

# A New Public-Key Cryptosystem via Mersenne Numbers

Divesh Aggarwal\*    Antoine Joux†    Anupam Prakash‡    Miklos Santha§

November 30, 2017

## 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  $T, R$ , find two  $n$ -bit integers  $F, G$  each of Hamming weight at most  $h$  such that  $T = F \cdot R + G$  modulo  $p$ , under the promise that they exist.

## 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; it remains the most popular scheme to date.

Shor [Sho97] gave a quantum algorithm that solves the abelian hidden subgroup problem and as a result solves both discrete logarithms and factoring. At the time, this was not considered a real threat against practical cryptographic schemes since quantum computers were far from being a reality. However, given the recent advances in quantum technologies, 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].

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

---

\*School of Computing and CQT, NUS.

†Chaire de Cryptologie de la Fondation de l'UPMC; Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 UMR 7606, Paris, France

‡School of Physical and Mathematical Sciences, Nanyang Technological University and Centre for Quantum Technologies, National University of Singapore, Singapore.

§IRIF, Université Paris Diderot, CNRS, 75205 Paris, France; and Centre for Quantum Technologies, National University of Singapore Singapore 117543

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 other promising proposals for public-key cryptosystems.

It is worthwhile to note that even though the concept of public-key cryptography was introduced four decades ago, creating public-key cryptographic schemes is a difficult task and until recently, very few proposals were able to withstand cryptanalysis. Moreover, in practice, systems based on factoring and discrete logarithms — two problems which collapse in front of quantum computers — dominated the market. The threat posed by quantum computers is reviving the search for new cryptosystems. Assessing the true security of such systems is a major cryptanalytic task for the coming years.

## 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 a simple and 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  $T$  appears pseudorandom, i.e., it seems 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  $\mathcal{E} : \{0, 1\}^k \rightarrow \{0, 1\}^n$  and decoding function  $\mathcal{D} : \{0, 1\}^n \rightarrow \{0, 1\}^k$ . 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  $m$  with high probability. For more details on our scheme and the underlying security assumption, we refer the reader to Section 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 smallness needs to be somewhat preserved under the arithmetic operations. 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, using only an elementary mathematical structure. 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.

Our first proposal using this structure [AJPS17] only allowed us to encrypt a single bit at a time. This inefficiency forced us to choose parameters that turned out to be vulnerable [BCGN17, dBDJdW17]. In the present proposal, we describe a variation that permits us to encrypt many bits at a time. This allows in turn to choose much larger parameters which resist the attacks in [BCGN17] and [dBDJdW17], even in their Groverized quantum form. The best quantum attack mentioned in these articles has a complexity of the order  $2^h$  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 cryptosystems of this type can be easily vulnerable to chosen-ciphertext attacks, 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 section 2 we introduce some preliminaries about Mersenne primes and security definitions. In section 3 we provide a semantically secure basic bit by bit encryption scheme. In section 4 we give a semantically secure blockwise encryption scheme. In section 5, we prove the semantic security for the scheme presented in section 4. In section 6 we discuss the known cryptanalytic attacks against this scheme. In section 7 we give the final key encapsulation scheme secure against chosen ciphertext attacks in the random oracle model. In sections 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  $\delta$ , then there is a probabilistic polynomial time distinguisher  $D'$  that distinguishes between  $X$  and  $Y$  with advantage  $\delta$ .*

