LP Solutions of Vectorial Integer Subset Sums – Cryptanalysis of Galbraith's Binary Matrix LWE

Gottfried Herold and Alexander May

Horst Görtz Institute for IT-Security
Ruhr-University Bochum, Germany
Faculty of Mathematics
gottfried.herold@rub.de, alex.may@rub.de

Abstract. We consider Galbraith's space efficient LWE variant, where the $(m \times n)$-matrix A is binary. In this binary case, solving a vectorial subset sum problem over the integers allows for decryption. We show how to solve this problem using (Integer) Linear Programming. Our attack requires only a fraction of a second for all instances in a regime for m that cannot be attacked by current lattice algorithms. E.g. we are able to solve 100 instances of Galbraith's small LWE challenge $(n, m) = (256, 400)$ all in a fraction of a second. We also show under a mild assumption that instances with $m \leq 2n$ can be broken in polynomial time via LP relaxation. Moreover, we develop a method that identifies weak instances for Galbraith's large LWE challenge $(n, m) = (256, 640)$.

Keywords: Binary matrix LWE, Linear Programming, Cryptanalysis

1 Introduction

Over the last decade, the Learning with Errors (LWE) problem [16] has proved to be extremely versatile for the construction of various cryptographic primitives. Since LWE is as hard as worst-case lattice problems, it is consider one of the most important post-quantum candidates. Let us recall that an LWE instance consists of a random $(m \times n)$-matrix $\mathbf{A}$ with elements from $\mathbb{Z}_q$ and an m-dimensional vector $\boldsymbol{b} \in \mathbb{Z}_q^m$, where $\boldsymbol{b} = \mathbf{A}\boldsymbol{s} + \boldsymbol{e} \bmod q$ with a secret random $\boldsymbol{s} \in \mathbb{Z}_q^n$ and where the entries of $\boldsymbol{e} \in \mathbb{Z}_q^m$ are from a discretized normal distribution.

The LWE decisional problem is to distinguish $(\mathbf{A}, \boldsymbol{b})$ from $(\mathbf{A}, \boldsymbol{u})$ for random $\boldsymbol{u} \in \mathbb{Z}_q^m$. While LWE has some intriguing hardness properties, it is known that one has to choose quite large n in order to reach a desired security level against lattice reduction attacks. This in turn makes the size of LWE instances $(\mathbf{A}, \boldsymbol{b})$, and thus the size of public keys, undesirably large. For practical reasons, people therefore looked into various variants of LWE, such as ring-LWE [13,14], LWE with short secret [15,2] or LWE with short error [15,10]. Recently, some special instances of ring-LWE were identified to have serious weaknesses [6,4], but these instances were not suggested for cryptographic use. Moreover, it was shown that

LWE with binary secrets and errors can be attacked in slightly subexponential time $2^{\mathcal{O}(n/\log\log n)}$ by a BKW-type algorithm[11], where LWE dimension $n = 128$ was practically broken within half a day. Also, LWE with binary secret leads to more efficient lattice attacks [3]. While choosing special variants of LWE seems to slightly decrease the security, the improved attacks do not substantially endanger the security of these variants in general.

In this paper, we look at another LWE variant due to Galbraith [8]. In this variant, $\mathbf{A}$ is replaced by a *binary* matrix. This makes Galbraith's variant very tempting for low-weight devices that are not capable of storing a sufficiently large LWE instance.

In [8], Galbraith instantiates Regev's encryption system [16] with his binary matrix $\mathbf{A}$ and suggests to use the parameters $(n, m, q) = (256, 640, 4093)$ that were originally proposed by Lindner and Peikert [12] for Regev's original scheme. Galbraith also gives a thorough security analysis based on lattices, where in his experiments he fixes n and tries to break encryption for increasing m. Based on this analysis, he concludes that instances with $m \geq 400$ might be hard to break with lattice techniques.

For Regev's original scheme, security follows from hardness of LWE for appropriate parameters; this is not automatically the case for binary matrix $\mathbf{A}$ without changing parameters. For Galbraith's choices, in order to break encryption, one can solve an equation of the form $\boldsymbol{u}\mathbf{A} = \boldsymbol{c}_1$ for a known matrix $\mathbf{A} \in \{0,1\}^{m\times n}$, some known ciphertext component $\boldsymbol{c}_1 \in \mathbb{Z}^n$ and some unknown vector $\boldsymbol{u} \in \{0,1\}^m$. In other words, one has to find a subset of all rows of $\mathbf{A}$ that sums to $\boldsymbol{c}_1$. We call this problem therefore a *vectorial integer subset sum*. If the unknown vector $\boldsymbol{u}$ is short, a vectorial integer subset sum can certainly be solved by finding a closest vector in some appropriate lattice. This is the standard analysis that was carried out in [8] against this avenue of attack.

However, a vectorial integer subset sum is by its definition also an Integer Linear Programming (ILP) problem. Namely, we are looking for an integral solution $\boldsymbol{u} \in \mathbb{Z}^m$ of m linear equations over the integers. While it is known that ILP is in general NP-hard, it is also known that in many cases removing the integrality constraint on $\boldsymbol{u}$ provides a lot of useful information about the problem. Removing the integrality constraint is called a *LP relaxation* of the problem. Without integrality constraints, the resulting problem can be solved in polynomial time, using e.g. the ellipsoid method [9].

