_{\boldsymbol{X}^{(\kappa-1)}} = XY - \textstyle\sum_j b_j \cdot X_j^{(1)} \cdot Y \\ - \textstyle\sum_j b'_j \cdot X_j^{(\kappa-1)} \cdot X + \sum_{j,k} b_j b'_k \cdot X_j^{(1)} \cdot X_k^{(\kappa-1)}.$$

Note that the noise of $[[X]_{\boldsymbol{X}^{(1)}} \cdot [Y]_{\boldsymbol{X}^{(\kappa-1)}}]_{\boldsymbol{X}^{(\kappa)}}$ is bounded by $2\rho+\alpha+2\log_2(\gamma')+2$ and $\eta > \kappa(2\alpha+2\rho+\lambda+2\log_2 n+3)$, and therefore, we can adapt Proposition 2. Therefore, if we know the ϕ-value of each term, we can compute the ϕ-value of XY. Finally, Lemma 3 enables one to compute $\phi([X]_{\boldsymbol{X}^{(1)}} \cdot [Y]_{\boldsymbol{X}^{(\kappa-1)}})$. The second and third terms of the right hand side can be computed using $[X_j^{(1)}]_{\boldsymbol{X}^{(1)}}$, $[X_j^{(\kappa-1)}]_{\boldsymbol{X}^{(\kappa-1)}}$, and we know the ϕ-value of the last one. Since we perform zero-testings for $O(\gamma^2)$ encodings of zero, the complexity becomes $\widetilde{O}(\gamma^3)$. □

Note that the above Lemma can be applied to a level-t encoding X and a level-$(\kappa - t)$ encoding of zero Y. The proof is exactly the same, except for the indexes.

4.2.3 Constructing Matrix Equations over $\mathbb{Q}$ We reach the final stage. The following theorem is the result.

Theorem 1. *Given the public instances in [CLT15] and $\boldsymbol{p}_{zt}$, one can find all the secret parameters given in [CLT15] in $\widetilde{O}(\kappa^{\omega+4}\lambda^{2\omega+6})$ bit computations with $\omega \leq 2.38$.*

Proof. We construct a matrix equation by collecting several ϕ-values of the product of level-0, 1 and $(\kappa-1)$ encodings. Let c, X, and Y be a level-0, 1, and $(\kappa-1)$ encoding, respectively, and additionally we assume Y is an encoding of zero. Let us express them as

$$\begin{aligned} c &= \mathsf{CRT}_{(p_i)}(c_i), \\ X &= \mathsf{CRT}_{(p_i)}\left(\frac{x_i}{z}\right) = x_i \left[z^{-1}\right]_{p_i} + q_i p_i, \\ Y &= \mathsf{CRT}_{(p_i)}\left(\frac{y_i g_i}{z^{\kappa-1}}\right) = \sum_{i=1}^{n} y_i \left[\frac{g_i}{z^{\kappa-1}}\left({p_i}^*\right)^{-1}\right]_{p_i} \cdot {p_i}^* + a x_0. \end{aligned}$$

Assume that the size of each is less than $2x_0$. The product of c and X can be written as $cX = c_i x_i \left[z^{-1}\right]_{p_i} + q'_i p_i$ for some integer q'_i.

By multiplying cX and Y, we have

$$\begin{aligned} & cXY \\ &= \sum_{i=1}^{n} \left(c_i x_i y_i \left[z^{-1}\right]_{p_i} \left[\frac{g_i}{z^{\kappa-1}}\left(\frac{x_0}{p_i}\right)^{-1}\right]_{p_i} \cdot \frac{x_0}{p_i} + y_i \left[\frac{g_i}{z^{\kappa-1}}\left(\frac{x_0}{p_i}\right)^{-1}\right]_{p_i} q'_i x_0 \right) + (cX)(a x_0) \\ &= \sum_{i=1}^{n} c_i x_i y_i u_i + \sum_{i=1}^{n} (c_i x_i y_i s_i + y_i \theta_i q'_i) x_0 + acX x_0, \end{aligned}$$

where $\theta_i = \left[\frac{g_i}{z^{\kappa-1}}\left(\frac{x_0}{p_i}\right)^{-1}\right]_{p_i}, \theta_i \left[z^{-1}\right]_{p_i} \frac{x_0}{p_i} = u_i + s_i x_0$ for some integer $s_i \in \mathbb{Z}$. Then, we can obtain $\phi(cXY) = \sum_{i=1}^{n} c_i x_i y_i v_i + \sum_{i=1}^{n}(c_i x_i y_i s_i + y_i \theta_i q_i')v_0 + acXv_0$ by Lemma 4.