## 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  $\text{Ham}(y)$ . Let  $\text{seq} : \mathbb{Z}_p \rightarrow \{0, 1\}^n$  be the map which to  $x \in \mathbb{Z}_p$  associates the binary string  $\text{seq}(x)$  representing the representative of  $x$  in the interval  $[0, p - 1]$ . The map  $\text{int} : \{0, 1\}^n \rightarrow \mathbb{Z}_p$  sends a string  $y$  into the integer represented by  $y$  modulo  $p$ . Clearly  $\text{seq}$  and  $\text{int}$  are inverse bijections between  $\mathbb{Z}_p$  and  $\{0, 1\}^n \setminus \{1^n\}$ , and  $\text{int}(1^n) = 0$ . 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  $y + y' = \text{seq}(\text{int}(y) + \text{int}(y'))$ , and let  $y \cdot y' = \text{seq}(\text{int}(y) \cdot \text{int}(y'))$ . 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  $(-1) \cdot y = -y = \text{seq}(-(\text{int}(y)))$ . Observe that addition is invariant by rotation, that is if  $\text{rot}_k(y)$  denotes the circular rotation of  $y$  by  $k$  positions to the left, then  $\text{rot}_k(y + y') = \text{rot}_k(y) + \text{rot}_k(y')$ .

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

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

*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  $\text{Ham}(B) = 1$ . 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  $\text{Ham}(A + B) = \text{Ham}(A) - j + 1 \leq \text{Ham}(A) + 1$ .

Let  $\text{Ham}(B) = k > 1$ . Then we can decompose  $B$  as  $B_1 + B_2$ , where  $\text{Ham}(B_1) = k - 1$  and  $\text{Ham}(B_2) = 1$ . By the previous claim and the induction hypothesis we get:

$$\text{Ham}(A + B) = \text{Ham}((A + B_1) + B_2) \leq \text{Ham}(A + B_1) + 1 \leq \text{Ham}(A) + (k - 1) + 1,$$

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  $B_1 + \dots + B_k$ , 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  $A \cdot B_i = \text{rot}_{j_i-1}(A)$ . Thus  $\text{Ham}(A \cdot B_i) = \text{Ham}(A)$ , and by distributivity we get  $A \cdot B = A \cdot B_1 + \dots + A \cdot B_k$ . 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

### 2.2.1 Public-Key Encryption

A public key encryption scheme comprises three algorithms: the key generation algorithm  $\text{KeyGen}$ , the encryption algorithm  $\text{Enc}$ , and the decryption algorithm  $\text{Dec}$ . The  $\text{KeyGen}$  algorithm outputs a public-key  $\text{pk}$ , and a secret key  $\text{sk}$ . The encryption algorithm  $\text{Enc}$  takes as input a message  $m$ , and  $\text{pk}$ , and outputs a ciphertext  $C$ . The decryption algorithm takes as input a ciphertext  $C$  and  $\text{sk}$ , 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  $m$ ,  $\Pr[\text{Dec}(\text{sk}, \text{Enc}(\text{pk}, m)) = m] \geq 1 - \delta$ , where the probability is over the randomness of  $\text{pk}, \text{sk}$  and the encryption algorithm.

We denote the security parameter by  $\lambda$ . All other parameters including key lengths and ciphertext size are given as polynomially bounded functions of  $\lambda$ .

**Definition 1.** The public-key encryption scheme  $PKE = (\text{KeyGen}, \text{Enc}, \text{Dec})$  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  $\text{pk}$ , the advantage for distinguishing  $C_0 = \text{Enc}(\text{pk}, m_0)$  and  $C_1 = \text{Enc}(\text{pk}, m_1)$  is at most  $\frac{\text{poly}(|C_i|)}{2^\lambda}$  for some polynomial  $\text{poly}$ .

**Definition 2.** The public-key encryption scheme  $PKR = (\text{KeyGen}, \text{Enc}, \text{Dec})$  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  $\text{pk}$ , the advantage for distinguishing  $C_0 = \text{Enc}(\text{pk}, m_0)$  and  $C_1 = \text{Enc}(\text{pk}, m_1)$  is at most  $\frac{\text{poly}(|C_i|)}{2^\lambda}$  for some polynomial  $\text{poly}$  under the assumption that the distinguisher does not query the oracle with  $C_0$  or  $C_1$ .

