Improved Lattice-Based Attack on Mersenne Low Hamming Ratio Search Problem*

Mengce Zheng[1] and Wei Yan[2]

[1] Zhejiang Wanli University, Ningbo, China
mengce.zheng@gmail.com
[2] National University of Defense Technology, Hefei, China
yan.wei2023@nudt.edu.cn

Abstract. This paper investigates the Mersenne number-based AJPS cryptosystem, with a particular focus on its associated hard problem. Specifically, we aim to enhance the existing lattice-based attack on the Mersenne low Hamming ratio search problem. Unlike the previous approach of directly employing lattice reduction algorithm, we apply the lattice-based method to solving polynomial equations derived from the above problem. We extend the search range for vulnerabilities in weak keys and increase the success probability of key recovery attack. To validate the efficacy and accuracy of our proposed improvements, we conduct numerical computer experiments. These experiments serve as a concrete validation of the practicality and effectiveness of our improved attack.

Keywords: Attack · Mersenne number · Weak key · Low Hamming weight · Lattice

1 Introduction

Background. At Crypto 2018, Aggarwal et al. [AJPS18] introduced the AJPS cryptosystem, an innovative variant of the NTRU public-key cryptosystem [HPS98]. In their novel approach, integers characterized by sparse binary representation are employed as secret keys, diverging from the conventional use of polynomials with small coefficients. Notably, the AJPS cryptosystem is conjectured to possess inherent resilience against potential quantum threats.

The fundamental architecture of the AJPS cryptosystem unfolds as follows. Consider a Mersenne number designated as $p = 2^n - 1$, where n is a prime. The algebraic structure denoted as $\mathbb{Z}/p\mathbb{Z}$ can be elegantly mapped onto a set of n-bit strings, with 1^n aligning with 0^n. Leveraging the arithmetic operations conducted modulo p, a profound connection emerges between integers modulo p and binary strings of length n. The key generation process involves the random selection of elements f and g from $\mathbb{Z}/p\mathbb{Z}$, with each element having a predetermined Hamming

*This work was supported by the National Natural Science Foundation of China, grant number 62002335, and Ningbo Young Science and Technology Talent Cultivation Program, grant number 2023QL007.

weight $w \approx \sqrt{n}$. It is necessary that g possess a multiplicative inverse within $\mathbb{Z}/p\mathbb{Z}$. The ensuing public key h is defined as f/g, rendering an n-bit string characterized by an arbitrary Hamming weight. Meanwhile, the private key corresponds to f and g.

The AJPS cryptosystem is divided into a basic bit-by-bit encryption scheme and a key encapsulation mechanism scheme. Expanding upon the former, consider a Mersenne prime $p = 2^n - 1$, and introduce two random integers f and g, both residing within $\mathbb{Z}/p\mathbb{Z}$. Moreover, these integers f and g each possess a Hamming weight of w with a constraint guided by the relationship $n > 4w^2$. The public key pk is expressed as $h = f/g \pmod{p}$, while the private key sk is established as g. The encryption procedure involves the utilization of two random integers a and b with a Hamming weight of w. Encrypting one bit m is accomplished through

$$c = (-1)^m \cdot (a \cdot h + b).$$

Upon decryption, the computation of $d = \mathsf{Ham}(c \cdot g)$ is conducted, leading to the output of '0' if $d \leq 2w^2$, and '1' otherwise. This decryption procedure leverages the property that $c \cdot g$ exhibits distinct Hamming weights based on the value of m. The core relation is

$$c \cdot g = (-1)^m \cdot (a \cdot h \cdot g + b \cdot g) = (-1)^m \cdot (a \cdot f + b \cdot g),$$

thus resulting in a Hamming weight of at most $2w^2$ if $m = 0$.

Transitioning to the key encapsulation mechanism scheme, the instantiation involving error correcting codes is required. In this scheme, n and w should satisfy the constraint $n > 10w^2$. By introducing a random integer r modulo p, the establishment of public and private keys is denoted by $\mathsf{pk} := (r, t) = (r, f \cdot r + g)$ and $\mathsf{sk} := f$. For encrypting a message $m \in \{0,1\}^w$, the first step involves the generation of random integers $a,\ b_1,\ b_2$ modulo p, all featuring a Hamming weight of w. Subsequently, employing the encoding algorithm $\mathcal{E} : \{0,1\}^w \to \{0,1\}^n$ associated with an error correcting code $(\mathcal{E}, \mathcal{D})$, the ciphertext (c_1, c_2) is produced as

$$(c_1, c_2) = (a \cdot r + b_1, (a \cdot t + b_2) \oplus \mathcal{E}(m)).$$

The decryption process is executed through the calculation of

$$\mathcal{D}((f \cdot c_1) \oplus c_2) = \mathcal{D}((f \cdot c_1) \oplus (a \cdot t + b_2) \oplus \mathcal{E}(m)),$$

where $\mathcal{D}$ represents the corresponding decoding algorithm. This decryption leverages the property that $f \cdot c_1$ and $a \cdot t + b_2$ exhibit a low Hamming distance, thereby facilitating the recovery of m with a high probability. The core relation is

$$f \cdot c_1 = f \cdot a \cdot r + f \cdot b_1 = a \cdot (t - g) + f \cdot b_1 = (a \cdot t + b_2) - a \cdot g - b_2 + b_1 \cdot f,$$

indicating the low Hamming weight difference between $f \cdot c_1$ and $a \cdot t + b_2$.

While AJPS related ideas have been employed in cryptographic framework or algorithms such as [NZH19, FX20, BCSV20], a more comprehensive investigation into its security remains imperative. After the initial proposal by Aggarwal et al. [AJPS17], the focus shifted to the vulnerability of the AJPS cryptosystem,

as addressed by Beunardeau et al. [BCGN17]. They introduced a lattice-based attack that could work in time complexity of $O(2^{2w})$. Expanding upon this lattice-based approach, a subsequent study [dBDJdW18] not only delved into the details of lattice-based attack but also proposed an alternative meet-in-the-middle strategy using locality-sensitive hash functions. Their work demonstrated that the lattice-based attack surpasses the efficiency of the meet-in-the-middle one.

The security analysis of the AJPS cryptosystem rests on the foundation of two challenging problems. The first, referred to as the Mersenne low Hamming ratio search problem (MLHRSP), plays a pivotal role in the recovery of an unknown private key from a known public key.

Problem 1 (MLHRSP)**.** *Consider an n-bit Mersenne prime $p = 2^n - 1$ and a positive integer w. Let f and g be two n-bit random strings characterized by a Hamming weight of w. The objective is to extract the values of f and g from the information provided by the equation $h = f/g \pmod p$ with a given h.*

The second challenge, termed the Mersenne low Hamming combination search problem (MLHCSP), is equally significant in the context of recovering an unknown private key from a given public key.

