New Public-Key Crypto-System EHT

Alessandro Budroni   Igor Semaev

Department of Informatics, University of Bergen, Norway

{alessandro.budroni, igor.semaev}@uib.no

March 1, 2021

Abstract

In this note, an LWE problem with a hidden trapdoor is introduced. It is used to construct an efficient public-key crypto-system EHT. The new system is significantly different from LWE based NIST candidates like FrodoKEM. The performance of EHT compares favorably with FrodoKEM.

1 Introduction

The LWE (Learning with Errors) problem was introduced by Regev in [13], where an LWE based public-key encryption was described. The problem was there proved to be hard assuming the hardness of computing shortest non-zero vectors in general lattices. Since then several lattice based public-key crypto-systems were invented, see [12]. The NIST Post-Quantum Standardization Process stimulated interest for developing new quantum computer resistant public-key protocols. A number of submission to this competition are LWE or Ring LWE based, see [11]. In this note an LWE problem with a hidden trapdoor is introduced. It is used to construct a new efficient public-key crypto-system EHT.

2 LWE problem

Let $n \leq m$ be positive integers and q be a prime, bounded by a fixed polynomial in n for large n. Let A be an integer $m \times n$-matrix of rank n modulo q and $e = (e_1, \ldots, e_m)$ be a column vector with entries generated independently according to a non-uniform distribution

$$p[0], p[1], \ldots, p[q-1] \tag{1}$$

on residues modulo q. Also let $x = (x_1, \ldots, x_n)$ be a column vector of integers and $Ax - e \equiv y \mod q$, where $y = (y_1, \ldots, y_m)$. The problem is to find $x \mod q$ given A, y, q and the distribution (1). The solution is unique for large enough $m > n$ depending on (1). Commonly, a discrete normal (also called Gaussian) distribution $p_\sigma[a]$ with mean 0 and variance $\approx \sigma^2$ is used. There are several ways to determine $p_\sigma[a]$ on integers a. One may define it as

$$p_\sigma[a] = \frac{e^{-a^2/2\sigma^2}}{\sum_{b\in\mathbb{Z}} e^{-b^2/2\sigma^2}}. \tag{2}$$

However, in practice, it is more convenient to generate a real b according to an ordinary continuous normal distribution with mean 0 and variance σ^2 and then round b to the nearest integer a. If the standard deviation $\sigma = o(q)$ as q grows, one may assume that the distribution has its support on a such that $|a| \le (q-1)/2$. That defines a distribution P_σ on the residues $0, 1, \ldots, q-1$ modulo q.

3 Contributions

Let k be a positive integer. The public key of EHT is an $kn \times n$ matrix A over residues modulo q, constructed in Section 4.5 and the distribution P_σ introduced in Section 2. Let x be a plain-text block. The cipher-text block y is then computed as $y = Ax - e$, where the entries of e are independently generated according to the distribution P_σ. In order to recover the plain-text block $x = (x_1, \ldots, x_n)$ given the cipher-text $y = (y_1, \ldots, y_{kn})$ one has to solve an LWE problem. The private key consists of three matrices B, T, C described in Section 4.2 such that $A = C^{-1}TB$ modulo q. With the private key the plain-text x is recovered by a statistical procedure in Section 4.7. The decryption failure probability may be taken different by varying k, see Sections 4.8 and 7.3. For instance, for using in KEM (Key Encapsulating Mechanisms) k may be chosen to make the probability at most 10^{-5}, while for PKE (Public Key Encryption) one can make the probability 10^{-10}. With that choice the system is functional and more efficient, that is encryption/decryption work faster, than some of the NIST candidates with comparable security level. Explicit EHT parameters to fit three security levels are in Section 6. The comparison with the NIST candidate FrodoKEM is in Section 7. The decryption failure probability in EHT is higher than that in the most submissions to the NIST competition. However, we have not found any efficient attacks that exploit the property. There are two main approaches to the EHT cryptanalysis: find plain-text given cipher-text and find private key given public key. In Section 5 the results of the cryptanalysis are presented. The asymptotic complexity of breaking the crypto-system depends on the parameters, see Section 4.1, and is generally exponential in n for large n.

The EHT crypto-system was invented by Semaev, who also analysed the decryption failure probability in Sections 4.8 and 7.3, multiple encryptions of the same plain-text block in Section 5.2 and equivalent key recovering in Section 5.3.3. The choice of EHT optimised parameters in Section 6 and all computer experiments including those in Section 7 are due to Budroni. Also, Budroni discovered key-recovery attacks in Section 5.3.2.

4 EHT Parameters and Encryption/Decryption

In this section the EHT crypto-system is described in detail.

4.1 Parameters

Let $n, q, \sigma, k, \lambda, \gamma$ be positive integer parameters, where $q = n^{c_q}, \sigma = n^{c_\sigma}, \lambda = n^{c_\lambda}$ for some c_q, c_σ, c_λ, where $c_q > c_\sigma + c_\lambda$ and k is to be defined later.

4.2 Private Key

The system private key consists of three matrices B, T, C.

1. The matrix B is an integer $n \times n$ matrix of rank n modulo q, whose rows are $B_1, B_2, \dots, B_n$.

2. The matrix T is an integer $kn \times n$ matrix

$$T = \begin{pmatrix} t_{11} & 0 & \dots & 0 \\ \dots & & & \\ t_{k1} & 0 & \dots & 0 \\ 0 & t_{12} & \dots & 0 \\ \dots & & & \\ 0 & t_{k2} & \dots & 0 \\ \dots & & & \\ 0 & 0 & \dots & t_{1n} \\ \dots & & & \\ 0 & 0 & \dots & t_{kn} \end{pmatrix},$$

   where the entries $t_{ij}, 1 \le i \le k, 1 \le j \le n$ are non-zero modulo q. To reduce the decryption failure probability, in particular, to accept an incorrect plain-text block the following condition may be satisfied for every $1 \le j \le n$:

   (a) the residues $t_{ij}, 1 \le i \le k$ are different,

   (b) for every $a \ne 0 \mod q$ not all the residues $at_{ij} \mod q, 1 \le i \le k$ are close to 0.