By plugging $q_i' = \frac{1}{p_i}(cX - c_i x_i [z^{-1}]_{p_i})$ into the equation, we obtain

$$\begin{aligned}\phi(cXY) &= \sum_{i=1}^{n} y_i(v_i + s_i v_0 - \frac{\theta_i v_0}{p_i}[z^{-1}]_{p_i})c_i x_i + \sum_{i=1}^{n} y_i \frac{\theta_i v_0}{p_i} cX + a v_0 cX \\ &= \sum_{i=1}^{n} y_i w_i c_i x_i + \sum_{i=1}^{n} y_i w_i' cX + a v_0 cX,\end{aligned}$$

where $w_i = v_i + s_i v_0 - \frac{\theta_i}{p_i}[z^{-1}]_{p_i} v_0$ and $w_i' = \frac{\theta_i v_0}{p_i}$. It can be written (over $\mathbb{Q}$) as

$$\phi(cXY) = \begin{pmatrix} y_1 & y_2 & \cdots & y_n & a \end{pmatrix} \begin{pmatrix} w_1 & & & 0 & w_1' \\ & w_2 & & & w_2' \\ & & \ddots & & \vdots \\ & & & w_n & w_n' \\ 0 & & & & v_0 \end{pmatrix} \begin{pmatrix} c_1 x_1 \\ c_2 x_2 \\ \vdots \\ c_n x_n \\ cX \end{pmatrix}. \tag{6}$$

Since $p_i w_i = p_i(v_i + s_i v_0) - \theta_i \left[z^{-1}\right]_{p_i} v_0 \equiv -\theta_i \left[z^{-1}\right]_{p_i} v_0 \not\equiv 0 \bmod p_i$, w_i is not equal to zero. Therefore, $v_0 \prod_{i=1}^{n} w_i \neq 0$ and thus the matrix in Equation (6) is non singular. By applying Equation (6) to various X, Y, taking for $0 \leq j, k \leq n$

$$\begin{aligned} X &= [X_j^{(1)}]_{\boldsymbol{X}^{(1)}} = \mathsf{CRT}_{(p_i)}\left(\frac{x_{ij}}{z}\right), \\ Y &= [X_k^{(\kappa-1)}]_{\boldsymbol{X}^{(\kappa-1)}} = \sum_{i=1}^{n} y_{ik}\theta_i \frac{x_0}{p_i} + a_k x_0, \end{aligned}$$

we finally obtain the matrix equation

$$\begin{aligned} \boldsymbol{W}_c &= \begin{pmatrix} y_{10} & \cdots & y_{n0} & a_0 \\ & & & \\ & \ddots & & \vdots \\ & & & \\ y_{1n} & \cdots & y_{nn} & a_n \end{pmatrix} \begin{pmatrix} w_1 & & & 0 & w_1' \\ & w_2 & & & w_2' \\ & & \ddots & & \vdots \\ & & & w_n & w_n' \\ 0 & & & & v_0 \end{pmatrix} \begin{pmatrix} c_1 & & & & 0 \\ & c_2 & & & \\ & & \ddots & & \\ & & & c_n & \\ 0 & & & & c \end{pmatrix} \begin{pmatrix} x_{10} & \cdots & x_{1n} \\ & & \\ & \ddots & \vdots \\ x_{n0} & & x_{nn} \\ X_0 & \cdots & X_n \end{pmatrix} \\ &= \quad \boldsymbol{Y} \qquad\qquad \boldsymbol{W} \qquad\qquad \mathsf{diag}(c_1, \cdots, c_n, c) \qquad\qquad \boldsymbol{X}. \end{aligned}$$

We perform the same computation on $c = 1$, which is a level-0 encoding of $\mathbf{1} = (1, 1, \cdots, 1)$, and then, it implies

$$\boldsymbol{W}_1 = \boldsymbol{Y} \cdot \boldsymbol{W} \cdot \boldsymbol{I} \cdot \boldsymbol{X}.$$

From $\boldsymbol{W}_c$ and $\boldsymbol{W}_1$, we have a matrix that is similar to $\mathsf{diag}(c_1, \cdots, c_n, c)$:

$$\boldsymbol{W}_1^{-1} \cdot \boldsymbol{W}_c = \boldsymbol{X}^{-1} \cdot \mathsf{diag}(c_1, \cdots, c_n, c) \cdot \boldsymbol{X}.$$

Then, by computing the eigenvalues of $\boldsymbol{W}_1^{-1} \cdot \boldsymbol{W}_c$, we have $c_1, \cdots, c_n$, satisfying $p_i | (c - c_i)$ for each i. Using an additional level-0 encoding c', we obtain $\boldsymbol{W}_1^{-1} \cdot \boldsymbol{W}_{c'}$, and therefore, $c'_1, \cdots, c'_n$ with $p_i | (c' - c'_i)$ for each i. Computing $\gcd(c - c_i, c' - c'_i)$ gives the secret prime p_i.