Problem 2 (MLHCSP)**.** *Consider an n-bit Mersenne prime $p = 2^n - 1$, a positive integer w, and a uniformly random n-bit string r. Let f and g be two n-bit random strings with a Hamming weight of w. The objective is to extract the values of f and g given $(r,\ t) = (r,\ f \cdot r + g \pmod p)$.*

Alongside the lattice-based attack and meet-in-the-middle attack mentioned above, other possible attack types have been presented in [BT19, TD20, BCSV23]. We study lattice-based cryptanalysis in this paper and briefly describe two representative attacks as follows.

Beunardeau et al. [BCGN17] handle MLHRSP based on a key insight: when f and g satisfy $f, g < \sqrt{p}$, the equation $h = f/g \pmod p$ can be exploited to deduce f and g. This recovery is facilitated by employing the lattice reduction algorithm in a 2-dimensional lattice. The attack achieves private key recovery from a public key with a probability of 2^{-2w}. To delve deeper, consider the construction of a 2-dimensional lattice Λ generated by

$$\begin{pmatrix} 1 & h \\ 0 & p \end{pmatrix}.$$

The lattice determinant is $\det(\Lambda) = p$, aligning with the Gaussian heuristic, thus it has a vector of norm approximately $\sqrt{p}$. Therefore, the vector (g, f) resides as a short vector within this lattice. When $f < \sqrt{p}$ and $g < \sqrt{p}$ hold simultaneously, recovery of f and g ensues with an approximate $(1/2)^{2w}$ probability, driven by their Hamming weight of w.

Furthermore, a refined attack can apply to the bit-by-bit encryption using the equation $c = (-1)^m \cdot (a \cdot h + b)$. When both a and b satisfy $a < \sqrt{p}$ and $b < \sqrt{p}$, the use of lattice reduction algorithm in a 3-dimensional lattice leads to recovery of a, b, and the plaintext bit m. Expanding this attack to MLHCSP with the equation

$t = f \cdot r + g \pmod{p}$ is similar and the attack's success probability remains the same.

It is essential to recognize that Beunardeau et al.'s attack leads to recovery of weak keys, extracting a private key from known public key with a 2^{-2w} probability. This approach is further developed through random partition technique using higher-dimensional lattices. Thus, the private key can be recovered from any public key with a time complexity of $O(2^{2w})$.

Coron-Gini's Attack [CG20] is a modified version of Beunardeau et al.'s attack targeting the key encapsulation mechanism. In contrast to extracting the private key, this attack breaks the indistinguishability of ciphertexts. To be specific, given a public key (r, t) and a ciphertext (c_1, c_2), this attack effectively differentiates between $m = 0$ and $m \neq 0$. When $m = 0$, one has $\mathcal{E}(m) = 0$, which yields $c_1 = a \cdot r + b_1$ and $c_2 = a \cdot t + b_2$. Provided a, b_1, and b_2 are all less than $p^{2/3}$, recovery of a, b_1, and b_2 through lattice reduction algorithm is feasible. Consequently, the attack's success probability is $(2/3)^{3w} \approx 2^{-1.75w}$, outperforming the original success probability. By applying a similar random partition technique, the attack complexity to compromise the indistinguishability of any ciphertext can be reduced to $O(2^{1.75w})$.

Our Contribution. We concentrate on an enhanced examination of lattice-based cryptanalysis related to MLHRSP, a challenging hard problem in the realm of AJPS. The emphasis lies in refining existing attack strategy and addressing unbalanced scenarios that arise when $f < \sqrt{p} < g$ or $g < \sqrt{p} < f$, instead of solely focusing on the balanced case where both f and g are below $\sqrt{p}$. Through this, we aim to augment the effectiveness of current attacks. An additional insight relates to the utilization of lattice reduction algorithm, i.e., the LLL algorithm under Gaussian heuristic in previous lattice-based attacks. To be specific, we recognize the unexplored advantage of the LLL algorithm in solving modular polynomial equations associated with MLHRSP.

We start by revisiting the key equation of MLHRSP, that is $h = f/g \pmod{p}$. This equation can be transformed into a bivariate modular polynomial equation as $x_1 - hx_2 \equiv 0 \pmod{p}$, where the desired root $(x_1^\star, x_2^\star)$ corresponds to (f, g). This allows us to apply lattice-based solving strategy without confining our attack to the previous f and g constraints. Consequently, the unbalanced scenarios like $f < \sqrt{p} < g$ or $g < \sqrt{p} < f$ become tractable, expanding the range of exploitable weak keys.

Moreover, our proposed attack increases the success probability from 2^{-2w} to $\sqrt{\pi} w^{3/2} 2^{-2w-1}$, improving Beunardeau et al.'s attack by a factor of $\sqrt{\pi} w^{3/2}/2$. To validate the correctness and efficiency of our attack, we provide a numerical attack instance that succeeds under our proposed strategy while failing under the previous one.

Organization. This paper is structured as follows. In Section 2, we provide essential preliminaries, including the lattice-based method for solving modular polynomial equations. Section 3 presents our improved attack along with detailed success probability analysis. The experimental results for validating our proposed

attack are presented in Section 4. Finally, we draw our conclusions in Section 5.

2 Preliminaries

We present the fundamental concepts required for our attack. These include the lattice reduction algorithm, i.e., the LLL algorithm proposed by Lenstra, Lenstra, and Lovász [LLL82], and Coppersmith's lattice-based method [Cop96, Cop97], which was later refined as Howgrave-Graham's lemma [How97]. Additionally, a solving condition essential for finding the root of polynomial equations is introduced. For a more comprehensive understanding, interested readers can refer to [May03, May10].

Let us begin by defining lattice Λ as the set of all integer linear combinations of linearly independent vectors $\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_\omega \in \mathbb{R}^n$. In other words, a lattice can be expressed as

$$\Lambda = \left\{ \sum_{i=1}^{\omega} z_i \vec{b}_i : z_i \in \mathbb{Z} \right\}.$$

The lattice determinant denoted as $\det(\Lambda)$ is calculated as $\sqrt{\det(BB^{\mathsf{T}})}$, where each $\vec{b}_i$ is considered as a row vector of the basis matrix B. When dealing with a full-rank lattice with $\omega = n$, the lattice determinant becomes $\det(\Lambda) = |\det(B)|$.

The LLL algorithm [LLL82] is a core mathematical tool for efficiently finding approximately short lattice vectors. As proven by [May03], the LLL algorithm yields a reduced basis $(\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_\omega)$ with the following property, where $\|\vec{v}_i\|$ denotes the Euclidean norm of vector $\vec{v}_i$.