*Note:* As usual, these are asymptotic definitions of security for a family of system when  $\lambda$  goes to infinity. It is also useful to consider concrete definitions of security when giving explicit instances of a cryptosystem. One difficulty is that concrete security definitions lose the (classical) computation model independence that is provided by the asymptotic definition thanks to the polynomial factors. Moreover, concrete definitions easily allow for non uniform adversaries which are excluded from the asymptotic ones. Basically, the requirement is that, in a given computing model, any adversary with running time at most  $T$ , memory (including code size) at most  $M$  and advantage  $\delta$  should satisfy:

$$\frac{TM}{\delta} \leq 2^\lambda.$$

For simplicity of presentation, through the rest of the paper, we stick to the asymptotic security definitions.

### 2.2.2 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  $\mathbf{pk}$ , and a secret key  $\mathbf{sk}$ . The encapsulation algorithm `Encaps` takes as input a public key  $\mathbf{pk}$  to produce a ciphertext  $C$  and a key  $K \in \mathcal{K}$ . The decapsulation algorithm `Decaps` takes as input a ciphertext  $C$  and  $\mathbf{sk}$ , and outputs a key  $K'$  or a special symbol  $\perp$  indicating rejection. We say that the KEM is  $(1 - \delta)$ -correct if

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

where the probability is over the randomness of  $\mathbf{pk}, \mathbf{sk}$  and the encapsulation algorithm.

Again, we denote the security parameter by  $\lambda$ . All other parameters including key lengths and ciphertext size are given as polynomially bounded functions of  $\lambda$ .

**Definition 3.** The key-encapsulation mechanism  $KEM = (\text{KeyGen}, \text{Encaps}, \text{Decaps})$  is said to be *semantically secure* if for any probabilistic polynomial time distinguisher, given the public key  $\mathbf{pk}$ , the advantage for distinguishing  $(C, K_0)$  and  $(C, K_1)$ , where  $(C, K_0) \leftarrow \text{Encaps}(\mathbf{pk})$  and  $K_1$  is uniform and independent of  $C$  is at most  $\frac{\text{poly}(|C|, |K_0|)}{2^\lambda}$  for some polynomial  $\text{poly}$ .

**Definition 4.** The key-encapsulation mechanism  $KEM = (\text{KeyGen}, \text{Encaps}, \text{Decaps})$  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  $\mathbf{pk}$ , the advantage for distinguishing  $(C, K_0)$  and  $(C, K_1)$ , where  $(C, K_0) \leftarrow \text{Encaps}(\mathbf{pk})$  and  $K_1$  is uniform and independent of  $C$  is at most  $\frac{\text{poly}(|C|, |K_0|)}{2^\lambda}$  for some polynomial  $\text{poly}$  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  $h$  such that  $4h^2 < n \leq 16h^2$ , the advantage of any probabilistic polynomial time adversary running in time  $\text{poly}(n)$  in attempting to distinguish between

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

is at most  $O(2^{-h})$ , where  $R_1, R_2, R_3, R_4$  are independent and uniformly random  $n$ -bit strings, and  $A, B$ , 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  $(-R_1, A \cdot R_1 + B)$ . Since for a random  $R_1$ ,  $-R_1$  is also random, 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.

Informally, the *Mersenne Low Hamming Ratio Assumption* states that given an  $n$ -bit Mersenne prime  $p = 2^n - 1$ , and an integer  $h$ , it is difficult to distinguish between the quotient  $\text{seq}(\frac{\text{int}(A)}{\text{int}(B)})$  of two independently chosen  $n$ -bit strings  $A$  and  $B$  of Hamming weight  $h$  and a uniformly random  $n$ -bit string.

### 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  $\lambda$ , choose a Mersenne prime  $p = 2^n - 1$  and an integer  $h$  such that  $\binom{n}{h} \geq 2^\lambda$  and  $4h^2 < n \leq 16h^2$ .
- Choose  $F, G$  to be two independent  $n$ -bit strings chosen uniformly at random from all  $n$ -bit strings of Hamming weight  $h$ .
- Set  $\text{pk} := H = \text{seq}(\frac{\text{int}(F)}{\text{int}(G)})$ , and  $\text{sk} := G$ .

