A New Public-Key Cryptosystem via Mersenne Numbers

Divesh Aggarwal[1(✉)], Antoine Joux[2], Anupam Prakash[3,4], and Miklos Santha[4,5]

[1] School of Computing and Centre for Quantum Technologies,
National University of Singapore, Singapore, Singapore
divesh.aggarwal@gmail.com
[2] Chaire de Cryptologie de la Fondation SU, Sorbonne Université,
Institut de Mathématiques de Jussieu-Paris Rive Gauche, Inria,
CNRS, Univ Paris Diderot, Paris, France
[3] School of Physical and Mathematical Sciences,
Nanyang Technological University, Singapore, Singapore
[4] Centre for Quantum Technologies,
National University of Singapore, Singapore, Singapore
[5] IRIF, Université Paris Diderot, CNRS, Paris, France

Abstract. In this work, we propose a new public-key cryptosystem whose security is based on the computational intractability of the following problem: Given a Mersenne number $p = 2^n - 1$, where n is a prime, a positive integer h, and two n-bit integers T, R, decide whether their exist n-bit integers F, G each of Hamming weight less than h such that $T = F \cdot R + G$ modulo p.

1 Introduction

1.1 Motivation

Since the seminal work of Diffie and Hellman [DH76] which presented the fundamentals of public-key cryptography, one of the most important goal of cryptographers has been to construct secure and practically efficient public-key cryptosystems. Rivest, Shamir, and Adleman [RSA78] came up with the first practical public-key cryptosystem based on the hardness of factoring integers, and it remains the most popular scheme till date.

Shor [Sho97] gave a quantum algorithm that solves the abelian hidden subgroup problem and as a result solves both discrete logarithms and factoring. Back in 1994, this was not considered a real threat to the practical cryptographic schemes since quantum computers were far from being a reality. However, given the recent advances in quantum computing, there is serious effort in both the industry and the scientific community to make information security systems resistant to quantum computing. In fact, the National Institute of Standards and Technology (NIST) is now beginning to prepare for the transition into quantum-resistant cryptography and has announced a project where they are accepting submissions for quantum-resistant public-key cryptographic algorithms [NIS17].

H. Shacham and A. Boldyreva (Eds.): CRYPTO 2018, LNCS 10993, pp. 459–482, 2018.
https://doi.org/10.1007/978-3-319-96878-0_16

In the recent years, some presumably quantum-safe public-key cryptosystems have been proposed in the literature. Perhaps the most promising among these are those based on the hardness of lattice problems like Learning with Errors (LWE) based cryptosystems [Reg09], Ring-LWE based cryptosystems [LPR10] and NTRU [HPS98]. While these cryptosystems have so far resisted any classical or quantum attacks, it cannot be excluded that such attacks are possible in the future. In fact, there have been some, albeit unsuccessful, attempts at a quantum algorithm solving the LWE problem [ES16]. In particular, there is no unifying complexity-theoretic assumption (like NP-hardness) that relates the difficulty of breaking all these cryptosystems. Thus, it is desirable to come up with promising new proposals for public-key cryptosystems.

It is worthwhile to note that even though the concept of public-key cryptography was introduced four decades ago, the number of existing public-key cryptographic schemes whose hardness does not depend on the hardness of factoring or finding short vectors in lattices is not very large [KLC+00, McE78, LvTMW09, GWO+13, NS97]. This is not an exhaustive list but it illustrates the various approaches that have been tried. The rarity of proposals for potentially quantum safe public key cryptosystems further motivates the problem of constructing such cryptosystems.

1.2 Our Cryptosystem

Our cryptosystem is based on arithmetic modulo so called Mersenne numbers, i.e., numbers of the form $p = 2^n - 1$, where n is a prime. These numbers have an extremely useful property: For any number x modulo p, and $y = 2^z$, where z is a positive integer, $x \cdot y$ is a cyclic shift of x by z positions and thus the Hamming weight of x is unchanged under multiplication by powers of 2. Our encryption scheme is based on the simple observation that, given a uniformly random n-bit string R, when we consider $T = F \cdot R + G \pmod{p}$, where the binary representation of F and G modulo p has low Hamming weight, then T looks pseudorandom, i.e., it is hard to obtain any non-trivial information about F, G from R, T.

The public-key is chosen to be the pair (R, T), and the secret key is the string F. The encryption scheme also requires an efficient error correcting code with encoding function $\mathcal{E} : \{0,1\}^k \to \{0,1\}^n$ and decoding function $\mathcal{D} : \{0,1\}^n \to \{0,1\}^k$. In order to encrypt a message $m \in \{0,1\}^k$, the encryption algorithm chooses three random numbers A, B_1, B_2 of low Hamming weight modulo p and then outputs

$$C := (C_1, C_2),$$

where $C_1 = A \cdot R + B_1$, and $C_2 = (A \cdot T + B_2) \oplus \mathcal{E}(m)$ where $\oplus$ denotes the bitwise XOR operation. Given the private key, one can compute

$$C_2^* := C_1 \cdot F = (A \cdot T + B_2) - A \cdot G - B_2 + B_1 \cdot F.$$

Since A, B_1, B_2, F, G have low Hamming weight, the Hamming distance between $A \cdot T + B_2$ and C_2^* is expected to be low, and so we get that $\mathcal{D}(C_2 \oplus C_2^*)$ is equal to to m with high probability. For more details on our scheme and the underlying security assumption, we refer the reader to Sects. 4 and 5.

1.3 Related Work

The Mersenne cryptosystem can be seen as belonging to a family that started with the Ntru cryptosystem and as been instantiated in many ways [HPS98, Reg09, LPR10, MTSB13]. The common idea behind all these cryptosystems is to work with elements in a ring which are hidden by adding some small noise. This notion of smallest needs to be somewhat preserved under the arithmetic operation. At the same time, it should be somewhat unnatural and not fully compatible with the ring structure in order to lead to hard problems.

Our goal in designing the Mersenne cryptosystem was to find a very simple instantiation of this paradigm based on the least complicated ring we could find. This led us to consider numbers modulo a prime together with the Hamming weight to measure smallest. In this context, it is natural to restrict ourselves to Mersenne primes, since reduction modulo such a prime cannot increase Hamming weights. Moreover, our cryptosystem relies on a conceptually simpler ring of numbers modulo a prime and its description only requires very elementary mathematics.

Our first proposal using this structure [AJPS17] only allowed us to encrypt a single bit at a time. The security parameters in [AJPS17] were based on the assumption that there is no attack on the cryptosystem that runs faster than the trivial attack that runs in time $\binom{n}{h}$. Subsequent works showed that this assumption was incorrect. In particular, [BCGN17] showed a non-trivial guess-and-determine attack based on a low-dimension lattice reduction subroutine that runs in time $(2+\varepsilon+o(1))^{2h}$ for some small constant ε, and [dBDJdW17] gave a meet-in-the-middle attack that runs in time $O\left(\binom{n-1}{h-1}^{1/2}\right)$ on classical computers and $O\left(\binom{n-1}{h-1}^{1/3}\right)$ on quantum machines. While these attacks could be circumvented by choosing the parameter h to be as large as the security parameter, this would make our cryptosystem inefficient.

Fortunately, in the present proposal, we are able to overcome this difficulty. We describe a variant that allows us to encrypt many bits at a time. This allows in turn to choose much larger parameters which resist the attacks in [BCGN17, dBDJdW17], even in their Groverized quantum form while still maintaining the efficiency of our cryptosystem. Such quantum attacks would have complexity larger than 2^h where h is the Hamming weight we allow for low Hamming weight numbers. This explains our choice of h to be equal to the desired quantum security level.

Since it is well-known that a cryptosystem of this type can be easily vulnerable to chosen-ciphertext attack, it is extremely important to bind them together with a CCA-secure wrapper. We chose to present the system as a key encapsulation mechanism because this makes the design of the CCA wrapper very simple.

1.4 Organization of the Paper

In Sect. 2 we introduce some preliminaries about Mersenne primes and security definitions. In Sect. 3 we provide a semantically secure basic bit by bit encryption scheme. In Sect. 4 we give a semantically secure blockwise encryption scheme. In Sect. 5, we prove the semantic security for the scheme presented in Sect. 4. In Sect. 6 we discuss the known cryptanalytic attacks against this scheme. In Sect. 7 we give the final key encapsulation scheme secure against chosen ciphertext attacks in the random oracle model. In Sects. 8 and 9 we provide an instantiation for the error correcting codes used in our encryption/key encapsulation schemes.