Using $p_1, \cdots, p_n$, we can recover all the remaining parameters. By the definition of y and $X_j^{(1)}$, the equation $y / [X_j^{(1)}]_{x_0} \equiv (r_i g_i + 1)/(r_{ij}^{(1)} g_i) \bmod p_i$ is satisfied. Since $r_i g_i + 1$ and $r_{ij}^{(1)} g_i$ are smaller than $\sqrt{p_i}$ and are co-prime, one can recover them by rational reconstruction up to the sign. Therefore, we can obtain g_i by computing the gcd of $r_{i0}^{(1)} g_i, \cdots, r_{im}^{(1)} g_i$. Moreover, using $r_{ij}^{(1)} g_i$ and $[X_j^{(1)}]_{x_0}$, we can compute $[z]_{p_i}$ for each i and therefore z. Any other parameters are computed using z, g_i, and p_i.

Our attack consists of the following arithmetics: computing $\phi(X_j^{(\kappa)})$, $\phi(X_j^{(1)} \cdot X_k^{(\kappa-1)})$, constructing a matrix $\boldsymbol{W}_c$ and $\boldsymbol{W}_1$, matrix inversing and multiplying, and computing eigenvalues and the greatest common divisor. All of these are bounded by $\widetilde{O}(\gamma^3 + n^\omega \gamma) = \widetilde{O}(\kappa^6 \lambda^9)$ bit computations with $\omega \leq 2.38$. For this algorithm to succeed, we need a property that $\boldsymbol{W}_1$ is non-singular. If we use the fact that the rank of a matrix $\boldsymbol{A} \in \mathbb{Z}^{(n+1)\times(n+1)}$ can be computed in time $\widetilde{\mathcal{O}}\left((n+1)^\omega \log \|\boldsymbol{A}\|_\infty\right)$ (see [Sto09]), we can find that $\boldsymbol{X}, \boldsymbol{Y} \cdot \boldsymbol{W} \in \mathbb{Q}^{(n+1)\times(n+1)}$ are non-singular in $\widetilde{\mathcal{O}}(2(\gamma + \log \ell)(n^\omega \log N)) = \widetilde{\mathcal{O}}(\kappa^{\omega+4} \lambda^{2\omega+6})$ by considering another $(n+1)$ subsets of $X_0^{(1)}, \cdots, X_{\gamma'}^{(1)}$ for X and also for Y. Therefore, the total complexity of our attack is $\widetilde{\mathcal{O}}(\kappa^{\omega+4} \lambda^{2\omega+6})$. □

4.3 Determinant Attack

4.3.1 On the Impact of Recovering x_0 If x_0 is known, CLT15 essentially collapses to CLT13. In particular, all encodings can be reduced modulo x_0 so ladders are no longer needed. What is more, all $\omega_{i,j}$'s from the CHLRS attack can be reduced modulo $v_0 = x_0 \boldsymbol{p}_{zt} \bmod N$, which effectively removes the new noise a. As a direct consequence the CHLRS attack goes through and all secret parameters are recovered (cf. [CLT15, Section 3.3]). Moreover ladder elements reduced by x_0 provide low-level encodings of zero even if the scheme itself does not. Also note that the CHLRS attack is quite efficient as it can be performed modulo any prime larger than the values we are trying to recover, *i.e.* larger than $2^{2\rho}$.

4.3.2 Recovering x_0 when an Exact Multiple is Known The authors of [CLT15] propose an optimized version of their scheme, where a multiple qx_0 of x_0 is provided in the public parameters. The size of q is chosen such that qx_0 is about the same size as N. Ladders at levels below κ are no longer necessary: every encoding can be reduced modulo qx_0 without altering encoded values or

increasing any noise. The ladder at level κ is still needed as a preliminary to zero-testing, however it does not need to go beyond qx_0, which makes it much smaller. In the end this optimization greatly reduces the size of the public key and speeds up computations, making the scheme much more practical (cf. Section 4.3.4).

In this scenario, note that qx_0 may be regarded as an encoding of 0 at level κ (and indeed every level). Moreover by construction it is small enough to be reduced by the ladder at level κ with a valid computation (*i.e.* with low enough noise for every intermediate encoding involved that the scheme operates as desired and zero-extraction is correct). As a direct consequence we have:

$$\phi(qx_0) = qv_0$$

and so we can recover q as $q = \gcd(qx_0, \phi(qx_0))$, and get $x_0 = qx_0/q$. This attack has been verified on the reference implementation, and recovers x_0 instantly.

Remark. qv_0 is larger than N by design, so that it cannot be computed simply as $qx_0\boldsymbol{p}_{zt} \bmod N$ due to modular reductions (cf. [CLT15, Section 3.4]). The point is that our computation of ϕ is over the integers and not modulo N.