We show under a mild assumption on $\mathbf{A}$ that the vectorial subset sum problem can for parameters $m \leq 2n$ be solved by its LP relaxation (with success probability $\frac{1}{2}$). More precisely, the LP solution has the property that it is already integral. This in turn means that vectorial integer subset sums with $m \leq 2n$ can be solved in polynomial time. In practice, we are able to solve instances with $n = 256$ and $m \leq 2n$ in a fraction of a second. Notice that this is already a regime for m that seems to be infeasible to reach with current lattice reduction algorithms.

However, $m \leq 2n$ does not quite suffice to break Galbraith's $(n, m) = (256, 640)$-challenge in practice. Namely, when we look at instances with $m > 2n$

the success probability of our MATLAB ILP solver drops quite quickly – when we allow only some fixed, small computation time. Yet, when looking at a large number of instances of our vectorial integer subset sums, we realize experimentally that there is still a significant number of weak instances that are vulnerable to LP relaxation with some additional tricks (such as e.g. the cutting plane method). More concretely, we are able to show that at least 1 out of 2^{15} instances of Regev-type encryptions with $(n, m) = (256, 640)$ can be solved in about 30 minutes. Interestingly, we are able to compute a simple score for every instance I that accurately predicts whether I is indeed weak – based on an estimation of the volume of the search space that comes from the LP relaxation. We find that such a quick test for identifying weak instances I is a quite remarkable property of Linear Programming. We are not aware of a similar property for other cryptanalytic methods. We hope that our results motivate more cryptanalytic research using (Integer) Linear Programming.

Note that our attack breaks Galbraith's instantiation of LWE encryption with binary matrices, but does not break binary LWE itself. Due to that, our attack allows ciphertext recovery, but not key recovery.

Our paper is organized as follows. In Section 2, we recall Galbraith's scheme and its cryptanalysis challenges. In Section 3, we model vectorial integer subset sums in form of an Integer Linear Programming. We attack instances with $m \leq 2n$ in Section 4 and show that they actually admit a polynomial time attack. In Section 5, we show how to identify weak instances for large m and we present our experimental results for Galbraith's large challenge $(n, m) = (256, 640)$.

2 Galbraith's Binary Matrix LWE

Let us briefly recall Regev's LWE encryption scheme. Let q be prime. One chooses a public $\mathbf{A} \in_R \mathbb{Z}_q^{m \times n}$ and a private $\boldsymbol{s} \in_R \mathbb{Z}_q^n$. One then compute $\boldsymbol{b} = \mathbf{A}\boldsymbol{s} + \boldsymbol{e} \bmod q$, where the e_i are sampled from a discrete normal distribution with mean 0 and standard deviation σ. The public key consists of $(\mathbf{A}, \boldsymbol{b})$.

For encrypting some message $M \in \{0, 1\}$, one chooses a random nonce $\boldsymbol{u} \in_R \{0, 1\}^m$ and computes the ciphertext

$$\boldsymbol{c} = (\boldsymbol{c}_1, c_2) = (\boldsymbol{u}\mathbf{A} \bmod q, \langle \boldsymbol{u}, \boldsymbol{b} \rangle + M \lfloor \tfrac{q}{2} \rfloor \bmod q) \in \mathbb{Z}_q^n \times \mathbb{Z}_q.$$

For decryption to 0 respectively 1, one checks whether $c_1\boldsymbol{s} - c_2$ is closer to 0 respectively $\frac{q}{2}$.

After analyzing lattice attacks, Lindner and Peikert [12] suggest to use the parameters

$$(n, m, q) = (256, 640, 4093)$$

for medium security level and estimate that these parameters offer roughly 128-bit security. However, for these parameters the public key $(\mathbf{A}, \boldsymbol{b})$ has already 247 kilobytes, which is way too much for constrained devices.

Therefore, Galbraith [8] suggested to construct the public matrix $\mathbf{A}$ with **binary entries** simply from the seed of a PRNG. All that one has to store in

this case is the seed itself, and the vector $\boldsymbol{b}$. A similar trick is also used in other contexts to shorten the public key size [5].

Moreover, Galbraith gives a thorough security analysis of his LWE variant, based on its lattice complexity. In his security analysis he considers the problem of recovering the nonce $\boldsymbol{u}$ from

$$\boldsymbol{c}_1 = \boldsymbol{u}\mathbf{A}. \tag{1}$$

Notice that since now $\mathbf{A} \in \{0,1\}^{m \times n}$, every entry of $\boldsymbol{c}_1$ is an inner product of two random binary length-m vectors. Thus, the entries of c_1 are random variables from a binomial distribution $B(m, \frac{1}{4})$ with expected value $\frac{m}{4}$. Since $\frac{m}{4} \ll q$, the equality $\boldsymbol{c}_1 = \boldsymbol{u}\mathbf{A}$ does not only hold modulo q, but also over the integers.

Hence, recovering $\boldsymbol{u}$ from $(\boldsymbol{c}_1, \mathbf{A})$ can be seen as a *vectorial integer subset sum* problem. Once $\boldsymbol{u}$ is recovered, one can easily subtract $\langle \boldsymbol{u}, \boldsymbol{b} \rangle$ from c_2 and thus recover the message m. Hence, solving the vectorial integer subset sum problem gives a *ciphertext only message recovery attack*.

