Choosing Parameters for NTRUEncrypt

Jeff Hoffstein[1], Jill Pipher[1], John M. Schanck[2,3], Joseph H. Silverman[1], William Whyte[3], and Zhenfei Zhang[3](✉)

[1] Brown University, Providence, USA
{jhoff,jpipher,jhs}@math.brown.edu
[2] University of Waterloo, Waterloo, Canada
[3] Security Innovation, Wilmington, USA
{wwhyte,zzhang,jschanck}@securityinnovation.com

Abstract. We describe a method for generating parameter sets, and calculating security estimates, for NTRUEncrypt. Our security analyses consider lattice attacks, the hybrid attack, subfield attacks, and quantum search. Analyses are provided for the IEEE 1363.1-2008 product-form parameter sets, for the NTRU Challenge parameter sets, and for two new parameter sets. These new parameter sets are designed to provide $\geq$ 128-bit post-quantum security.

Keywords: Public-key cryptography/NTRUEncrypt · Cryptanalysis · Parameter derivation

1 Introduction and Notation

In this note we will assume some familiarity with the details and notation of NTRUEncrypt. The reader desiring further background should consult standard references such as [11,12,16]. The key parameters are summarized in Table 1. Each is, implicitly, a function of the security parameter λ.

NTRUEncrypt uses a *ring of convolution polynomials*; a polynomial ring parameterized by a prime N, and an integer q, of the form $R_{N,q} = (\mathbb{Z}/q\mathbb{Z})[X]/(X^N - 1)$. The subscript will be dropped when discussing generic properties of such rings. We denote multiplication in R by $*$. An NTRUEncrypt public key is a generator for a cyclic R-module of rank 2, and is denoted $(1, h)$. The private key is an element of this module which is "small" with respect to a given norm and is denoted (f, g). Ring elements are written in the monomial basis. When an element of $\mathbb{Z}/q\mathbb{Z}$ is lifted to $\mathbb{Z}$, or reduced modulo p, it is identified with its unique representative in $[-q/2, q/2) \cap \mathbb{Z}$. The aforementioned norm is the 2-norm on coefficient vectors:

$$\left\| \sum_{i=0}^{N-1} a_i x^i \right\|^2 = \sum_{i=0}^{N-1} a_i^2.$$

An extended version of the paper is available at [10].

H. Handschuh (Ed.): CT-RSA 2017, LNCS 10159, pp. 3–18, 2017.
DOI: 10.1007/978-3-319-52153-4_1

Table 1.

Primary NTRUEncrypt parameters	
N, q	Ring parameters $R_{N,q} = \mathbb{Z}_q[X]/(X^N - 1)$.
p	Message space modulus.
d_1, d_2, d_3	Non-zero coefficient counts for product form polynomial terms.
d_g	Non-zero coefficient count for private key component g.
d_m	Message representative Hamming weight constraint

This norm is extended to elements $(a, b) \in R \oplus R$ as

$$\|(a, b)\|^2 = \|a\|^2 + \|b\|^2.$$

There is a large degree of freedom in choosing the structure of the private key. In previous parameter recommendations [9,16] the secret polynomials f and g have been chosen uniformly from a set of binary or trinary polynomials with a prescribed number of non-zero coefficients. These are far from the only choices. The provably secure variant of NTRUEncrypt by Stehlé and Steinfeld [17], samples f and g from a discrete Gaussian distribution, and the NTRU-like signature scheme BLISS [6] samples its private keys from a set of polynomials with a prescribed number of ± 1s and ± 2s. The reasons for such choices are varied: binary polynomials were believed to allow for a small q parameter, but the desire to increase resistance against the hybrid combinatorial attack of [15] motivated the use of larger sample spaces in both NTRUEncrypt and BLISS. In the provably secure variant the public key must be computationally indistinguishable from an invertible ring element chosen uniformly at random. The discrete Gaussian distribution has several nice analytic properties that simplify the proof of such a claim, and sampling from such a distribution is reasonably efficient.

Our parameter choices use product-form polynomials for f and for the blinding polynomial, r, used during encryption. First introduced to NTRUEncrypt in [13], product form polynomials allow for exceptionally fast multiplication in R without the use of the Fourier transform.

An extended version of the paper is available at [10] which includes the following:

– a more detailed description of NTRUEncrypt algorithms;
– a survey of other known attacks and the security level against those attacks;
– tables that list suggested q parameter; and parameters for the NTRU challenge [2];
– some additional analysis for the hybrid attack.

2 General Considerations

2.1 Ring Parameters

The only restrictions on p and q are that they generate coprime ideals of $\mathbb{Z}[X]/(X^N - 1)$. In this document we will fix $p = 3$ and only consider q that are a power of 2. This choice is motivated by the need for fast arithmetic modulo q, and by the impact of p on decryption failure probability (see Sect. 6).

For NTRUEncrypt we take N to be prime. Many ideal lattice cryptosystems use the ring $\mathbb{Z}_q[X]/(X^{2^n} + 1)$ primarily because $X^{2^n} + 1$ is irreducible over the rationals. Some complications arise from using a reducible ring modulus, but these are easily remedied.