4.3.3 Recovering x_0 in the General Case We now return to the non-optimized version of the scheme, where no exact multiple of x_0 is provided in the public parameters.

The second step of our attack recovers x_0 using a matrix product similar to the CHLRS attack (cf. Section 3.2), except we start with families of $n+1$ encodings rather than n. That is, assume that for some t we have $n+1$ level-t small encodings (a_i) of any value, and $n+1$ level-$(\kappa - t)$ small encodings (b_i) of zero. This is easily achievable for one-round multi-party Diffie-Hellman (cf. Section A.2), e.g. choose $t = 1$, then pick $(n+1)$ level-1 encodings (a_i) of zero from the public parameters, and let $b_i = a'_i y^{\kappa-2}$ for a'_i another family of $(n+1)$ level-1 encodings of zero and y any level-1 encoding, where the product is ladder-reduced at each level. In other applications of the multilinear map, observe that ladder elements provide plenty of small encodings of zero, as each ladder element can be reduced by the elements below it to form a small encoding of zero. Thus the necessary conditions to perform both our attack to recover x_0, and the follow-up CHLRS attack to recover other secret parameters once x_0 is known, are very lax. In this respect CLT15 is weaker than CLT13.

Let $a_{i,j} = a_i z \bmod p_j$, *i.e.* $a_{i,j}$ is the j-th value "$r_j g_j + m_j$" associated with a_i. Likewise for $i \leq n$, let $r_{i,j} = b_i z^{\kappa-1}/g_j \bmod p_j$, *i.e.* $r_{i,j}$ is the j-th value "r_j" associated with b_i (recall that b_i is an encoding of zero, so $m_j = 0$). Now compute:

$$\omega_{i,j} \triangleq \phi(a_i b_j).$$

If we look at the $\omega_{i,j}$'s modulo v_0 (which is unknown for now), everything behaves as in CLT13 since the new noise term av_0 disappears, and the ladder reduction at level κ is negated by the integer extraction procedure. Hence, sim-

ilar to Section 3.2, we have:

$$\omega_{i,j} \bmod v_0 = \sum_k a_{i,k} r_{j,k} v_k \bmod v_0. \tag{7}$$

Again, equation (7) may be seen as a matrix product. Indeed, define Ω as the $(n+1) \times (n+1)$ integer matrix with entries $\omega_{i,j}$, let A be the $(n+1) \times n$ matrix with entries $a_{i,j}$, let R be the $(n+1) \times n$ matrix with entries $r_{i,j}$, and finally let V be the $n \times n$ diagonal matrix with diagonal entries v_i. Then (7) may be rewritten modulo v_0:

$$\Omega = A \cdot V \cdot R^{\mathrm{T}} \qquad \text{in } \mathbb{Z}_{v_0}.$$

Since A and R are $(n+1) \times n$ matrices, this implies that Ω is not full-rank when embedded into $\mathbb{Z}_{v_0}$. As a consequence v_0 divides $\det(\Omega)$, where the determinant is computed over the integers. Now we can build a new matrix Ω' in the same way using a different choice of b_i's, and recover v_0 as $v_0 = \gcd(\det(\Omega), \det(\Omega'))$. Finally we get $x_0 = v_0/\boldsymbol{p}_{zt} \bmod N$ (note that $N \gg x_0$ by construction).

The attack has been verified on the reference implementation with reduced parameters.

Remark. As pointed out above, Ω cannot be full-rank when embedded into $\mathbb{Z}_{v_0}$. Our attack also requires that it *is* full-rank over $\mathbb{Q}$ (whp). This holds because while Ω can be nicely decomposed as a product when viewed modulo v_0, the "remaining" part of Ω, that is $\Omega - (\Omega \bmod v_0)$ is the matrix of the terms av_0 for each $\omega_{i,j}$, and the value a does have the nice structure of $\omega_{i,j} \bmod v_0$. This is by design, since the noise a was precisely added in CLT15 in order to defeat the matrix product structure of the CHLRS attack.

4.3.4 Attack Complexity It is clear that the attack is polynomial, and asymptotically breaks the scheme. In this section we provide an estimate of its practical complexity. When an exact multiple of x_0 is known, the attack is instant as mentioned in Section 4.3.2, so we focus on the general case from Section 4.3.3.

In the general case, a ladder of encodings of size $\ell \approx \gamma$ is published at every level[4]. Using the scheme requires κ ladder reductions, *i.e.* $\kappa\ell$ additions of integers of size γ. Since there are κ users, this means the total computation incurred by using the scheme is close to $\kappa^2\gamma^2$. For the smallest 52-bit instance, this is already $\approx 2^{46}$. Thus using the scheme a hundred times is above the security parameter. This highlights the importance of the optimization based on publishing qx_0, which makes the scheme much more practical. More importantly for our current purpose, this makes it hard to propose an attack below the security parameters.

