The Mersenne Low Hamming Combination Search Problem can be reduced to an ILP Problem

Alessandro Budroni, Andrea Tenti

Universitet i Bergen, Norway
{alessandro.budroni,andrea,tenti}@uib.no

Abstract. In 2017, Aggarwal, Joux, Prakash, and Santha proposed an innovative NTRU-like public-key cryptosystem that was believed to be quantum resistant, based on Mersenne prime numbers $q = 2^N - 1$. After a successful attack designed by Beunardeau, Connolly, Géraud, and Naccache, the authors revised the protocol which was accepted for Round 1 of the Post-Quantum Cryptography Standardization Process organized by NIST. The security of this protocol is based on the assumption that a so-called Mersenne Low Hamming Combination Search Problem (MLHCombSP) is hard to solve. In this work, we present a reduction of MLHCombSP to Integer Linear Programming (ILP). This opens new research directions for assessing the concrete robustness of such cryptosystem. In particular, we uncover a new family of weak keys, for whose our attack runs in polynomial time.

Keywords: Post-Quantum Cryptography, Public-Key Cryptography, Integer Linear Programming, Mersenne-Based Cryptosystem

1 Introduction

In [1], Aggarwal, Joux, Prakash, and Santha introduced a new public-key encryption scheme similar to the NTRU cryptosystem [2] that employs the properties of Mersenne numbers.

A Mersenne number is an integer $q = 2^N - 1$ so that N is prime. One can associate to each element in the ring $\mathbb{Z}_q$ a binary string representing $0 \leq a < q$ of the class $[a] \in \mathbb{Z}_q$. The secret key is a pair of elements F and $G \in \mathbb{Z}_q$ with Hamming weight $h < \sqrt{N/10}$. Let R be chosen at random from $\mathbb{Z}_q$; the public key is given by the pair $(R, T \equiv RF + G \mod q)$. The security assumption (and the mathematical problem that supports the robustness of this cryptosystem) is that it is hard to recover F and G, knowing only R and T. This assumption is called *Mersenne Low Hamming Combination Search Problem* (MLHCombSP).

The version in [1] is the second iteration of the cryptosystem, first presented in [3]. The security assumptions were based on a problem similar to MLHCombSP and called *Mersenne Low Hamming Ratio Search Problem* (MLHRatioSP). That system has been successfully attacked by Beunardeau et al. in [4].

The attack is performed via a series of calls to an SVP-oracle. Its complexity has been estimated by de Boer et al. in [5]. They also showed that a Meet-in-the-Middle attack is possible using locality-sensitive hashing, which improves upon brute force. However, Beunardeau et al. attack turned out to be the most effective of the two. After the publications of these works, Aggarwal et al. revised the protocol [1] to prevent the above attacks from being effective against the full-scale cipher.

This protocol has been accepted to the Round 1 of the Post-Quantum Cryptography Standardization Process organized by NIST. However, it does not appear among the proposals for Round 2.

1.1 Our Contribution/Outline

In this work we present a non-trivial reduction of the underlying mathematical problem of [1] to a relatively low-dimensional Integer Linear Programming (ILP) instance. The secret key is a solution of the resulting ILP instance with probability p, that depends on the size of F and G.

In section 2, we introduce notation and related work. Furthermore, we recap the Beunardeau et al. attack against [3] with a generalization to the MLH-CombSP. Section 3 describes our reduction together with the success probability analysis. There, we describe a variation in the description of the ILP to be solved, that allows some flexibility for the attacker. In particular, one can perform a trade-off between the success probability of the attack and the dimension of the resulting ILP. The application of this trade-off is shown by two examples. In section 4, we describe a new family of weak keys (F, G) and the probability of such a pair to appear. These keys are characterized by a long sequence of zeros in their bit-wise representation. The family is obtained by performing two independent rotations on F and G. After these rotations, F and G become small and easy to recover. In this way the size of the set of the weak keys increases. For example, for $N = 1279$ and $h = 17$ (parameters used in [4]), a random key is weak in the sense of Beunardeau et al. with probability $\sim 2^{-34}$. In what follows, we estimate that a random key is weak with probability $\sim 2^{-11}$.

2 Preliminaries

Definition 1 *Let N be a prime number and let $q = 2^N - 1$. Then q is called a Mersenne number. If q is also prime, then it is called Mersenne prime number.*

Let $\mathrm{seq}_N : \{0, ..., q-1\} \to \{0,1\}^N$ be the map which associates to each A the corresponding N-bits binary representation $\mathrm{seq}_N(A)$ with most-significant bit to the left.

Let us consider an integer $0 \leq B < q$, seq_N maps $[B] \in \mathbb{Z}_q$ to the N-bits binary representation of B. We define the *Hamming weight* $w(B)$ of B as the Hamming weight of $\mathrm{seq}_N(B)$, i.e. the number of 1s in $\mathrm{seq}_N(B)$.