   The conditions are satisfied for randomly chosen t_{ij} with high probability.

3. The matrix C is an integer $kn \times kn$ matrix whose rows $C_1, \dots, C_{kn}$ are of norm at most λ. For a correct decryption, the dot-products $C_u C_v$ for $1 \le u < v \le kn$ have to be very small, say bounded by γ in absolute value. We assume C is invertible modulo q. The matrix C may be constructed with one of the following methods.

4.3 Three Methods to Construct the Matrix C

In this section we present methods to construct the matrix C.

1. To define a row C_s one may take a random subset $\{i_1, \dots, i_{\lambda^2}\}$ of $\{1, 2, \dots, kn\}$ of size λ^2 and put

$$C_{sj} = \begin{cases} \pm 1, & j \in \{i_1, \dots, i_{\lambda^2}\}, \\ 0, & j \notin \{i_1, \dots, i_{\lambda^2}\}. \end{cases}$$

   The average of the dot-products $C_u C_v, u \ne v$ is 0.

2. One can use combinatorial configurations(incidence structures). For instance, we may use a protective plane over a finite field of size p. The number of points on the plane is $p^2+p+1 = nk$, there are $p^2 + p + 1$ lines on the plane, each line has $p + 1 = \lambda^2$ points, each two lines have exactly one point in common. The points on the plane and the lines may be permuted with secret permutations P and Q accordingly. We take lines as subsets $\{i_1, \dots, i_{\lambda^2}\}$ and choose the rows of C by inscribing 1 or -1 to the points on the line and 0 to all other points. For this construction $\gamma = 1$.

Table 1: Permutation P

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$T(x)$	14	1	10	9	6	13	5	16	15	2	12	3	7	11	8	4

3. Let $\lambda^2 = 2^s$ and $kn = r2^s$ for some integers r, s and H be a Hadamard matrix of size $2^s \times 2^s$. The rows and columns of H are indexed by binary s-strings a and b respectively. The entry of H in the row a and the column b is $(-1)^{ab}$, where ab is the dot-product of a, b. Let P, Q be secret $nk \times nk$ permutation matrices. We set $C = P(H \otimes I_r)Q$, where I_r is a unity matrix of size $r \times r$. That is

$$C = P \begin{pmatrix} H & 0 & \dots & 0 \\ 0 & H & \dots & 0 \\ & & \dots & \\ 0 & 0 & \dots & H \end{pmatrix} Q, \qquad C^{-1} = 2^{-s} Q^{-1} \begin{pmatrix} H & 0 & \dots & 0 \\ 0 & H & \dots & 0 \\ & & \dots & \\ 0 & 0 & \dots & H \end{pmatrix} P^{-1}$$

as $H^2 = 2^s\, I_{2^s}$. The rows of H are orthogonal, so the rows of C are orthogonal too and $\gamma = 0$. In order to avoid weak keys found in Section 5.3.2, instead of P one uses a permutation P^* defined in Section 4.4.

In all cases the rows of C are of norm λ. We will need to invert the matrix C modulo q to construct the system public key. For the first two methods the inversion is rather slow as the matrix size $kn \times kn$ may be large. For the third method the inversion is trivial. The experiments in Section 7 were conducted with EHT based on the third method for constructing C.

4.4 Permutation P^*

The rows of the matrix C may be split into chunks of size k as $C_{1+k(i-1)}, \dots, C_{ki}$ for $i = 1, \dots, n$. Assume that two rows with the same positions for non-zero entries fall into one chunk. According to the cryptanalysis in Section 5.3.2, they may be recovered faster than in the general case and this may lead to a weak key of the crypto-system. For instance, let $n = 8, k = 2, \lambda^2 = 4$ and P be defined by Table 1, and $C = P(H \otimes I_r)Q$ as above. The pairs of rows (C_5, C_6), (C_7, C_8) and (C_9, C_{10}) have the same non-zero positions. One needs to choose $P = P^*$ to avoid this. That is easy to do if $\lambda^2 | n$.

One first defines a mapping $M : \{1, 2, \dots, nk\} \to \{1, 2, \dots, n\}$ such that the values $M(l + (j-1)\lambda^2)$ are different for $l = 1, \dots, \lambda^2$ and every fixed j from $1 \le j \le nk/\lambda^2$. Then one defines a permutation P^* on $1, 2, \dots, nk$ by the rule $P^*(l + (j-1)\lambda^2) \in \{1 + k(i-1), \dots, ki\}$, where $i = M(l + (j-1)\lambda^2)$. For instance, the mapping M and the permutation P^* may be defined by Table 2. The number of possible mappings M and therefore the number of possible permutations P^* is very large. So this restriction does not affect the security.

4.5 Public Key

The public key is an integer $kn \times n$ matrix $A \equiv C^{-1}TB \mod q$ and σ.

Table 2: Mapping M and Permutation P^*

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$M(x)$	1	2	3	4	5	6	7	8	1	3	5	7	2	4	6	8
$P^*(x)$	1	3	5	7	9	11	13	15	2	6	10	14	4	8	12	16

4.6 Encryption

Let x be a plain-text encoded by a column vector of size n of residues modulo q. We may assume the plain-text has some natural redundancy. That significantly reduces the probability of accepting an incorrect plain-text block x as correct at the decryption. If the plain-text does not have natural redundancy then it may be artificially introduced. For the parameters chosen in Section 6 we need at most $2\log_2 q$ bits redundancy, which may be provided with a linear $\mathbb{F}_q$-code of length n and dimension $n-2$. This only negligibly affects the security of the crypto-system. Another option is to use Optimal Asymmetric Encryption Padding (OAEP), see [3].

The cipher-text is $y \equiv Ax - e \mod q$, where e is a column vector of size m whose entries are residues modulo q independently generated according to the distribution P_σ defined by (2).