Lemma 1. *The LLL algorithm outputs a reduced basis $(\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_\omega)$ of a given ω-dimensional lattice Λ satisfying*

$$\|\vec{v}_i\| \leq 2^{\frac{\omega(\omega-1)}{4(\omega+1-i)}} \det(\Lambda)^{\frac{1}{\omega+1-i}}, \quad \text{for } i = 1, 2, \ldots, \omega.$$

Its time complexity is polynomial in ω and in logarithmic maximal input vector component.

An important lemma introduced by Howgrave-Graham [How97] provides a principle for determining whether the root of a modular polynomial equation also corresponds to a root over the integers. This lemma concerns an integer polynomial $g(x_1, \ldots, x_n) := \sum c_{i_1,\ldots,i_n} x_1^{i_1} \cdots x_n^{i_n}$, and its norm $\|g(x_1, \ldots, x_n)\| := \sqrt{\sum |c_{i_1,\ldots,i_n}|^2}$.

Lemma 2. *Let $g(x_1, \ldots, x_n) \in \mathbb{Z}[x_1, \ldots, x_n]$ be an integer polynomial, consisting of at most ω monomials. Let $R, X_1, \ldots, X_n$ be given positive integers. If the two following conditions are satisfied:*

(1) $g(x_1^\star, \ldots, x_n^\star) \equiv 0 \pmod{R}$, *for* $|x_1^\star| \leq X_1, \ldots, |x_n^\star| \leq X_n$,

(2) $\|g(X_1 x_1, \ldots, X_n x_n)\| < R/\sqrt{\omega}$.

Then $g(x_1^\star, \ldots, x_n^\star) = 0$ holds over the integers.

Combining the LLL algorithm's outputs with Howgrave-Graham's lemma, we can efficiently solve modular/integer polynomial equations. Suppose that we have calculated the first ℓ many reduced vectors $(\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_\ell)$, due to

$$\|\vec{v}_i\| \leq 2^{\frac{\omega(\omega-1)}{4(\omega+1-i)}} \det(\Lambda)^{\frac{1}{\omega+1-i}} \leq 2^{\frac{\omega(\omega-1)}{4(\omega+1-\ell)}} \det(\Lambda)^{\frac{1}{\omega+1-\ell}}, \text{ for } i = 1, 2, \ldots, \ell.$$

The key to success lies in satisfying the condition

$$2^{\frac{\omega(\omega-1)}{4(\omega+1-\ell)}} \det(\Lambda)^{\frac{1}{\omega+1-\ell}} < \frac{R}{\sqrt{\omega}}.$$

It reduces to

$$\det(\Lambda) < R^{\omega+1-\ell} 2^{-\frac{\omega(\omega-1)}{4}} \omega^{-\frac{\omega+1-\ell}{2}}.$$

We always have $\ell < \omega \ll R$ and hence it further leads to $\det(\Lambda) < R^{\omega-\epsilon}$ with a tiny error term ϵ. We finally derive the following asymptotic solving condition as

$$\det(\Lambda) < R^{\omega}, \tag{1}$$

which allows us to effectively solve given modular/integer polynomial equations.

The lattice-based solving strategy consists of the following stages. Initially, we generate a set of shift polynomials using the provided polynomial $f(x_1, \ldots, x_n)$ and estimated upper bounds $X_1, \ldots, X_n$. These shift polynomials are specifically designed to share a common root modulo R. Subsequently, we generate a lattice by converting the coefficient vectors of each shift polynomial $g_i(X_1 x_1, \ldots, X_n x_n)$ into row vectors of a lattice basis matrix. Utilizing the LLL algorithm, we then obtain the first few reduced vectors. These vectors are further transformed into integer polynomials $h_i(x_1, \ldots, x_n)$. Once we ensure that the resulting integer polynomials $h_i(x_1, \ldots, x_n)$ are algebraically independent, the equation system can be effectively solved using trivial methods, thus extracting the desired root $(x_1^\star, \ldots, x_n^\star)$.

The generation of lattice stands as a pivotal stage and several studies like [BM05, JM06, HM08, TK13, LZPL15] have focused on constructing an elegant lattice basis matrix with optimized solving conditions. Additionally, the extraction of the common root can be accomplished using resultant computation or Gröbner basis computation [BWK93]. The running time primarily depends on computing the reduced lattice basis and recovering the desired root, both of which can be efficiently achieved in polynomial time with respect to the inputs.

We note that the lattice-based solving strategy is a heuristic approach, as there is no assurance that the derived integer polynomials will always be algebraically independent. However, in the realm of lattice-based attacks, it is commonly assumed that the polynomials obtained through the LLL algorithm possess algebraic independence. While some limited research may contradict this assumption, it is widely accepted and, for the sake of efficiency, we adopt the following assumption throughout this paper. We assume that the obtained integer polynomials are algebraically independent, facilitating the efficient recovery of their common root.

3 Improved Lattice-Based Attack

We present the formulation of MLHRSP in modular polynomial equation form. Given $h = f/g \pmod{p}$ with a known h, we derive a bivariate polynomial

$f(x_1, x_2) := x_1 - hx_2$, yielding the modular equation

$$f(x_1, x_2) \equiv 0 \pmod{p}, \tag{2}$$

with the root $(x_1^\star, x_2^\star) = (f, g)$. To solve this bivariate homogeneous linear equation, we employ two distinct lattice-based solving strategies described in [HM08, Theorem 3] and [LZPL15, Theorem 7], respectively.

Basic Strategy. We present a polynomial-time attack employing the lattice-based solving strategy used in [HM08, Theorem 3]. We denote the upper bounds of the desired root (x_1, x_2) as $X_1 = p^{\xi_1}$ and $X_2 = p^{\xi_2}$ respectively. Defining shift polynomials for a predetermined positive integer s, and non-negative integers i_1 and i_2,

$$g_{[i_1,i_2]}(x_1, x_2) := x_2^{i_2} f^{i_1}(x_1, x_2) p^{s-i_1}, \quad 0 \le i_1 + i_2 \le s.$$

Therefore, parameter R indicated in the lattice-based solving strategy is equal to p^s.