[4] As the level increases, it is possible to slightly reduce the size of the ladder. Indeed the acceptable level of noise increases with each level, up to ρ_f at level κ. As a consequence it is possible to leave a small gap between ladder elements as the level increases. For instance if the base level of noise is 2ρ for ladder elements, then at level κ it is possible to leave a gap of roughly $\rho_f - 2\rho - \log \ell$ bits between ladder elements. We disregard this effect, although it slighly improves our complexity.

As a result, what we propose in terms of complexity evaluation is the following. For computations that compare directly to using the multilinear scheme, we will tally the complexity as the number of operations equivalent to using the scheme, in addition to the bit complexity. For unrelated operations, we will count the number of bit operations as usual.

There are two steps worth considering from a complexity point of view: computing Ω and computing its determinant. In practice both steps happen to have comparable complexity. Computing the final gcd is negligible in comparison using a subquadratic algorithm [Möl08], which is practical for our parameter size.

Computing Ω. As a precomputation, in order to compute ϕ, the integer extraction of ladder elements at level κ needs to be computed. This requires ℓ integer extractions, where $\ell \leq \gamma$. Computing Ω itself requires $(n+1)^2$ integer extractions of a single product. Each integer extraction requires 1 multiplication, and 2ℓ additions (as well as ℓ multiplications by small scalars). For comparison, using the multilinear scheme for one user requires 1 multiplication and ℓ additions on integers of similar size. Thus overall computing Ω costs about $\gamma + n^2$ times as much as simply *using* the multilinear scheme. For the 52-bit instance proposed in [CLT15] for instance, this means that if it is practical to use the scheme about a million times, then it is practical to compute Ω. Here by using the scheme we mean one (rather than κ^2) ladder reduction, so the bit complexity is $\mathcal{O}(\gamma^3 + n^2\gamma^2)$.

Computing the Determinant. Let n denote the size of a matrix Ω (it is $(n+1)$ in our case but we will disregard this), and β the number of bits of its largest entry. When computing the determinant of an integer matrix, one has to carefully control the size of the integers appearing in intermediate computations. It is generally possible to ensure that these integers do not grow past the size of the determinant. Using Hadamard's bound this size can be upper bounded as $\log(\det(\Omega)) \leq n(\beta + \frac{1}{2}\log n)$, which can be approximated to $n\beta$ in our case, since β is much larger than n.[5]

As a result, computing the determinant using "naive" methods requires $\mathcal{O}(n^3)$ operations on integers of size up to $n\beta$, which results in a complexity $\widetilde{\mathcal{O}}(n^4\beta)$ using fast integer multiplication (but slow matrix multiplication). The asymptotic complexity is known to be $\widetilde{\mathcal{O}}(n^\omega\beta)$ [Sto05]; however we are interested in the complexity of practical algorithms. Computing the determinant can be reduced to solving the linear system associated with Ω with a random target vector: indeed the determinant can then be recovered as the least common denominator of the (rational) solution vector[6]. In this context the fastest algorithms use p-adic lifting [Dix82], and an up-to-date analysis using fast arithmetic in [MS04] gives a complexity $\mathcal{O}(n^3\beta\log^2\beta\log\log\beta)$ (with $\log n = o(\beta)$).[7]

[5] This situation is fairly unusual, and in the literature the opposite is commonly assumed; algorithms are often optimized for large n rather than large β.

[6] In general extra factors may appear, but this is not relevant for us.

[7] This assumes a multitape Turing machine model, which is somewhat less powerful than a real computer.

For the concrete instantiations of one-round multipartite Diffie-Hellman implemented in [CLT15], this yields the following complexities:

Security parameter:	52	62	72	80
Building Ω:	2^{60}	2^{66}	2^{74}	2^{82}
Determinant:	2^{57}	2^{66}	2^{74}	2^{81}

Thus, beside being polynomial, the attack is actually coming very close to the security parameter as it increases to 80 bits.[8]

Acknowledgement. We would like to thank Damien Stehlé and the authors of CLT13 and CLT15 Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi for fruitful discussions and remarks. The authors of the Seoul National University, Jung Hee Cheon, Changmin Lee, and Hansol Ryu, were supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2014R1A2A1A11050917).

References

[BF01] Dan Boneh and Matt Franklin. Identity-based encryption from the Weil pairing. In *Advances in Cryptology–CRYPTO 2001*, pages 213–229. Springer, 2001.

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

[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 attacks on multilinear maps and their limitations. In *Advances in Cryptology–CRYPTO 2015*, pages 247–266. Springer, 2015.

[CHL+15] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In *Advances in Cryptology–EUROCRYPT 2015*, pages 3–12. Springer, 2015.