**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

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

**Decryption.** The decryption algorithm computes  $d = \text{Ham}(C \cdot G)$ . 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.

In the next section, we describe a scheme for encrypting longer message blocks at once.

### 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  $\lambda$ , choose a Mersenne prime  $p = 2^n - 1$  such that  $h = \lambda$  and  $16h^2 \geq n > 10h^2$ .
- 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  $\text{pk} := (R, F \cdot R + G) := (R, T)$ , and  $\text{sk} := F$ .

**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,

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

Here  $\mathcal{E} : \{0, 1\}^\lambda \rightarrow \{0, 1\}^n$  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  $\Pr_{(C_1, C_2) \leftarrow \text{Enc}(\text{pk}, m)}[\mathcal{D}((F \cdot C_1) \oplus C_2) = m] \geq 1 - \delta$ , where the probability is over the randomness of the encryption algorithm and the choice of  $\text{pk}, \text{sk}$ . For concrete instantiations of error-correcting codes that satisfy this for a small enough  $\delta$ , see Section 8.

## 5 Semantic Security of the Cryptosystem

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

**Theorem 1.** *The encryption scheme  $(\text{Enc}, \text{Dec})$  described in Section 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  $T = F \cdot R + G$ .

In the course of this proof, we are using many polynomial functions. In order to lighten the notation, all of them are denoted by  $\text{poly}$ . However, it should be understood that all these functions are possibly distinct and the reader should mentally replace them by  $\text{poly}_1, \text{poly}_2, \dots$

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

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

This implies that for any message  $m$ ,

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

since  $R, R', R'', R'''$  and  $R, R', R'', R''' \oplus \mathcal{E}(m)$  are identically distributed. This implies the required semantic security. The final inequality comes from the fact that  $n$  is polynomial in  $\lambda$  in our choice of parameters.  $\square$

## 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 6** (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  $\text{Ham}(R, R')$  is close to the distribution of  $\text{Ham}(R, T)$  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  $f_R(X) = \frac{\text{Ham}(X, R) - n/2}{\sqrt{n/4}}$  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<sup>+</sup>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  $H = \text{seq}(\frac{\text{int}(F)}{\text{int}(G)}) \bmod P$  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  $H/P$ .

**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 decomposition 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 Section 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. As a consequence, the Groverized version would require to perform lattice reduction at the bottom of Grover's to implement it. 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 cryptanalytic 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 Section 8.

**Meet in the middle attack.** In [dBDJdW17], an efficient meet-in-the-middle attack that makes use of locality-sensitive hash functions is also mentioned. 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  $Y = FR + G \pmod q$ , 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 Section 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 section 6.2 that the semantically secure cryptosystem described in Section 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 Section 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  $\text{Enc}, \text{Dec}$  be the encryption and decryption algorithms as defined in Section 4. In addition to this, our transformation uses three random oracles  $\mathcal{H}_1, \mathcal{H}_2$  and  $\mathcal{H}_3$  that all take as input a  $\lambda$ -bit string and output a uniformly random string of Hamming weight  $h$  on  $n$  bits. It is quite easy to construct instantiation for  $\mathcal{H}_1, \mathcal{H}_2$  and  $\mathcal{H}_3$  using an expandable hash function, we omit the details here.

**Key Generation.** The key generation is identical to the semantically secure cryptosystem and produces  $\text{pk} := R, T := F \cdot R + G$ , and  $\text{sk} := F$  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  $\text{pk} = (R, T)$ , the algorithm  $\text{Encaps}$  proceeds as follows:

1. Pick a uniformly random  $\lambda$ -bit string  $K$ .
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. Output  $C, K$ .

**Decapsulation.** Given a ciphertext  $C = (C_1, C_2)$ , and  $\text{sk} = F$ , the decapsulation algorithm  $\text{Decaps}$  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  $K'$ , else output  $\perp$ .