2 Preliminaries

Notations. For any distinguisher D that outputs a bit $b \in \{0,1\}$, the distinguishing advantage to distinguish between two random variables X and Y is defined as:

$$\Delta^D(X\ ;\ Y) := |\Pr[D(X) = 1] - \Pr[D(Y) = 1]|.$$

The following lemma is well known and easy to see.

Lemma 1. *Given a probabilistic polynomial time computable function f on two random variables X and Y, if there is a probabilistic polynomial time distinguisher D that distinguishes between $f(X)$ and $f(Y)$ with advantage δ, then there is a probabilistic polynomial time distinguisher D' that distinguishes between X, and Y with advantage δ.*

2.1 Mersenne Numbers and Mersenne Primes

Let n be a positive integer, and let $p = 2^n - 1$. When n is a prime, p is called a Mersenne number, and if $2^n - 1$ is itself a prime number, then it is called a Mersenne prime. Note that if n is a composite number of the form $n = k\ell$, then $2^k - 1$ and $2^\ell - 1$ divide p, and hence p is not a prime. The smallest Mersenne primes are

$$2^2 - 1, 2^3 - 1, 2^5 - 1, 2^7 - 1, 2^{13} - 1, 2^{17} - 1, \dots$$

We denote by $\mathbb{Z}_p$ the ring of integers modulo p. We index binary strings from right to left, that is for $x \in \{0,1\}^n$ we write x as $x_n \dots x_1$. The Hamming weight of an n-bit string y is the total number of 1's in y and is denoted by $\mathsf{Ham}(y)$. Let seq : $\mathbb{Z}_p \to \{0,1\}^n$ be the map which to $x \in \mathbb{Z}_p$ associates the binary string $\mathrm{seq}(x)$ representing x. The map $\mathrm{int} : \{0,1\}^n \to \mathbb{Z}_p$ sends a string y into the integer represented by y modulo p. Clearly seq and int are inverse functions between $\mathbb{Z}_p$ and $\{0,1\}^n \setminus \{1^n\}$, and $\mathrm{int}(1^n) = 0$. We use this bijection between $\mathbb{Z}_p$ and $\{0,1\}^n \setminus \{1^n\}$ to define *addition* and *multiplication* over $\{0,1\}^n$ in the natural way: for $y, y' \in \{0,1\}^n$, let $y + y' = \mathrm{seq}(\mathrm{int}(y) + \mathrm{int}(y'))$, and let $y \cdot y' = \mathrm{seq}(\mathrm{int}(y) \cdot \mathrm{int}(y'))$. It is easy to see that both operations remain associative and commutative, and the distributivity of the multiplication over the addition also holds. We also set $(-1) \cdot y = -y = \mathrm{seq}(-(\mathrm{int}(y))$. Observe that addition is invariant by rotation, that is if $\mathrm{rot}_k(y)$ denotes the circular rotation of y by k positions to the left, then $\mathrm{rot}_k(y + y') = \mathrm{rot}_k(y) + \mathrm{rot}_k(y')$.

Lemma 2. *Let $p = 2^n - 1$. For all $A, B \in \{0,1\}^n$, we have*

1. $\mathsf{Ham}(A + B) \leq \mathsf{Ham}(A) + \mathsf{Ham}(B)$.
2. $\mathsf{Ham}(A \cdot B) \leq \mathsf{Ham}(A) \cdot \mathsf{Ham}(B)$.
3. *If $A \neq 0^n$ then $\mathsf{Ham}(-A) = n - \mathsf{Ham}(A)$.*

Proof.

1. If $A = 1^n$ the result is obviously true. When $A \neq 1^n$, we prove the result by induction on the Hamming weight of B. If $B = 0^n$ the statement is obviously true.
   For the induction step we first prove the claim when $\mathsf{Ham}(B) = 1$. Let i be the index on which B takes the value 1. Since addition is invariant by rotation, we may assume that $i = 1$ and thus $B = 0^{n-1}1$. A can be written as $C01^j$ for some $0 \leq j \leq n-1$, and $A + B = C10^j$. Thus $\mathsf{Ham}(A + B) = \mathsf{Ham}(A) - j + 1 \leq \mathsf{Ham}(A) + 1$.
   Let $\mathsf{Ham}(B) = k > 1$. Then we can decompose B as $B_1 + B_2$, where $\mathsf{Ham}(B_1) = k - 1$ and $\mathsf{Ham}(B_2) = 1$. By the previous claim and the induction hypothesis we get:
   $$\mathsf{Ham}(A+B) = \mathsf{Ham}((A+B_1)+B_2) \leq \mathsf{Ham}(A+B_1)+1 \leq \mathsf{Ham}(A)+(k-1)+1,$$
   and the result follows.
2. If $B = 0^n$ the statement is obviously true. Otherwise, for some $k \geq 1$, we can decompose B as $B_1 + \cdots + B_k$, where each B_i has Hamming weight 1, for $1 \leq i \leq k$. Let j_i be the index of the position where B_i takes the value 1. Then $A \cdot B_i = \mathrm{rot}_{j_i - 1}(A)$. Thus $\mathsf{Ham}(A \cdot B_i) = \mathsf{Ham}(A)$, and by distributivity we get $A \cdot B = A \cdot B_1 + \cdots + A \cdot B_k$. The result then follows from part (1).
3. If $A \neq 0^n$ then $-A$ is the binary string obtained from A by replacing 0's by 1's and 1's by 0's.

□

2.2 Security Definitions

Public-Key Encryption. A public key encryption scheme comprises three algorithms: the key generation algorithm KeyGen, the encryption algorithm Enc, and the decryption algorithm Dec. The KeyGen algorithm outputs a public-key pk, and a secret key sk. The encryption algorithm Enc takes as input a message m, and pk, and outputs a ciphertext C. The decryption algorithm takes as input a ciphertext C and sk, and outputs a message m' or a special symbol $\perp$ indicating rejection. We say that the encryption scheme is $1-\delta$ correct if for all m, $\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) = m] \geq 1-\delta$, where the probability is over the randomness of pk, sk and the encryption algorithm.

We denote the security parameter by λ. All other parameters including key lengths and ciphertext size are polynomial functions of λ.

Definition 1. The public-key encryption scheme

$$PKE = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$$

is said to be *semantically secure* if for any probabilistic polynomial time distinguisher and any pair of messages m_0, m_1 of equal length, given the public key pk, the advantage for distinguishing $C_0 = \mathsf{Enc}(\mathsf{pk}, m_0)$ and $C_1 = \mathsf{Enc}(\mathsf{pk}, m_1)$ is at most $\frac{\text{poly}(|C_i|)}{2^\lambda}$ for some polynomial poly.

Definition 2. The public-key encryption scheme

$$PKE = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$$

is said to be *secure under chosen ciphertext attacks* if for any probabilistic polynomial time distinguisher that is given access to an oracle that decrypts any given ciphertext, the following holds: For any pair of messages m_0, m_1 of equal length, given the public key pk, the advantage for distinguishing $C_0 = \mathsf{Enc}(\mathsf{pk}, m_0)$ and $C_1 = \mathsf{Enc}(\mathsf{pk}, m_1)$ is at most $\frac{\text{poly}(|C_i|)}{2^\lambda}$ for some polynomial poly under the assumption that the distinguisher does not query the oracle with C_0 or C_1.

Key Encapsulation Mechanism. A key-encapsulation mechanism (KEM) comprises three algorithms: the key generation algorithm KeyGen, the encapsulation algorithm Encaps, and the decapsulation algorithm Decaps, and a key space $\mathcal{K}$. The KeyGen algorithm outputs a public-key pk, and a secret key sk. The encapsulation algorithm Encaps takes as input a public key pk to produce a ciphertext C and a key $K \in \mathcal{K}$. The decapsulation algorithm Decaps takes as input a ciphertext C and sk, and outputs a key K' or a special symbol $\perp$ indicating rejection. We say that the KEM is $(1-\delta)$-correct if

$$\Pr[\mathsf{Decaps}(\mathsf{sk}, C) = K \;:\; (C, K) \leftarrow \mathsf{Encaps}(\mathsf{pk})] \geq 1-\delta,$$

where the probability is over the randomness of pk, sk and the encapsulation algorithm.

We denote the security parameter by λ. All other parameters including key lengths and ciphertext size are polynomial functions of λ.

Definition 3. The key-encapsulation mechanism

$$KEM = (\mathsf{KeyGen}, \mathsf{Encaps}, \mathsf{Decaps})$$

is said to be *semantically secure* if for any probabilistic polynomial time distinguisher, given the public key pk, the advantage for distinguishing (C, K_0) and (C, K_1), where $(C, K_0) \leftarrow \mathsf{Encaps}(\mathsf{pk})$ and K_1 is uniform and independent of C is at most $\frac{\text{poly}(|C|,|K_0|)}{2^\lambda}$ for some polynomial poly.