Our attack involves transforming the coefficient vectors of $g_{[i_1,i_2]}(X_1 x_1, X_2 x_2)$ into row vectors of a lattice basis matrix B. Before that, we establish the monomial order and polynomial order. The former order $\prec$ corresponds to $x_1^{i_1} x_2^{i_2} \prec x_1^{i_1'} x_2^{i_2'}$ if

$$i_1 + i_2 < i_1' + i_2' \quad \text{or} \quad i_1 + i_2 = i_1' + i_2',\ i_1 < i_1'.$$

The latter order $\prec$ corresponds to $g_{[i_1,i_2]} \prec g_{[i_1',i_2']}$ if

$$i_1 + i_2 < i_1' + i_2' \quad \text{or} \quad i_1 + i_2 = i_1' + i_2',\ i_1 < i_1'.$$

Moreover, the leading monomial of $g_{[i_1,i_2]}(x_1, x_2)$ is $x_1^{i_1} x_2^{i_2} p^{s-i_1}$. Representing derived coefficient vectors from $g_{[i_1,i_2]}(X_1 x_1, X_2 x_2)$ as $\vec{b}_i$ for $i = 1, \ldots, \omega$, we generate a lattice

$$\Lambda = \left\{ \sum_{i=1}^{\omega} z_i \vec{b}_i : z_i \in \mathbb{Z} \right\}.$$

The lattice dimension ω can be calculated as

$$\omega = \sum_{i_1=0}^{s} \sum_{i_2=0}^{s-i_1} 1 = \frac{(s+1)(s+2)}{2}.$$

We provide an illustrative example of the lattice basis matrix B when setting $s = 2$. The shift polynomials are listed as follows.

$$\begin{aligned}
g_{[0,0]}(x_1, x_2) &= x_2^0 f^0(x_1, x_2) p^2 = p^2, \\
g_{[0,1]}(x_1, x_2) &= x_2^1 f^0(x_1, x_2) p^2 = p^2 x_2, \\
g_{[1,0]}(x_1, x_2) &= x_2^0 f^1(x_1, x_2) p^1 = p x_1 - hp x_2, \\
g_{[0,2]}(x_1, x_2) &= x_2^2 f^0(x_1, x_2) p^2 = p^2 x_2^2, \\
g_{[1,1]}(x_1, x_2) &= x_2^1 f^1(x_1, x_2) p^1 = p x_1 x_2 - hp x_2^2, \\
g_{[2,0]}(x_1, x_2) &= x_2^0 f^2(x_1, x_2) p^0 = x_1^2 - 2h x_1 x_2 + h^2 x_2^2.
\end{aligned}$$

Thus, substituting x_i with X_i, we construct the following $\omega \times \omega$ lattice basis matrix B that is

$$\left(\begin{array}{c|cccccc} & 1 & x_2 & x_1 & x_2^2 & x_1x_2 & x_1^2 \\ \hline g_{[0,0]} & p^2 & 0 & 0 & 0 & 0 & 0 \\ g_{[0,1]} & 0 & p^2X_2 & 0 & 0 & 0 & 0 \\ g_{[1,0]} & 0 & -hpX_2 & pX_1 & 0 & 0 & 0 \\ g_{[0,2]} & 0 & 0 & 0 & p^2X_2^2 & 0 & 0 \\ g_{[1,1]} & 0 & 0 & 0 & -hpX_2^2 & pX_1X_2 & 0 \\ g_{[2,0]} & 0 & 0 & 0 & h^2X_2^2 & -2hX_1X_2 & X_1^2 \end{array}\right).$$

The corresponding matrix diagonal elements are $X_1^{i_1}X_2^{i_2}p^{s-i_1}$ for $0 \leq i_1 + i_2 \leq s$.

Following the lattice-based solving strategy, we calculate the lattice determinant $\det(\Lambda) = p^{s_p}X_1^{s_1}X_2^{s_2}$, where the respective exponents s_p, s_1 and s_2 are computed as

$$\begin{aligned} s_p &= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1}(s-i_1) = \frac{s(s+1)(s+2)}{3}, \\ s_1 &= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1} i_1 = \frac{s(s+1)(s+2)}{6}, \\ s_2 &= \sum_{i_1=0}^{s}\sum_{i_2=0}^{s-i_1} i_2 = \frac{s(s+1)(s+2)}{6}. \end{aligned}$$

This relates to the derived solving condition (1), i.e., $\det(\Lambda) < R^{\omega}$ with $R = p^s$, which yields

$$p^{\frac{s(s+1)(s+2)}{3}}(X_1X_2)^{\frac{s(s+1)(s+2)}{6}} < p^{s\cdot\frac{(s+1)(s+2)}{2}}.$$

With x_1 and x_2 bounded by $X_1 = p^{\xi_1}$ and $X_2 = p^{\xi_2}$ respectively, we simplify the exponents over p to obtain

$$\frac{1}{3} + \frac{1}{6}\cdot(\xi_1 + \xi_2) < \frac{1}{2},$$

which further reduces to

$$\xi_1 + \xi_2 < 1. \tag{3}$$

Improved Strategy. We show another polynomial-time attack employing the lattice-based solving strategy mentioned in [LZPL15, Theorem 7]. The upper bounds of the desired root (x_1, x_2) are denoted by $X_1 = p^{\xi_1}$ and $X_2 = p^{\xi_2}$ respectively. Defining shift polynomials for a predetermined positive integer s, and a non-negative integer i,

$$g_i(x_1, x_2) := x_2^{s-i}f^i(x_1, x_2)p^{s-i}, \quad 0 \leq i \leq s.$$

Thus, parameter R indicated in the lattice-based solving strategy is equal to p^s.

Our attack involves transforming the coefficient vectors of $g_i(X_1x_1, X_2x_2)$ into row vectors of a lattice basis matrix B. We define the same monomial order and polynomial order as in the previous basic strategy. Moreover, the leading monomial of $g_i(x_1, x_2)$ now is $x_1^i x_2^{s-i} p^{s-i}$. Representing derived coefficient vectors from $g_i(X_1x_1, X_2x_2)$ as $\vec{b}_i$ for $i = 1, \ldots, \omega$, we generate a lattice

$$\Lambda = \left\{ \sum_{i=1}^{\omega} z_i \vec{b}_i : z_i \in \mathbb{Z} \right\}.$$

The lattice dimension ω can be calculated as

$$\omega = \sum_{i=0}^{s} 1 = s + 1.$$

Note that we shall construct a lower-dimensional lattice compared to the previous one, which is a significant advantage. We provide an illustrative example of the lattice basis matrix B when setting $s = 2$ as before. The shift polynomials are listed as follows.

$$\begin{aligned}
g_0(x_1, x_2) &= x_2^2 f^0(x_1, x_2) p^2 = p^2 x_2^2, \\
g_1(x_1, x_2) &= x_2^1 f^1(x_1, x_2) p^1 = p x_1 x_2 - hp x_2^2, \\
g_2(x_1, x_2) &= x_2^0 f^2(x_1, x_2) p^0 = x_1^2 - 2h x_1 x_2 + h^2 x_2^2.
\end{aligned}$$

Thus, substituting x_i with X_i, we construct the following $\omega \times \omega$ lattice basis matrix B that is

$$\left(\begin{array}{c|ccc}
 & x_2^2 & x_1x_2 & x_1^2 \\ \hline
g_0 & p^2X_2^2 & 0 & 0 \\
g_1 & -hpX_2^2 & pX_1X_2 & 0 \\
g_2 & h^2X_2^2 & -2hX_1X_2 & X_1^2
\end{array}\right).$$

The corresponding matrix diagonal elements are $X_1^i X_2^{s-i} p^{s-i}$ for $0 \le i \le s$.