For prime N the ring modulus factors into irreducibles over $\mathbb{Q}$ as

$$X^N - 1 = (X - 1)\Phi_N(X)$$

where $\Phi_N(X)$ is the N^{th} cyclotomic polynomial. To maximize the probability that a random f is invertible in $R_{N,q}$ we should ensure that $\Phi_N(X)$ is irreducible modulo 2, i.e. we should choose N such that (2) is inert in the N^{th} cyclotomic field. Such a choice of N ensures that f is invertible so long as $f(1) \neq 0 \pmod{2}$. It is not strictly necessary that $\Phi_N(X)$ be irreducible modulo 2, and one may allow a small number of high degree factors while maintaining a negligible probability of failure. Reasonable primes is provided in the full version of the paper [10]. Similar considerations apply for other choices of q.

2.2 Private Key, Blinding Polynomial, and Message Parameters

The analysis below will be considerably simpler if we fix how the values d_1, d_2, d_3, and d_g will be derived given N and q.

We set the notation:

$$\begin{aligned}
\mathcal{T}_N &= \{\text{trinary polynomials}\} \\
\mathcal{T}_N(d, e) &= \left\{ \begin{array}{l} \text{trinary polynomials with exactly} \\ d \text{ ones and } e \text{ minus ones} \end{array} \right\} \\
\mathcal{P}_N(d_1, d_2, d_3) &= \left\{ \begin{array}{l} \text{product form polynomials} \\ A_1 * A_2 + A_3 : A_i \in \mathcal{T}_N(d_i, d_i) \end{array} \right\}.
\end{aligned}$$

If N is fixed we will write $\mathcal{T}$, $\mathcal{T}(d, e)$, and $\mathcal{P}(d_1, d_2, d_3)$ instead.

A *product form private key* is of the form $(f, g) = (1 + pF, g)$ with $F \in \mathcal{P}_N(d_1, d_2, d_3)$ and $g \in \mathcal{T}_N(d_g + 1, d_g)$. Note that f must be invertible in $R_{N,q}$ for the corresponding public key $(1, h) = (1, f^{-1}g)$ to exist. The parameters recommended in this document ensure that, when F is sampled uniformly from $\mathcal{P}_N(d_1, d_2, d_3)$, the polynomial $1 + pF$ will always be invertible. One may optionally check that g is invertible, although this is similarly unnecessary for appropriately chosen parameters.

In order to maximize the size of the key space, while keeping a prescribed number of ± 1s in g, we take $d_g = \lfloor N/3 \rceil$. The expected number of non-zero coefficients in f is $4d_1d_2 + 2d_3$. In order to roughly balance the difficulty of the search problems for f and g (Sect. 4), we take $d_1 \approx d_2 \approx d_3$ with $d_1 = \lfloor \alpha \rceil$ where α is the positive root of $2x^2 + x - N/3$. This gives us $2d_1d_2 + d_3 \approx N/3$.

A Hamming weight restriction is placed on message representatives to avoid significant variation in the difficulty of message recovery. Message representatives are trinary polynomials; we require that the number of $+1$s, -1s, and 0s each be greater than d_m. The procedure for choosing d_m is given in Sect. 5.

3 Review of the Hybrid Attack

We consider the hybrid attack [15] to be the strongest attack against NTRUEncrypt, and believe that cost estimates for the hybrid attack give a good indication of the security of typical NTRUEncrypt parameter sets. Information on other attacks can be found from the full version of the paper [10].

Suppose one is given an NTRU public key $(1, h)$ along with the relevant parameter set. This information determines a basis for a lattice $\mathcal{L}$ of rank $2N$ generated by the rows of

$$L = \left(\begin{array}{c|c} qI_N & 0 \\ \hline H & I_N \end{array} \right) \tag{1}$$

wherein the block H is the circulant matrix corresponding to h, i.e. its rows are the coefficient vectors of $x^i * h$ for $i \in [0, N-1]$. The map $(1, h)R_{N,q} \to \mathcal{L}/q\mathcal{L}$ that sends $(a, b) \mapsto (b_0, \ldots, b_{N-1}, a_0, \ldots, a_{N-1})$ is an additive group isomorphism that preserves the norm defined in Eq. 1. As such, if one can find short vectors of $\mathcal{L}$ one can find short elements of the corresponding NTRU module.

The determinant of L is $\Delta = q^N$, giving us a Gaussian expected shortest vector of length $\lambda_1 \approx \sqrt{qN/\pi e}$, though the actual shortest vector will be somewhat smaller than this. A pure lattice reduction attack would attempt to solve Hermite-SVP[1] with factor $\lambda/\Delta^{1/2N} = \sqrt{N/\pi e}$, which is already impractical for N around 100. The experiments of [8] support this claim, they were able to find short vectors in three NTRU lattices with $N = 107$ and $q = 64$ that were generated using binary private keys. Only one of these was broken with BKZ alone, the other two required a heuristic combination of BKZ on the full lattice and BKZ on a projected lattice of smaller dimension with block sizes between 35 and 41.

Consequently the best attacks against NTRUEncrypt tend to utilize a combination of lattice reduction and combinatorial search. In this section we will review one such method from [15], known as the hybrid attack.