**Theorem 2.** *Assume that  $\mathcal{H}$  is a random oracle and that the scheme from Section 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  $C^* = (C_1^*, C_2^*)$ . 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  $C^* = (C_1^*, C_2^*)$  computes  $K^* = \mathcal{D}((F \cdot C_1^*) \oplus C_2^*)$ , and then calls the encapsulation algorithm with input  $\tilde{K}$  to obtain  $\tilde{C} = (\tilde{C}_1, \tilde{C}_2)$ . If  $\tilde{C} = C^*$ , then the oracle outputs  $\tilde{K}$ , and the oracle outputs  $\perp$  otherwise.

If the random oracle was previously queried with the seed  $\tilde{K}$  by the adversary, then since the encapsulation procedure is a deterministic function of  $\tilde{K}$ , 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  $\tilde{K}$ , then we have that  $\tilde{C}_1 = A \cdot R + B_1$ , where  $A = \mathcal{H}_1(\tilde{K})$  and  $B_1 = \mathcal{H}_2(\tilde{K})$ . Since  $\mathcal{H}_2$  is a random oracle,  $A, B$  are assumed to be independent of everything else, and hence the probability that the decapsulation oracle does not output  $\perp$  is at most

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

□

## 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 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 Section 6, the best known efficient attack on our cryptosystem succeeds runs in time more than  $O(2^{2h})$ . We assume, somewhat conservatively, that even with future advancements in cryptanalysis of our scheme and quantum computers, the running time cannot be improved beyond  $O(2^h)$ . Under this assumption, we set  $h$  to be the security parameter  $\lambda$ . 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. Moreover, we want an error corrector code allowing us to encode  $k$  bits, with  $k$  close to 256.

### 8.1 Instantiation based on Deterministic Error-Correction Codes

We also need the following result. it is proved in Section 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  $\Delta$  and for every  $\varepsilon > 0$ ,*

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

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  $\varepsilon \in (0, 1)$ ,

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

and

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

Using union bound, and triangle inequality, we get that

$$\begin{aligned} \Pr[\text{Ham}(F \cdot (A \cdot R + B_1), A \cdot (F \cdot R + G) + B_2) \geq (4h^2 + 2h)(1 + \varepsilon)] &\leq 2^{-(2h^2 - 1)(\varepsilon - \ln(1 + \varepsilon))} \\ &\leq 2^{-(2h^2 - 1)(\varepsilon^2/2 - \varepsilon^3/3)} \\ &\leq 2^{-(2h^2 - 1)(\varepsilon^2/6)}, \end{aligned}$$

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  $\varepsilon$  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  $n \geq 2^{22} = 64h^2$ , and using Reed-Muller Codes [HCA<sup>+</sup>10], we can encode a message of length

$$k = \binom{22}{0} + \binom{22}{1} + \binom{22}{2} = 254$$

just two bits short of 256 and corrects up to  $\frac{n}{8} > 8h^2$  errors. Thus, choosing  $\varepsilon = \frac{8h^2}{4h^2 + 2h} - 1$  gives an instantiation of our scheme with decryption error as low as  $2^{-h^2/4}$ .

One can choose more sophisticated error correcting codes like dual BCH codes to obtain a construction with small decryption error and a smaller  $n$ . However, this will likely still be far from optimal. 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, which could perhaps lead to even better error correction as we discuss in the next section.