Following the lattice-based solving strategy, we calculate the lattice determinant $\det(\Lambda) = p^{s_p} X_1^{s_1} X_2^{s_2}$, where the respective exponents s_p, s_1 and s_2 are computed as

$$\begin{aligned}
s_p = s_2 &= \sum_{i=0}^{s} (s - i) = \frac{s(s+1)}{2}, \\
s_1 &= \sum_{i=0}^{s} i = \frac{s(s+1)}{2}.
\end{aligned}$$

This relates to the derived solving condition (1), i.e., $\det(\Lambda) < R^{\omega}$ with $R = p^s$, which yields

$$(pX_1X_2)^{\frac{s(s+1)}{2}} < p^{s\cdot(s+1)}.$$

With x_1 and x_2 bounded by $X_1 = p^{\xi_1}$ and $X_2 = p^{\xi_2}$ respectively, we simplify the exponents over p and have

$$\frac{1}{2} \cdot (1 + \xi_1 + \xi_2) < 1,$$

which further leads to

$$\xi_1 + \xi_2 < 1.$$

Interestingly, we are able to obtain the same condition with a lower lattice dimension and smaller lattice determinant (if the parameter s is set the same).

We discuss how to effectively find the desired root in our attack. Suppose we have several integer polynomials $h_i(x_1, x_2)$ derived from the proposed lattice-based strategy, where $h_i(x_1^\star, x_2^\star) = 0$ is satisfied. Their common root $(x_1^\star, x_2^\star)$ can be recovered using resultant computation. Otherwise, we compute the greatest common divisor $h(x_1, x_2)$ of $h_i(x_1, x_2)$ and turn to solve it. As $f(x_1, x_2)$ is homogeneous, we assume the homogeneity of $h(x_1, x_2)$ and introduce a new variable τ defined as $\tau := x_1/x_2$. With this, we define $\bar{h}(\tau) := h(x_1, x_2)/x_2^\delta$, where δ is a known constant, and ensure that $\bar{h}(x_1^\star/x_2^\star) = 0$. We can determine $x_1^\star/x_2^\star$ by employing trivial methods to extract the rational roots of $\bar{h}(\tau)$. Suppose $x_1^\star$ and $x_2^\star$ are coprime, we can finally deduce their values from the numerator and denominator of the derived root τ. In the validating experiments, we use the Gröbner basis computation to derive the solution $(x_1^\star, x_2^\star)$ more efficiently.

Regarding time complexity, it primarily relies on the polynomial-time LLL algorithm, which is polynomial in both s and $\log(p^s)$. Given that s is a fixed integer, the attack's time complexity is a polynomial of $\log p = n$. We present Proposition 1 as a conclusion of our improved attack.

Proposition 1. *Let $p = 2^n - 1$ be an n-bit Mersenne prime and w be a positive integer. Let f and g bounded by $f \leq p^{\xi_1}$ and $g \leq p^{\xi_2}$, denote two unknown n-bit random strings characterized by a Hamming weight of w. Given h with the equation $h = f/g \pmod p$, then f and g can be efficiently recovered in time polynomial in n if $\xi_1 + \xi_2 < 1$.*

It is worth noting that the condition $\xi_1 + \xi_2 < 1$ is identical to $X_1 X_2 < p$ and also $f \cdot g < p$. This condition covers the previous attack result where both f and g are less than $\sqrt{p}$. Furthermore, our advancement serves to extend the attack constraint for f and g, thereby significantly broadening the potential range of applicability.

Success Probability Analysis. We proceed to conduct a theoretical analysis of the success probability associated with our proposed attack. To simplify the subsequent examination, we will base our calculations on the representations of f and g using bit strings. When considering the scenario where w is approximately $\sqrt{n}$, let Pr_1 denote the previous success probability of Beunardeau et al.'s attack. Given that f and g are both less than $\sqrt{p}$, namely their w many '1' bits are chosen from low $\lfloor n/2 \rfloor$ bits, the expression for Pr_1 can be formulated as follows.

$$\mathsf{Pr}_1 = \frac{\binom{\lfloor n/2 \rfloor}{w}\binom{\lfloor n/2 \rfloor}{w}}{\binom{n}{w}\binom{n}{w}} = \left(\frac{\lfloor n/2 \rfloor!(n-w)!}{n!(\lfloor n/2 \rfloor - w)!}\right)^2 \approx 2^{-2w}.$$

Furthermore, we introduce Pr_2 to represent the success probability associated with our improved attack. Our aim is to compute the value of Pr_2, which is expressed as

$$\mathsf{Pr}_2 = \sum_{t=w}^{n-w} \frac{\binom{t}{w}\binom{n-t}{w}}{\binom{n}{w}\binom{n}{w}},$$

and concurrently determine the improvement ratio denoted by $\mathsf{r} := \mathsf{Pr}_2/\mathsf{Pr}_1$.

From combinatorial mathematics [GKP94, Page 169], it can be seen that the following combinatorial identity holds.

$$\sum_{t=m}^{n-r} \binom{t}{m}\binom{n-t}{r} = \binom{n+1}{m+r+1}.$$

Therefore, we obtain

$$\mathsf{Pr}_2 = \frac{\binom{n+1}{2w+1}}{\binom{n}{w}\binom{n}{w}}.$$

Due to Stirling's approximation and $w \approx \sqrt{n}$ when n tends to infinity, the improvement rate r can be further calculated as follows.