Lemma 1 *Let $k \geq 0$ be a positive integer, let A be an N-bits number, and let $q = 2^N - 1$. Then $\text{seq}_N(2^k A \mod q)$ corresponds to a rotation of $\text{seq}_N(A)$ of k positions to the left and $\text{seq}_N(2^{-k} A \mod q)$ corresponds to a rotation of k positions to the right.*

Proof. We prove it by induction on k. Write $\text{seq}_N(A) = (A_{N-1}, ..., A_1, A_0)$, where A_{N-1} is the most significant bit of A. Then we can represent A as

$$A = A_{N-1} \cdot 2^{N-1} + ... + A_1 \cdot 2 + A_0.$$

If we multiply A by 2 modulo q we obtain

$$\begin{aligned} 2 \cdot A &\equiv A_{N-1} \cdot 2^N + A_{N-2} \cdot 2^{N-1} + ... + A_1 \cdot 2^2 + A_0 \cdot 2 \mod q \\ &\equiv A_{N-2} \cdot 2^{N-1} + ... + A_1 \cdot 2^2 + A_0 \cdot 2 + A_{N-1} \mod q. \end{aligned}$$

Then $\text{seq}_N(2 \cdot A) = (A_{N-2}, ..., A_0, A_{N-1})$, i.e. the left rotation of 1 position of $\text{seq}_N(A)$.
By inductive hypothesis, $\text{seq}_N(2^k \cdot A)$ corresponds to the left rotation of k positions of $\text{seq}_N(A)$, then $\text{seq}_N(2^{k+1} \cdot A) = \text{seq}_N(2 \cdot 2^k \cdot A)$ corresponds to the left rotation of one position of $\text{seq}_N(2^k \cdot A)$, that is the left rotation of $k+1$ positions of $\text{seq}_N(A)$. The case right rotations of $\text{seq}_N(A)$ follows trivially.

The security of the Aggarwal et al. cryptosystem [1] relies on the assumption that the following two problems are hard to solve.

Mersenne Low Hamming Ratio Search Problem Let $q = 2^N - 1$ be a Mersenne prime number, $h < N$ an integer, F and G two integers chosen at random from the set of N-bit numbers with Hamming weight h. Let $H < q$ be the non-negative integer such that

$$H \equiv \frac{F}{G} \mod q. \tag{1}$$

The *Mersenne Low Hamming Ratio Search Problem* (MLHRatioSP) is to find (F, G) given h and H.

Mersenne Low Hamming Combination Search Problem Let $q = 2^N - 1$ be a Mersenne prime number, $h < N$ an integer, R a random N-bit number, and F, G integers chosen at random from the set of N-bits numbers with Hamming weight h. Let $T < q$ be the non-negative integer such that

$$RF + G \equiv T \mod q. \tag{2}$$

The *Mersenne Low Hamming Combination Search Problem* (MLHCombSP) is to find (F, G) given h and the pair (R, T).

In [3], the authors suggest to choose N and h to be such that $\binom{N-1}{h-1} \geq 2^\lambda$ and $4h^2 < N$, for a desired λ-bit security level. After the publications of the attacks by Beunardeau et al. [4] and De Boer et al. [5], the authors revised the choice of the parameters to be such that $h = \lambda$ and $10h^2 < N$, see [1].

2.1 Previous Attacks

Brute force attack In [3], Aggarwal et al. showed that a brute force attack to the MLHRatioSP would require $\binom{N-1}{h-1}$ trials. One assumes that one of the two secret numbers, say F, has a 1 in the most significant bit (condition that can be obtained by a rotation of $\text{seq}_N(F)$). Then one should check, for every N-bits number with 1 as most significant bit and weight h, whether the corresponding G through relation (1) has weight h. This approach does not apply to the MLHCombSP, which instead requires $\binom{N}{h}$ trials.

Meet-in-the-Middle attack De Boer et al. [5] showed that a Meet-in-the-Middle attack to MLHRatioSP is possible using locality-sensitive hashing with complexity $\tilde{O}\left(\sqrt{\binom{N-1}{h-1}}\right)$ on classical computers and $\tilde{O}\left(\sqrt[3]{\binom{N-1}{h-1}}\right)$ on quantum computers.

Weak Keys and Lattice attack Following the parameters' setting in [3], Beunardeau et al. found a weak key attack to the MLHRatioSP for the case when both F and G happen to have bits set to 1 only in their right halves, i.e. $F, G < \sqrt{2^N}$ [4]. This event happens with probability approximately 2^{-2h}, for $h << N$.

Following the above idea, Beunardeau et al. also presented a more general attack to the MLHRatioSP which consists of guessing a decomposition of F and G into windows of bits such that all the '1's are "close" to the right-most bit of such windows. Then F and G can be recovered through a lattice reduction algorithm such as LLL [6]. Even if Beunardeau et al. showed that this attack practically hits the security estimations in [3], they did not present any clear asymptotic analysis of its complexity. However, de Boer et al. [5], computed the complexity of this attack.

In [1], the authors stated that the above attack likely generalizes to the MLHCombSP case. Building directly on the work presented in [5], we show in the next subsection that this is true. However we refer the reader to [4] and [5] for a more detailed description.