Definition 4. The key-encapsulation mechanism

$$KEM = (\mathsf{KeyGen}, \mathsf{Encaps}, \mathsf{Decaps})$$

is said to be *secure under chosen ciphertext attacks* if for any probabilistic polynomial time distinguisher that is given access to the decapsulation oracle and the public key pk, the advantage for distinguishing (C, K_0) and (C, K_1), where $(C, K_0) \leftarrow \mathsf{Encaps}(\mathsf{pk})$ and K_1 is uniform and independent of C is at most $\frac{\text{poly}(|C|,|K_0|)}{2^\lambda}$ for some polynomial poly under the assumption that the distinguisher does not query the oracle with C.

2.3 Security Assumptions

The semantic security of our encryption scheme is based on the following assumption.

Definition 5. The *Mersenne Low Hamming Combination Assumption* states that given an n-bit Mersenne prime $p = 2^n - 1$, and an integer h, the advantage of any probabilistic polynomial time adversary running in time poly(n) in attempting to distinguish between

$$\left(\begin{bmatrix} R_1 \\ R_2 \end{bmatrix}, \begin{bmatrix} R_1 \\ R_2 \end{bmatrix} \cdot A + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \right) \text{ and } \left(\begin{bmatrix} R_1 \\ R_2 \end{bmatrix}, \begin{bmatrix} R_3 \\ R_4 \end{bmatrix} \right)$$

is at most $\frac{\text{poly}(n)}{2^\lambda}$, where R_1, R_2, R_3, R_4 are independent and uniformly random n-bit strings, and A, B_1, B_2, are independently chosen n-bit strings each having Hamming weight h.

We note that the assumption has some striking similarity to the learning with errors assumption by Regev [Reg09], where A corresponds to the secret, and B_1, B_2 correspond to the small error. The Mersenne Low Hamming Combination Assumption, in particular implies that one cannot obtain any useful information about A, B from the pair $(R_1, A \cdot R_1 + B)$. Notice that if the pair $(R_1, A \cdot R_1 + B)$ is assumed to be pseudorandom, then so is the pair $(R_1, A \cdot (-R_1) + B)$, and so one cannot obtain any useful information about A, B from the pair $(R_1, -A \cdot R_1 + B)$. The Mersenne Low Hamming Ratio Assumption is a homogeneous version of this assumption in the sense that we state that no useful information about A, B can be obtained from $(R_1, -A \cdot R_1 + B)$ given that $-A \cdot R_1 + B = 0$. It is required for the semantic security of the bit-by-bit encryption scheme that we describe in the next section, and was introduced in a previous version of this paper.

Definition 6. *The* Mersenne Low Hamming Ratio Assumption *states that given an n-bit Mersenne prime $p = 2^n - 1$, and an integer h, the advantage of any probabilistic polynomial time adversary running in time poly(n) in attempting to distinguish between* $\mathrm{seq}(\frac{\mathrm{int}(A)}{\mathrm{int}(B)})$ *and R is at most $\frac{\mathrm{poly}(n)}{2^\lambda}$, where R is a uniformly random n-bit string, and A, B, are independently chosen n-bit strings each having Hamming weight h.*

3 Basic Bit-by-Bit Encryption

In the following, we describe a basic encryption scheme to encrypt a single bit $b \in \{0, 1\}$.

Key Generation.

- Given the security parameter λ, choose a Mersenne prime $p = 2^n - 1$ and an integer h such that $\binom{n}{h} \geq 2^\lambda$ and $4h^2 < n$.
- Choose F, G to be two independent n-bit strings chosen uniformly at random from all n-bit strings of Hamming weight h.
- Set $\mathsf{pk} := H = \mathrm{seq}(\frac{\mathrm{int}(F)}{\mathrm{int}(G)})$, and $\mathsf{sk} := G$.

Encryption. The encryption algorithm chooses two independent strings A, B uniformly at random from all strings with Hamming weight h. A bit b is encrypted as

$$C = \mathsf{Enc}(\mathsf{pk}, b) := (-1)^b \left(A \cdot H + B\right).$$

Decryption. The decryption algorithm computes $d = \mathsf{Ham}(C \cdot G)$. If $d \leq 2h^2$, then output 0; if $d \geq n - 2h^2$, then output 1. Else output $\perp$.

For the correctness of the decryption note that $C \cdot G = (-1)^b \cdot (A \cdot F + B \cdot G)$ which, by Lemma 2, has Hamming weight at most $2h^2$ if $b = 0$, and at least $n - 2h^2$ if $b = 1$.

The basic bit-by-bit encryption scheme can be viewed as a simple proposal for a cryptosystem based on arithmetic modulo the Mersenne primes, however it is not efficient with respect to ciphertext size. Since this is not our final proposed encryption scheme, we do not analyze its security although it will easily follow from the Mersenne Low Hamming Ratio and the Mersenne Low Hamming Combination Assumption and appeared in a previous version of this paper. In the next section, we describe a scheme for encrypting longer message blocks.

4 Our Main Semantically Secure Public-Key Cryptosystem

It is reasonable to choose the message block length to be the same as the security parameter in practice. For this reason, we describe below a scheme for encrypting a message block $m \in \{0, 1\}^\lambda$.

Key Generation.

- Given the security parameter λ, choose a Mersenne prime $p = 2^n - 1$ such that $h = \lambda$ and $n > 10h^2$.
- Let F, G to be two independent n-bit strings chosen uniformly at random from all n-bit strings of Hamming weight h. Let R be a uniformly random n-bit string.
- Set $\mathsf{pk} := (R, F \cdot R + G) := (R, T)$, and $\mathsf{sk} := F$.

Encryption. The encryption algorithm chooses three strings A, B_1, B_2 independently and uniformly at random from all strings with Hamming weight h. Let $(\mathcal{E}, \mathcal{D})$ be the encoding and decoding algorithms of an error correcting code that we choose later. The message $m \in \{0,1\}^\lambda$ is encrypted as,

$$\mathsf{Enc}(\mathsf{pk}, m) := (C_1, C_2) := (A \cdot R + B_1, (A \cdot T + B_2) \oplus \mathcal{E}(m)).$$

Here $\mathcal{E} : \{0,1\}^\lambda \to \{0,1\}^n$ is a suitably chosen error correcting code and $\oplus$ denotes the bitwise XOR operation.

Decryption. The decryption algorithm computes $\mathcal{D}((F \cdot C_1) \oplus C_2)$.

In order to say that the scheme is $(1-\delta)$-correct, we need to choose the error correcting code such that $\Pr_{(C_1,C_2)\leftarrow \mathsf{Enc}(\mathsf{pk},m)}[\mathcal{D}((F \cdot C_1) \oplus C_2) = m] \geq 1 - \delta$, where the probability is over the randomness of the encryption algorithm and the choice of pk, sk. For concrete instantiations of error correcting codes that satisfy this for a small enough δ, see Sect. 8.

5 Semantic Security of the Cryptosystem

In this section, we prove the semantic security of the PKE scheme in Sect. 4.

Theorem 1. *The encrpytion scheme* $(\mathsf{Enc}, \mathsf{Dec})$ *described in Sect. 4 is semantically secure under the Mersenne Low Hamming Combination Assumption.*

Proof. In the following, let $A, B_1, B_2, F, G, R, R', R'', R'''$ be independently chosen such that A, B_1, B_2, F, G are chosen uniformly from all strings of Hamming weight h, and R, R', R'', R''' are uniformly random strings. Let $T = F \cdot R + G$. By the Mersenne Low Hamming Combination Assumption, for any probabilistic polynomial time distinguisher D running in time $\text{poly}(n)$,

$$\Delta^D(R, T\ ;\ R, R') \leq \frac{\text{poly}(n)}{2^\lambda}.$$

Now, from Lemma 1, we have that for any probabilistic polynomial time distinguisher D' running in time $\text{poly}(n)$,

$$\Delta^{D'}(R, T, A \cdot R + B_1, A \cdot T + B_2; R, R', A \cdot R + B_1, A \cdot R' + B_2) \leq \frac{\text{poly}(n)}{2^\lambda}.$$

Again, by the Mersenne Low Hamming Combination Assumption, we have that

$$\Delta^{D'}(R, R', A \cdot R + B_1, A \cdot R' + B_2 \ ; \ R, R', R'', R''') \leq \frac{\text{poly}(n)}{2^{\lambda}}.$$

Using the triangle inequality, we get that

$$\Delta^{D'}(R, T, A \cdot R + B_1, A \cdot T + B_2 \ ; \ R, R', R'', R''') \leq \frac{2\,\text{poly}(n)}{2^{\lambda}}.$$

This implies that for any message m,

$$\Delta^{D'}(R, T, A \cdot R + B_1, A \cdot T + B_2 \oplus \mathcal{E}(m) \ ; \ R, R', R'', R''') \leq \frac{2\,\text{poly}(n)}{2^{\lambda}},$$

since R, R', R'', R''' and $R, R', R'', R''' \oplus \mathcal{E}(m)$ are identically distributed. This implies the required semantic security. □

6 Analysis of Our Security Assumption

6.1 Attempts at Cryptanalysis

In this section, we mention the known approaches to break our security assumption and thereby mention the conjectured security guarantee for our scheme. For cryptanalysis, it is often more convenient to talk about search problems. We introduce the following search problem whose solution would imply an attack on our cryptosystem.