[CLR15] Jung Hee Cheon, Changmin Lee, and Hansol Ryu. Cryptanalysis of the new CLT multilinear maps. Cryptology ePrint Archive, Report 2015/934, 2015. http://eprint.iacr.org/.

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

[CLT15] Jean-Sebastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. New multilinear maps over the integers. In *Advances in Cryptology–CRYPTO 2015*, pages 267–286. Springer, 2015.

[Cor15] Jean-Sebastien Coron. Cryptanalysis of GGH15 multilinear maps. Cryptology ePrint Archive, Report 2015/1037, 2015. http://eprint.iacr.org/.

[8] We may note in passing that in a random-access or log-RAM computing model [Für14], which is more realistic than the multitape model, the estimated determinant complexity would already be slightly lower than the security parameter.

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

[Dix82] John D. Dixon. Exact solution of linear equations using P-adic expansions. *Nümerische Mathematik*, 40(1):137–141, 1982.

[Für14] Martin Fürer. How fast can we multiply large integers on an actual computer? In *LATIN 2014: Theoretical Informatics*, pages 660–670. Springer, 2014.

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *Eurocrypt*, volume 7881, pages 1–17. Springer, 2013.

[GGH+13b] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 40–49. IEEE, 2013.

[GGH15] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In *Theory of Cryptography*, pages 498–527. Springer, 2015.

[Hal15a] Shai Halevi. Cryptographic graded-encoding schemes: Recent developments. TCS+ online seminar, available at https://sites.google.com/site/plustcs/past-talks/20150318shaihaleviibmtjwatson, 2015.

[Hal15b] Shai Halevi. Graded encoding, variations on a scheme. Technical report, Cryptology ePrint Archive, Report 2015/866, 2015. http://eprint. iacr. org, 2015.

[HJ15] Yupu Hu and Huiwen Jia. Cryptanalysis of GGH map. Technical report, Cryptology ePrint Archive, Report 2015/301, 2015.

[HSW13] Susan Hohenberger, Amit Sahai, and Brent Waters. Full domain hash from (leveled) multilinear maps and identity-based aggregate signatures. In *Advances in Cryptology–CRYPTO 2013*, pages 494–512. Springer, 2013.

[Jou00] Antoine Joux. A one round protocol for tripartite Diffie–Hellman. In *Algorithmic number theory*, pages 385–393. Springer, 2000.

[MF15] Brice Minaud and Pierre-Alain Fouque. Cryptanalysis of the new multilinear map over the integers. Cryptology ePrint Archive, Report 2015/941, 2015. http://eprint.iacr.org/.

[Möl08] Niels Möller. On Schönhage's algorithm and subquadratic integer GCD computation. *Mathematics of Computation*, 77(261):589–607, 2008.

[MS04] Thom Mulders and Arne Storjohann. Certified dense linear system solving. *Journal of Symbolic Computation*, 37(4):485–510, 2004.

[Sha85] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in cryptology*, pages 47–53. Springer, 1985.

[Sto05] Arne Storjohann. The shifted number system for fast linear algebra on integer matrices. *Journal of Complexity*, 21(4):609 – 650, 2005. Festschrift for the 70th Birthday of Arnold Schonhage.

[Sto09] Arne Storjohann. Integer matrix rank certification. In *Proceedings of the 2009 International Symposium on Symbolic and Algebraic Computation*, pages 333–340. ACM, 2009.

[VDGHV10] Marten Van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *Advances in cryptology–EUROCRYPT 2010*, pages 24–43. Springer, 2010.

[Zim15] Joe Zimmerman. How to obfuscate programs directly. In *Advances in Cryptology-EUROCRYPT 2015*, pages 439–467. Springer, 2015.

A Short Introduction to Multilinear Maps

In this section we give a brief introduction to multilinear maps to make our article self-contained. In particular we only consider symmetric multilinear maps. We refer the interested reader to [GGH13a, Hal15b] for a more thorough presentation.

A.1 Multilinear Maps and Graded Encoding Schemes

Cryptographic multilinear maps were introduced by Boneh and Silverberg [BS03], as a natural generalization of bilinear maps stemming from pairings on elliptic curves, which had found striking new applications in cryptography [Jou00, BF01, ...]. A (symmetric) multilinear map is defined as follows.