2.2 The Beunardeau et al. attack on MLHCombSP

Since F is taken at random among the N-bits numbers with Hamming weight h, w.h.p. the '1' valued bits of $\text{seq}_N(F)$ do not appear in big clusters along the N possible positions. One then computes an interval-like partition $\mathcal{P}$ of $\{0, \ldots, N-1\}$ at random, i.e. each set of $\mathcal{P}$ is of the form $\{a, a+1, \ldots, b-1, b\}$, with $0 \leq a < b < N$. Let all '1' valued bits of $\text{seq}_N(F)$ fall in the right-half of one of the sets of $\mathcal{P}$. Then, each set of $\mathcal{P}$ corresponds to a binary substring of $\text{seq}_N(F)$, corresponding in turn to a "small" number. Therefore, the array of these numbers can be seen as a representation of F.

Let $\mathcal{P} = \{P_1, .., P_k\}$ and $\mathcal{Q} = \{Q_1, ..., Q_l\}$ be two interval-like partitions of $\{0, ..., N-1\}$ and $(R, T) \in \mathbb{Z}_q^2$ be public parameters of an MLHCombSP

instance. Let p_i, q_i be the smallest elements of P_i, Q_i respectively. We consider the following integer lattice.

$$\mathcal{L}_{\mathcal{P},\mathcal{Q},R,T} = \left\{ (x_1, ..., x_k, y_1, ..., y_l, u) \mid R \cdot \sum_{i=1}^{k} 2^{p_i} \cdot x_i + \sum_{j=1}^{l} 2^{q_i} \cdot y_i - uT \equiv 0 \mod q \right\}$$

The above defined lattice $\mathcal{L}_{\mathcal{P},\mathcal{Q},R,T}$ has volume $\det(\mathcal{L}_{\mathcal{P},\mathcal{Q},R,T}) = q$ and rank $d = k + l + 1$. Let $(F, G) \in \mathbb{Z}_q^2$ be such that $w(F) = w(G) = h$ and $RF + G \equiv T$ as in a MLHCombSP instance. Define the vector

$$\boldsymbol{s} = (f_1, ..., f_k, g_1, ..., g_l, 1) \in \mathcal{L}_{\mathcal{P},\mathcal{Q},R,T},$$

where $0 \leq f_i < 2^{|P_i|}$ and $0 \leq g_j < 2^{|Q_j|}$ are the unique natural numbers such that $\sum_{i=1}^{k} f_i \cdot 2^{p_i} = F$ and $\sum_{j=1}^{l} g_j \cdot 2^{q_j} = G$, where $|\cdot|$ denotes the cardinality operator. One wishes to find the vector $\boldsymbol{s}$ by a lattice reduction algorithm applied to $\mathcal{L}_{\mathcal{P},\mathcal{Q},R,T}$.

The lattice $\mathcal{L}_{\mathcal{P},\mathcal{Q},R,T}$ is very similar to the one defined in [5] for the MLHRatioSP and their success probability analysis of the attack holds for this case too. Therefore the following conclusions follow directly from the work of de Boer et al.

Given two partitions $\mathcal{P}$ and $\mathcal{Q}$ of $\{0, ..., N-1\}$ with block size at least $N/d + \Theta(\log N)$, where $d = k + l + 1$ with $k = |P|$ and $l = |Q|$. The success probability of finding the vector $\boldsymbol{s} \in \mathcal{L}_{\mathcal{P},\mathcal{Q},R,T}$ using a SVP-oracle is $2^{-2h+o(1)}$.

Remark 1 *The above attack is actually a simplified version of the attack of Beunardeau et al. Indeed, a more general attack can be made by considering the variation of partition sizes and the fraction of each partition block. This variant of the attack has success probability* $2^{-(2+\delta)h+o(1)}$*, for some small constant* $\delta > 0$ *[5].*

Remark 2 *In practice, instead of an SVP-oracle, the LLL algorithm [6] which has polynomial complexity is used. This decreases the overall complexity of the attack, but the success probability is decreased too [5].*

The above attack was made against the parameters setting contained in the first version of Aggarwal et al. work. However, as already mentioned, in the most recent version of their work the authors revisited the protocol in order to withstand it.

2.3 Integer Linear Programming

An *Integer Linear Programming* (ILP) problem in his *canonical form* is defined as follows. Given a matrix $A \in \mathbb{Q}^{m \times n}$ and two vectors $\boldsymbol{c} \in \mathbb{Q}^n$ and $\boldsymbol{b} \in \mathbb{Q}^m$, minimize (or maximise) the quantity

$$\boldsymbol{c}^T \boldsymbol{x}$$

subject to

$$\begin{cases} A\boldsymbol{x} \leq \boldsymbol{b}, \\ \boldsymbol{x} \geq 0, \\ \boldsymbol{x} \in \mathbb{Z}^n \end{cases}$$

The number n is called the *dimension* of the ILP. An *ILP-oracle* is an oracle that solves any ILP instance.

Solving a general ILP is proved to be NP-hard [7]. Nevertheless, understanding the complexity of specific families of ILP problems is not an easy task: it can widely vary from case to case [8]. For example, when the problem can be reduced to a simple *Linear Programming* problem, it is proved that it has polynomial complexity [9]. H. Lenstra also provided a polynomial algorithm for certain ILP problems [10].

Nowadays there exists families of ILP solving algorithms, for example *Branch and Bound* [11], *Lagrange relaxation* [12], *Column Generation* [13], and the *Cutting Planes* [14], whose implementations [15, 16] are able to solve in practice relatively challenging instances.