$$\begin{aligned}
\mathsf{r} &= \frac{\binom{n+1}{2w+1}}{\binom{A}{w}\binom{A}{w}} \\
&= \frac{(n+1)!(w!(A-w)!)^2}{(2w+1)!(n-2w)!(A!)^2} \\
&\approx \frac{\sqrt{2\pi(n+1)}(\frac{n+1}{\mathrm{e}})^{n+1} \cdot 2\pi w(\frac{w}{\mathrm{e}})^{2w} \cdot 2\pi(A-w)(\frac{A-w}{\mathrm{e}})^{2A-2w}}{\sqrt{2\pi(2w+1)}(\frac{2w+1}{\mathrm{e}})^{2w+1} \cdot \sqrt{2\pi(n-2w)}(\frac{n-2w}{\mathrm{e}})^{n-2w} \cdot 2\pi A(\frac{A}{\mathrm{e}})^{2A}} \\
&= \frac{\sqrt{2\pi}(n+1)^{n+\frac{3}{2}} \cdot w^{2w+1} \cdot (A-w)^{2A-2w+1}}{(2w+1)^{2w+\frac{3}{2}} \cdot (n-2w)^{n-2w+\frac{1}{2}} \cdot A^{2A+1}} \\
&\approx \frac{\sqrt{2\pi}(w^2+1)^{w^2+\frac{3}{2}} \cdot w^{2w+1} \cdot (\frac{w^2}{2}-w)^{w^2-2w+1}}{(2w+1)^{2w+\frac{3}{2}} \cdot (w^2-2w)^{w^2-2w+\frac{1}{2}} \cdot (\frac{w^2}{2})^{w^2+1}} \\
&= \frac{\sqrt{2\pi}(\frac{1}{2})^{w^2-2w+1}}{(\frac{1}{2})^{w^2+1} \cdot 2^{2w+\frac{3}{2}}} \cdot \frac{(w^2+1)^{w^2+\frac{3}{2}}}{(w^2)^{w^2+1}} \cdot \frac{(w^2-2w)^{w^2-2w+1}}{(w^2-2w)^{w^2-2w+\frac{1}{2}}} \cdot \frac{w^{2w+1}}{(w+\frac{1}{2})^{2w+\frac{3}{2}}} \\
&= \frac{\sqrt{\pi}}{2} \cdot \frac{(w^2+1)^{w^2+1+\frac{1}{2}}}{(w^2)^{w^2+1}} \cdot (w^2-2w)^{\frac{1}{2}} \cdot \frac{w^{2w+1}}{(w+\frac{1}{2})^{2w+1+\frac{1}{2}}} \\
&= \frac{\sqrt{\pi}}{2} \cdot \frac{(w^2+1)^{\frac{1}{2}}(w^2-2w)^{\frac{1}{2}}}{(w+\frac{1}{2})^{\frac{1}{2}}} \cdot \left(1+\frac{1}{w^2}\right)^{w^2+1} \cdot \left(1-\frac{1}{2w+1}\right)^{2w+1} \\
&\approx \frac{\sqrt{\pi}}{2} \cdot w^{\frac{3}{2}} \cdot \mathrm{e} \cdot \mathrm{e}^{-1} \\
&= \frac{\sqrt{\pi}}{2} w^{\frac{3}{2}},
\end{aligned}$$

where $A := \lfloor \frac{n}{2} \rfloor$ for simplicity. Therefore, we obtain the success probability Pr_2 of our improved attack that is approximately equal to $\sqrt{\pi}w^{3/2}2^{-2w-1}$.

4 Validating Experiments

To validate the validity and effectiveness of our improved attack on MLHRSP, which exploits Proposition 1 by the basic and improved strategies, we conducted a series of numerical experiments. These experiments were performed on a computer running a 64-bit Windows 10 operating system with Ubuntu 22.04 installed on WSL 2. The system had a CPU operating at 2.80 GHz and 16 GB of RAM. The experiments were conducted using `SageMath` [The23] with `Python`, and the parameters for generating the experimental instances were randomly chosen.

We generated the MLHRSP instances with suggested parameters $p = 2^n - 1$, n, and w in each experiment. Based on two randomized integers f and g satisfying $f \cdot g < p$, we then derived the corresponding public key h using its key equation $h = f/g \pmod{p}$. Furthermore, we gradually increased f and g to achieve larger ξ_1 and ξ_2 for performing a successful key recovery attack. We provide an open source implementation of the proposed attacks and the source code is available at https://github.com/MengceZheng/MLHRSP. Using this implementation to execute the key recovery attacks, we selected a suitable parameter s to construct a lattice. Moreover, we ran 5 trials and ensured a 100% attack success rate for each of the different experimental parameter settings.

The experimental results are presented in Table 1. The n and w columns indicate the specific parameters of the MLHRSP instances. The ξ_1 and ξ_2 columns present the experimental results on bounds of randomly generated f and g. The lattice settings are controlled by s, and the lattice dimension is provided in the ω column. The average time consumption of the proposed key recovery attack is recorded in the Time column and measured in seconds.

During each experiment, we collected sufficient polynomials that satisfied the solvable requirements after running the LLL algorithm. As indicated in Table 1, the running time increases as the lattice dimension ω or the modulus p becomes larger. The reason is that it is mainly influenced by the lattice dimension and the lattice basis matrix entries. Moreover, we observe that the more unbalanced the private keys are, the more time consuming the attack is. In more detail, the time consumption of lattice reduction and root extraction is roughly a few seconds.

We obtained several integer polynomials by transforming the derived vectors into polynomials and then calculated their greatest common divisor $h(x_1, x_2)$. The integer polynomial $h(x_1, x_2)$ was always of a particular homogeneous form $a_1x_1 - a_2x_2$. Therefore, we obtained the desired root $(x_1^\star, x_2^\star) = (a_2, a_1)$ assuming f and g were coprime. Furthermore, we used a more efficient mathematical tool, namely the Gröbner basis computation to directly extract the solution $(x_1^\star, x_2^\star)$. Then we recovered f and g, which allows us to break the AJPS cryptosystem. The experimental results reached the theoretical bounds by constructing lattices of low dimension, where the lowest dimension can be down to 3. Additionally, we provide the following toy examples to aid in numerical understanding.

Example 1. We provide a numerical example to illustrate key recovery attack utilizing Proposition 1 on the AJPS cryptosystem with the basic strategy. In this example, we consider a toy scenario where we have set $n = 521$ and hence $p = 2^{521} - 1$, and we are working with $w = 10$. We assume that two unbalanced