Definition 1 (Multilinear Map [BS03]). *Given two groups $\mathbb{G}, \mathbb{G}_T$ of the same prime order, a map $e : \mathbb{G}^\kappa \to \mathbb{G}_T$ is a* κ-multilinear map *iff it satisfies the following two properties:*

1. *for all $a_1, \ldots, a_\kappa \in \mathbb{Z}$ and $x_1, \ldots, x_\kappa \in \mathbb{G}$,*
$$e(x_1^{a_1}, \ldots, x_\kappa^{a_\kappa}) = e(x_1, \ldots, x_\kappa)^{a_1 \cdots a_\kappa}$$
2. *if g is a generator of $\mathbb{G}$, then $e(g, \ldots, g)$ is a generator of $\mathbb{G}_T$.*

A natural special case are *leveled* multilinear maps:

Definition 2 (Leveled Multilinear Map [HSW13]). *Given $\kappa + 1$ groups $\mathbb{G}_1, \ldots, \mathbb{G}_\kappa, \mathbb{G}_T$ of the same prime order, and for each $i \leq \kappa$, a generator $g_i \in \mathbb{G}_i$, a* κ-leveled multilinear map *is a set of bilinear maps $\{e_{i,j} : \mathbb{G}_i \times \mathbb{G}_j \to G_{i+j} | i, j, i + j \leq \kappa\}$ such that for all i, j with $i + j \leq \kappa$, and all $a, b \in \mathbb{Z}$:*
$$e_{i,j}(g_i^a, g_j^b) = g_{i,j}^{ab}.$$

Similar to public-key encryption [DH76] and identity-based cryptosystems [Sha85], multilinear maps were originally introduced as a compelling target for cryptographic research, without a concrete instantiation [BS03]. The first multilinear map was built ten years later in the breakthrough construction of Garg, Gentry and Halevi [GGH13a]. More accurately, what the authors proposed was a *graded encoding scheme*, and to this day all known cryptographic multilinear maps constructions are actually variants of graded encoding schemes [Hal15b]. For this reason, and because both constructions have similar expressive power, the term "multilinear map" is used in the literature in place of "graded encoding scheme", and we follow suit in this article.

Graded encoding schemes are a relaxed definition of leveled multilinear map, where elements x_i^a for $x_i \in \mathbb{G}_i, a \in \mathbb{Z}$ are no longer required to lie in a group.

Instead, they are regarded as "encodings" of a ring element a at level i, with no assumption about the underlying structure. Formally, encodings are thus defined as general binary strings in $\{0,1\}^*$. In the following definition, $S_i^{(\alpha)}$ should be regarded as the set of encodings of a ring element α at level i.

Definition 3 (Graded Encoding System [GGH13a]). *A κ-graded encoding system consists of a ring R and a system of sets $\mathcal{S} = \{S_i^{(\alpha)} \subset \{0,1\}^* | \alpha \in R, 0 \leq i \leq \kappa\}$, with the following properties:*

1. *For each fixed i, the sets $S_i^{(\alpha)}$ are pairwise disjoint as α spans R.*
2. *There is an associative binary operation '$+$' and a self-inverse unary operation '$-$' on $\{0,1\}^*$ such that for every $\alpha_1, \alpha_2 \in R$, every $i \leq \kappa$, and every $u_1 \in S_i^{(\alpha_1)}, u_2 \in S_i^{(\alpha_2)}$, it holds that:*

$$u_1 + u_2 \in S_i^{(\alpha_1+\alpha_2)} \quad \text{and} \quad -u_1 \in S_i^{(-\alpha_1)}$$

   where $\alpha_1 + \alpha_2$ and $-\alpha_1$ are addition and negation in R.
3. *There is an associative binary operation '$\times$' on $\{0,1\}^*$ such that for every $\alpha_1, \alpha_2 \in R$, every $i_1, i_2 \in \mathbb{N}$ such that $i_1 + i_2 \leq \kappa$, and every $u_1 \in S_{i_1}^{(\alpha_1)}, u_2 \in S_{i_2}^{(\alpha_2)}$, it holds that $u_1 \times u_2 \in S_{i_1+i_2}^{(\alpha_1 \cdot \alpha_2)}$. Here $\alpha_1 \cdot \alpha_2$ is the multiplication in R, and $i_1 + i_2$ is the integer addition.*

Observe that a leveled multilinear map is a graded encoding system where $R = \mathbb{Z}$ and, with the notation from the definitions, $S_i^{(\alpha)}$ contains the single element g_i^α. Also note that the behavior of addition and multiplication of encodings with respect to the levels i is the same as that of a graded ring, hence the *graded* qualifier.

All known constructions of graded encoding schemes do not fully realize the previous definition, insofar as they are "noisy"[9]. That is, all encodings have a certain amount of noise; each operation, and especially multiplication, increases this noise; and the correctness of the scheme breaks down if the noise goes above a certain threshold. The situation in this regard is similar to somewhat homomorphic encryption schemes.

A.2 Multilinear Map Procedures

The exact interface offered by a multilinear map, and called upon when it is used as a primitive in a cryptographic scheme, varies depending on the scheme. However the core elements are the same. Below we reproduce the procedures for manipulating encodings defined in [CLT15], which are a slight variation of [GGH13a].