We would like to stress that this attack does not allow for key recovery of $\boldsymbol{s}$. We also note that in Regev's original scheme, the security proof shows IND-CPA security assuming that the LWE problem is hard. For this reduction, we need that c_1 is essentially independent of $\mathbf{A}$, which is proven using the Leftover Hash Lemma by setting parameters sufficiently large. In particular, $\boldsymbol{u}$ is required to have sufficient entropy and Eq. (1) has many solutions for $\boldsymbol{u}$ in Regev's non-binary scheme, whereas the parameters in Galbraith's binary scheme are set such that $\boldsymbol{u}$ is the unique solution to Eq. (1). Due to that, our attack does not give an attack on binary LWE. In fact, binary LWE was shown to be at least as secure as standard LWE in [1], provided n is increased by a factor $\mathcal{O}(\log q)$. Consequently, it seems unlikely that the attack extends to binary LWE.

2.1 Previous Cryptanalysis and Resulting Parameter Suggestions

In his security analysis, Galbraith attacks the vectorial integer subset sum by lattice methods. Namely, he first finds an arbitrary integer solution $\boldsymbol{w} \in \mathbb{Z}^m$ with $\boldsymbol{c}_1 = \boldsymbol{w}\mathbf{A}$. Then he solves CVP with target vector $\boldsymbol{w}$ in the lattice

$$L = \{\boldsymbol{v} \in \mathbb{Z}^m \mid \boldsymbol{v}\mathbf{A} \equiv 0 \bmod q\}.$$

Let $\boldsymbol{v}$ be a CVP-solution, then we usually have $\boldsymbol{u} = \boldsymbol{w} - \boldsymbol{v}$.

Galbraith reports that for $n = 256$ and $m \in [260, 340]$, the CVP-method works well. He further conjectures that with additional tricks one should be able to handle values up to $m = 380$ or 390, but that "it would be impressive to solve cases with $m > 400$ without exploiting weeks or months of computing resources".

Based on his analysis, Galbraith raised the two following cryptanalysis challenges:

- C1 with $(n, m) = (256, 400)$: The goal is to compute $\boldsymbol{u}$ from $(\mathbf{A}, c_1)$ in less than a day on an ordinary PC.

– C2 with $(n, m) = (256, 640)$: The goal is mount an attack using current computing facilities that would take less than a year.

According to Galbraith, breaking C1 should be interpreted "as causing embarrassment to the author", while C2 should be considered a "total break".

3 Modeling our Vectorial Integer Subset Sum as an Integer Linear Program

In the canonical form of an Integer Linear Program (ILP), one is given *linear constraints*

$$\mathbf{A}'\boldsymbol{x} \leq \boldsymbol{b}', \boldsymbol{x} \geq 0 \text{ and } \boldsymbol{x} \in \mathbb{Z}^m,$$

for which one has to maximize a *linear objective function* $\langle \boldsymbol{f}, \boldsymbol{x} \rangle$ for some $\boldsymbol{f} \in \mathbb{R}^m$ that can be freely chosen.

Notice that it is straightforward to map our vectorial integer subset sum problem $\boldsymbol{u}A = \boldsymbol{c}_1$ from Eq. (1) into an ILP. Namely, we define the inequalities

$$\begin{aligned} \mathbf{A}^\mathsf{T}\boldsymbol{u} &\leq \boldsymbol{c}_1 \\ -\mathbf{A}^\mathsf{T}\boldsymbol{u} &\leq -\boldsymbol{c}_1 \text{ and} \\ u_i &\leq 1 \text{ for all } i = 1, \ldots, m. \\ u_i &\geq 0 \text{ for all } i = 1, \ldots, m. \end{aligned} \tag{2}$$

We can for simplicity chose $\boldsymbol{f} = \mathbf{0}$, since we are interested in *any* feasible solution to Eq. (2), and it is not hard to see that by the choice of our parameters our solution $\boldsymbol{u}$ is a unique feasible solution. Namely, look at the map

$$\begin{aligned} \{0,1\}^m &\to \left(B\big(m, \tfrac{1}{4}\big)\right)^n, \\ \boldsymbol{u} &\mapsto \boldsymbol{u}\mathbf{A}, \end{aligned}$$

where $X \sim B(m, \frac{1}{4})$ is a binomially distribution random variable with m experiments and $\Pr[X = 1] = \frac{1}{4}$ for each experiment. Notice that the j^{th} entry, $1 \leq j \leq n$, of $\boldsymbol{u}\mathbf{A}$ can be written as $u_1 a_{1,j} + \ldots + u_m a_{m,j}$, where we have the event X_i that $u_i a_{i,j} = 1$ iff $u_i = a_{i,j} = 1$, i.e. with probability $\frac{1}{4}$. Hence, we can model the entries of $\boldsymbol{u}\mathbf{A}$ as random variables from $B(m, \frac{1}{4})$.