## 8.2 Instantiation based on Repetition Codes

In the previous section we considered Reed Muller 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  $\rho$  (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  $\rho$ , and decode 1 if the number of 1s in this block of length  $\rho$  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 (say below  $2^{-128}$ ), 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. However, implementing an instantiation of this random permutation 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

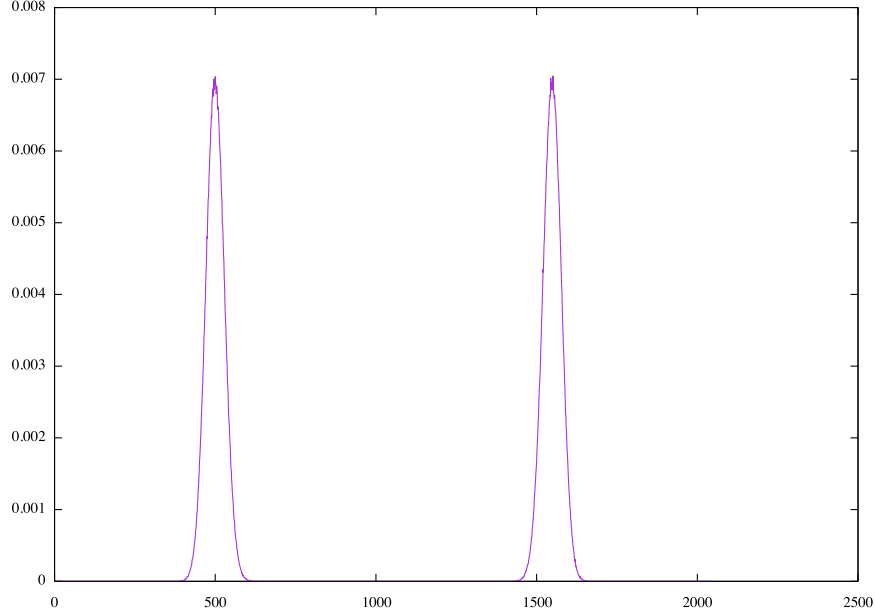


Figure 1: Density distribution of Hamming Weights during decryption

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 Figure 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  $\sigma$  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 Figure 2, where the two Gaussian are superposed with the measured data.

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

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

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

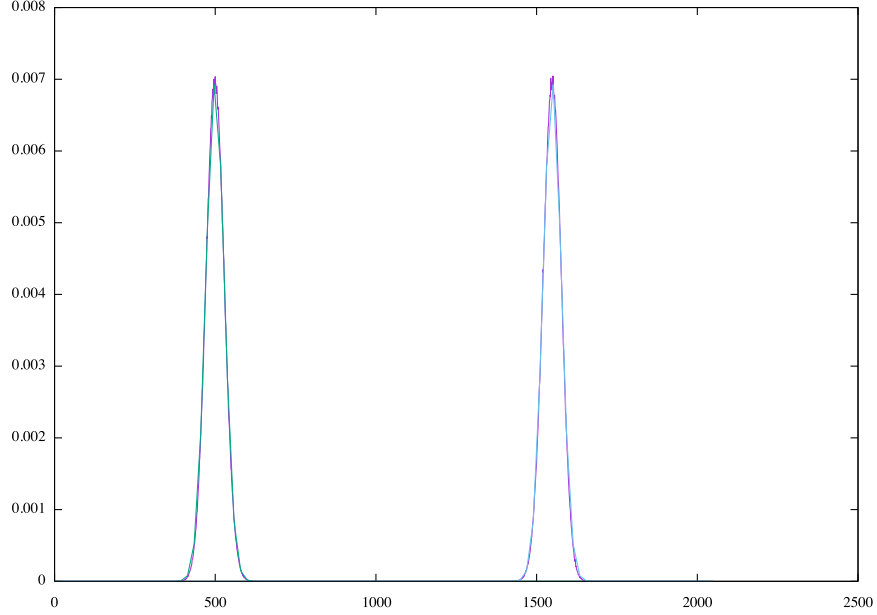


Figure 2: Density distribution with fitted Gaussians

## 9 Proof of Theorem 3

Let  $x$  be an arbitrary  $n$ -bit string of Hamming weight  $\Delta$ , for some positive integer  $\Delta$ . We can decompose  $x$  as  $x_1 + \dots + x_\Delta$  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  $j_1 < \dots < j_\Delta$ . 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  $Y_i = \text{Ham}(U, U^i) - \text{Ham}(U, U^{i-1})$ . 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  $U^\Delta$ .