3 ILP Reduction

Let R, T be two random elements of $\mathbb{Z}_q^*$. We define the map $\varphi : \mathbb{Z}_q \to \mathbb{Z}_q$ sending $X \mapsto -RX+T$. Any point on the graph of φ, namely $\{(X, \varphi(X))\}_{X \in \mathbb{Z}_q}$, satisfying the condition that both coordinates have Hamming weight equal to h is a solution to the MLHCombSP. We denote such condition as *the graph condition*.

We notice that φ is bijective, for it is the combination of two bijective functions (i.e. multiplication times a nonzero element of a field and sum with an element of the underlying group). This means that for any subset $\mathcal{U} \subseteq \mathbb{Z}_q$, the restriction $\varphi_{|\mathcal{U}}$ is injective. Hence, $|\operatorname{Im}(\varphi_{|\mathcal{U}})| = |\mathcal{U}|$.

Let $\mathcal{V}$ be another subset of $\mathbb{Z}_q$, and assume that φ behaves like a random bijection from $\mathbb{Z}_q$ to itself. The probability that a random element of $\operatorname{Im}(\varphi_{|\mathcal{U}})$ is in $\mathcal{V}$ is given by $\frac{|\mathcal{V}|}{2^N-1}$. Hence the expected size of $\operatorname{Im}(\varphi_{|\mathcal{U}}) \cap \mathcal{V}$ is given by the mean of the Hypergeometric distribution [17] in $|\mathcal{U}|$ draws, from a population of size $2^N - 1$ that contains $|\mathcal{V}|$ objects that yield a success. That is:

$$\mathbb{E}(|\operatorname{Im}(\varphi_{|\mathcal{U}}) \cap \mathcal{V}|) = \frac{|\mathcal{U}||\mathcal{V}|}{2^N - 1}. \tag{3}$$

Remark 3 *The expectation* (3) *is obtained assuming that φ is a random bijection of $\mathbb{Z}_q$. We verified experimentally that this is accurate also for $\varphi(X) = -RX + T$.*

Let $\mathcal{U}, \mathcal{V} \subset \mathbb{Z}_q$ be such that $F \in \mathcal{U}$ and $G \in \mathcal{V}$. Then, (F, G) is a solution of the system of constraints

$$\begin{cases} T - Rx \equiv y \mod q, \\ x \in \mathcal{U}, \\ y \in \mathcal{V}. \end{cases} \tag{4}$$

For every fixed instance of $x \in \{0, \ldots, q-1\}$, there is exactly one $a \in \mathbb{Z}$ that satisfies $0 \leq T + aq - Rx < q$. It is possible to represent (4) in terms of integers:

$$\begin{cases} T + qa - Rx = y, \\ x \in \mathcal{U}, \\ y \in \mathcal{V}. \end{cases} \tag{5}$$

Assume that $\mathcal{U} = \{\mathrm{l}_{x_3}, \mathrm{l}_{x_3}+1, \ldots, \mathrm{u}_{x_3}-1, \mathrm{u}_{x_3}\}$ and $\mathcal{V} = \{\mathrm{l}_y, \mathrm{l}_y+1, \ldots, \mathrm{u}_y-1, \mathrm{u}_y\}$, for some integers $\mathrm{l}_{x_3}, \mathrm{u}_{x_3}, \mathrm{l}_y, \mathrm{u}_y < q$, and that $F \in \mathcal{U}$ and $G \in \mathcal{V}$. Assume also that (5) has a unique solution. Then, one can use an ILP solver to recover (F, G) from the following ILP instance. Minimize the quantity in the integer variables x_1, x_2, x_3, y:

$$Tx_1 + qx_2 - Rx_3 + 0y \tag{6}$$

with constraints

$$\begin{cases} y = Tx_1 + qx_2 - Rx_3, \\ x_1 = 1, \\ \mathrm{l}_{x_3} \leq x_3 \leq \mathrm{u}_{x_3}, \\ \mathrm{l}_y \leq Tx_1 + qx_2 - Rx_3 \leq \mathrm{u}_y\,. \end{cases} \tag{7}$$

Note that y is a redundant variable because it is set to be integer by default (T, q and R are integers) and it takes the value of the minimized quantity. Therefore the ILP (6) with constraints (7) has dimension 3.

Remark 4 *Our approach consists of looking for settings where the ILP has only one solution. For this reason, one can use any linear combination of the variables x_1, x_2, x_3, y in (6) to define the quantity to minimize.*

Finding good choices on $\mathcal{U}$ and $\mathcal{V}$ (i.e. small and containing F and G with high probability) is, in general, a difficult task. One can exploit the fact that F has weight exactly h to establish the following ILP problem in the integer variables $x_1, x_2, x_3, n_1, \ldots, n_N, y$:

$$Tx_1 + qx_2 - Rx_3 + 0n_1 + \cdots + 0n_N + 0y, \tag{8}$$

with constraints

$$\begin{cases} y = Tx_1 + qx_2 - Rx_3, \\ x_1 = 1, \\ x_3 = \sum_{i=1}^{N} n_i 2^{i-1}, \\ 0 \leq n_i \leq 1, \quad \text{for } i = 1, \ldots, N \\ \sum_{i=1}^{N} n_i = h, \\ \mathrm{l}_y \leq y \leq \mathrm{u}_y\,. \end{cases} \tag{9}$$

Using these constraints results in having $\mathcal{U}$ of size $|\mathcal{U}| = \binom{N}{h}$. On the other hand, the dimension of the ILP to be solved increased from 3 to $N+3$ (the variable y is redundant). In subsection 3.1, we show how to obtain a trade-off between the number of variables of the ILP to be solved or the size of $\mathcal{U}$.