For the usual parameter choice $q > m$, the solution $\boldsymbol{u}$ of Eq. (2) is unique as long as this map is injective, i.e. as long as the entropy of $\left(B(m, \frac{1}{4})\right)^n$ is larger than m. The entropy of the binomial distribution $\left(B(m, \frac{1}{4})\right)^n$ is roughly $\frac{n}{2}\log_2(\frac{3}{8}\pi e m)$. Thus, one can compute for which m we obtain unique solutions $\boldsymbol{u}$. Choosing e.g. $n = 256$, we receive unique $\boldsymbol{u}$ for $m \leq 1500$. Hence, in the remaining paper we can safely assume unique solutions to our vectorial subset sum problem.

4 Attacking $m \leq 2n$: Solving Challenge C1

We ran 100 instances of Eq. (2) on an ordinary 2.8 GHz laptop with $n = 256$ and increasing m. We used the ILP solver from MATLAB 2015, which was stopped whenever it did not find a solution after time $t_{\max} = 10$ seconds. We found that the success probability of our attack dropped from 100% at $m = 490$ to approximately 1% at $m = 590$, cf. Table 1. The largest drop of success probability takes place slightly after $m = 2n$.

For comparison, we also solved the LP relaxation, i.e. Eq. (2) without integrality constraint on $\boldsymbol{u}$. This is much faster than ILP, so we solved 1000 instances for each m. We checked whether the returned non-integral solution matched our desired integral solution for $\boldsymbol{u}$, in which case we call a run successful. The success rate of LP relaxation is also given in Table 1.

It turns out that Galbraith's small C1 challenge can already solely be solved by its LP relaxation. Since LP relaxation is only the starting point for ILP, it does not come as a surprise that ILP has a slightly larger success rate. However, it is impressive that LP relaxation alone is already powerful enough to solve a significant fraction of all instances.

m	400	450	480	490	500	510	**512**	520
Success(ILP)	100%	100%	100%	100%	96%	83%	79%	63%
Success(LP)	100%	99.6%	93.3%	82.3%	68.8%	55.6%	**48.1%**	35.4%
m	530	540	550	560	570	580	590	600
Success(ILP)	60%	32%	25%	12%	3%	1%	1%	0%
Success(LP)	19.8%	11.0%	4.5%	1.9%	0.8%	0.3%	0%	0%

Table 1. Success probability for solving Eq. (2) for $n = 256$. We used MATLAB 2015 and restricted to $t_{\max} = 10$ seconds for the ILP.

We now give a theoretical justification for the strength of LP relaxation, showing that under some mild heuristic, for $m \leq 2n$, the solution of the LP relaxation is unique. Since, by construction, we know that there is an integral solution $\boldsymbol{u}$ to Eq. (2), uniqueness of the solution directly implies that the LP solver has to find the desired $\boldsymbol{u}$.

In the following lemma, we replace our linear constraints from $\mathbf{A}$ by some random linear constraints from some matrix $\bar{\mathbf{A}}$ over the reals. This will give us already uniqueness of the solution $\boldsymbol{u}$. Afterwards, we will argue why replacing $\bar{\mathbf{A}}$ back by our LWE matrix $\mathbf{A}$ should not affect the lemma's statement.

Lemma 1. *Let $\boldsymbol{u} \in \{0,1\}^{2n}$. Let $\bar{\mathbf{A}} \in \mathbb{R}^{n \times 2n}$ be a random matrix, whose rows are uniformly distributed on the sphere around $\mathbf{0} \in \mathbb{R}^{2n}$. Then*

$$\Pr[\nexists \boldsymbol{x} \in (\mathbb{R} \cap [0,1])^{2n} \mid \bar{\mathbf{A}}\boldsymbol{x} = \bar{\mathbf{A}}\boldsymbol{u}, \boldsymbol{x} \neq \boldsymbol{u}] = \frac{1}{2}.$$

Proof. Let us look at the $2n$-dimensional unit cube $U_{2n} = \{\boldsymbol{x} \in (\mathbb{R} \cap [0,1])^{2n}\}$. Obviously $\mathbf{0}, \boldsymbol{u} \in U_{2n}$, both lying at corners of U_{2n}. Now, let us assume wlog. that $\boldsymbol{u} = \mathbf{0}$ (which can be achieved by reflections). Let H be the hyperplane defined by the kernel of $\bar{\mathbf{A}}$.

Since $\bar{\mathbf{A}}$ is randomly chosen from $\mathbb{R}^{n \times 2n}$, it has full rank n with probability 1: since we chose the entries of $\bar{\mathbf{A}}$ from the reals $\mathbb{R}$, we avoid any problems that might arise from co-linearity. Thus, H as well as its orthogonal complement $H^\perp$ have dimension n. Notice that $H^\perp = \mathrm{Im}(\bar{\mathbf{A}}^\mathsf{T})$. By construction, both H and $H^\perp$ intersect U_{2n} in the corner $\mathbf{0} = \boldsymbol{u}$. We are interested whether one of the hyperplanes goes through U_{2n}.

The answer to this question is given by Farkas' Lemma [7], which tells us that *exactly one* of H and $H^\perp$ passes through U_{2n}. Notice first that not both can pass through U_{2n}. Now assume that H intersects U_{2n} only in the zero point $\mathbf{0}$. Then Farkas' Lemma tells us that there is a vector in its orthogonal complement $H^\perp$ that fully intersects U_{2n}. Notice that again by having vectors over the reals, the intersection $H^\perp \cap U_{2n}$ is n-dimensional.