We recall that we want to prove Theorem 3, namely:

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

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

*Proof.* The string  $U^\Delta$  is constructed from  $U$  in  $\Delta$  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  $\Delta$ , for all integers  $s, y_1, \dots, y_{\Delta-1}$ , we have*

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



*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  $\Delta$ , we will determine  $\Pr[Y_\Delta = r | Y_1, \dots, Y_{\Delta-1} | Y_1 = y_1, \dots, Y_{\Delta-1} = y_{\Delta-1}]$ , and then we will sum up these values. Let  $Z^{\Delta-1}$  denote the event  $Y_1 = y_1, \dots, Y_{\Delta-1} = y_{\Delta-1}$ .

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  $U_{r-1} = \dots = U_2 = 1$ . 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  $t_\Delta$  as  $n+1$  if  $i$  is a wrap-around step for some  $1 \leq i \leq \Delta-1$ . Otherwise, let  $t_\Delta$  be the smallest integer such that all but the first (from right)  $t_\Delta$  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  $U_{t_\Delta-1}^{\Delta-1} = \dots = U_{j_\Delta-1}^{\Delta-1} = 0$ . In addition, if  $t_\Delta \leq n$  then  $U_{t_\Delta}^{\Delta-1} = 1$ .

**Case 1:**  $j_\Delta \leq t_\Delta - 1$ . Since  $j_\Delta > j_{\Delta-1}$ , we have then  $U_{j_\Delta}^{\Delta-1} = 0$ . Therefore  $U^{\Delta-1}$  and  $U^\Delta$  differ only in one position, implying  $Y_\Delta$  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  $U_1^{\Delta-1} = \dots = U_1^1 = U_{t_\Delta} = 0$  and  $U_{t_\Delta}^{\Delta-1} = 1$ .

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

Step  $\Delta$  is not a wrap-around step when  $U_\ell^{\Delta-1} = 0$  and  $U_{j_\Delta}^{\Delta-1} = \dots = U_{\ell-1}^{\Delta-1} = 1$ , 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  $U_{j_\Delta}^{\Delta-1} = \dots = U_{\ell-1}^{\Delta-1} = 1$ , which happens with probability  $2^{-(\ell-j_\Delta+1)}$ . In that case  $U^{\Delta-1}$  and  $U^\Delta$  differ in positions  $j_\Delta, \dots, \ell$ , where  $U^{\Delta-1}$  coincides with  $U$  and  $U^\Delta$  differs from  $U$ , implying  $Y_\Delta = \ell - j_\Delta + 1$ . When  $j_\Delta = t_\Delta$ ,  $\ell$  must be at least  $j_\Delta + 1$ , and the condition is equivalent to  $U_\ell = 0$  and  $U_{j_\Delta+1}^{\Delta-1} = \dots = U_{\ell-1}^{\Delta-1} = 1$ , which happens with probability  $2^{-(\ell-j_\Delta)}$ . In that case  $U^{\Delta-1}$  and  $U^\Delta$  differ in positions  $j_\Delta + 1, \dots, \ell$ , where  $U^{\Delta-1}$  coincides with  $U$  and  $U^\Delta$  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  $r \in \{2, \dots, n - j_\Delta - 1, n - j_\Delta + 1\}$  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  $r \in \{2, \dots, n - j_\Delta - 2, n - j_\Delta\}$  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  $r \geq s$ .  $\square$

Observe that  $\text{Ham}(U, U+x) = Y_1 + \dots + Y_\Delta$ . In order to bound the tail of  $Y_1 + \dots + Y_\Delta$ , we introduce the independent random variables  $X_1, \dots, X_\Delta$ , 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  $Y_1 + \dots + Y_\Delta$  can be upper bounded by the tail of  $X_1 + \dots + X_\Delta$ .