Table 1: Experimental results of the key recovery attacks on MLHRSP

n	w	ξ_1	ξ_2	basic strategy			improved strategy		
				s	ω	Time[†]	s	ω	Time[†]
521	10	0.5	0.5	7	36	0.506 s	7	8	0.116 s
		0.4	0.6	7	36	0.550 s	7	8	0.130 s
		0.3	0.7	7	36	0.646 s	7	8	0.135 s
2203	20	0.5	0.5	5	21	3.906 s	5	6	3.675 s
		0.2	0.8	5	21	4.043 s	5	6	3.757 s
		0.1	0.9	5	21	4.255 s	5	6	3.869 s
3217	25	0.5	0.5	3	10	20.237 s	3	4	17.095 s
		0.65	0.35	3	10	21.489 s	3	4	18.125 s
		0.75	0.25	3	10	22.826 s	3	4	19.024 s
4253	30	0.5	0.5	3	10	52.097 s	2	3	48.097 s
		0.35	0.65	3	10	53.312 s	2	3	48.648 s
		0.25	0.75	3	10	54.229 s	2	3	49.052 s
9689	45	0.5	0.5	3	10	1558.614 s	3	4	1528.035 s
		0.35	0.65	3	10	1606.880 s	3	4	1572.536 s
		0.15	0.85	3	10	1638.942 s	3	4	1598.409 s
11213	50	0.5	0.5	3	10	3013.204 s	3	4	2886.811 s
		0.4	0.6	3	10	3047.022 s	3	4	2906.314 s
		0.3	0.7	3	10	3071.708 s	3	4	2927.524 s
		0.2	0.8	3	10	3106.117 s	3	4	2954.749 s
		0.1	0.9	3	10	3127.506 s	3	4	2976.037 s
19937[‡]	70	0.5	0.5	2	6	30 015.984 s	2	3	28 950.599 s
		0.4	0.6	2	6	30 052.953 s	2	3	28 965.218 s
		0.3	0.7	2	6	30 112.863 s	2	3	28 995.330 s
		0.2	0.8	2	6	30 164.167 s	2	3	29 140.652 s
		0.1	0.9	2	6	30 202.318 s	2	3	29 232.141 s
23209[‡]	75	0.5	0.5	2	6	58 070.592 s	2	3	56 919.641 s
		0.6	0.4	2	6	58 136.852 s	2	3	57 138.886 s
		0.7	0.3	2	6	58 206.983 s	2	3	57 237.149 s
		0.8	0.2	2	6	58 285.447 s	2	3	57 334.223 s
		0.9	0.1	2	6	58 359.950 s	2	3	57 449.701 s

[†] This recorded the time consumption including lattice creation, lattice reduction, integer equation recovery, and root extraction.

[‡] One trial was performed in each experiment setting using faster implementation for efficiency and the corresponding running time was estimated.

secret parameters f and g are less than 2^{52} and 2^{469}, respectively. The specific values for this example instance are as follows.

$$\begin{aligned}
p &= 6864797660130609714981900799081393217269435300143305409394\backslash\\
&\quad 34591855431833976560521225596406614545549772963113914808580\backslash\\
&\quad 3712198799971664381257402829111505715 1,\\
h &= 1543439057814335566199090676920692033224754421507304942585\backslash\\
&\quad 69202850224261065191872563678854092758603703688697463484625\backslash\\
&\quad 64501247468814335627928089217185572713 07.
\end{aligned}$$

To conduct our basic key recovery attack, we set $s = 3$ to construct a 10-dimensional lattice. After less than one second, we successfully extract the desired root $(x_1^\star, x_2^\star)$. The obtained root values are as follows.

$$\begin{aligned}
x_1^\star &= 2323306724327516,\\
x_2^\star &= 3810786357988350189060986105119376014388521856126817435524 58\backslash\\
&\quad 94383907241277495073869577899508005333776992979958056441975 9\backslash\\
&\quad 766509943792567582721.
\end{aligned}$$

Thus, f and g are recovered as follows.

$$\begin{aligned}
f &= 2323306724327516,\\
g &= 3810786357988350189060986105119376014388521856126817435524 58\backslash\\
&\quad 94383907241277495073869577899508005333776992979958056441975 9\backslash\\
&\quad 766509943792567582721.
\end{aligned}$$

It can be easily verified that f, g, h and p do satisfy the key generation of the AJPS cryptosystem, confirming the success of applying Proposition 1 to the Mersenne low Hamming ratio search problem. Moreover, we confirm that the previous attack using a 2-dimensional lattice is invalid.

Example 2. We provide another numerical example to illustrate key recovery attack utilizing Proposition 1 on the AJPS cryptosystem with the improved strategy. In this example, we also consider a toy scenario where we have set $n = 521$ and hence $p = 2^{521} - 1$, and we are working with $w = 10$. We assume that two unbalanced secret parameters f and g are less than 2^{390} and 2^{131}, respectively. The specific values for this example instance are as follows.

$$\begin{aligned}
p &= 6864797660130609714981900799081393217269435300143305409394\backslash\\
&\quad 34591855431833976560521225596406614545549772963113914808580\backslash\\
&\quad 3712198799971664381257402829111505715 1,\\
h &= 1572150789080668564831092970656228267003440076918436663480 46\backslash\\
&\quad 62254863868955497221377995510173655193880368160325759015546 7\backslash\\
&\quad 99820960569234015039700407499048529 59.
\end{aligned}$$

To conduct our improved key recovery attack, we set $s = 2$ to construct a 3-dimensional lattice. After less than one second, we successfully extract the desired root $(x_1^\star, x_2^\star)$. The obtained root values are as follows.

$$\begin{aligned} x_1^\star &= 1300266205068726834352664408895617566470233273297795364782547236924976239227630672771919293698599597927345039798697984, \\ x_2^\star &= 6805647541242865778020847533805397294081. \end{aligned}$$

Thus, f and g are recovered as follows.

$$\begin{aligned} f &= 1300266205068726834352664408895617566470233273297795364782547236924976239227630672771919293698599597927345039798697984, \\ g &= 6805647541242865778020847533805397294081. \end{aligned}$$

It can be verified that f, g, h and p do satisfy the key generation of the AJPS cryptosystem, confirming the success of applying Proposition 1 to the Mersenne low Hamming ratio search problem. Furthermore, in contrast to the failure of the previous attack on unbalanced f and g using a 2-dimensional lattice, we were able to successfully recover them using a 3-dimensional lattice.

5 Concluding Remarks

We revisit the Mersenne number-based AJPS cryptosystem, delving deep into the associated hard problems it presents. Our goal centers on enhancing the existing lattice-based attack targeting the Mersenne low Hamming ratio search problem. Our improved attack adopts a specific lattice-based solving strategy, tailored for solving bivariate polynomial equations. This results in two notable enhancements to our key recovery attack. Firstly, we expand the attack range of susceptible scenarios, amplifying our capacity to uncover vulnerabilities in weak keys. Secondly, we increase the attack's success probability when considering unbalanced attack cases. Furthermore, we conduct a series of numerical experiments to validate the practicality and effectiveness of our improved attack.

The major limitation of our improved lattice-based attack on MLHRSP is that it cannot be applied when facing the enhanced key generation algorithm. To be precise, our proposed attack is unavailable when one discards and resamples f and g again if both of them fall within our attack range. However, the previous attack [BCGN17] using the random partition technique is still effective. Hence, future research should be undertaken to explore how to incorporate a similar random partition technique into our improved lattice-based attack.

References

[AJPS17] Divesh Aggarwal, Antoine Joux, Anupam Prakash, and Miklos Santha. A new public-key cryptosystem via mersenne numbers. Cryptology ePrint Archive, Paper 2017/481, 2017.