The rough idea is as follows. One first chooses $N_1 < N$ and extracts a block, L_1, of $2N_1 \times 2N_1$ coefficients from the center of the matrix L defined in Eq. 1.

[1] In practice q has a strong impact on the effectiveness of pure lattice reduction attacks as well. For large q the relevant problem becomes Unique-SVP which appears to be somewhat easier than Hermite-SVP. Conservative parameter generation should ensure that it is difficult to solve Hermite-SVP to within a factor of $q/\Delta^{1/2N} = \sqrt{q}$.

The rows of L_1 are taken to generate a lattice $\mathcal{L}_1$.

$$\left(\begin{array}{c|c} qI_N & 0 \\ \hline H & I_N \end{array}\right) = \left(\begin{array}{c|c|c} qI_{r_1} & 0 & 0 \\ \hline * & L_1 & 0 \\ \hline * & * & I_{r_2} \end{array}\right)$$

A lattice reduction algorithm is applied to find a unimodular transformation, U', such that $U'L_1$ is reduced, and an orthogonal transformation, Y', is computed such that $U'L_1Y' = T'$ is in lower triangular form. These transformations are applied to the original basis to produce a basis for an isomorphic lattice:

$$T = ULY = \left(\begin{array}{c|c|c} I_{r_1} & 0 & 0 \\ \hline 0 & U' & 0 \\ \hline 0 & 0 & I_{r_2} \end{array}\right) \left(\begin{array}{c|c|c} qI_{r_1} & 0 & 0 \\ \hline * & L_1 & 0 \\ \hline * & * & I_{r_2} \end{array}\right) \left(\begin{array}{c|c|c} I_{r_1} & 0 & 0 \\ \hline 0 & Y' & 0 \\ \hline 0 & 0 & I_{r_2} \end{array}\right) = \left(\begin{array}{c|c|c} qI_{r_1} & 0 & 0 \\ \hline * & T' & 0 \\ \hline * & * & I_{r_2} \end{array}\right).$$

Notice that $(g, f)Y$ is a short vector in the resulting lattice.

In general it is not necessary for the extracted block to be the central $2N_1 \times 2N_1$ matrix, and it is sometimes useful to consider blocks shifted s indices to the top left along the main diagonal. Let $r_1 = N - N_1 - s$ be the index of the first column of the extracted block and $r_2 = N + N_1 - s$ be the index of the final column. The entries on the diagonal of T will have values $\{q^{\alpha_1}, q^{\alpha_2}, \ldots, q^{\alpha_{2N}}\}$, where $\alpha_1 + \cdots + \alpha_{2N} = N$, and the α_i, for i in the range $[r_1, r_2]$, will come very close to decreasing linearly. That is to say, L_1 will roughly obey the geometric series assumption (GSA). The rate at which the α_i decrease can be predicted very well based on the root Hermite factor achieved by the lattice reduction algorithm used.[2] Clearly $\alpha_i = 1$ for $i < r_1$ and $\alpha_i = 0$ for $i > r_2$. By the analysis in [10] we expect

$$\alpha_{r_1} = \frac{1}{2} + \frac{s}{2N_1} + 2N_1 \log_q(\delta) \tag{2}$$

$$\alpha_{r_2} = \frac{1}{2} + \frac{s}{2N_1} - 2N_1 \log_q(\delta), \tag{3}$$

and a linear decrease in-between. The profile of the basis will look like one of the examples in Fig. 1.

By a lemma of Furst and Kannan (Lemma 1 in [15]), if $y = uT + x$ for vectors u and x in $\mathbb{Z}^{2N}$, and $-T_{i,i}/2 < x_i \le T_{i,i}/2$, then reducing y against T with Babai's nearest plane algorithm will yield x exactly. Thus if v is a shortest vector in $\mathcal{L}$ and $\alpha_{r_2} > \log_q(2\|v\|_\infty)$, it is guaranteed that v can be found by enumerating candidates for its final $K = 2N - r_2$ coefficients. Further knowledge about v can also diminish the search space. For example, if it is known that there is a trinary vector in $\mathcal{L}$, and $\alpha_{r_2} > \log_q(2)$, then applying Babai's nearest plane algorithm to some vector in the set $\{(0|v')T - (0|v') : v' \in \mathcal{T}_K\}$ will reveal it.

The optimal approach for the attacker is determined by the balancing the cost of combinatorial search on K coordinates against the cost of lattice reduction that results in a sufficiently large α_{2N-K}. Unsurprisingly, naïve enumeration of the possible v' is not optimal.

[2] A lattice reduction algorithm that achieves root Hermite factor δ returns a basis with $\|\boldsymbol{b}_1\|_2 \approx \delta^n \det(\Lambda)^{1/n}$.