4.7 Decryption

By multiplying the both sides of $y \equiv Ax - e \mod q$ with C one gets $z \equiv Tb - Ce \mod q$, where $z = Cy = (z_1, \ldots, z_{kn}), b = Bx = (b_1, \ldots, b_n)$ are column vectors of length kn and n respectively. The entries of Ce have discrete normal distributions $P_{\sigma\lambda} = (p_{\sigma\lambda}[0], \ldots, p_{\sigma\lambda}[q-1])$ with mean 0 and standard deviation $\approx \sigma\lambda = o(q)$, as $c_q > c_\sigma + c_\lambda$. By the definition of the matrix T in Section 4.2,

$$C_{1+k(i-1)}e \equiv t_{1i}b_i - z_{1+k(i-1)} \mod q,$$

$$\ldots$$

$$C_{k+k(i-1)}e \equiv t_{ki}b_i - z_{k+k(i-1)} \mod q.$$

One now finds $b_i = B_i x, 1 \le i \le n$ with a statistical procedure. For each residue a the value of the statistic

$$S_i(a) = \sum_{j=1}^{k} \ln(q\, p_{\sigma\lambda}[t_{ji}a - z_{j+k(i-1)}]) \tag{3}$$

is computed, where $t_{ij}a - z_{j+k(i-1)}$ is taken modulo q. One decides $b_i = a$ if $S_i(a) > 0$. In practice, it may be more convenient to use an ordinary continuous normal density $p_{\sigma\lambda}[x] = e^{-x^2/2(\sigma\lambda)^2}/\sqrt{2\pi(\sigma\lambda)^2}$ for residues x modulo q such that $|x| \le (q-1)/2$. Also, one may decide $b_i = a$ for a residue a which maximises $S_i(a)$, however, in this case the decryption failure probability is difficult to predict. By a decryption failure we mean rejecting the correct plain-text block x or accepting an incorrect one. If k is large enough, then the decryption is unique and correct with high probability as it follows from the next Section.

4.8 Decryption Failure Probability

The asymptotic analysis in Section 7.3 shows that k may be taken constant for large n if $c_\sigma + c_\lambda < c_q$. In this section we find explicit expressions for the decryption failure probability. The distribution

$P_{\sigma\lambda}$ may be approximated by an ordinary continuous normal density with mean 0 and standard deviation $\sigma\lambda$, so

$$\ln(q\,p_{\sigma\lambda}[x]) \approx \ln(\frac{q}{\sigma\lambda\sqrt{2\pi}}) - \frac{x^2}{2(\sigma\lambda)^2}. \tag{4}$$

Then $S_i(a) \approx \sum_{j=1}^{k} \ln(\frac{q}{\sigma\lambda\sqrt{2\pi}}) - \frac{x_j^2}{2(\sigma\lambda)^2}$, where $x_j \equiv t_{ji}a - z_{j+k(i-1)} \mod q$. Let $\delta^2 = 2k(\sigma\lambda)^2 \ln(\frac{q}{\sigma\lambda\sqrt{2\pi}})$. According to the test $S_i(a) > 0$, one decides $b_i = a$ if

$$\sum_{j=1}^{k} x_j^2 < \delta^2. \tag{5}$$

We consider two cases. First, let $b_i = a$. Then $x_1, \ldots, x_k$ are independently generated with the distribution $P_{\sigma\lambda}$. The success (that is to accept $b_i = a$) probability is

$$\beta = \mathbf{Pr}(S_i(a) > 0) \approx \mathbf{Pr}(\chi_k^2 < (\delta/\sigma\lambda)^2),$$

where χ_k^2 denotes a random variable distributed as χ-square with k degrees of freedom. The probability to reconstruct correctly all $b_i = B_i x$ and therefore to recover the correct x is then $\beta_1 = \beta^n$. Second, let $b_i \neq a$. Then

$$x_j \equiv t_{ji}a - z_{j+k(i-1)} \equiv t_{ji}(a - b_i) + C_{j+k(i-1)}e \mod q.$$

So we can assume $x_1, \ldots, x_k$ are independently and uniformly distributed over residues modulo q. By (5), the probability to decide $b_i = a$ for an incorrect a is

$$\alpha \approx \frac{\mathrm{Vol}_k(\delta)}{q^k} = \frac{\pi^{k/2}\,\delta^k}{\Gamma(k/2+1)\,q^k},$$

where $\mathrm{Vol}_k(\delta)$ is the volume of a k-ball of radius δ. Therefore, the probability to accept at least one incorrect $b_i = a$ is at most $\alpha_1 = nq\alpha$. If the plain-text block x is a codeword in a linear $\mathbb{F}_q$-code of length n and dimension $n-2$ the probability to accept an incorrect plain-text block is at most $\alpha_1 q^{-2} = n\alpha/q$. Therefore, the decryption failure (to reject the correct x or to accept an incorrect one) probability is at most $(1-\beta_1) + \alpha_1 q^{-2} \approx (1-\beta_1)$ for the parameters in Section 6. So the most probable failure is to reject both the correct and incorrect plain-text blocks x and the probability of this is around $(1-\beta_1)$. Obviously, the probability to reject the correct x and to accept an incorrect one is much smaller.

Let $q = 1021, n = 128, \sigma = 5.105, \lambda = 4$. The following Figure 1 shows the experimental probabilities $1-\beta_1$ and α_1 of the decryption failures for a variety of $k = 7, \ldots, 13$ in comparison with their estimates. The estimates are rather accurate upper bounds for the actual probabilities.

4.9 Complexity Parameters

The size of the public key is kn^2 residues modulo q. The encryption cost (to encrypt $n\log_2 q$ bits of the plain-text) is kn^2 multiplications modulo q. For binary x the cost (to encrypt n bits of the plain-text) is kn^2 additions modulo q. The decryption cost is $kn\lambda^2$ additions modulo q to compute $z = Cy$. This may be improved to $kn\log_2(\lambda^2)$ additions with the fast Walsh-Hadamard transform. Then it takes knq additions modulo q to compute nq values of the statistic, and n^2 operations to solve the linear equation system $b_i = B_i x, i = 1, \ldots, n$ to recover x, where the inversion of B modulo q is precomputed. The matrix B may be chosen to accelerate the solution of the system to $< n^2$ operations.