[AJPS18] Divesh Aggarwal, Antoine Joux, Anupam Prakash, and Miklos Santha. A new public-key cryptosystem via mersenne numbers. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part III*, volume 10993 of *Lecture Notes in Computer Science*, pages 459–482. Springer, 2018.

[BCGN17] Marc Beunardeau, Aisling Connolly, Rémi Géraud, and David Naccache. On the hardness of the mersenne low hamming ratio assumption. In Tanja Lange and Orr Dunkelman, editors, *Progress in Cryptology - LATINCRYPT 2017 - 5th International Conference on Cryptology and Information Security in Latin America, Havana, Cuba, September 20-22, 2017, Revised Selected Papers*, volume 11368 of *Lecture Notes in Computer Science*, pages 166–174. Springer, 2017.

[BCSV20] Carl Bootland, Wouter Castryck, Alan Szepieniec, and Frederik Vercauteren. A framework for cryptographic problems from linear algebra. *J. Math. Cryptol.*, 14(1):202–217, 2020.

[BCSV23] Carl Bootland, Wouter Castryck, Alan Szepieniec, and Frederik Vercauteren. SoK: On the security of cryptographic problems from linear algebra. *Mathematical Cryptology*, 3(1):52–95, Jul. 2023.

[BM05] Johannes Blömer and Alexander May. A tool kit for finding small roots of bivariate polynomials over the integers. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 251–267. Springer, 2005.

[BT19] Alessandro Budroni and Andrea Tenti. The mersenne low hamming combination search problem can be reduced to an ILP problem. In Johannes Buchmann, Abderrahmane Nitaj, and Tajje-eddine Rachidi, editors, *Progress in Cryptology - AFRICACRYPT 2019 - 11th International Conference on Cryptology in Africa, Rabat, Morocco, July 9-11, 2019, Proceedings*, volume 11627 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2019.

[BWK93] Thomas Becker, Volker Weispfenning, and Heinz Kredel. *Gröbner bases - a computational approach to commutative algebra*, volume 141 of *Graduate texts in mathematics*. Springer, 1993.

[CG20] Jean-Sébastien Coron and Agnese Gini. Improved cryptanalysis of the AJPS mersenne based cryptosystem. *J. Math. Cryptol.*, 14(1):218–223, 2020.

[Cop96] Don Coppersmith. Finding a small root of a univariate modular equation. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, volume 1070 of *Lecture Notes in Computer Science*, pages 155–165. Springer, 1996.

[Cop97] Don Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *J. Cryptol.*, 10(4):233–260, 1997.

[dBDJdW18] Koen de Boer, Léo Ducas, Stacey Jeffery, and Ronald de Wolf. Attacks on the AJPS mersenne-based cryptosystem. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*, volume 10786 of *Lecture Notes in Computer Science*, pages 101–120. Springer, 2018.

[FX20] Houda Ferradi and Keita Xagawa. Post-quantum provably-secure authentication and MAC from mersenne primes. In Stanislaw Jarecki, editor, *Topics in Cryptology - CT-RSA 2020 - The Cryptographers' Track at the RSA Conference 2020, San Francisco, CA, USA, February 24-28, 2020, Proceedings*, volume 12006 of *Lecture Notes in Computer Science*, pages 469–495. Springer, 2020.

[GKP94] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics: A Foundation for Computer Science, 2nd Edtion.* Addison-Wesley, 1994.

[HM08] Mathias Herrmann and Alexander May. Solving linear equations modulo divisors: On factoring given any bits. In Josef Pieprzyk, editor, *Advances in Cryptology - ASIACRYPT 2008, 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 7-11, 2008. Proceedings*, volume 5350 of *Lecture Notes in Computer Science*, pages 406–424. Springer, 2008.

[How97] Nick Howgrave-Graham. Finding small roots of univariate modular equations revisited. In Michael Darnell, editor, *Cryptography and Coding, 6th IMA International Conference, Cirencester, UK, December 17-19, 1997, Proceedings*, volume 1355 of *Lecture Notes in Computer Science*, pages 131–142. Springer, 1997.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In Joe Buhler, editor, *Algorithmic Number Theory, Third International Symposium, ANTS-III, Portland, Oregon, USA, June 21-25, 1998, Proceedings*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer, 1998.

[JM06] Ellen Jochemsz and Alexander May. A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants. In Xuejia Lai and Kefei Chen, editors, *Advances in Cryptology - ASIACRYPT 2006, 12th International Conference on the Theory and Application of Cryptology and Information Security, Shanghai, China, December 3-7, 2006, Proceedings*, volume 4284 of *Lecture Notes in Computer Science*, pages 267–282. Springer, 2006.

[LLL82] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[LZPL15] Yao Lu, Rui Zhang, Liqiang Peng, and Dongdai Lin. Solving linear equations modulo unknown divisors: Revisited. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part I*, volume 9452 of *Lecture Notes in Computer Science*, pages 189–213. Springer, 2015.

[May03] Alexander May. *New RSA vulnerabilities using lattice reduction methods*. PhD thesis, University of Paderborn, 2003.

[May10] Alexander May. Using LLL-reduction for solving RSA and factorization problems. In Phong Q. Nguyen and Brigitte Vallée, editors, *The LLL Algorithm - Survey and Applications*, Information Security and Cryptography, pages 315–348. Springer, 2010.

[NZH19] Jiehui Nan, Mengce Zheng, and Honggang Hu. Post-quantum pseudorandom functions from mersenne primes. In Bazhong Shen, Baocang Wang, Jinguang Han, and Yong Yu, editors, *Frontiers in Cyber Security - Second International Conference, FCS 2019, Xi'an, China, November 15-17, 2019, Proceedings*, volume 1105 of *Communications in Computer and Information Science*, pages 128–142. Springer, 2019.

[TD20] Marcel Tiepelt and Jan-Pieter D'Anvers. Exploiting decryption failures in mersenne number cryptosystems. In Keita Emura and Naoto Yanai, editors, *Proceedings of the 7th on ASIA Public-Key Cryptography Workshop, APKC@AsiaCCS 2020, Taipei, Taiwan, October 6, 2020*, pages 45–54. ACM, 2020.

[The23] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.5)*, 2023. `https://www.sagemath.org`.

[TK13] Atsushi Takayasu and Noboru Kunihiro. Better lattice constructions for solving multivariate linear equations modulo unknown divisors. In Colin Boyd and Leonie Simpson, editors, *Information Security and Privacy - 18th Australasian Conference, ACISP 2013, Brisbane,*

Australia, July 1-3, 2013. Proceedings, volume 7959 of *Lecture Notes in Computer Science*, pages 118–135. Springer, 2013.