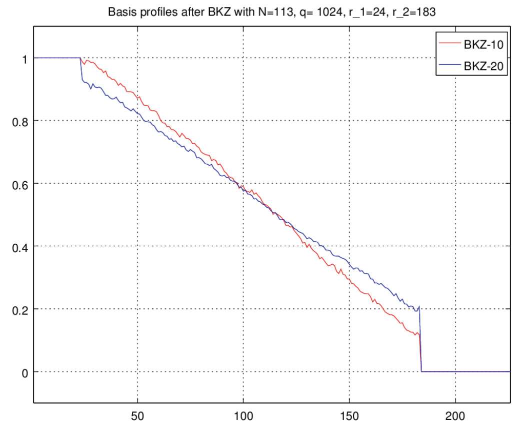

Fig. 1. Log length of i^{th} Gram-Schmidt vector, $\log_q(\|b_i^*\|)$.

4 Meet in the Middle Search

The adaptation of meet-in-the-middle search algorithms to the structure of binary NTRU keys is due to Odlyzko and described in [14]. Generalizations to other private key types are described by Howgrave-Graham in [15]; this is the presentation we follow here. The key idea is to decompose the search space S as $S \subseteq S' \oplus S'$ for some set S' such that $|S'| \approx \sqrt{|S|}$. If s_1 and s_2 are elements of S' such that $s_1 + s_2 = f$, and (f, g) is an element with small coefficients in the NTRU module generated by $(1, h)$, then $(s_1, s_1 * h) = (f, g) - (s_2, s_2 * h)$. In particular, when the coefficients of g are trinary, this implies that $s_1 * h \approx -s_2 * h$ coordinate-wise.

Under the assumption that all approximate collisions can be detected, a meet in the middle search on the full product form NTRUEncrypt key space would require both time and memory of order $O(\sqrt{|\mathcal{P}_N(d_1, d_2, d_3)|})$.

A meet in the middle search is also possible on a basis that has been preprocessed for the hybrid attack as in Eq. 2. The assumption that all approximate collisions can be detected will turn out to be untenable in this case, however, in the interest of deriving conservative parameters we will assume that this complication does not arise. Let $\Pi : \mathbb{Z}^N \to \mathbb{Z}^K$ be a projection[3] onto K coordinates of $\mathbb{Z}^N$. Let $P_\Pi = \{v\Pi : v \in \mathcal{P}_N(d_1, d_2, d_3)\}$. The f component of the private key is guaranteed to appear in P_Π, so the expected time and memory required for the attack is $O(\sqrt{|P_\Pi|})$. That said, estimating the size of P_Π is non-trivial.

We may also consider an adversary who attempts this attack on the lattice corresponding to $(1, h^{-1})$ and searches for the g component of the private key instead. This may in fact be the best strategy for the adversary, because while $|\mathcal{P}_N(d_1, d_2, d_3)| < |\mathcal{T}_N(d_g + 1, d_g)|$ for parameters of interest to us, the presence

[3] We will abuse notation slightly and allow Π to act on elements of R by acting on their coefficient vectors lifted to $\mathbb{Z}^N$.

of coefficients not in $\{-1,0,1\}$ in product form polynomials leads to a large increase in the relative size of the projected set.

In either case we assume that it is sufficient for the adversary to search for trinary vectors, and that they may limit their search to a projection of $\mathcal{T}_N(d,e)$ for some (d,e). When targeting g we have $d = d_g + 1$, $e = d_g$, and when targeting f we have that both d and e are approximately $2d_1d_2 + d_3$. Clearly when $d = e = N/3$, and $N \gg K$, we should expect that the projection of a uniform random element of $\mathcal{T}_N(d,e)$ onto K coordinates will look like a uniform random element of $\mathcal{T}_K$. For such parameters, the size of the set that must be enumerated in the meet-in-the-middle stage is $\approx 3^{K/2}$.

For $d \neq N/3$, or for large K, not all trinary sequences are equally likely, and the adversary may choose to target a small set of high probability sequences. Consequently we must estimate the size of the set of elements that are typical under the projection. Fix N, K, Π, d, and e and let $\mathcal{S} = \mathcal{T}_N(d,e)$. Let $p : \mathcal{T}_K \to \mathbb{R}$ be the probability mass function on $\mathcal{T}_K$ induced by sampling an element uniformly at random from $\mathcal{S}$ and projecting its coefficient vector onto the set of K coordinates fixed by Π. We will estimate the size of the search space in the hybrid attack as, roughly, $2^{H(p)}$, where $H(p)$ is the Shannon entropy of p.

Let $\mathcal{S}_\Pi(a,b)$ be the subset of $\mathcal{S}$ consisting of vectors, v, such that $v\Pi$ has exactly a coefficients equal to $+1$ and b coefficients equal to -1. By the symmetry of $\mathcal{S}$ under coordinate permutations we have that $p(v\Pi) = p(v'\Pi)$ for all pairs $v, v' \in \mathcal{S}_\Pi(a,b)$. We choose a fixed representative of each type: $v_{a,b} = v\Pi$ for some $v \in \mathcal{S}_\Pi(a,b)$, and write

$$p(v_{a,b}) = \frac{1}{\binom{K}{a}\binom{K-a}{b}} \frac{|\mathcal{S}_\Pi(a,b)|}{|\mathcal{S}|} = \frac{\binom{N-K}{d-a}\binom{N-K-d+a}{d-b}}{\binom{N}{d}\binom{N-d}{d}}.$$

As there are exactly $\binom{K}{a}\binom{K-a}{b}$ distinct choices for $v_{a,b}$ this gives us:

$$H(p) = -\sum_{v \in \mathcal{T}_K} p(v)\log_2 p(v) = -\sum_{0 \le a,b \le d} \binom{K}{a}\binom{K-a}{b} p(v_{a,b})\log_2 p(v_{a,b}). \tag{4}$$

The size of the search space is further decreased by a factor of N since $x^i * g$ is likely to be a distinct target for each $i \in [0, N-1]$. Hence in order to resist the hybrid meet-in-the-middle attack we should ensure

$$\frac{1}{2}(H(p) - \log_2(N)) \ge \lambda.$$

The only variable not fixed by the parameter set itself in Eq. 4 is K. In order to fix K we must consider the cost of lattice reduction.