By the randomness of B, the orientation of H in $\mathbb{R}^{2n}$ is uniformly random, and hence the same holds for the orientation of $H^\perp$. Since H and $H^\perp$ share exactly the same distribution, and since by Farkas' Lemma exactly one out of both has a trivial intersection with U_{2n}, we have

$$\Pr[H \cap U_{2n} = \{\boldsymbol{u}\}] = \Pr[H^\perp \cap U_{2n} = \{\boldsymbol{u}\}] = \frac{1}{2}.$$

Let $\boldsymbol{b} = \bar{\mathbf{A}}\boldsymbol{u} = \mathbf{0}$. Since $H = \ker(\bar{\mathbf{A}})$, it follows that $\boldsymbol{u}$ is a unique solution to the equation $\bar{\mathbf{A}}\boldsymbol{x} = \boldsymbol{b}$ in the case that H has trivial intersection with U_{2n}. □

Theorem 1. *Under the heuristic assumption that our matrix $\mathbf{A}^\mathsf{T}$ behaves like a random $(n \times m)$-matrix, whose rows are uniformly distributed on the sphere around 0^m, LP relaxation solves Eq.* (2) *in polynomial time for all $m \leq 2n$.*

Proof. Notice that the case $m = 2n$ follows directly from Lemma 1, since LP relaxation has to find the unique solution $\boldsymbol{u}$, and its running time is polynomial using e.g. the ellipsoid method. For the case $m < 2n$ we can simply append $2n - m$ additional columns to $\mathbf{A}^\mathsf{T}$, and add a random subset of these to $\boldsymbol{c}_1$.

Now let us say a word about the heuristic assumption from Theorem 1. Our assumption requires that the discretized $\mathbf{A}^\mathsf{T}$ defines a random orientation of a hyperplane just as $\bar{\mathbf{A}}$. Since $\mathbf{A}^\mathsf{T}$ has by definition only positive entries, its columns always have non-negative inner product with the all-one vector 1^n. This minor technical problem can be fixed easily by centering the entries of $\mathbf{A}^\mathsf{T}$ around 0 via the following transformation of Eq. (2):

First, guess the Hamming weight $w = \sum_{i=1}^{m} u_i$. Then subtract $(\frac{1}{2}, \ldots, \frac{1}{2})$ from every column vector of $\mathbf{A}^\mathsf{T}$ and finally subtract $\frac{w}{2}$ from every entry of $\boldsymbol{c}_1$. After this transformation $\mathbf{A}^\mathsf{T}$ has entries uniform from $\{\pm\frac{1}{2}\}$ and should fulfill the desired heuristic assumption of Theorem 1.

5 Attacking $m = 640$: Solving Challenge C2

In order to tackle the $m = 640$ challenge, we could in principle proceed as in the previous section, identify a weak instance for e.g. $m = 590$, brute-force guess 50 coordinates of $\boldsymbol{u}$ and run each time an ILP solver for 10 seconds.

However, we found out experimentally that even in dimension $m = 640$ the density of weak instances is not negligible. Hence, it seems to be much more effective to identify weak instances than to brute-force coordinates. So in the following we try to identify what makes particular instances weak.

We follow the paradigm that an ILP is the easier to solve, the more the LP relaxation "knows about the problem". In particular, we expect that a problem is easy to solve if the solution polytope P of the LP relaxation of Eq. (2) is small. In the extreme case, if $P = \{\boldsymbol{u}\}$, then the problem can be solved by the LP solver alone (cf. Thm. 1). To quantify the size of the solution space in an easy-to-compute way, we compute the length of a random projection of P. It turns out that this length, henceforth called *score* gives a very good prediction on the hardness of an instance.

More concretely, for an instance $I = (\mathbf{A}, \boldsymbol{c})$, we choose a vector $\boldsymbol{r}$ with random direction. Then we maximize and minimize the linear objective function $\langle \boldsymbol{r}, \boldsymbol{u} \rangle$ under the linear constraints given by the LP relaxation of Eq. (2) and consider their difference D. Clearly, $S_{\boldsymbol{r}} := \frac{D}{\|\boldsymbol{r}\|}$ is the length of the orthogonal projection of P onto the span of $\boldsymbol{r}$. Formally, the *score* of an instance I wrt. to some direction $\boldsymbol{r}$ is defined as follows.

Definition 1. *Let $I = (\mathbf{A}, \boldsymbol{c})$ be an instance. Consider the solution polytope P of the LP relaxation of Eq. (2), i.e. P is defined as $P = [0,1]^m \cap \{\boldsymbol{x} \mid \mathbf{A}^\mathsf{T}\boldsymbol{x} = \boldsymbol{c}\}$. Let $\boldsymbol{r} \in \mathbb{R}^m$. Then the score $S_{\boldsymbol{r}}$ is defined via*

$$\begin{aligned} f_{\max} &:= \max_{\boldsymbol{x} \in P} \langle \boldsymbol{r}, \boldsymbol{x} \rangle \\ f_{\min} &:= \min_{\boldsymbol{x} \in P} \langle \boldsymbol{r}, \boldsymbol{x} \rangle \\ S_{\boldsymbol{r}} &:= \frac{f_{\max} - f_{\min}}{\|\boldsymbol{r}\|} \end{aligned} \tag{3}$$

Note that $S_{\boldsymbol{r}}$ can be computed by solving two LP problems, hence in polynomial time.