3.1 Merging Bits

To reduce the number of variables of the ILP (8), one can merge more than one bit within each variable n_i. Consider the 2^s-ary representation of $x_3 = \sum_{i=1}^{\lceil N/s \rceil} 2^{s(i-1)} n_i$. For $s = 2$, each n_i can assume values in $\{0, 1, 2, 3\}$ and the total weight of x_3 varies between h and $2h$, as we prove in Proposition 1. Compared to the setting in (8) and (9), this results in an increase of the size of $\mathcal{U}$, and decrease of the number of variables: from $N+3$ to $\lceil N/2 \rceil + 3$.

Example 1 *Let $F = (00010011)$ and $h = 3$. By merging bits in pairs and assuming the ILP gives the correct solution, one gets $n_1 = (00)$, $n_2 = (01)$, $n_3 = (00)$, $n_4 = (11)$. The total sum results in $n_1 + n_2 + n_3 + n_4 = 4 \leq 2h = 6$.*

Using this method, it is possible to merge an arbitrary number of bits together. Let $S = \lceil N/s \rceil$. Consider the following system of linear inequalities.

$$\begin{cases} T + aq - Rx = y, \\ 2^h - 1 \leq y \leq 2^N - 2^{N-h}, \\ x = \sum_{i=1}^{S} 2^{s(i-1)} n_i, \\ 0 \leq n_i \leq 2^s - 1, \text{ for } 1 \leq i \leq S, \\ h \leq \sum_{i=1}^{S} n_i \leq 2^{s-1} h. \end{cases} \tag{10}$$

We prove in Proposition 1 that a solution $(X, \varphi(X))$ satisfying the graph condition is also a solution to the system of inequalities (10) and, therefore, it can be obtained via an ILP-oracle. Generally, choosing larger s implies a decrease of the probability that the ILP-oracle will return the correct solutions because the number of solutions satisfying the conditions increase.

Proposition 1 *Let $F, G \in \mathbb{Z}_q$ such that $\varphi(F) = G$ and the Hamming weight of $\mathrm{seq}_N(F)$ is h. Then there exists an integer solution $(x, y, a, n_1, \ldots, n_S)$ that solves the system (10), with $x = F$ and $y = G$.*

Proof. The first equation and the first inequality of (10) are satisfied by the definition of φ and by the fact that y is of weight h. The second equation and the second inequality represent the fact that we are writing x in base 2^s. Hence the only remaining thing to prove is that the last inequality holds.

Let $F = F_0 2^0 + \ldots + F_{N-1} 2^{N-1}$. We notice that $n_i = \sum_{j=0}^{s-1} F_{(i-1)s+j} 2^j$. For the fact that $\sum_{i=0}^{N-1} F(i) = h$, we conclude that

$$\sum_{i=1}^{S} n_i = \sum_{i=1}^{S} \sum_{j=0}^{s-1} F_{(i-1)s+j} 2^j \geq \sum_{i=1}^{S} \sum_{j=0}^{s-1} F_{(i-1)s+j} = h.$$

We prove the second inequality by induction on h. For $h = 1$, n_i is a string of weight 1 of s bits. That is at most 2^{s-1}.

Assuming that the inequality holds for $h-1$. If $n_i \leq 2^{s-1}$ for every i, the inequality is satisfied. Hence we assume that there exists one j for which $n_j > 2^{s-1}$. This means that the Hamming weight of $\mathrm{seq}_s(n_j) \geq 2$. Then one gets:

$$\sum_i n_i \leq 2^s + \sum_{i \neq j} n_i.$$

The sum of the Hamming weights of $\mathrm{seq}_s(n_j)$, $j \neq i$ is at most $h-2$. By inductive hypothesis, it follows that

$$\sum_i n_i \leq 2^s + 2^{s-1}(h-2) = 2^{s-1}h.$$

The following Proposition determines the size of $\mathcal{U}$ that one obtains from considering the constraints in (10).

Proposition 2 *Let $\mathcal{U}$ be the set containing all $0 \leq F < q$, whose 2^s-ary representation $F = \sum_{i=1}^{S} n_i 2^{i-1}$ satisfies $0 \leq n_i < 2^s$, for $1 \leq i \leq S$ and $h \leq \sum_{i=1}^{S} n_i \leq 2^{s-1}h$. Then*

$$|\mathcal{U}| = \sum_{d=h}^{2^{s-1}h} l_{2^s}(S, d),$$

where $l_t(n, d)$ is the number of integer solutions to $z_1 + \ldots + z_n = d$, $0 \leq z_i < t$.