The block to be reduced is of size $(r_2 - r_1) \times (r_2 - r_1)$ where $r_2 = 2N - K$ and $r_1 = \lambda$. Recall that $s = N - (r_1 + r_2)/2$, and $N_1 = (r_2 - r_1)/2$. Having fixed these parameters we can use Eq. 3 to determine the strength of the lattice reduction needed to ensure that α_{r_2} is sufficiently large to permit recovery of a trinary vector. In particular, we need $\alpha_{r_2} = \frac{N_1+s}{2N_1} - 2N_1\log_q(\delta) \ge \log_q(2)$, which implies that

$$\log_2(\delta) \leq \frac{N_1 + s}{4N_1^2}\log_2(q) - \frac{1}{2N_1}. \tag{5}$$

Translating the required root Hermite factor, δ, into a concrete bit-security estimate is notoriously difficult. However there seems to be widespread consensus on the values that are currently out of reach for common security parameters. As such one might use the following step function as a first approximation:

$$\delta^*(\lambda) = \begin{cases} 1.009 & \text{if } \lambda \leq 60 \\ 1.008 & \text{if } 60 < \lambda \leq 80 \\ 1.007 & \text{if } 80 < \lambda \leq 128 \\ 1.005 & \text{if } 128 < \lambda \leq 256 \\ 1 & \text{otherwise.} \end{cases}$$

A more refined approach involving a BKZ simulator, from [4], is used in Sect. 8.1.

Rewriting Eq. 5 in terms of N, q, r_1 and K we define:

$$\log_2(\eta(N,q,r_1,K)) = \frac{(N - r_1)\log_2(q)}{4N^2 - 4N(K + r_1) + (K^2 + 2r_1K + r_1^2)} - \frac{1}{2N - (K + r_1)}.$$

Conclusion: A parameter set resists hybrid meet-in-the-middle attacks on private keys if Eq. 4 is satisfied and

$$1 < \eta(N,q,r_1,K) \leq \delta^*(r_1). \tag{6}$$

5 Rejecting Sparse (and Dense) Message Representatives

The parameter sets in this paper specify the exact number of 1's and -1's in each of r_1, r_2, r_3, which reveals the quantity $r(1)$, that is, the polynomial r evaluated at 1. As an encrypted message has the form $e = pr * h + m$, the value $m(1)$ modulo q is revealed by the known quantities $r(1), e(1), h(1)$. The value $m(1)$ in turn reveals the difference between the number of 1's and the number of -1's in the message representative.

We assume the message representative is uniformly distributed over $\mathcal{T}_N$. The expected value of $m(1)$ is zero, but for large $|m(1)|$, the size of the search space for m decreases, making a meet in the middle search for (r, m) easier. We assume that the adversary observes a very large number of messages and can freely condition their attack on the value of $m(1)$ regardless of the probability that a uniform random message representative takes that value.

In addition, we forbid the number of $+1$s, -1s, or 0s to be less than a given parameter, d_m. The choice of d_m depends primarily affects resistance against hybrid combinatorial attacks, but d_m also has an impact on decryption failure probability, as will be discussed in Sect. 6.

The calculation for determining resistance against hybrid combinatorial attacks is very similar to that leading up to Eq. 4, but there are two key differences. First, in Sect. 4 we were primarily concerned with validating the security of the obvious choice $d_g = \lfloor N/3 \rceil$. Here we will need to search for d_m. Second, having fixed d_m we need to condition the distribution of projected elements on the value of $m(1)$.

The search space for d_m can be constrained by imposing an arbitrary upper bound on the probability of a failure. Such a failure is roughly as expensive as a full encryption, so d_m should be chosen to ensure that failures are rare.

Let $I(d_m) = \{(i,j) : d_m \leq i < (N - 2d_m),\ d_m \leq j < (N - d_m - i)\}$. We will only consider d_m satisfying:

$$2^{-10} \geq 1 - 3^{-N} \left(\sum_{(i,j) \in I(d_m)} \binom{N}{i} \binom{N-i}{j} \right)$$