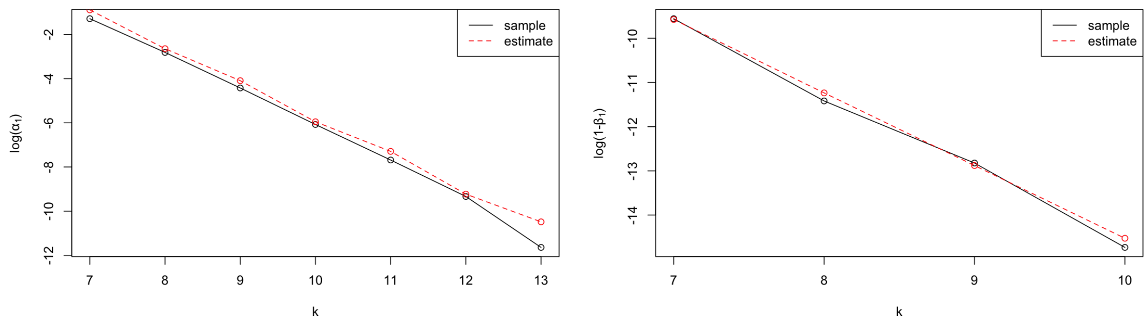

Figure 1: Experimental and estimated error probabilities α_1 and $1-\beta_1$

4.10 Variant

To construct private/public key one may take

$$T = \begin{pmatrix} t_{11} & 0 & \dots & 0 \\ \dots & & & \\ t_{k1} & 0 & \dots & 0 \\ t_{k+1\,1} & t_{1\,2} & \dots & 0 \\ \dots & & & \\ t_{2k\,1} & t_{k\,2} & \dots & 0 \\ \dots & & & \\ t_{(n-1)k+1\,1} & t_{(n-2)k+1\,2} & \dots & t_{1\,n} \\ \dots & & & \\ t_{nk\,1} & t_{(n-1)k\,2} & \dots & t_{k\,n} \end{pmatrix}.$$

The decryption algorithm is then easy to adjust.

5 Cryptanalysis

We consider two approaches to the cryptanalysis of LWE-Trapdoor crypto-system: find plain-text from the cipher-text and recover private key from the public key.

5.1 Plain-Text Recovering

To reconstruct the plain-text x from the cipher-text y one has to solve an LWE problem $y \equiv Ax - e \mod q$, where A is a matrix of size $kn \times n$. According to [6] the complexity of this problem is exponential $2^{O(n)}$ with both lattice-base algorithms and amplified BKW if the structure of A is not taken into account.

Let's consider the so called primal attack. One may assume that $A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix}$, where A_1 is a matrix of size $n \times n$, invertible modulo q and the matrix A_2 is of size $(kn - n) \times n$, and let $y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$ and $e = \begin{pmatrix} e_1 \\ e_2 \end{pmatrix}$ accordingly. Then $A_1 x - e_1 = y_1, A_2 x - e_2 = y_2$. So $x = A_1^{-1} e_1 + y_1$ and

$A_2A_1^{-1}e_1 - e_2 - y_2 + A_2A_1^{-1}y_1 = 0$. The vector $w = (e, 1)$ satisfies $A'w \equiv 0 \mod q$, where A' is a horizontal concatenation of $A_2A_1^{-1}, -I_{kn-n}, -y_2 + A_2A_1^{-1}y_1$. The size of A' is $(kn - n) \times (kn + 1)$.

Let $m \leq kn$ be a parameter and let B be a matrix constructed with m rows of A' ignoring some zero columns. The size of B is $m \times (m + n + 1)$. Then $A'w \equiv 0 \mod q$ implies $Bv \equiv 0 \mod q$, where $v = (f, 1)$ and f is a sub-vector of e of length $m+n$. Thus v belongs to a lattice of dimension $m + n + 1$ and of volume $\mathrm{Vol} = q^m$.

The expected norm of v is at most $\sqrt{\sigma^2(m + n) + 1} \approx \sigma\sqrt{m + n}$. The Block Korkine-Zolotarev (BKZ) algorithm is applied to reduce the basis of the lattice and recover the shortest non-zero vector, which is likely to be v. Then x is computed by solving a system of linear equations modulo q. The reduction algorithm calls Shortest Vector Problem (SVP) oracle for a lattice of a smaller dimension $\beta \leq m$ a polynomial number of times [5]. The cost of one call is $\geq 2^{0.292\beta}$ operations with a sieving algorithm according to [7]. One takes the smallest β to satisfy the inequalities

$$\sqrt{\beta}\sigma \leq \delta^{2\beta-d}\,\mathrm{Vol}^{1/d}, \quad \delta = \Big((\pi\beta)^{1/\beta}\beta/2\pi e\Big)^{1/2(\beta-1)} > 1.$$

The sieve algorithms may benefit from the Grover's quantum search and this pushes the complexity down to $2^{0.265b}$ [10, 8]. As remarked in [1], since these algorithms require building a list of lattice vectors of size of $2^{0.2075b}$, it is plausible to believe that the best quantum SVP algorithm would run in $2^{0.2075b}$.

The so called dual attack [13] is based on finding a large number of short vectors u, v such that $uA \equiv v \mod q$. One then applies a statistical test to recover the entries of x. The dual attack is generally inferior compared to the primal attack, see [2].