Since $S_{\boldsymbol{r}}$ quantifies the search space for the ILP, instances with small score should be easier to compute. For $m = 640$, we computed the scores of 2^{19} instances, which took approximately 1 second per instance.

Independence of $\boldsymbol{r}$ and Reliability of Our Score. We experimentally confirm that for a given instance I, the value of $S_{\boldsymbol{r}}$ is mainly a function of I and does not depend significantly on the particular choice of $\boldsymbol{r}$. Therefore, we choose the fixed vector $\boldsymbol{r} = (1, \ldots, 1, -1, \ldots, -1)$ for $\boldsymbol{r}$ with exactly $\frac{m}{2}$ ones and $\frac{m}{2}$ -1's. We use the score $S = S_{\boldsymbol{r}}$ for this particular choice of $\boldsymbol{r}$ and sort instances according to S.

We confirm that the score S is a very good predictor for the success of ILP solvers and the success probability drops considerably at some cutoff value for S. E.g. for $m = 520$ and within a 10 second time limit, we find that we can solve

- $> 99\%$ of instances with $S \leq 1.22$,
- 60% of instances with $1.22 \leq S \leq 1.54$ and
- $< 3\%$ of instances with $S > 1.54$.

Distribution of S. Average values for S can be found in Table 2. Fig. 1 shows the distribution of S. Note that while the distribution looks suspiciously Gaussian for $m = 640$, there is a considerable negative skewness and the tail distribution towards 0 is much fatter than for a Gaussian (cf. Fig. 2). This fat tail enables us to find a significant fraction of weak instances even for large m.

Notice that a score $S = 0$ basically means that LP relaxation finds the solution.

m	400	450	480	490	500	510	512	520
average of S	0	0.002	0.07	0.22	0.43	0.69	0.83	1.15
m	530	540	550	560	570	580	590	600
average of S	1.76	2.16	2.74	3.16	3.60	4.04	4.34	4.80
m	610	620	630	640				
average of S	5.18	5.52	5.83	6.18				

Table 2. Average values for S for $n = 256$ and varying m. We used 1000 instances for each m.

Results for $m = 640$. We generated a large number $N = 2^{19}$ of instances with $n = 256$, $m = 640$, and tried to solve only those 271 instances with the lowest score S, which in our case meant $S < 3.2$. We were able to solve 16 out of those 271 weakest instances in half an hour each. We found 15 instances with $S < 2.175$, of which we solved 12. The largest value of S, for which we could solve an instance, was $S \approx 2.6$.

Fixing Coordinates. Let us provide some more detailed explanation why an ILP solver works well on instances with small score S. Consider some $\boldsymbol{r} \in \{0, \pm 1\}^m$ of low Hamming weight $|\boldsymbol{r}|_1 = w$, so $\|\boldsymbol{r}\| = \sqrt{w}$. Heuristically, we expect that $S_{\boldsymbol{r}}$ should be approximately S, as $S_{\boldsymbol{r}}$ mainly depends on the instance and not on the choice of $\boldsymbol{r}$. Of course, for a vector $\boldsymbol{r} \in \{0, \pm 1\}^m$ with low Hamming weight we have

$$S_{\boldsymbol{r}} = \frac{1}{\sqrt{w}} \Big(\max_{\boldsymbol{x} \in P} \langle \boldsymbol{r}, \boldsymbol{x} \rangle - \min_{\boldsymbol{x} \in P} \langle \boldsymbol{r}, \boldsymbol{x} \rangle \Big) \leq \frac{1}{\sqrt{w}} \Big(\max_{\boldsymbol{x} \in [0,1]^m} \langle \boldsymbol{r}, \boldsymbol{x} \rangle - \min_{\boldsymbol{x} \in [0,1]^m} \langle \boldsymbol{r}, \boldsymbol{x} \rangle \Big) = \sqrt{w},$$

but that only means we should expect $S_{\boldsymbol{r}}$ to be even smaller. Since we know that for the true integer solution $\boldsymbol{u}$, we have $\langle \boldsymbol{r}, \boldsymbol{u} \rangle \in \mathbb{Z}$, we can add the *cuts*