Let K be the value derived in Sect. 4. Fix Π and let $\mathcal{S}(e_1, e_2; a, b)$ be the set of projections of elements of $\mathcal{T}(e_1, e_2)$ with a ones and b minus ones. Let $\mathcal{M}$ be the subset of $\mathcal{T}_N$ satisfying the d_m constraint. Let $p : \mathcal{T}_K \times \mathbb{Z} \times \mathbb{Z} \to \mathbb{R}$ be the probability mass function given by

$$p(v, e_1, e_2) = \text{Prob}_{m \leftarrow_\$ \mathcal{M}} \left(m\Pi = v \text{ and } m \in \mathcal{T}_N(e_1, e_2) \right),$$

i.e. $p(v, e_1, e_2)$ is the probability that an m sampled uniformly from $\mathcal{M}$ has e_1 ones, e_2 minus ones, and is equal to v under projection. If the information leakage from $m(1)$ determined e_1 and e_2 then we could use essentially the same analysis as Sect. 4 and our security estimate would be

$$\frac{1}{2} \min_{(e_1,e_2) \in I(d_m)} H(p_{d_m} | e_1, e_2).$$

However the adversary only learns $m(1) = e_1 - e_2$, so we will account for their uncertainty about whether $m \in \mathcal{T}(e_1, e_2)$ given $m(1) = e_1 - e_2$.

The marginal distribution on e_1 and e_2 conditioned on the event $m(1) = y$ is

$$\begin{aligned} q(e_1, e_2 | m(1) = y) &= \sum_{v \in \mathcal{T}_K} p(v, e_1, e_2 | m(1) = y) \\ &= \binom{N}{e_1} \binom{N - e_1}{e_2} \left(\sum_{\substack{i - j = y \\ (i,j) \in I(d_m)}} \binom{N}{i} \binom{N-i}{j} \right)^{-1}. \end{aligned}$$

Conclusion: As such we will consider a parameter set secure against hybrid meet-in-the-middle attacks on messages provided that:

$$\lambda \ \leq \ \min_{y} \min_{\substack{e_1 - e_2 = y \\ (e_1, e_2) \in I(d_m)}} \frac{1}{2} H(p | e_1, e_2) - \log_2 q(e_1, e_2 | m(1) = y). \tag{7}$$

Evaluating this expression is considerably simplified by noting that local minima will be found at the extremal points: $|e_1 - e_2| = N - 3d_m$ and $e_1 = e_2 \approx N/3$.

Note that unlike the estimate in Sect. 4 we do not include a $-\log_2(N)$ term to account for rotations of m.

6 Estimating the Probability of Decryption Failure

As remarked earlier, in order for decryption to succeed the coefficients of

$$a = p * (r * g + m * F) + m$$

must have absolute value less than $q/2$.

Assuming $p \in \mathbb{Z}$, and trinary g and m, the triangle inequality yields:

$$\|a\|_\infty \leq p\left(\|r\|_1\|g\|_\infty + \|F\|_1\|m\|_\infty\right) + 1 = p\left(\|r\|_1 + \|F\|_1\right) + 1.$$

Thus with product form r and F decryption failures can be avoided entirely by ensuring $(q-2)/2p > 8d_1d_2 + 4d_3$. However, since ciphertext expansion scales roughly as $N\log_2(q)$, it can be advantageous to consider probabilistic bounds as well. The probability

$$\text{Prob}\left(\text{a given coefficient of } r * g + m * F \text{ has absolute value } \geq c\right)$$

can be analyzed rather well by an application of the central limit theorem. This was done for the case of trinary r, g, m, F in [9]. Here we provide a modified analysis for the case where the polynomials r and F take a product form. In particular, we assume that $r = r_1 * r_2 + r_3$, $F = F_1 * F_2 + F_3$, where each r_i and F_i has exactly d_i coefficients equal to 1, d_i coefficients equal to -1, and the remainder equal to 0.

Let X_k denote a coefficient of $r * g + m * F$. The spaces from which r and m are drawn are invariant under permutations of indices, so the probability that $|X_k| > c$ does not depend on the choice of k.[4] Note that X_k has the form

$$X_k = (r_1 * r_2 * g)_k + (r_3 * g)_k + (F_1 * F_2 * m)_k + (F_3 * m)_k,$$

and each term in the sum is itself a sum of either $4d_1d_2$ or $2d_3$ (not necessarily distinct) coefficients of g or m. For instance, $(r_1 * r_2 * g)_k = \sum_{i,j}(r_1)_i(r_2)_j(g)_{(k-i-j)}$ and only the $4d_1d_2$ pairs of indices corresponding to non-zero coefficients of r_1 and r_2 contribute to the sum. We can think of each index pair as selecting a sign $\epsilon(i)$ and an index $a(i)$ and rewrite the sum as $(r_1 * r_2 * g)_k = \sum_{i=1}^{4d_1d_2} \epsilon(i)(g)_{a(i)}$. While the terms in this sum are not formally independent (since a may have repeated indices, and g has a prescribed number of non-zero coefficients) extensive experiments show that the variance of $(r_1 * r_2 * g)_k$ is still well approximated by treating $(g)_{a(i)}$ as a random coefficient of g, i.e. as taking a non-zero value with probability $(2d_g+1)/N$:

$$\mathbb{E}\left[(r_1 * r_2 * g)_k^2\right] \approx \sum_{i=1}^{4d_1d_2} \mathbb{E}\left[(\epsilon(i)(g)_{a(i)})^2\right] = \sum_{i=1}^{4d_1d_2} \mathbb{E}\left[(g)_{a(i)}^2\right] = 4d_1d_2 \cdot \frac{2d_g+1}{N}$$

Nearly identical arguments can be applied to compute the variances of the other terms of Eq. 6, although some care must be taken with the terms involving m. While an honest party will choose m uniformly from the set of trinary

[4] The X_k for different k have the same distribution, but they are not completely independent. However, they are so weakly correlated as to not affect our analysis.

polynomials, m could be chosen adversarily to maximize its Hamming weight and hence the probability of a decryption failure. Due to the d_m constraint (Sect. 5), the number of non-zero coefficients of m cannot exceed $N - d_m$. As such we model the coefficients of m as taking ± 1 each with probability $(1 - d_m/N)$ and 0 with probability d_m/N.