In a nutshell, the scheme relies on a trusted third party that generates the instance (and is typically no longer needed afterwards). Users of the instance (that

[9] In fact the question of achieving the functionality of multilinear maps without noise may be regarded as an important open problem [Zim15].

is, everyone but the generating trusted third party) cannot encode nor decode arbitrary encodings: they can only combine existing encodings using addition, negation and multiplication, and subject to the limitation that the level of an encoding cannot exceed κ. The power of the multilinear map comes from the zero-testing (resp. extraction) procedure, which allows users to test whether an encoding at level κ encodes zero (resp. roughly get a λ-bit "hash" of the value encoded by a level-κ encoding).

Here users are also given access to random level-0 encodings, and have the ability to re-randomize encodings, as well as promote any encoding to a higher-level encoding of the same element. These last functionalities are tailored towards the application of multilinear maps to one-round multi-party Diffie-Hellman. In general different applications of multilinear map require different subsets of the procedures below, and sometimes variants of them.

$\mathsf{instGen}(1^\lambda, 1^\kappa)$: the randomized instance procedure takes as input the security parameter λ, the multilinearity level κ, and outputs the public parameters $(\mathsf{pp}, \boldsymbol{p}_{zt})$, where pp is a description of a κ-graded encoding system as above, and $\boldsymbol{p}_{zt}$ is a zero-test parameter (see below).

$\mathsf{samp}(\mathsf{pp})$: the randomized sampling procedure takes as input the public parameters pp and outputs a level-0 encoding $u \in S_0^{(\alpha)}$ for a nearly uniform $\alpha \in R$.

$\mathsf{enc}(\mathsf{pp}, i, u)$: the possibly randomized encoding procedure takes as input the public parameters pp, a level $i \leq \kappa$, and a level-0 encoding $u \in S_0^{\alpha}$ for some $\alpha \in R$, and outputs a level-i encoding $u' \in S_i^{(\alpha)}$.

$\mathsf{reRand}(\mathsf{pp}, i, u)$: the randomized rerandomization procedure takes as input the public parameters pp, a level $i \leq \kappa$, and a level-i encoding $u \in S_i^{\alpha}$ for some $\alpha \in R$, and outputs another level-i encoding $u' \in S_i^{(\alpha)}$ of the same α, such that for any $u_1, u_2 \in S_i^{(\alpha)}$, the output distributions of $\mathsf{reRand}(\mathsf{pp}, i, u_1)$ and $\mathsf{reRand}(\mathsf{pp}, i, u_2)$ are nearly the same.

$\mathsf{neg}(\mathsf{pp}, u)$: the negation procedure is deterministic and that takes as input the public parameters pp, and a level-i encoding $u \in S_i^{(\alpha)}$ for some $\alpha \in R$, and outputs a level-i encoding $u' \in S_i^{(-\alpha)}$.

$\mathsf{add}(\mathsf{pp}, u_1, u_2)$: the addition procedure is deterministic and takes as input the public parameters pp, two level-i encodings $u_1 \in S_i^{(\alpha_1)}, u_2 \in S_i^{(\alpha_2)}$ for some $\alpha_1, \alpha_2 \in R$, and outputs a level-i encoding $u' \in S_i^{(\alpha_1+\alpha_2)}$.

$\mathsf{mult}(\mathsf{pp}, u_1, u_2)$: the multiplication procedure is deterministic and takes as input the public parameters pp, two encodings $u_1 \in S_i^{(\alpha_1)}, u_2 \in S_j^{(\alpha_2)}$ of some $\alpha_1, \alpha_2 \in R$ at levels i and j such that $i + j \leq \kappa$, and outputs a level-$(i + j)$ encoding $u' \in S_{i+j}^{(\alpha_1 \cdot \alpha_2)}$.

$\mathsf{isZero}(\mathsf{pp}, u)$: the zero-testing procedure is deterministic and takes as input the public parameters pp, and an encoding $u \in S_\kappa^{(\alpha)}$ of some $\alpha \in R$ at the maximum level κ, and outputs 1 if $\alpha = 0$, 0 otherwise, with negligible probability of error (over the choice of $u \in S_\kappa^{(\alpha)}$).

$\mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, u)$: the extraction procedure is deterministic and takes as input the public parameters pp, the zero-test parameter $\boldsymbol{p}_{zt}$, and an encoding $u \in S_\kappa^{(\alpha)}$ of some $\alpha \in R$ at the maximum level κ, and outputs a λ-bit string s such that:

1. For $\alpha \in R$ and $u_1, u_2 \in S_\kappa^{(\alpha)}$, $\mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, u_1) = \mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, u_2)$.
2. The distribution $\{\mathsf{ext}(\mathsf{pp}, \boldsymbol{p}_{zt}, v) | \alpha \leftarrow R, v \in S_\kappa^{(\alpha)}\}$ is nearly uniform over $\{0,1\}^\lambda$.