Definition 7 (Mersenne Low Hamming Combination Search Problem). *For an n-bit Mersenne number $p = 2^n - 1$ and an integer h, given tuple $(R, FR+G \pmod{p})$ where R is a uniformly random n-bit string and F, G have Hamming weight h, find F, G.*

For the remainder of the paper, we call this problem $\mathcal{P}$. It is easy to see that if one can efficiently solve the problem $\mathcal{P}$, then one can break the assumption in Definition 5, and hence the security of our cryptosystem. It is therefore important to study the hardness of this problem.

Hamming Distance Distribution. Let R be a uniformly random n bit string and $Y = FR+G$ where F, G are chosen uniformly at random from n bit strings with Hamming weight h. A basic test for the assumption that Y is pseudorandom given R is to check that the distribution of $\mathsf{Ham}(R, R')$ is close to the distribution of $\mathsf{Ham}(R, T)$ where T is a uniformly random n bit string.

If R is a fixed string and X is a uniformly random n bit string, the random variable $f_R(X) = \frac{\mathsf{Ham}(X,R) - n/2}{\sqrt{n/4}}$ is approximated by the standard normal random variable $N(0, 1)$. We generated R at random and then obtained samples $Y =$

$FR + G$ where F, G are uniformly distributed over strings of Hamming weight $\sqrt{n}$. A quantile-quantile plot of $f_R(Y_i)$ against samples from $N(0, 1)$ is close to a straight line and does not show significant deviations from normality.

One could also perform more advanced statistical tests, such as the NIST suite [RSN+01] to verify the pseudorandomness of Y given R. However, in the context of cryptographic schemes, such tests only serve as sanity checks and it is preferable to focus on dedicated cryptanalysis.

Weak key attack. Following the appearance of a preliminary version of this paper, [BCGN17] found a weak key attack on the Mersenne Low Hamming Ratio search problem where given $H = \mathrm{seq}(\frac{\mathrm{int}(F)}{\mathrm{int}(G)}) \mod P$ with F, G having low Hamming weight, the goal is to find F and G.

The weak key attack of [BCGN17] is based on rational reconstruction. If all the bits of F and G are in the right half of the bits, then both F and G are smaller than $\sqrt{P}$ and they can easily be recovered using a continued fraction expansion of H/P. The weak key attack also extends to the Mersenne Low Hamming Combination search problem, we choose parameters such that the success probability for this attack is negligible.

Generalization using LLL. The authors of the above weak attack also proposed in [BCGN17] a generalization based on guessing a decomposition of F and G into windows of bits such that in any window all the '1's are on the right. Using such a decompostion and replacing the use of continued fraction by LLL in relatively small dimension they can recover F and G from any compatible window decomposition.

A careful analysis of this method and its cost is presented in [dBDJdW17] and concludes that its running time is $2^{(2+\epsilon)h}$ for some small constant h. For simplicity, we assume that the cost of this attack is 2^{2h} on a classical computer.

Even if this attack was developed for the homogeneous Mersenne Low Hamming Ratio assumption, it is likely that it generalizes to the Mersenne Low Hamming Combination Assumption. We thus assume that it is the case. To the best of our knowledge, this is the most efficient known attack on our security assumption and the security parameters proposed in Sect. 8 have been revised to withstand it.

Quantum Speedup via Grover's Algorithm. With access to a quantum computer, one could use Grover's algorithm [Gro96] to obtain a quadratic speedup over the above attack.

Note that the attack performs a lattice reduction step for each guess of window decomposition and concludes that they are correct if the lattice reduction step succeeds. The Groverized version of the algorithm would prepare a superposition over possible guesses of window decompositions, use a unitary operator that performs lattice reduction to mark the good guesses for window decomposition and then amplify the success probability using Grover's search. This would certainly need very sophisticated universal quantum computers and it may well be infeasible for near term quantum devices. However, in view of this potential quantum attack and potential cryptananalytic improvements, we take this attack into account. With this constraint, our cryptosystem can only be secure if we make sure that h is at least equal to the desired security level. For simplicity, we just set $h = \lambda$ and assume that the best possible attack on the Mersenne Low Hamming Combination problem has complexity at least 2^h to derive security estimates in Sect. 8.

Meet in the middle attack. A recent work [dBDJdW17] gave a non-trivial meet-in-the-middle attack that makes use of locality-sensitive hash functions. Its complexity is $O\left(\binom{n-1}{h-1}^{1/2}\right)$ on classical computers and $O\left(\binom{n-1}{h-1}^{1/3}\right)$ on quantum machines.

For our choice of parameters, this is much bigger than 2^h and thus doesn't affect the security level.

Attacking the system if n is not a prime. We mention here that it seems quite important to choose $2^n - 1$ to be a prime for our cryptosystem. There is at least a partial attack when n is not prime. Indeed if n_0 divides n, then $q = 2^{n_0} - 1$ divides $p = 2^n - 1$, and also F, G have Hamming weight at most h modulo q. Thus, given $Y = FR+G \mod q$, one can try to guess the secret key G modulo q, which can be done in $\sqrt{\binom{n_0}{h}}$ time using a quantum algorithm. This also reveals F modulo q and we can likely use it to guess F, G modulo p much faster than the attacks that work in the prime case.

6.2 Active Attacks

Active attacks and/or decryption errors attacks are powerful tools that can be used to attack our bit-by-bit encryption. We recall that the basic idea of such attacks is to ask for the decryption of incorrectly formed ciphertext and use the answers to recover information about the key.

For example, incorrect ciphertexts can be obtained by picking a random bitstring, by modifying a valid one or encrypting in a non conformant way. Here, we review the attack in the context of a single bit, but it is important to note that the encryption of many bits remain vulnerable to such attacks, even if plaintext redundancy in the style of OAEP paddings [Sho02] is added. We show in Sect. 7 how to withstand such attacks using appropriate checks of ciphertext validity.

For simplicity, assume that we have access to a decryption oracle. Forming pseudo ciphertexts of the form $A^*H + B^*$ with A^* and B^* with low but not conformant Hamming weights can leak information about the private key. In particular, one might incrementally add '1' bits into B^* (or A^*) until decryption transitions from 0 to $\perp$. We did not concretely write down a full working attack along this line, but it is clear that our encryption scheme would be vulnerable to such attacks.

7 Mersenne Key Encapsulation Mechanism

Since we have seen in Sect. 6.2 that the semantically secure cryptosystem described in Sect. 4 cannot offer resistance to chosen-ciphertext attack, we need to integrate it into a more complex scheme with this ability. A first approach would be to use an existing generic transformation for this purpose. However, this is not a simple matter, indeed, systems such as OAEP or REACT [OP01] perform checks at the plaintext level and thus cannot protect against the attack strategy of Sect. 6.2. The Naor-Yung paradigm [NY90, CHK10] would be more suitable but the introduction of dual-encryption and non-interactive proofs is too costly for our purpose.

In this section, we specify a full cryptosystem that achieves this level of resistance using a transformation specifically designed for our encryption scheme. We present our cryptosystem as a key encapsulation mechanism. It can be turned into an public key encryption scheme using a standard transformation.

Let $\mathsf{Enc}, \mathsf{Dec}$ be the encryption and decryption algorithms as defined in Sect. 4. In addition to this, our transformation uses a random oracle $\mathcal{H}$ that takes as input λ-bit strings, and outputs a uniformly random string that is long enough to compute a λ-bit string, and three n-bit strings, each chosen uniformly over all strings of Hamming weight h, and such that all four strings are independent. Let $\mathcal{H}_0(k), \mathcal{H}_1(k), \mathcal{H}_2(k), \mathcal{H}_3(k)$ be the four such outputs obtained from the random oracle on input k. As usual, every output is randomly selected whenever a fresh query is asked.