With these considerations the variance of $(r_1 * r_2 * g)_k + (r_3 * g)_k$ is found to be $\sigma_1^2 = (4d_1d_2 + 2d_3) \cdot \frac{2d_g+1}{N}$, and the variance of $(F_1 * F_2 * m)_k + (F_3 * m)_k$ is found to be $\sigma_2^2 = (4d_1d_2 + 2d_3) \cdot (1 - \frac{d_m}{N})$. Both terms are modeled as sums of i.i.d. random variables, and the d_i are chosen such that $4d_1d_2 + 2d_3 \approx 2N/3$, so for sufficiently large N the central limit theorem suggests that each term will have a normal distribution. Finally X_k can be expected to be distributed according to the convolution of these two normal distributions, which itself is a normal distribution with variance

$$\sigma^2 = \sigma_1^2 + \sigma_2^2 = (4d_1d_2 + 2d_3) \cdot \frac{N - d_m + 2d_g + 1}{N}.$$

The probability that a normally distributed random variable with mean 0 and standard deviation σ exceeds c in absolute value is given by the complementary error function, specifically $\texttt{erfc}(c/(\sqrt{2}\sigma))$. Applying a union bound, the probability that any of the N coefficients of $r * g + m * f$ is greater than c is bounded above by $N \cdot \texttt{erfc}(c/(\sqrt{2}\sigma))$.

Conclusion: To have negligible probability of decryption failure with respect to the security parameter, λ, we require

$$N \cdot \texttt{erfc}((q-2)/(2\sqrt{2} \cdot p \cdot \sigma)) < 2^{-\lambda} \tag{8}$$

where $\sigma = \sigma(N, d_1, d_2, d_3, d_g, d_m)$ as in Eq. 6.

7 Product Form Combinatorial Strength

The search space for a triple of polynomials F_1, F_2, F_3 where each polynomial F_i has d_i 1's and d_i -1's is of size:

$$|\mathcal{P}_N(d_1, d_2, d_3)| = \binom{N}{d_1}\binom{N-d_1}{d_1}\binom{N}{d_2}\binom{N-d_2}{d_2}\binom{N}{d_3}\binom{N-d_3}{d_3}.$$

Thus a purely combinatorial meet-in-the-middle search on product form keys can be performed in time and space $O(\sqrt{|\mathcal{P}_N(d_1, d_2, d_3)|/N})$, where we have divided by N to account for the fact that rotations of a given triple are equivalent.

Finally, one could construct a $3N$ dimensional lattice attack by considering the lattice generated by linear combinations of the vectors $(1, 0, f_1 * h), (0, 1, h), (0, 0, q)$, where each entry corresponds to N entries in the lattice. The vector (f_2, f_3, g) will be a very short vector, but the increase of the dimension of the lattice by N, without any corresponding increase in the determinant of the lattice, leads to a considerably harder lattice reduction problem. As this attack also requires a correct guess of f_1 we will not consider it further.

8 Explicit Algorithm for Computing Parameters

Algorithm 1 determines the smallest recommended N that allows for k bit security. Additional details, such as recommendations on how to efficiently perform the search in Line 16, may be found in our implementation available at [1].

8.1 Sample Parameter Generation

We will ignore the implicit outer loop over security parameters and consider the case of $N = 401$ starting from Line 3.

Our recommendations for the key structure suggests taking $d_g = 134$, $d_1 = 8$, $d_2 = 8$, $d_3 = 6$. Taking $d_m = 102$ satisfies Eq. 5 with a probability of $2^{-10.4}$ of rejecting a message representative due to its coefficient sum. A direct meet-in-the-middle attack on the product form key space will involve testing approximately 2^{145} candidates. As this is an upper bound on the security of the parameter set we will ensure that our decryption failure probability is less than 2^{-145}. This implores us to take $q = 2048$, for which there is, by Eq. 8, a decryption failure probability of 2^{-217}.

In order to finish the parameter derivation we need a tighter estimate on its security. It may be significantly less than 145, in which case we may be able to reduce q.

We estimate the security of the parameter set by minimizing the adversary's expected cost over choices of the hybrid attack parameter K. Equation 4 specifies, for each K, the root Hermite factor, δ, that must be reached during the lattice reduction phase of the hybrid attack in order for the combinatorial stage to be successful. We use the BKZ-2.0 simulator of [4] to determine the blocksize and number of rounds of BKZ that will be required to reach root Hermite factor.