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

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

*Proof.* We prove it by induction on  $\Delta$ . 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:

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

The first inequality follows from Lemma 3 and the second inequality from the inductive hypothesis. For the third equality we have used that  $X_\Delta$  is independent from the random variables  $Y_i$ .  $\square$

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

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

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

Putting together Lemmas 4 and 5, we immediately obtain our bound on the Hamming distance of  $U$  and  $U_\Delta$ , which concludes the proof.  $\square$

## 10 Conclusion

In this paper, we propose a simple new public-key encryption scheme. As with many other public-key cryptosystems, the security of our cryptosystem relies on an unproven assumption introduced in Definition 5. In Section 6.1, we summarized the known cryptanalytic attacks against this scheme. Since the proposed cryptosystem is based on a relatively new assumption, it requires more cryptanalytic effort before one can become reasonably confident about its security.

## 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] Divesh Aggarwal, Antoine Joux, Anupam Prakash, and Miklos Santha. A new public-key cryptosystem via Mersenne numbers. *Cryptology ePrint Archive, Report 2017/481, version:20170530.072202*, 2017.
- [BCGN17] Marc Beunardeau, Aisling Connolly, Rémi Géraud, and David Naccache. On the hardness of the Mersenne Low Hamming Ratio assumption. Technical report, Cryptology ePrint Archive, 2017/522, 2017.
- [CHK10] Ronald Cramer, Dennis Hofheinz, and Eike Kiltz. A twist on the Naor-Yung paradigm and its application to efficient CCA-secure encryption from hard search problems. In *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, pages 146–164, 2010.
- [dBDJdW17] Koen de Boer, Léo Ducas, Stacey Jeffery, and Ronald de Wolf. Attacks on the ajps cryptosystem. (Personal communication of the preprint), November 2017.
- [DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.
- [ES16] Lior Eldar and Peter W. Shor. An efficient quantum algorithm for a variant of the closest lattice-vector problem. *arXiv preprint arXiv:1611.06999*, 2016.
- [Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.
- [HCA<sup>+</sup>10] Prahladh Harsha, Moses Charikar, Matthew Andrews, Sanjeev Arora, Subhash Khot, Dana Moshkovitz, Lisa Zhang, Ashkan Aazami, Dev Desai, Igor Gorodezky, et al. Limits of approximation algorithms: Pcps and unique games (dimacs tutorial lecture notes). *arXiv preprint arXiv:1002.3864*, 2010.
- [HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph Silverman. NTRU: A ring-based public key cryptosystem. *Algorithmic number theory*, pages 267–288, 1998.

- [Jan17] Svante Janson. Tail bounds for sums of geometric and exponential variables. *arXiv preprint arXiv:1709.08157*, 2017.
- [LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 1–23. Springer, 2010.
- [MTSB13] Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo Barreto. MDPC-McEliece: New McEliece variants from moderate density parity-check codes. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 2069–2073. IEEE, 2013.
- [NIS17] NIST. Post quantum crypto project. <http://csrc.nist.gov/groups/ST/post-quantum-crypto/>, 2017. Accessed: 2017-05-19.
- [NY90] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing, STOC '90*, pages 427–437, New York, NY, USA, 1990. ACM.
- [OP01] Tatsuaki Okamoto and David Pointcheval. REACT: rapid enhanced-security asymmetric cryptosystem transform. In *Topics in Cryptology - CT-RSA 2001, The Cryptographer's Track at RSA Conference 2001, San Francisco, CA, USA, April 8-12, 2001, Proceedings*, pages 159–175, 2001.
- [Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 56(6):Art. 34, 40, 2009.
- [RSA78] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
- [RSN<sup>+</sup>01] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, and Elaine Barker. A statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, DTIC Document, 2001.
- [Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM journal on computing*, 26(5):1484–1509, 1997.
- [Sho02] Victor Shoup. OAEP reconsidered. *J. Cryptology*, 15(4):223–249, 2002.