Key Generation. The key generation is identical to the semantically secure cryptosystem and produces $\mathsf{pk} := R, T := F \cdot R + G$, and $\mathsf{sk} := F$ where R is a uniformly random n-bit string, and F, G are chosen uniformly at random from n-bit strings of Hamming weight h.

Key Encapsulation. Given the public key $\mathsf{pk} = (R, T)$, the algorithm Encaps proceeds as follows:

1. Pick a uniformly random λ-bit string K.
2. Let $S = \mathcal{H}_0(K)$.
3. Let $A = \mathcal{H}_1(K)$, $B_1 = \mathcal{H}_2(K)$, and $B_2 = \mathcal{H}_3(K)$.
4. Let $C = (C_1, C_2)$, where $C_1 = A \cdot R + B_1$, and $C_2 = \mathcal{E}(K) \oplus (A \cdot T + B_2)$.
5. Output C, S.

Decapsulation. Given a ciphertext $C = (C_1, C_2)$, and $\mathsf{sk} = F$, the decapsulation algorithm Decaps algorithm proceeds as follows:

1. Compute $K' = \mathcal{D}((F \cdot C_1) \oplus C_2)$.
2. Let $A' = \mathcal{H}_1(K')$, $B_1' = \mathcal{H}_2(K')$, and $B_2 = \mathcal{H}_3(K')$.
3. Let $C' = (C_1', C_2')$, where $C_1' = A' \cdot R + B_1'$, and $C_2' = \mathcal{E}(K') \oplus (A' \cdot T + B_2')$.

4. If $C = C'$, output $\mathcal{H}_0(K')$, else output $\perp$.

A proof of the CCA security of our transformation is nearly identical to that of [HHK17]. We include the proof below for completeness.

Theorem 2. *Assume that $\mathcal{H}$ is a random oracle and that the scheme from Sect. 4 is semantically secure. Then the above mentioned key encapsulation mechanism is secure against chosen-ciphertext attacks.*

Proof. We need to show that chosen-ciphertext queries are not helping the adversary, i.e. that they can be simulated without significantly degrading the adversary's advantage. Once this is done, the semantic security suffices to conclude our result.

For this, we consider the behavior of the decapsulation oracle when receiving a ciphertext $C^\star = (C_1^\star, C_2^\star)$. We want to conclude, that unless the ciphertext was produced by a procedure functionally equivalent to the encapsulation specification, the decapsulation oracle outputs $\perp$ with overwhelming probability.

The decapsulation oracle, on input $C^\star = (C_1^\star, C_2^\star)$ computes $K^\star = \mathcal{D}((F \cdot C_1^\star) \oplus C_2^\star)$, and then calls the encapsulation algorithm with input $\widetilde{K}$ to obtain $\widetilde{C} = (\widetilde{C}_1, \widetilde{C}_2)$. If $\widetilde{C} = C^\star$, then the oracle outputs $\mathcal{H}_0(\widetilde{K})$, and the oracle outputs $\perp$ otherwise.

If the random oracle was previously queried with the seed $\widetilde{K}$ by the adversary, then since the encapsulation procedure is a deterministic function of $\widetilde{K}$, the output of the decapsulation oracle could be efficiently simulated by the adversary. On the other hand, if the random oracle was never queried with the key $\widetilde{K}$, then we have that $\widetilde{C}_1 = A \cdot R + B_1$, where $A = \mathcal{H}_1(\widetilde{K})$ and $B_1 = \mathcal{H}_2(\widetilde{K})$. Since $\mathcal{H}_1, \mathcal{H}_2$ are random oracles, A, B are assumed to be independent of everything else, and hence the probability that the decapsulation oracle does not output $\perp$ is at most

$$\Pr[A \cdot R + B_1 = C_1^\star] = \Pr[B_1 = C_1^\star - A \cdot R] \leq \frac{1}{\binom{n}{h}}.$$

□

8 Instantiating Error Correcting Code in Our Scheme

In this section, we give a concrete choice of parameters, instantiate error correcting codes in our scheme, and analyze the probability of decryption error.

We will set the security parameter to $\lambda = 256$. This is the one of the most acceptable choices in the cryptographic community given the current computational powers.

As we discussed in Sect. 6, the best known efficient attack on our cryptosystem succeeds runs in time $O(2^{2h})$. We assume, somewhat conservatively, that even with future advancements in cryptanalysis of our scheme, the running time cannot be improved beyond $O(2^h)$. Under this assumption, we set h to be the security parameter λ. Thus, $\lambda = h = 256$. Also, in order to prevent against unforeseen attacks that exploit the factorization of p, we choose $p = 2^n - 1$ to be a Mersenne prime.

8.1 Instantiation Based on Deterministic Error-Correction Codes

We will need the following result. We prove this in Sect. 9.

Theorem 3. *Let U be a random variable having uniform distribution on strings of length n. For every n-bit string x of Hamming weight Δ and for every $\varepsilon > 0$,*

$$\Pr[\mathsf{Ham}(U, U + x) \geq 2(1+\varepsilon)\Delta] \leq 2^{-2\Delta(\varepsilon - \ln(1+\varepsilon))}.$$

We now bound the Hamming distance between $F \cdot (A \cdot R + B_1)$ and $A \cdot (F \cdot R + G) + B_2$. Using Theorem 3, and Lemma 2, we get that for any $\varepsilon \in (0, 1)$,

$$\Pr[\mathsf{Ham}(F \cdot (A \cdot R + B_1), F \cdot (A \cdot R)) \geq 2h^2(1+\varepsilon)] \leq 2^{-2h^2(\varepsilon - \ln(1+\varepsilon))},$$

and

$$\Pr[\mathsf{Ham}(A \cdot (F \cdot R + G) + B_2, F \cdot (A \cdot R)) \geq 2(h^2+h)(1+\varepsilon)] \leq 2^{-(2h^2+h)(\varepsilon - \ln(1+\varepsilon))}.$$

Using union bound, and triangle inequality, we get that

$$\Pr[\mathsf{Ham}(F \cdot (A \cdot R + B_1), A \cdot (F \cdot R + G) + B_2) \geq (4h^2 + 2h)(1+\varepsilon)]$$

is at most

$$2^{-(2h^2-1)(\varepsilon - \ln(1+\varepsilon))} \leq 2^{-(2h^2-1)(\varepsilon^2/2 - \varepsilon^3/3)} \leq 2^{-(2h^2-1)(\varepsilon^2/6)},$$

where the second to last inequality follows from the Taylor series expansion of $\ln(1+\varepsilon)$.

This our scheme is $1-\delta$-correct if the error correction code $(\mathcal{E}, \mathcal{D})$ corrects up to $(4h^2 + 2h)(1+\varepsilon)$ errors where ε is chosen such that $2^{-(2h^2-1)(\varepsilon^2/6)} < \delta$.

This implies that by choosing an appropriate error-correction code, we get that for any $\delta > 0$, and for $n = ch^2$, for a large enough constant c, our scheme is $1-\delta$-correct. In particular, we can instantiate our scheme with $n \geq 2^\ell - 1 = 2^{21} - 1 = 32h^2 - 1$, and using Dual-BCH Codes [MS77], we can encode a message of length $k = 256$, such that the parameter where $t = \lceil k/\ell \rceil = 13$, and the scheme corrects up to at least

$$\frac{n}{4} - \frac{(t-1) \cdot 2^{\ell/2}}{2} - 1 \geq 8h^2 - 40h$$

errors. Thus, choosing $\varepsilon = \frac{8h^2-40h}{4h^2+2h} - 1$ gives an instantiation of our scheme with decryption error as low as $2^{-h^2/4}$.

Notice that the bound on the Hamming weight of $F \cdot B_1$, $A \cdot G$, and also the bound on the Hamming distance in Theorem 3 is not tight, and perhaps it will be difficult to prove much tighter bounds. Moreover, the error distribution is randomized, and exploiting this fact could perhaps lead to better error correction as we discuss in the next section.

8.2 Instantiation Based on Repetition Codes

In the previous section we considered dual-BCH codes which correct a certain fraction of errors no matter how these errors are distributed. On the other hand we observe that for our particular application, the error is "quite" random, and even though this distribution is difficult to mathematically analyze, it is reasonable to conjecture that the error pattern is somewhat similar to the model where each bit is flipped with probability $q < \frac{(4h^2+2h)(1+\varepsilon)}{n}$. As we stated in the previous section, the bounds on the Hamming weight of $F \cdot B_1$ and $A \cdot G$, and also the bound in Theorem 3 are not tight, which means q will likely be sufficiently smaller than $\frac{4h^2+2h}{n}$.