Proof. Let d be one of the values of $\sum_{i=1}^{S} n_i$. For each d, we consider all the possible configurations of $n_1, \ldots, n_S$. Since each of these is bounded by $2^s - 1$, the number of legitimate configurations is $l_{2^s}(S, d)$.

Examples

In Table 3.1 and Table 3.2 we report the dimensions of the ILP instances, for two concrete parameter choices, resulting when varying s. For each case, we give the success probability of the attack that is computed as follows. Consider the set

$$\mathcal{V} = \{2^h - 1, 2^h, \ldots, 2^{N-t} - 2^{N-t-h}\},$$

with t satisfying $\log_2(|\mathcal{U}|) \leq t$, and $\mathcal{U}$ constructed as in Proposition 2 so that it contains F by default. Since $\log_2(|\mathcal{V}|) < N - t$, we have that $|\mathcal{U}||\mathcal{V}| < 2^N$. Therefore, because of the estimation in (3), if $G \in \mathcal{V}$, then we expect (10) to have a unique solution with $x = F$ and $y = G$. In this case, the private key (F, G) can be successfully retrieved with a query to an ILP-oracle and, therefore, the success probability of the attack is the probability that $G \in \mathcal{V}$.

Let E_G be the number of '0' valued bits before the first '1' in $\mathrm{seq}_N(G)$. One wants to compute the probability that, for a fixed s, $\log_2(|\mathcal{U}|) \leq E_G$. The random

variable E_G is distributed according to the negative hypergeometric distribution [18]:

$$Pr(E_G = t) = \frac{\binom{N-t-1}{N-t-h}}{\binom{N}{N-h}}.$$

In Table 3.1 and Table 3.2 the success probability and the number of ILP variables, computed as $\lceil N/s \rceil + 3$, are presented for a variety of s[1]. We notice that the parameters in Table 3.1 violate the guidelines given in [1]. These corresponds to the parameters choice of the first iteration of the protocol [3], and are the ones attacked by Beunardeau et al. [4].

s	Probability of success	Number of variables in ILP
1	$2^{-2.56}$	1282
2	$2^{-3.97}$	643
3	$2^{-6.13}$	430
4	$2^{-9.13}$	323
5	$2^{-12.94}$	259
6	$2^{-17.33}$	217
7	$2^{-21.73}$	186
8	$2^{-26.07}$	163
9	$2^{-30.47}$	146
10	$2^{-34.06}$	131

Table 3.1. parameters: $N = 1279$ and $h = 17$

Remark 5 *It is possible to increase the probability of success by taking into consideration the fact that G has weight h too. In this case, one would need to add more constraints to the set $\mathcal{V}$, as done for the set $\mathcal{U}$, resulting in an increase of the number of ILP variables.*

Remark 6 *The approach can be easily adjusted in order to solve the MLHRatioSP by taking $T = 0$.*

4 A new family of weak keys

In [4], a family of weak keys was introduced for the MLHRatioSP. Those were the ones for which all the '1' valued bits appeared in the right halves of $\mathrm{seq}_N(F)$

[1] The redundant variable y is not taken into account when computing the number of ILP variables.

s	Probability of success	Number of variables in ILP
1	$2^{-1.36}$	1282
2	$2^{-1.78}$	643
3	$2^{-2.80}$	430
4	$2^{-4.29}$	323
5	$2^{-6.26}$	259
6	$2^{-8.64}$	217
7	$2^{-11.18}$	186
8	$2^{-13.71}$	163
9	$2^{-16.27}$	146
10	$2^{-18.42}$	131

Table 3.2. parameters: $N = 1279$ and $h = 11$

and $\text{seq}_N(G)$. As noted in [1], one can break keys in this family by performing a rational reconstruction [19] of the quotient H defined by (1). A key in this family appears with probability approximately 2^{-2h}. Many keys which have a long sequence of zeros in the middle of their bit-sequence representation are not considered as weak keys in [4]. However, we show that this is a weakness that can be exploited.

Let $u, v \geq 0$ be two integers. The following transformation

$$(2^{(u-v)}R)(2^v F) + 2^u G \equiv 2^u T \mod q \tag{11}$$

gives a new instance of the MLHCombSP where the binary representation of the public values R and T are rotated by $u - v$ and u positions respectively. The secret values F and G are rotated by v and u positions. In practice, the transformation 11 allows us to rotate the binary representation of F and G by multiplying R and T times powers of 2. We will say that F is *minimized* by u rotations if

$$2^u F \mod q = \min_{\substack{0 \leq \tilde{u} < N \\ \tilde{u} \in \mathbb{N}}} 2^{\tilde{u}} F \mod q.$$

In particular, the binary representation of $2^u F$ has its longest sequence of consecutive zeros arranged to the left.

Our attack targets keys for which F and G contain long sequences of zeros in their binary representations. The approach consists of *minimizing* through rotations F and G using (11), then defining an ILP problem as in (6) with exceptionally tight bounds so that the solution is unique.