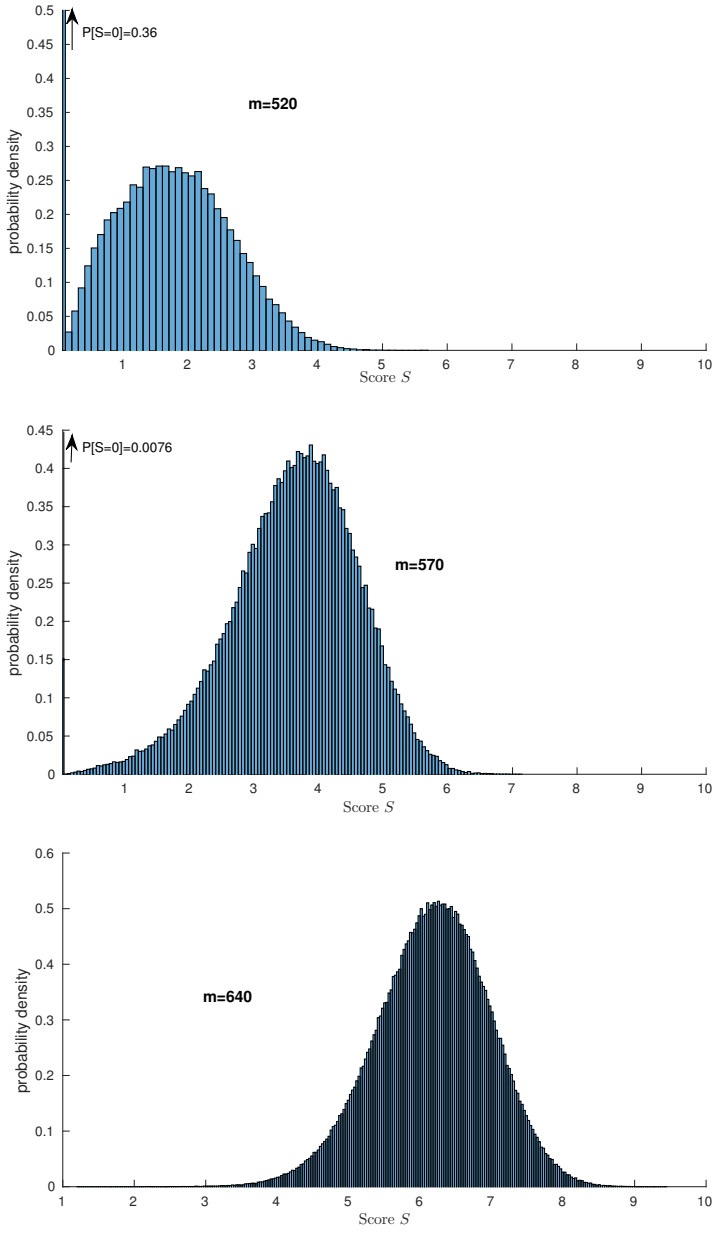

Fig. 1. pdf's of S for $n = 256$ and varying values of m. Note that the y-axis is cropped and does not show the true density at $S = 0$ (where the distribution technically does not even have a finite continuous density). We rather give the probability for $S = 0$. For $m = 640$, we never encountered an instance with $S = 0$.

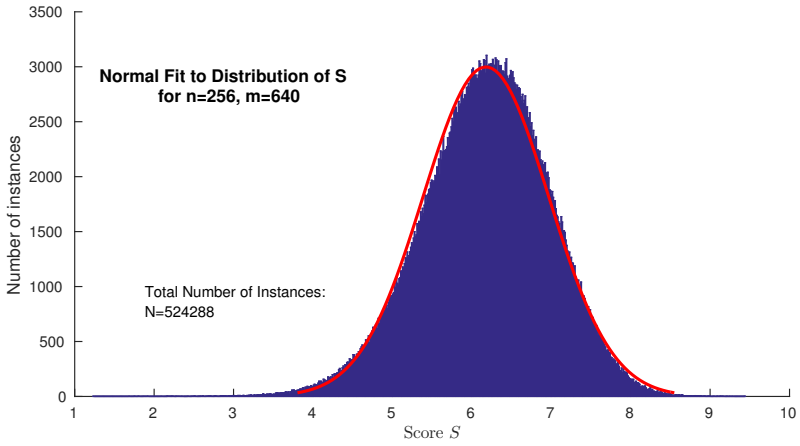

Fig. 2. Comparison of distribution of S for $n = 256$, $m = 640$ with a normal distribution. The distribution of S has negative skewness and a much fatter tail towards 0. Hence, we obtain more weak instances than we would expect from a normal distribution.

$\langle \boldsymbol{r}, \boldsymbol{u} \rangle \leq \lfloor f_{\max} \rfloor$ and $\langle \boldsymbol{r}, \boldsymbol{u} \rangle \geq \lceil f_{\min} \rceil$ to the set of equations, where $f_{\max}$ resp. $f_{\min}$ are the maximum resp. minimum computed for $S_{\boldsymbol{r}}$.

This is a special case of what is called *cut generation* in Integer Linear Programming. If $S_{\boldsymbol{r}} < \sqrt{w}$, i.e. $f_{\max} - f_{\min} < w$, then adding such a new inequality always makes the solution space of the LP relaxation smaller. In fact, such an inequality restricts the possible set that w out of the m variables u_i can jointly obtain. So if $S_{\boldsymbol{r}} < \sqrt{w}$ for many different $\boldsymbol{r}$, we get lots of sparse relations between the u_i. Such inequalities are called *good cuts*.

In particular, consider the case $w = 1$ and $\boldsymbol{r} = (0, 0, \ldots, 0, 1, 0, \ldots, 0)$, i.e. we maximize / minimize an individual variable u_i over P. If this maximum is < 1, we know that $u_i = 0$ holds and if the minimum is > 0, we know $u_i = 1$. So if $S_{\boldsymbol{r}} < 1$ holds for some $\boldsymbol{r}$ with $|\boldsymbol{r}|_1 = 1$, we can fix one of the u_i's and reduce the number of unknowns by one – which makes fixing further u_i's even easier. If the score S is small, we expect that the ILP solver can find lots of such good cuts, possibly even cuts with $w = 1$.