Thus if we choose $n > 10h^2$, and we encode each bit b of the message $m \in \{0,1\}^k$ using a repetition code of length ρ (where $k \cdot \rho < n$) as $bb \cdots b \in \{0^\rho, 1^\rho\}$, then we expect the number of bits flipped to be smaller than $\rho/2$ with very high probability. Thus, we could decrypt correctly by looking at blocks of length ρ, and decode 1 if the number of 1s in this block of length ρ is more than $\rho/2$, and is 0, otherwise.

We analyzed the error probability when we choose $n = 756839$, $k = 256$, and $\rho = 2048$. At the present time, we are unable to provide a tight rigorous analysis of the decryption error probability. In order to give a satisfactory bound, we would need either to enlarge the parameters (as discussed in the previous section) of the scheme again or to replace the very simple repetition encoding that we are using by a more complex one. One very simple option would be to combine the repetition encoding with a random permutation of the bits of C_2 which are used to mask the encoded value at encryption time. This random permutation could be built from C_1 using the XOF provided by NIST. However, this would make the cryptosystem too slow and add an extra layer of complexity that is really undesirable.

Thus, we propose a heuristic analysis of the decryption error probability. This analysis is based on the distribution of the Hamming weights that are encountered in the decryption blocks corresponding to a single bit. Since, with our choice of parameters every bit is encoded into $\rho = 2048$ bits, we want to see how often a bit might cross the Hamming weight 1024 boundary. It is easy to equip the code and count the Hamming weights encountered during decryption. We performed experiments involving 10000 of each key generation, encapsulation and decapsulation in order to collect the distribution shown in Fig. 1. We see that the distribution looks like a superposition of two Gaussian distributions one corresponding to encryptions of a 0 and one to encryptions of a 1. Our heuristic assumption is that the probability of decryption failure is very close to the one corresponding to these Gaussian distributions. More precisely, we fitted a Gaussian G_0 corresponding to zeroes by searching for best fitting values of p and σ in:

$$G_0(x) = \frac{1}{2\sigma\sqrt{2\pi}} e^{-\frac{(x-p)^2}{2\sigma^2}}.$$

Note the extra 1/2 compared to a usual normal distribution. This is due to the fact that half of the encrypted bits are zeroes and half are ones. By symmetry, the Gaussian distribution corresponding to ones is simply $G_1(x) = G_0(\rho - x)$. We found that taking $p = 499.6$ and $\sigma = 28.64$ yields the very good approximation shown on Fig. 2 where the two Gaussian are superposed with the measured data.

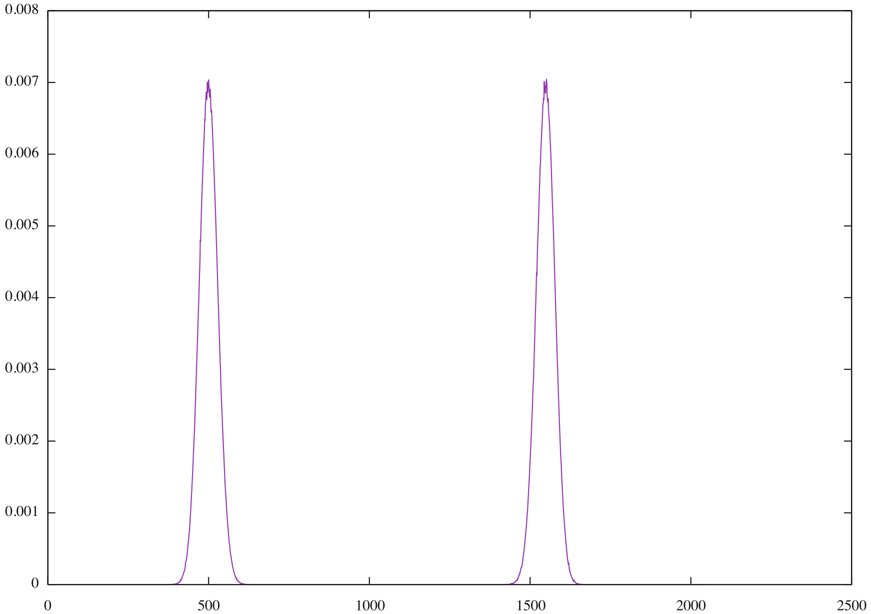

Fig. 1. Density distribution of Hamming Weights during decryption

As a consequence, the probability that a single bit crosses the 1024 boundary is approximated by:

$$0.5\,\mathrm{erfc}\left(\frac{1024 - p}{\sigma\sqrt{2}}\right) < 2^{-247}.$$

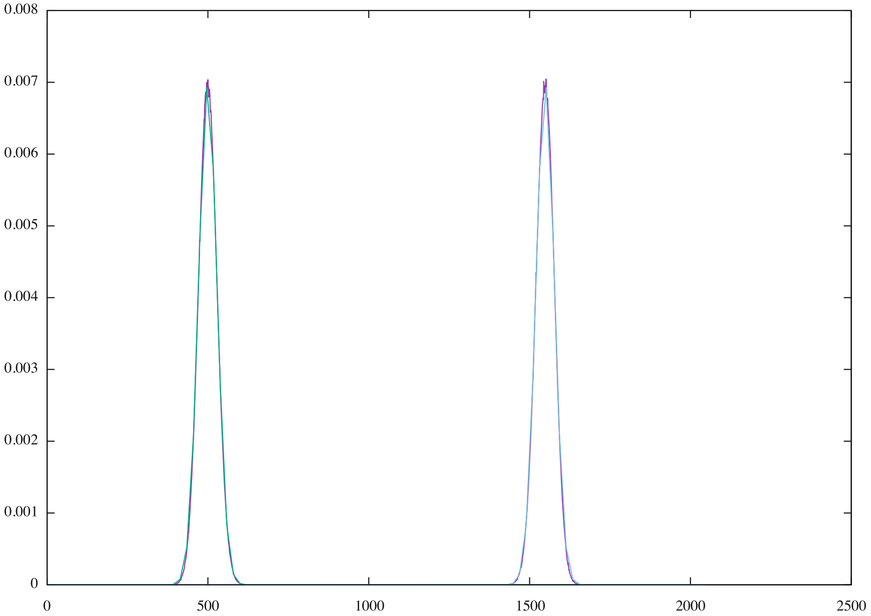

Fig. 2. Density distribution with fitted Gaussians

Since the encrypted value is formed of 256-bits, the overall probability of decryption failure can be heuristically upper bounded by 2^{-239}.

8.3 Further Efficiency Improvements

If we need to use repetition codes, we need n to be sufficiently large, say larger than $10h^2$, in order for the decryption error to be small. If we choose a smaller n (say $n \approx 4h^2$), and then use repetition codes and majority decoding as in the previous section, we expect that the Hamming distance between the message after decoding and the original message is small (but maybe non-zero). To get around this issue, we propose to modify our encoding procedure as follows. We encode a message $m \in \{0,1\}^k$ to a codeword $c_1 \in \{0,1\}^{n_1}$ using some efficient error correcting codes like BCH codes [MS77], and then encode c_1 to obtain a codeword $c_2 \in \{0,1\}^{n_2}$ with $n_2 \leq n$, using repetition codes, then the errors remaining after majority decoding can be corrected by decoding the modified BCH code. Notice that the choice of a smaller n does not alter the security of the scheme, since the security of the scheme depends on the parameter h.

Again we cannot rigorously analyse the decryption error probability, but we can obtain a heuristic analysis similar to the one in the previous section. Concretely, we obtain the following parameters.

Encoding $k = 256$ *bits, with* $n = 216091$. We can choose the next smaller Mersenne prime $p = 2^{216091} - 1$. In this case, we use a BCH code that encodes k-bit messages to $n_1 = 2^9 - 1 = 511$ bit messages. Each of these n_1 bits is encoded using a repetition code which repeats each bit 422 times. We again performed an experiment with 10000 key generation, encapsulation, and decapsulation and observed that the distribution of the Hamming weight for each 422 bits looks like a superposition of two Gaussian distributions one corresponding to encryptions of a 0 and one to encryptions of a 1. In particular, the Gaussian-like distribution corresponding to encryption of 1 has mean $\mu = 234.65$, and variance $\sigma^2 = 132.47$, and hence the probability of decoding a bit incorrectly under the heuristic assumption is

$$0.5\,\mathrm{erfc}(\frac{234.65 - 211}{\sigma\sqrt{2}}) < 0.02.$$

The BCH code corrects up to $\lfloor \frac{511-256}{9} \rfloor = 28$ errors [MS77]. Thus, assuming that the Hamming weight of each block of 422 bits is distributed independently, the probability that there is a decapsulation error is at most

$$\sum_{i=29}^{511} \binom{511}{i} \cdot 0.02^i \cdot 0.98^{511-i},$$

which can be estimated to be at most 2^{-25}.