Assume F and G have been *minimized* already as much as possible through rotations, and let E_F and E_G be respectively the length of the largest sequences of consecutive zeros of F and G. In this case, one can set $\mathcal{V} = \{2^{N-E_G-1}, 2^{N-E_G-1} + 1, \ldots, 2^{N-E_G} - 1\}$ so that $|\mathcal{V}| = 2^{N-E_G-1}$. Let $\mathcal{U} \subset \mathbb{F}_q$ be such that $|\mathcal{U}| < 2^{E_G+1}$ and $F \in \mathcal{U}$. Then, because of estimation (3), there is only one expected solution

to the system of constraints:

$$\begin{cases} T - Rx \equiv y \mod q, \\ x \in \mathcal{U}, \\ y \in \mathcal{V}. \end{cases} \tag{12}$$

and this solution is $x = F$, $y = G$ with high probability. Once found, one can rotate back the solutions to retrieve the original pair. More in general, one expects a unique solution when $E_F + E_G \geq N$, and this can be retrieved by solving an ILP instance with only three variables.

Let A be a random positive N-bits integer such that $w(A) = h$, and let E_A be the length of the longest sequence of zeros in its binary representation considering rotations. Let $k \leq N-h$ be a positive integer. Our goal is to evaluate the probability

$$Pr(E_A = k). \tag{13}$$

Computing the exact probability (13) involves a recursive formula and it is hard in practice. For this reason, we used the following approximation. Consider the set of tuples

$$\Omega_{h,N} = \left\{ (a_1, a_2 \ldots, a_h) | a_1 \geq a_i \geq 0 \text{ for } i > 0, \sum_{i=1}^{h} a_i = N - h \right\}.$$

One can use the elements of $\Omega_{h,N}$ to represents the distribution of zeros in $\text{seq}(A)$ after the best rotational shift (the one that minimizes A). In particular, $a_1, a_2, ..., a_h$ represents the length of each sequence of consecutive zeros as follows

$$\underbrace{0...0}_{a_1 \ times} \ 1 \ \underbrace{0...0}_{a_2 \ times} \ 1 \ ... \ 1 \ \underbrace{0...0}_{a_h \ times} \ 1.$$

Note that $a_1 \geq a_i$, for $i > 1$, ensures that we are considering already the best shift possible.

Proposition 3 *Let $(a_1, a_2, ..., a_h)$ be chosen uniformly at random from $\Omega_{h,N}$. Then*

$$Pr(a_1 = k) = \frac{l_{k+1}(h-1, N-h-k)}{\sum_{i=0}^{N-h} l_{i+1}(h-1, N-h-i)}, \tag{14}$$

where $l_t(n, d)$ is the number of integer solutions to $z_1 + \ldots + z_n = d$, $0 \leq z_i < t$.

Proof. Let $\mathcal{A} : \Omega_{h,N} \to \mathbb{Z}$ be the function that maps $(a_1, a_2, ..., a_h)$ to a_1. For $0 \leq k \leq N-h$, $\mathcal{A}_{h,N}^{-1}(k)$ is the subset of $\Omega_{h,N}$ containing the tuples of the form $(k, a_2, \ldots, a_h)$ under the condition that $a_j \leq k$ for every j and that $\sum_{j=2}^{h} a_j = N - h - k$. The number of such tuples is $l_{k+1}(h-1, N-h-k)$.

We verified experimentally that the above model approximates well (13). Proposition 3 allowed us to discover a new family of weak keys, namely, pairs

(F, G) such that $E_F + E_G \geq N$. We used (14) to derive the following formula

$$Pr(E_F+E_G \geq N) \approx \sum_{a=N}^{2(N-h)} \sum_{k=0}^{a} \frac{l_{k+1}(h-1, N-h-k) l_{a-k+1}(h-1, N-h-k)}{\left(\sum_{i=0}^{N-h} l_{i+1}(h-1, N-h-i)\right)^2}.$$

Note that E_G and E_F are upper bounded by $N-h$, and their lower bound is $N/h-1$ if h divides N, $\lfloor N/h \rfloor$ otherwise. For $N = 1279$ and $h = 17$, the expected length of the longest sequence of zeros is approximately 256. For these parameters, there is approximately one key every 2^{11} with $E_F + E_G \geq N$, and such keys are weak as explained below. This improves upon Beunardeau et al. work for which 1 over 2^{34} keys is weak.

Let C_{ILP} be the cost of solving a system with 3 variables and unique solution as in (12). To retrieve the key, one must perform up to N^2 rotations to find the one that minimizes F and G. Since E_F is unknown, for each rotation, one makes a query to the ILP-oracle for every possible value $E_F = k$ can take. One must try up to $N - h - \lfloor N/h \rfloor + 2$ possible k to find a unique solution to (12), where $\mathcal{U} = \{2^h - 1, 2^h, \ldots, 2^{N-k} - 2^{N-k-h}\}$ and $\mathcal{V} = \{2^h - 1, 2^h, \ldots, 2^k - 2^{k-h}\}$. The overall cost is $(N - h - \lfloor N/h \rfloor + 2)N^2 C_{ILP} < N^3 C_{ILP}$.