5.2 Multiple Encryptions

Assume that the same plain-text block x was encrypted s times with independently generated error vectors $e(i) = (e_{i1}, \dots, e_{im}), i = 1, \dots, s$. So s cipher-texts $y(i) = Ax - e(i)$ are available for the cryptanalysis, where $y(i) = (y_{i1}, \dots, y_{im})$ and $|y_{ij}| < q/2$. Let's fix an index j in $1, \dots, m$ and let A_j be a row in A. Then $y_{ij} = A_jx - e_{ij} \mod q$ are taken from a discrete normal distribution with mean $A_jx \mod q$ and variance σ^2. So $\sum_{i=1}^{s} y_{ij}/s$ is normally distributed with mean $A_jx \mod q$ and variance σ^2/s. If $s = O(\sigma^2)$, then one recovers $A_jx \mod q$ as $\sum_{i=1}^{s} y_{ij}/s \approx A_jx \mod q$. One does this for some n values of j and then solves a system of linear equations to recover x.

A kind of DOS (denial of service) attack is applicable to recover x with this method. To protect against that sort of attacks, each plain-text block may be made dependent on a counter value or time stamp before encryption.

5.3 Equivalent and Private Key Recovering

5.3.1 Framework

Let C, T, B be any triplet satisfying the restrictions of Section 4.2 and $CA = TB$. Let the norm of the rows of C is at most μ. If μ is not significantly larger than λ, then C, T, B is the system equivalent private key and it may be used to decrypt data. However, if μ is significantly larger than λ, then the decryption fails as it is not unique. The equality $CA = TB$ implies

$$C_{1+k(i-1)}A = t_{1i}B_i, \dots, C_{k+k(i-1)}A = t_{ki}B_i,$$

for $i = 1, \ldots, n$ and the rows of C and B. Let's assume $t_{1i} = 1$ and eliminate B_i from the equations (if $t_{1i} \neq 1$, then we eliminate $t_{1i}B_i$). The vector $\bar{C}_i = (C_{1+k(i-1)}, \ldots, C_{k+k(i-1)})$ is of size k^2n and it satisfies

$$\bar{C}_i\bar{A} \equiv 0 \mod q, \quad \bar{A} = \begin{pmatrix} t_{2i}A & t_{3i}A & \ldots & t_{ki}A \\ -A & 0 & \ldots & 0 \\ 0 & -A & \ldots & 0 \\ \ldots & & & \\ 0 & 0 & \ldots & -A \end{pmatrix}, \tag{6}$$

where $\bar{A}$ is a matrix of size $k^2n \times (k-1)n$. Therefore, $\bar{C}_i$ belongs to a lattice of volume $q^{n(k-1)}$ and of dimension k^2n for some residues $t_{2i}, \ldots, t_{ki}$. The norm of $\bar{C}_i$ is at most $\mu\sqrt{k}$.

5.3.2 Private Key Recovering

The rows of C have to satisfy (6) and so

$$(C_{1+k(i-1)}, C_{j+k(i-1)}) \begin{pmatrix} t_{ji}A \\ -A \end{pmatrix} \equiv 0 \mod q. \tag{7}$$

One has to find all pairs of non-collinear vectors C_1, C_2 of length kn with λ^2 non-zero entries ± 1 each such that $tC_1A - C_2A \equiv 0 \mod q$ for some residues t. Then C is easy to recover and the matrices T and B are easy to recover too.

Brute Force Attack. One brute forces such C_1, C_2 and finds collinear rows (matches) C_1A, C_2A. To this end one divides (scales) C_1A, C_2A by their left most non-zero entries and searches for all the matches. The number of trials is $2^{2\lambda^2}\binom{kn}{\lambda^2}\binom{kn-\lambda^2}{\lambda^2}$.

Time-Memory-Trade-Off. One defines the table of all scaled C_1A. The table is sorted to find all the matches. At the expense of memory size $V = 2^{\lambda^2}\binom{kn}{\lambda^2}$, the complexity is $O(V \log V)$.

Lattice Based Method. Let $h \leq kn$ be an integer parameter and $S_1, S_2 \subseteq \{1, 2, \ldots, kn\}$ be two subsets of size h. Let A_i be a sub-matrix constructed with h rows of A whose indices are in S_i. If the positions of non-zero entries of C_i are in S_i, then $tC_1A_1 - C_2A_2 \equiv 0 \mod q$. Therefore (C_1, C_2) belongs to a lattice of dimension $2h$ and of volume q^n for a residue t modulo q. The norm of (C_1, C_2) is $\lambda\sqrt{2}$. For each t one tries random subsets S_1, S_2 and applies BKZ algorithm until (C_1, C_2) is found. The number of BKZ applications to recover all the rows of C is $q\left(\binom{kn}{h}/\binom{kn-\lambda^2}{h-\lambda^2}\right)^2$ on the average. One finds h to minimise the overall cost.

Possible Weak Keys. Assume that the non-zero positions of $C_1 = C_{j_1+k(i-1)}$ and $C_2 = C_{j_2+k(i-1)}$ are the same for some $1 \leq j_1 < j_2 \leq k$ and $1 \leq i \leq n$. This event happens quite often with the third method for constructing C in Section 4.3 when a random row permutation P is used and $\lambda^2 > \sqrt{n}$. To recover C_1 and C_2 one tries only one subset $S \subseteq \{1, 2, \ldots, kn\}$ of size h not two. One applies BKZ to a lattice of dimension $2h$ and volume q^n to recover C_1, C_2.

The vector C_1A is collinear to the row B_i of B. So $C_{j+k(i-1)}A \in \langle B_i \rangle = \langle C_1A \rangle$ for every $1 \leq j \leq k$. Let the non-zero positions of $C_{j+k(i-1)}$ be in a subset $S_1 \subseteq \{1, 2, \ldots, kn\}$ of size $h_1 \geq \lambda^2$ and let A_1 be a sub-matrix of A constructed with h_1 rows of A whose indices are in S_1. Then $C_{j+k(i-1)}A_1 \in \langle B_i \rangle$. The vector $C_{j+k(i-1)}$ of norm λ belongs to a lattice of dimension h_1 and of volume q^{n-1}. One guesses the subset S_1 and runs BKZ to recover

$C_{j+k(i-1)}$. With a similar approach one finds all the rows of C. The values h, h_1 are chosen to minimise the overall cost. The method is significantly faster than all the previous methods of cryptanalysis. However, the event does not happen for any i if the row permutation P^* instead of P is used to construct C, see Section 4.4. The system does not then admit such weak keys.