Encoding $k = 256$ *bits, with* $n = 86243$*, and* $h = 128$. We cannot choose $n = 86243$ if $h = 256$, since n must be significantly larger than h^2 for the scheme to work. However, if we are willing to relax the security requirement to 128-bit security, then we can choose a much smaller Mersenne prime, and the scheme is extremely efficient. In particular, we can choose the Mersenne prime $p = 2^{86243} - 1$. Again, the BCH code encodes k-bit messages to $n_1 = 2^9 - 1 = 511$ bit messages. Each of these n_1 bits is encoded using a repetition code which repeats each bit 168 times. We again performed experiment with 10000 key generation, encapsulation, and decapsulation and again observed that the distribution of the Hamming weight for each 168 bits looks like a superposition of two Gaussian distributions one corresponding to encryptions of a 0 and one to encryptions of a 1. The mean and variance of this distribution are $\mu = 104.55$, and $\sigma^2 = 68.91$, respectively, and hence the probability of decoding a bit incorrectly under the heuristic assumption is

$$0.5\,\mathrm{erfc}(\frac{234.65 - 211}{\sigma\sqrt{2}}) < 0.005.$$

Thus, assuming that the Hamming weight of each block of 168 bits is distributed independently, the probability that there is a decapsulation error is at most

$$\sum_{i=29}^{511} \binom{511}{i} \cdot 0.005^i \cdot 0.995^{511-i},$$

which can be estimated to be at most 2^{-60}.

9 Proof of Theorem 3

Let x be an arbitrary n-bit string of Hamming weight Δ, for some positive integer Δ. We can decompose x as $x_1 + \ldots x_\Delta$ where for all $1 \leq i \leq \Delta$, the string x_i has Hamming distance 1 whose single 1 bit is in position j_i, and $j_1 < \ldots < j_\Delta$. Let $U = U^0$ be the random variable which takes an n-bit binary string with uniform distribution. For $1 \leq i \leq \Delta$, we define the random variables $U^i = U^{i-1} + x_i$ and $Y_i = \mathsf{Ham}(U, U^i) - \mathsf{Ham}(U, U^{i-1})$. The main result in this section is an upper bound the tail of the random variable measuring the Hamming distance of U and $U + x$, that is U and U^Δ.

Theorem 4. *Let U be a random variable having uniform distribution on strings of length n. For every n-bit string x of Hamming weight Δ and for every $\varepsilon > 0$,*

$$\Pr[\mathsf{Ham}(U, U + x) \geq 2(1 + \varepsilon)\Delta] \leq 2^{-2\Delta(\varepsilon - \ln(1+\varepsilon))}.$$

Observe from the Taylor series $\ln(1 + \varepsilon) = \varepsilon - \frac{\varepsilon^2}{2} + \frac{\varepsilon^3}{3} - \ldots$ that, for small ε, we can well approximate the right hand side of the above inequality by $2^{-\Delta\varepsilon^2}$.

Proof. The string U^h is constructed from U in Δ steps, where in every step we add a new string of Hamming weight 1 to the string obtained in the previous steps. Our first lemma bounds the tail of the random variable measuring the increase in the Hamming distance in one step.

Lemma 3. *For every n-bit string x of Hamming weight Δ, and for all integers $s, y_1, \ldots, y_{\Delta-1}$, we have*

$$\Pr[Y_\Delta \geq s | Y_1 = y_1, \ldots, Y_{\Delta-1} = y_{\Delta-1}] \leq \min\{1, 2^{-(s-1)}\}.$$

Proof. Observe that for $s \leq 1$ the statement is trivial, therefore we only consider $s \geq 2$. For every $r \geq s$, and for every Δ, we well determine $\Pr[Y_\Delta = r | Y_1, \ldots, Y_{\Delta-1} | Y_1 = y_1, \ldots, Y_{\Delta-1} = y_{\Delta-1}]$, and then we will sum up these values. Let $Z^{\Delta-1}$ denote the event $Y_1 = y_1, \ldots, Y_{\Delta-1} = y_{\Delta-1}$.

Since addition, Hamming distance and the uniform distribution are invariant under rotation, we can suppose without generality that $x_1 = 1$. Under the condition that U and x don't have a 1 in the same position, $Y_\Delta = 1$ with probability 1, and the statement follows. Therefore we can work under the condition that U and x have a common 1, and we can suppose, again without loss of generality, that $U_1 = 1$.

First we consider the case $\Delta = 1$. For $2 \leq r \leq n$, the random variable Y_1 is r when $U_r = 0$ and $U_{r-1} = \ldots = U_2 = 1$. Thus $\Pr[Y_1 = r] = 2^{-r+1}$.

We suppose now that $\Delta \geq 2$. We say that $i \leq \Delta$ is a wrap-around step if $U_n^{i-1} = 1$ and $U_n^i = 0$. Observe that in that case U_1^{i-1} and U_1^i are different. We define the random variable t_Δ as $n + 1$ if i is a wrap-around step for some

$1 \leq i \leq \Delta - 1$. Otherwise, let t_Δ be the smallest integer such that all but the first (from right) t_Δ bits are the same in U and $U^{\Delta-1}$. Since the single 1 bit in $x_{\Delta-1}$ is in position $j_{\Delta-1}$, it follows from the definition that $t_\Delta \geq j_{\Delta-1}$, and that $U^{\Delta-1}_{t_\Delta-1} = \ldots = U^{\Delta-1}_{j_{\Delta-1}} = 0$. In addition, if $t_\Delta \leq n$ then $U^{\Delta-1}_{t_\Delta} = 1$.

Case 1: $j_\Delta \leq t_\Delta - 1$. Since $j_\Delta > j_{\Delta-1}$, we have then $U^{\Delta-1}_{j_\Delta} = 0$. Therefore $U^{\Delta-1}$ and U^Δ differ only in one position, implying Y_Δ is never more than 1.

Case 2: $j_\Delta \geq t_\Delta$. Then $t_\Delta \leq n$ and therefore none of the previous steps was a wrap-around step. Thus $U^{\Delta-1}_1 = \ldots = U^1_1 = U_{t_\Delta} = 0$ and $U^{\Delta-1}_{t_\Delta} = 1$.

Step Δ is a wrap-around step when $U^{\Delta-1}_{j_\Delta} = \ldots = U^{\Delta-1}_n = 1$. When $j_\Delta > t_\Delta$, this is equivalent to $U_{j_\Delta} = \ldots = U_n = 1$, and happens with probability $2^{-(n-j_\Delta+1)}$. In that case $U^{\Delta-1}$ and U^Δ differ in positions $1, j_\Delta \ldots, n$. Among these positions U^Δ and U differ at $j_\Delta, \ldots, n$ but $U^\Delta_1 = U_1 = 1$, and therefore $Y_\Delta = n - j_\Delta$. When $j_\Delta = t_\Delta$, this is equivalent to $U_{j_\Delta+1} = \ldots = U_n = 1$, and happens with probability $2^{-(n-j_\Delta)}$. In that case $U^{\Delta-1}$ and U^Δ differ in positions $1, j_\Delta \ldots, n$. Among these positions U^Δ and U differ at $j_\Delta + 1, \ldots, n$ but $U^\Delta_1 = U_1 = 1$, and therefore $Y_\Delta = n - j_\Delta - 1$.

Step Δ is not a wrap-around step when $U^{\Delta-1}_\ell = 0$ and $U^{\Delta-1}_{j_\Delta} = \ldots = U^{\Delta-1}_{\ell-1} = 1$, for some $j_\Delta \leq \ell \leq n$. This never happens when $j_\Delta = n$. When $t_\Delta < j_\Delta < n$, this is equivalent to $U_\ell = 0$ and $U^{\Delta-1}_{j_\Delta} = \ldots = U^{\Delta-1}_{\ell-1} = 1$, which happens with probability $2^{-(\ell-j_\Delta+1)}$. In that case $U^{\Delta-1}$ and U^Δ differ in positions $j_\Delta, \ldots, \ell$, where $U^{\Delta-1}$ coincides with U and U^Δ differs from U, implying $Y_\Delta = \ell - j_\Delta + 1$. When $j_\Delta = t_\Delta$, ℓ must be at least $j_\Delta + 1$, and the condition is equivalent to $U_\ell = 0$ and $U^{\Delta-1}_{j_\Delta+1} = \ldots = U^{\Delta-1}_{\ell-1} = 1$, which happens with probability $2^{-(\ell-j_\Delta)}$. In that case $U^{\Delta-1}$ and U^Δ differ in positions $j_\Delta + 1, \ldots, \ell$, where $U^{\Delta-1}$ coincides with U and U^Δ differs from U, implying $Y_\Delta = \ell - j_\Delta$.