Indeed, in all instances that we could solve, some variables could be fixed by such good cuts with $w = 1$. For dimensions $m \leq 550$, most instances that were solved by the ILP could be solved by such cuts alone.

In fact, we preprocessed our 271 weak instances for $m = 640$ by trying to fix each individual coordinate. This alone was sufficient to determine an average of > 100 individual coordinates of the solution $\boldsymbol{u}$ for $S < 2.175$, and in one case it was sufficient to completely solve the problem.

6 Conclusion

According to Galbraith's metric for the challenge C2 in Section 3, the results of Section 5 can be seen as total break for binary matrix LWE. On the other hand, one could easily avoid weak instances I by simply rejecting weak I's during ciphertext generation. This would however violate the idea of lightweight encryption with binary matrix LWE.

Still, during our experiments we got the feeling that the vectorial integer subset sum problem gets indeed hard for large m, even for its weakest instances. So Galbraith's variant might be safely instantiated for large m, but currently we find it hard to determine m's that fulfill a concrete security level of e.g. 128 bit. One possibility to render our attack inapplicable is to change parameters such that modular reductions mod q occur in Eq. (1), since our attack crucially relies on the fact that we work over $\mathbb{Z}$. Note here that while there are standard ways to model modular reduction via ILP as $\boldsymbol{c}_1 = \boldsymbol{u}\mathbf{A} - \boldsymbol{k}q$, this renders LP relaxation useless: by allowing non-integral $\boldsymbol{k}$, we can choose any value for $\boldsymbol{c}_1, \boldsymbol{u}$.

Acknowledgements: The authors would like to thank Bernhard Esslinger and Patricia Wienen for comments.

References

1. D. Boneh, K. Lewi, H. W. Montgomery, and A. Raghunathan. Key homomorphic PRFs and their applications. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 410–428, Santa Barbara, CA, USA, Aug. 18–22, 2013. Springer, Heidelberg, Germany. 4
2. Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors. In D. Boneh, T. Roughgarden, and J. Feigenbaum, editors, *45th ACM STOC*, pages 575–584, Palo Alto, CA, USA, June 1–4, 2013. ACM Press. 1
3. J. A. Buchmann, F. Göpfert, R. Player, and T. Wunderer. On the hardness of LWE with binary error: Revisiting the hybrid lattice-reduction and meet-in-the-middle attack. *IACR Cryptology ePrint Archive*, 2016:89, 2016. 2
4. W. Castryck, I. Iliashenko, and F. Vercauteren. Provably weak instances of ring-lwe revisited. *Eurocrypt 16*, 2016. 1
5. J.-S. Coron, D. Naccache, and M. Tibouchi. Public key compression and modulus switching for fully homomorphic encryption over the integers. In D. Pointcheval and T. Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 446–464, Cambridge, UK, Apr. 15–19, 2012. Springer, Heidelberg, Germany. 4
6. Y. Elias, K. E. Lauter, E. Ozman, and K. E. Stange. Provably weak instances of ring-LWE. In R. Gennaro and M. J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 63–92, Santa Barbara, CA, USA, Aug. 16–20, 2015. Springer, Heidelberg, Germany. 1
7. J. Farkas. Theorie der einfachen Ungleichungen. *Journal für die reine und angewandte Mathematik (Crelle's Journal)*, 124:1–27, 1902. `http://resolver.sub.uni-goettingen.de/purl?GDZPPN002165023`. 7
8. S. D. Galbraith. Space-efficient variants of cryptosystems based on learning with errors. *https://www.math.auckland.ac.nz/ sgal018/pubs.html*, 2013. 2, 3

9. M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012. 2
10. T. Güneysu, V. Lyubashevsky, and T. Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In E. Prouff and P. Schaumont, editors, *CHES 2012*, volume 7428 of *LNCS*, pages 530–547, Leuven, Belgium, Sept. 9–12, 2012. Springer, Heidelberg, Germany. 1
11. P. Kirchner and P.-A. Fouque. An improved BKW algorithm for LWE with applications to cryptography and lattices. In R. Gennaro and M. J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 43–62, Santa Barbara, CA, USA, Aug. 16–20, 2015. Springer, Heidelberg, Germany. 2
12. R. Lindner and C. Peikert. Better Key Sizes (and Attacks) for LWE-Based Encryption. In *Topics in Cryptology - CT-RSA 2011*, volume 6558 of *Lecture Notes in Computer Science*, pages 319–339. Springer, 2011. 2, 3
13. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23, French Riviera, May 30 – June 3, 2010. Springer, Heidelberg, Germany. 1
14. V. Lyubashevsky, C. Peikert, and O. Regev. A toolkit for ring-LWE cryptography. In T. Johansson and P. Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 35–54, Athens, Greece, May 26–30, 2013. Springer, Heidelberg, Germany. 1
15. D. Micciancio and C. Peikert. Hardness of SIS and LWE with small parameters. In R. Canetti and J. A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 21–39, Santa Barbara, CA, USA, Aug. 18–22, 2013. Springer, Heidelberg, Germany. 1
16. O. Regev. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. In *STOC 2005*, pages 84–93. ACM, 2005. 1, 2