5.3.3 Equivalent Key Recovering

To construct an equivalent key one may try to find short vectors $\bar{C}_i$(say of norm $\leq \mu\sqrt{k}$) such that $\bar{C}_i\bar{A} \equiv 0 \mod q$, see (6). Thus $\bar{C}_i$ belongs to a lattice of dimension k^2n and volume $q^{(k-1)n}$. Then $B_i \equiv C_{1+k(i-1)}A$ thus constructing the system equivalent key. Since the size of $\bar{A}$ is very large, that seems a hard task.

For some $t_{2i}, \ldots, t_{ki}$ the lattice may contain very short vectors $\bar{C}_i$ which do not depend on A, see Section 5.3.4 below. The component vectors $C_{1+k(i-1)}, \ldots, C_{k+k(i-1)}$ are then collinear. The random variables $C_{1+k(i-1)}e, \ldots, C_{k+k(i-1)}e$ are dependent and the decryption fails.

5.3.4 Very short vectors in the lattice $\bar{C}\bar{A} \equiv 0 \mod q$

Let

$$\bar{I} = \begin{pmatrix} t_2I & t_3I & \ldots & t_kI \\ -I & 0 & \ldots & 0 \\ 0 & -I & \ldots & 0 \\ \ldots & & & \\ 0 & 0 & \ldots & -I \end{pmatrix},$$

for some non-zero residues $t_2, \ldots, t_k$, where I is a unity matrix of size $kn \times kn$ and so $\bar{I}$ is a matrix of size $k^2n \times k(k-1)n$. Let

$$\bar{R} = \begin{pmatrix} A & 0 & \ldots & 0 \\ 0 & A & \ldots & 0 \\ \ldots & & & \\ 0 & 0 & \ldots & A \end{pmatrix},$$

be a matrix of size $k(k-1)n \times (k-1)n$. Obviously, $\bar{A} = \bar{I}\bar{R}$ for $t_j = t_{ji}$. Let an integer vector

$$\bar{C} = (c_1, c_2, \ldots, c_{k^2n})$$

of size k^2n satisfy $\bar{C}\bar{I} \equiv 0 \mod q$, then $\bar{C}\bar{A} \equiv 0 \mod q$. We want to construct $\bar{C}$ with a low norm. The congruence $\bar{C}\bar{I} \equiv 0 \mod q$ implies

$$\left(c_i, c_{i+kn}, \ldots, c_{i+k(k-1)n}\right) \begin{pmatrix} t_2 & t_3 & \ldots & t_k \\ -1 & 0 & \ldots & 0 \\ 0 & -1 & \ldots & 0 \\ \ldots & & & \\ 0 & 0 & \ldots & -1 \end{pmatrix} \equiv 0 \mod q \tag{8}$$

for $i = 1, \ldots, kn$. Let $V_i = (c_i, c_{i+kn}, \ldots, c_{i+k(k-1)n})$. The vectors V_i belong to the same lattice $L(t_2, \ldots, t_k)$ of dimension k and volume q^{k-1} defined by (8). For the majority of $t_2, \ldots, t_k$ the shortest non-zero vector of the lattice has norm around $\sqrt{k}\, q^{1-1/k}$. Those vectors may be used to construct a vector $\bar{C}$ with norm around $\sqrt{k^2n}\, q^{1-1/k}$. That is much larger than $\lambda\sqrt{k}$, the norm of the hidden vectors $(C_{1+(i-1)k}, \ldots, C_{k+(i-1)k})$ used in the construction of the public matrix A. However, the lattice $L(t_2, \ldots, t_k)$ contains the vector $(1, t_2, \ldots, t_k)$ which is fairly short for small in absolute value non-zero residues $t_2, \ldots, t_k$. For instance, for $t_i = \pm 1$ the lattice contains a vector of norm $\sqrt{k}$. The following lemma shows that all very short vectors in the lattice are collinear.

Lemma 1. *Let* $(v_1, v_2, \ldots, v_k)$ *be a non-zero vector in* $L(t_2, \ldots, t_k)$*, where* $\gcd(v_1, \ldots, v_k) = 1$ *and* $|v_i| < \sqrt{q/2}$ *and let* $(w_1, w_2, \ldots, w_k)$ *be another vector there such that* $|w_i| < \sqrt{q/2}$*. Then*

$$(w_1, w_2, \ldots, w_k) = a(v_1, v_2, \ldots, v_k)$$

for an integer a*.*

Proof. As $v_i = t_i v_1$ and $w_i = t_i w_1$, we get $v_1 w_i - v_i w_1 \equiv 0 \mod q$ for $i = 2, \ldots, k$. Since, $|v_1 w_i - v_i w_1| < q$ we have $v_1 w_i = v_i w_1$. Therefore,

$$(w_1, w_2, \ldots, w_k) = \frac{w_1}{v_1}(v_1, v_2, \ldots, v_k)$$

As v_i are coprime, that is only possible when $v_1 | w_1$. That proves the lemma. □

So if the entries of every V_i are $< \sqrt{q/2}$, then $V_i = a_i(v_1, v_2, \ldots, v_k)$ for the same vector $(v_1, v_2, \ldots, v_k)$ and integer a_i. Then the vector $\bar{C} = (C_1, \ldots, C_k)$ satisfies $C_i = v_i(a_1, a_2, \ldots, a_{kn})$. Every entry of C_i is $< \sqrt{q/2}$ and may be fairly small, for instance, when $t_i = \pm 1$. In the latter case the norm of $\bar{C}$ is $\sqrt{k^2 n}$.

However, since $C_1, \ldots, C_k$ are collinear, the random variables $C_j e$ are dependent. The statistic (3) does not provide proper decryption for $b_i = B_i x$.