H. Lenstra introduced an algorithm [10] that solves the *decisional ILP problem* with a fixed number of variables in polynomial time in the size of the problem input. This problem consists in deciding whether there exists a solution to the ILP instance satisfying the constraints or not. One can use a *branch and bound*-like approach in combination with Lenstra's algorithm to solve our ILP instances in 3 variables. In particular, one may split $\mathcal{U}$ in two subsets $\mathcal{U}_1$ and $\mathcal{U}_2$, and apply Lenstra's algorithm to decide which one contains the (unique) solution. Assume this is $\mathcal{U}_1$. Then, one applies the same procedure to $\mathcal{U}_1$ and so on. It takes $\log(|\mathcal{U}|)$ steps to isolate and find the solution of the ILP instance. The overall cost C_{ILP} is therefore polynomial in N.

5 Conclusions and Future Work

This work introduces techniques to reduce the MLHCombSP to ILP. In Section 3, we show how to make a trade-off between the success probability of retrieving the private key via an ILP-oracle query and number of variables of the resulting ILP problem. In general, it is not easy to determine the complexity of an ILP instance. Unlike Linear Programming, the dimension of ILP is not determinant in establishing whether an instance is feasible or not to solve [20]. Therefore the size of the ILPs emerging from the reduction in Section 3 is not necessarily related to their hardness. Unfortunately, the vast majority of the ILP solvers available does not support big numbers arithmetic. This prevented us from performing noteworthy experiments ($N > 60$) on this reduction. With a dedicated implementation, it could be possible to perform such experiments and obtain empirical hints about the complexity of these ILP instances.

In Section 4, we introduce a new family of weak keys for which a key-recovery attack runs in polynomial time by solving ILP instances with only 3 variables

and only one expected solution. In particular, this family is significantly larger than the family of weak keys discovered by Beunardeu et al.

Bibliography

[1] D. Aggarwal, A. Joux, A. Prakash, and M. Santha, "A New Public-Key Cryptosystem via Mersenne Numbers," in *Advances in Cryptology – CRYPTO 2018*, vol. 10993 of *LNCS*, pp. 459–482, Springer, 2018.

[2] J. Hoffstein, J. Pipher, and J. H. Silverman, "NTRU: A Ring-based Public Key Cryptosystem," *Algorithmic Number Theory*, 12 1998.

[3] D. Aggarwal, A. Joux, A. Prakash, and M. Santha, "A New Public-Key Cryptosystem via Mersenne Numbers." Cryptology ePrint Archive, Report 2017/481 , version:20170530.072202, 2017.

[4] M. Beunardeau, A. Connolly, R. Géraud, and D. Naccache, "On the Hardness of the Mersenne Low Hamming Ratio Assumption." Cryptology ePrint Archive, Report 2017/522, 2017.

[5] K. de Boer, L. Ducas, S. Jeffery, and R. de Wolf, "Attacks on the AJPS Mersenne-Based Cryptosystem," in *Post-Quantum Cryptography*, vol. 10786 of *LNCS*, pp. 101–120, Springer, 2018.

[6] A. K. Lenstra, H. W. Lenstra, and L. Lovasz, "Factoring Polynomials with Rational Coefficients," *Mathematische Annalen*, vol. 261, no. 4, pp. 515–534, 1982.

[7] C. H. Papadimitriou, "On the Complexity of Integer Programming," in *Journal of ACM*, vol. 28, pp. 765–768, ACM, 1981.

[8] A. Schrijver, *Theory of Linear and Integer Programming*. John Wiley & Sons, 1986.

[9] L. Wolsey, *Integer Programming*. Wiley Series in Discrete Mathematics and Optimization, John Wiley & Sons, 1998.

[10] H. W. Lenstra, *Integer Programming with a Fixed Number of Variables*, vol. 8 of *Mathematics of Operations Research*. Informs, 1983.

[11] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell, *Branch-and-Bound Algorithms*, vol. 19 of *Discrete Optimization*. Elsevier Science Publishers, 2016.

[12] M. L. Fisher, "The Lagrangian Relaxation Method for Solving Integer Programming Problems," in *Management Science*, vol. 27, pp. 1–18, Informs, 1981.

[13] L. Appelgren, "A Column Generation Algorithm for a Ship Scheduling Problem," in *Transportation Science*, vol. 3, pp. 53–68, Informs, 02 1969.

[14] H. Marchand, A. Martin, R. Weismantel, and L. Wolsey, "Cutting Planes in Integer and Mixed Integer Programming," in *Discrete Applied Mathematics*, vol. 123, pp. 397–446, Elsevier Science Publishers, 2002.

[15] I. CPLEX Optimizer, "IBM ILOG CPLEX Optimization Studio," 2018.

[16] L. Gurobi Optimization, "Gurobi Optimizer Reference Manual," 2018.

[17] G. Casella and R. L. Berger, *Statistical Inference*, vol. 2. Duxbury Pacific Grove, 2002.

[18] K. J. Berry and P. W. Mielke Jr, "The Negative Hypergeometric Probability Distribution: Sampling Without Replacement from a Finite Population," in *Perceptual and Motor Skills*, vol. 86, pp. 207–210, SAGE, 1998.
[19] P. S. Wang, "A P-adic Algorithm for Univariate Partial Fractions," in *Proceedings of the Fourth ACM Symposium on Symbolic and Algebraic Computation*, SYMSAC 1981, pp. 212–217, ACM, 1981.
[20] D. Bertsimas and R. Weismantel, *Optimization Over Integers*. Dynamic Ideas, 2005.