All together, $\Pr[Y_\Delta = r | Z^{\Delta-1}] \neq 0$ in the following cases. When $j_\Delta > t_\Delta$, we have $\Pr[Y_\Delta = r | Z^{\Delta-1}] = 2^{-r}$ for $r \in \{2, \ldots n - j_\Delta - 1, n - j_\Delta + 1\}$ and $\Pr[Y_\Delta = n - j_\Delta] = 2^{-(n-j_\Delta+1)} + 2^{-(n-j_\Delta)}$. When $j_\Delta = t_\Delta$, we have $\Pr[Y_\Delta = r | Z^{\Delta-1}] = 2^{-r}$ for $r \in \{2, \ldots n - j_\Delta - 2, n - j_\Delta\}$ and $\Pr[Y_\Delta = n - j_\Delta - 1 | Z^{\Delta-1}] = 2^{-(n-j_\Delta)} + 2^{-(n-j_\Delta-1)}$. The statement follows by summing up these probabilities for $r \geq s$. □

Observe that $\mathsf{Ham}(U, U + x) = Y_1 + \ldots + Y_\Delta$. In order to bound the tail of $Y_1 + \ldots + Y_\Delta$, we introduce the independent random variables $X_1, \ldots X_\Delta$, where X_i is a geometric random variable with success probability $\frac{1}{2}$, for each $1 \leq i \leq \Delta$. This means that by definition, for every positive integer r, we have $\Pr[X_i = r] = 2^{-r}$. The definition immediately implies that for every integer s, we also have $\Pr[X_i \geq s] = \min\{1, 2^{-(s-1)}\}$. Our next lemma states that the tail of $Y_1 + \ldots + Y_\Delta$ can be upper bounded by the tail of $X_1 + \ldots + X_\Delta$.

Lemma 4. *For every n-bit string x of Hamming weight Δ, for every integer s,*

$$\Pr[Y_1 + \ldots + Y_\Delta \geq s] \leq \Pr[X_1 + \ldots + X_\Delta \geq s].$$

Proof. We prove it by induction on Δ. When $\Delta = 1$, from Lemma 3 we have

$$\Pr[Y_1 \geq s] \leq \min\{1, 2^{-(s-1)}\} = \Pr[X_1 \geq s].$$

When $\Delta \geq 2$, we have the following series of (in)equalities:

$$\begin{aligned}
\Pr[\sum_{i=1}^{\Delta} Y_i \geq s] &\leq \sum_{y_1,\ldots,y_{\Delta-1}} \Pr[Y_1 = y_1, \ldots, Y_{\Delta-1} = y_{\Delta-1}] \Pr[X_\Delta \geq s - \sum_{i=1}^{\Delta-1} y_i] \\
&= \Pr[\sum_{i=1}^{\Delta-1} Y_i + X_\Delta \geq s] \\
&= \sum_y \Pr[\sum_{i=1}^{\Delta-1} Y_i \geq y] \Pr[X_\Delta = s - y] \\
&\leq \sum_y \Pr[\sum_{i=1}^{\Delta-1} X_i \geq y] \Pr[X_\Delta = s - y] \\
&= \Pr[\sum_{i=1}^{\Delta} X_i \geq s].
\end{aligned}$$

The first inequality follows from Lemma 3 and the second inequality from the inductive hypothesis. For the third equality we have used that X_Δ is independent from the random variables Y_i. □

Our final lemma is a special case of Theorem 2.3 in the artice [Jan17] on tail bounds for sums of geometric and exponential variables.

Lemma 5 *[Jan17]. Let $X_1, \ldots X_\Delta$ be independent geometric random variables with success probability $\frac{1}{2}$, and let $\varepsilon > 0$. Then*

$$\Pr[\sum_{i=1}^{\Delta} X_i \geq 2(1+\varepsilon)\Delta] \leq 2^{-2\Delta(\varepsilon - \ln(1+\varepsilon))}.$$

Putting together Lemmas 4 and 5, we immediately obtain our bound on the Hamming distance of U and U_Δ, which concludes the proof. □

10 Conclusion

In this paper, we propose a simple new public-key encryption scheme. As with other public-key cryptosystems, the security of our cryptosystem relies on unproven assumptions mentioned in Definition 5. In Sect. 6.1, we summarized the known cryptanalytic attacks against this scheme. The proposed cryptosystem is based on a relatively new assumption, and it will require more cryptanalytic effort before one can be reasonably confident about the security assumption.

Acknowledgments. This research was partially funded by the Singapore Ministry of Education and the National Research Foundation, also through the Tier 3 Grant "Random numbers from quantum processes", MOE2012-T3-1-009. This work has been supported in part by the European Union's H2020 Programme under grant agreement number ERC-669891 and the French ANR Blanc program under contract ANR-12-BS02-005 (RDAM project). The second author is grateful to CQT where the work has started during his visit.

References

[AJPS17] Aggarwal, D., Joux, A., Prakash, A., Santha, M.: A new public-key cryptosystem via mersenne numbers. Cryptology ePrint Archive, Report 2017/481, version:20170530.072202 (2017)

[BCGN17] Beunardeau, M., Connolly, A., Géraud, R., Naccache, D.: On the hardness of the Mersenne Low Hamming Ratio assumption. Technical report, Cryptology ePrint Archive, 2017/522 (2017)

[CHK10] Cramer, R., Hofheinz, D., Kiltz, E.: A twist on the Naor-Yung paradigm and its application to efficient CCA-secure encryption from hard search problems. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 146–164. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11799-2_10

[dBDJdW17] de Boer, K., Ducas, L., Jeffery, S., de Wolf, R.: Attacks on the AJPS mersenne-based cryptosystem. Technical report, Cryptology ePrint Archive, Report 2017/1171 (2017). https://eprint.iacr.org/2017/1171

[DH76] Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976)

[ES16] Eldar, L., Shor, P.W.: An efficient quantum algorithm for a variant of the closest lattice-vector problem. arXiv preprint arXiv:1611.06999 (2016)

[Gro96] Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, pp. 212–219 (1996)

[GWO+13] Lize, G., Wang, L., Ota, K., Dong, M., Cao, Z., Yang, Y.: New public key cryptosystems based on non-abelian factorization problems. Secur. Commun. Netw. **6**(7), 912–922 (2013)

[HHK17] Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki-Okamoto transformation. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. LNCS, vol. 10677, pp. 341–371. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70500-2_12

[HPS98] Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: a ring-based public key cryptosystem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 267–288. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054868

[Jan17] Janson, S.: Tail bounds for sums of geometric and exponential variables. arXiv preprint arXiv:1709.08157 (2017)

[KLC+00] Ko, K.H., Lee, S.J., Cheon, J.H., Han, J.W., Kang, J., Park, C.: New public-key cryptosystem using braid groups. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 166–183. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44598-6_10

[LPR10] Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_1

[LvTMW09] Lempken, W., van Tran, T., Magliveras, S.S., Wei, W.: A public key cryptosystem based on non-abelian finite groups. J. Cryptol. **22**(2), 62–74 (2009)

[McE78] McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. Coding Thv **4244**, 114–116 (1978)

[MS77] MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes. Elsevier, New York (1977)

[MTSB13] Misoczki, R., Tillich, J.-P., Sendrier, N., Barreto, P.S.: MDPC-McEliece: new McEliece variants from moderate density parity-check codes. In: 2013 IEEE International Symposium on Information Theory Proceedings (ISIT), pp. 2069–2073. IEEE (2013)

[NIS17] NIST. Post quantum crypto project (2017). http://csrc.nist.gov/groups/ST/post-quantum-crypto/. Accessed 19 May 2017

[NS97] Naccache, D., Stern, J.: A new public-key cryptosystem. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 27–36. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-69053-0_3

[NY90] Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing, STOC 1990, pp. 427–437. ACM, New York (1990)

[OP01] Okamoto, T., Pointcheval, D.: REACT: rapid enhanced-security asymmetric cryptosystem transform. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 159–174. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45353-9_13

[Reg09] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM **56**(6), 34, 40 (2009)

[RSA78] Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM **21**(2), 120–126 (1978)

[RSN+01] Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E.: A statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, DTIC Document (2001)

[Sho97] Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**(5), 1484–1509 (1997)

[Sho02] Shoup, V.: OAEP reconsidered. J. Cryptol. **15**(4), 223–249 (2002)