We may admit some few of the vectors V_i with entries $\geq \sqrt{q/2}$ and the rest of the vectors V_i with entries $< \sqrt{q/2}$. Then the norms of $C_1, \ldots, C_k$ become significantly larger and there still is a significant statistical dependence between $C_j e$. So the decryption is not unique in this case either.

6 Proposed Parameters

In Table 3 we propose three sets of parameters matching Levels $1, 3, 5$ in the call of the NIST Post-Quantum Cryptography Standardization Process [11]. These parameters correspond to or exceed the brute-force security of AES-$128, 192, 256$ respectively. The parameters are chosen to minimise the encryption + decryption cost subject to the following constraints: the security provided according to the most efficient attack (primal attack for the underlying LWE problem, see Section 5.1) is close to the targeted security level, for implementation efficiency we set $q < 2^{16}$, where q is close to a power of 2, and n is a multiple of 64, and λ^2 is a power of 2, where $\lambda^2 | n$. Two sets of parameters for each security level reaching a decryption failure probability of at most 10^{-5}(type A) and 10^{-10}(type B) respectively are provided. The decryption failure probability $1 - \beta_1$ is computed according to Section 4.8.

The sizes of the public key A, the private key C, T, B, the plain-text x and the cipher- text y for each security level are presented in Table 4.

The approximate costs of the primal attack for each parameters' set are reported in Table 5. The number of samples m and BKZ block size b have been chosen to minimise the attack cost. **C**, **Q** and **P** correspond respectively to classical, quantum and best plausible quantum lattice sieve algorithm complexity according to Section 5.1.

In Table 6 we give the costs for the key-recovery attacks: brute-force, time-memory-trade-off and the lattice-based attacks according to Section 5.3.2.

Name	Level	n	k	q	λ^2	σ	$1-\beta_1$
EHT-light-A	1	256	16	1021	32	8.8	$7.4 \cdot 10^{-6}$
EHT-light-B		256	25	2039	32	14.5	$4.6 \cdot 10^{-11}$
EHT-medium-A	3	384	14	2039	32	13.5	$3.3 \cdot 10^{-6}$
EHT-medium-B		384	24	2039	32	13.5	$4.8 \cdot 10^{-11}$
EHT-high-A	5	448	17	2039	32	17.5	$5.2 \cdot 10^{-6}$
EHT-high-B		448	24	4091	32	27.0	$5.6 \cdot 10^{-11}$

Table 3: Proposed parameters.

Name	A	C, T, B	x	y
EHT-light-A	1310.7 kB	99.4 kB	320 B	5.1 kB
EHT-light-B	2252.8 kB	119.8 kB	352 B	8.8 kB
EHT-medium-A	2838.5 kB	227.7 kB	528 B	7.4 kB
EHT-medium-B	4866.0 kB	247.7 kB	528 B	12.7 kB
EHT-high-A	4691.5 kB	311.2 kB	616 B	10.5 kB
EHT-high-B	7225.3 kB	354.9 kB	672 B	16.1 kB

Table 4: Sizes in bytes (B) or kilobytes (kB).

7 Implementation and Performance

EHT encryption/decryption was implemented in C with no external dependencies. The objective was to have a proof-of-concept of the EHT, run experiments and make comparisons with other public-key protocols. By no means we claim this implementation to be ready for the real world, or to be resistant to side-channel attacks, etc. Also, even if the efficiency was one of our main foci, there is certainly room for further optimization, such as the use of AVX2.

We made two implementations to run the same algorithms with a difference in memory usage: only stack memory and a combination of stack and heap. The former provides with the best performances and is referred in this section. Both variants of the implementation are publicly available at `https://github.com/AlessandroBudroni/EHT-C`.

7.1 Performance

To evaluate the performance of EHT we run it for the benchmark parameters in Table 3. A machine with processor 3.60GHz Intel Core i7-7700 CPU, running Linux Mint 20 and with 32 GB of RAM was used. As a standard practice, TurboBoost was disabled during the tests. The code was compiled with `gcc -std=c11 -O3`.

The performance of the implementation is detailed in Table 7. Encryption is considerably faster than decryption and key generation routines. This was a design choice when determining the

Name	m	b	**C**	**Q**	**P**
EHT-light-A	420	468	136	124	97
EHT-light-B	471	478	139	126	99
EHT-medium-A	653	728	212	193	151
EHT-medium-B	653	728	212	193	151
EHT-high-A	788	956	279	253	198
EHT-high-B	835	947	276	251	196

Table 5: Primal attack costs in $\log_2$-scale.

Name	Lattice-based					Time-Memory	Bruteforce
	h	b	**C**	**Q**	**P**		
EHT-light-A	258	131	309	305	296	305	595
EHT-light-B	267	127	346	343	334	326	636
EHT-medium-A	314	115	311	308	299	318	620
EHT-medium-B	314	115	361	357	349	343	670
EHT-high-A	331	108	336	333	324	334	652
EHT-high-B	344	106	364	361	353	350	684

Table 6: Key recovery attacks cost in $\log_2$-scale.

parameters sets. The rationale was to keep the cost low on the client side, i.e. encryption, at the expense of increasing the cost on the server side, i.e. key-generation and decryption.

7.2 Comparison with FrodoKEM

FrodoKEM [2] is one of the candidates to the NIST Post-Quantum Standardization Process at Round 3 and the only one purely based on LWE left in the competition. It is a key encapsulation mechanism designed to allow two parties to agree on a master secret of 16, 24 or 32 bytes. On the other hand, EHT is a public-key crypto-system that allows to encrypt a relatively large number of bytes at once. A scenario on which both the protocols find an application is when two parties must agree on a batch of several keys. Due to the large plaintext load provided by EHT, it is enough to make only one encryption run to agree on a big master secret that can be used as the source for generating several keys. For example, EHT-light-B allows to exchange 349.25 bytes of plaintext, enough for generating 20 master secrets of 16 bytes each. In this scenario, EHT dominates over FrodoKEM in speed.

The experimental results reported in Table 8 support that claim. Both the algorithms were run the necessary number of times to encrypt and then decrypt (encapsulate and decapsualte) the same number of plain-text bits for each security level. The implementation of FrodoKEM available at `https://github.com/Microsoft/PQCrypto-LWEKE` with AVX2 enabled and *openssl* disabled was

Name	Plaintext Load	KeyGen	Encryption	Decryption	Total (Enc + Dec)
EHT-light-A	317.5 B	699188	2055	8677	10732
EHT-light-B	349.25 B	735385	3114	13322	16436
EHT-medium-A	525.25 B	2320390	3836	14739	18575
EHT-medium-B	525.25 B	2493254	6690	16843	23533
EHT-high-A	613.25 B	3518341	6286	29072	35358
EHT-high-B	669.00 B	3597042	8296	35192	43488

Table 7: Performance of EHT in thousands of cycles.

used. Plaintext load is expressed in kilobytes (kB), measured time is expressed in seconds (s) and it is the mean of 5 run. One can see that, at all security levels, EHT is considerably faster than FrodoKEM.

Level	Protocol	Plaintext Load	Repetitions	Time
1	EHT-light-A	20 kB	63	0.19 s
	EHT-light-B		58	0.26 s
	FrodoKEM-640-AES		1250	1.25 s
	FrodoKEM-640-SHAKE		1250	3.00 s
3	EHT-medium-A	30 kB	58	0.29 s
	EHT-medium-B		58	0.37 s
	FrodoKEM-976-AES		1250	2.44 s
	FrodoKEM-976-SHAKE		1250	6.35 s
5	EHT-high-A	40 kB	66	0.64 s
	EHT-high-B		60	0.71 s
	FrodoKEM-1344-AES		1250	4.26 s
	FrodoKEM-1344-SHAKE		1250	11.23 s

Table 8: Performance of EHT against FrodoKEM.

References

[1] E. Alkim, L. Ducas, T. Pöppelmann, P. Schwabe, *Post-quantum key exchange: A new hope*, in USENIX Conference on Security, pp. 327–343. SEC'16, USENIX Association, USA (2016).

[2] E. Alkim, J. Bos, L. Ducas, P. Longa, I. Mironov, M. Naehrig, V. Nikolaenko, C. Peikert, A. Raghunathan, D. Stebila, *FrodoKEM Learning With Errors Key Encapsulation,* Tech. rep. NIST (2020), available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions`.

[3] M. Bellare, P. Rogaway, *Optimal Asymmetric Encryption – How to encrypt with RSA. Extended abstract,* in Eurocrypt '94, LNCS vol. 950, Springer, 1995.

[4] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, second edition, John Wiley & Sons, 2006.

[5] G. Hanrot, X. Pujol, D. Stehlé, *Analyzing blockwise lattice algorithms using dynamical systems,* in CRYPTO 2011, LNCS vol. 7073, pp. 1–20, Springer 2011.

[6] G. Herod, E. Kirshanova and A. May, *On the asymptotic complexity of solving LWE*, Des. Codes Crypt. vol. 86(2018), pp. 55–83.

[7] T. Laarhoven, *Sieving for shortest vectors in lattices using angular locality-sensitive hashing,* in CRYPTO 2015, LNCS vol. 9215, pp. 3–22, Springer 2015.

[8] T. Laarhoven, *Search problems in cryptography*, available at `http://fs.fish.govt.nz/Page.aspx?pk=7\&sc=SUR`, January 2014.

[9] T. Laarhoven, A. Mariano, *Progressive lattice sieving*, in PQCrypto 2018, pp. 292–311, Springer, 2018.

[10] T. Laarhoven, M. Mosca, J. Pol, *Finding shortest lattice vectors faster using quantum search*, Des. Codes Crypt. 77(2015), pp. 375–400.

[11] NIST Post-Quantum Cryptography Standardization Process, `https://csrc.nist.gov`.

[12] C. Peikert, *A decade of lattice cryptography,* Foundations and Trends in Theoretical Computer Science, vol. 10, issue 4, pp. 283–424, 2016.

[13] O. Regev, *On lattices, learning with errors, random linear codes, and cryptography,* Journal of the ACM, vol. 56(2009), issue 6.

7.3 Appendix 1. Asymptotic value of k.

If $C_{1+k(i-1)}e, \dots, C_{k+k(i-1)}e$ are independently distributed, then $k = O(\ln q/\mathrm{Div}(P_{\sigma\lambda}||U))$ by Chernoff-Stein lemma, see [4], where $\mathrm{Div}(P_{\sigma\lambda}||U)$ is the Kullback-Leibler divergence of the distribution $P_{\sigma\lambda}$ and the uniform distribution U on residues modulo q. We have

$$\mathrm{Div}(P_{\sigma\lambda}||U) = \sum_a p_{\sigma\lambda}[a] \ln(q\, p_{\sigma\lambda}[a]) = E \ln(q\, p_{\sigma\lambda}[a]).$$

The distribution $P_{\sigma\lambda}$ may be approximated by an ordinary normal distribution with mean 0 and variance $(\sigma\lambda)^2$, see (4). Then $\mathrm{Div}(P_{\sigma\lambda}||U) \approx \ln(\frac{q}{\sigma\lambda\sqrt{2\pi}}) - \frac{1}{2} = \ln(\frac{q}{\sigma\lambda\sqrt{2\pi e}})$ and $k = O\left(\ln q/\ln(\frac{q}{\sigma\lambda\sqrt{2\pi e}})\right)$. If $c_\sigma + c_\lambda < c_q$ and they are constants, then k is bounded by a constant for large n. The vector Ce has approximately a multivariate normal distribution with the covariance matrix $\sigma^2 CC^T$. The expected value of non-diagonal entries of $\sigma^2 CC^T$ is bounded by a small parameter γ by the matrix C construction. When using a Hadamard matrix H for constructing C the non-diagonal entries are 0. So the statistical dependence of the entries of Ce is negligible and the asymptotic bound on k is correct.