To turn the blocksize and iteration count into a concrete security estimate we need estimates on the number of nodes visited per call to the enumeration subroutine of BKZ. Table 2 summarizes upper bounds given by Chen and Nguyen in [4] and in the full version of the same paper [5]. The estimates of the full version are significantly lower than the original, and have perhaps not recieved the same scrutiny. In what follows we will consider the implications of both estimates.

Table 2. Upper bounds on $\log_2$ number of nodes enumerated in one call to enumeration subroutine of BKZ-2.0 as reported in the original and full versions of the paper.

β	100	110	120	130	140	150	160	170	180	190	200	210	220	230	240	250
LogNodes(β) [5]	39	44	49	54	60	66	72	78	84	96	99	105	111	120	127	134

To facilitate computer search for parameters we fit curves to the estimates in Table 2, and following [4] we estimate the per-node cost, as 2^7 operations. The resulting predictions for the cost of the lattice reduction stage, in terms of the

Algorithm 1. NTRUEncrypt parameter generation

```
Input: Desired security level $k$.
 1: Let $n_j$ be the $j^{th}$ value, ordered by magnitude, from the list of first 100 primes
    $> 100$ for which $\mathrm{ord}_{(\mathbb{Z}/N\mathbb{Z})^*}(2) = (N-1)$, i.e. (2) is inert.
 2: Set $j = 1$.
 3: Set $N = n_j$.
 4: Set $d_g = \lfloor \frac{N}{3} \rceil$.
 5: Set $d_1 = \left\lceil \frac{1}{4}\left(\sqrt{1+\frac{8N}{3}} - 1\right) \right\rceil$ {The next integer above the positive root of $2x^2 +$
    $x - N/3$.}
 6: Set $d_2 = \left\lceil \left(\frac{N}{3} - d_1\right)/(2d_1) \right\rceil$.
 7: Set $d_3 = \max\left(\left\lceil \frac{d_1}{2} + 1 \right\rceil, \left\lceil \frac{N}{3} - 2d_1d_2 \right\rceil\right)$.
 8: Set $d_m$ to be the largest value satisfying Eq. 5.
 9: Set $k_1 = \left\lfloor \frac{1}{2}\log_2\left(|\mathcal{P}_N(d_1,d_2,d_3)|/N\right) \right\rfloor$. {Cost of direct combinatorial search gives
    an upper bound on the security.}
10: if $k_1 < k$ then
11:   Increment $j$.
12:   Goto line 3.
13: end if
14: Set $\sigma$ according to Eq. 6.

    $$\sigma = \left( (4d_1d_2 + 2d_3) \cdot \frac{N - d_m + 2d_g + 1}{N} \right)^{1/2}.$$

15: Set $q$ to be the smallest power of 2 satisfying

    $$N \cdot \mathtt{erfc}\left((q-2)/(6\sqrt{2}\sigma)\right) < 2^{-k_1}.$$

    {Estimate security}
16: Search for a hybrid parameter $K$ that minimizes the maximum of the cost estimates
    for hybrid attacks. Equation 4 gives the cost of the lattice reduction, and Eqs. 4
    and 7 give the cost of combinatorial search for key- and message-recovery attacks
    respectively. Let $k_2$ be the corresponding security estimate.
17: if $k > \min(k_1,\ k_2)$ then
18:   Increment $j$.
19:   Go to Line 3.
20: end if
21: Let $q' = q/2$.
22: if $N \cdot \mathtt{erfc}\left((q'-2)/(6\sqrt{2}\sigma)\right) < 2^{-k}$ then
23:   Set $q = q'$
24:   Repeat security estimate (Line 16) with modulus $q'$ and set $k_2$ equal to the
      result.
25:   Go to Line 17.
26: end if
Output: $[N, q, d_1, d_2, d_3, d_g, d_m]$.
```

blocksize, the dimension of the sublattice to be reduced, and number of rounds are thus:

$$\mathrm{LogNodes}(\beta) = 0.12081 \cdot \beta \log_2(\beta) - 0.42860 \cdot \beta$$
$$\mathrm{BKZCost}(dim, \beta, rounds) = \mathrm{LogNodes}(\beta) + \log_2(dimension \cdot rounds) + 7.$$

Finally our security estimate requires a search over K to balance the cost of lattice reduction against the cost of combinatorial search given by Eq. 4.

Fixing $K = 154$ the BKZ-2.0 simulator suggests that 10 rounds of BKZ-197 will achieve to the requisite $\delta = 1.0064$. The BKZCost estimate suggets that this reduction will require 2^{116} operations, matching the cost of 2^{116} given by Eq. 4 for the combinatorial search step.

We find that we cannot decrease q without violating the constraint on the decryption failure probability, and we are done.

The parameter set we have just (re-)derived originally appeared in the EESS #1 standard at the 112 bit security level. All four product-form parameter sets from EESS #1 are reviewed in Table 3 with security estimates following the above analysis. Note that while the algorithm in Sect. 8 rederives the $N = 401$ parameter set almost exactly (d_g is 133 in EESS #1), this is not true for the $N = 593$ and $N = 743$ parameter sets. In particular, all four of the published parameter sets take $q = 2048$, and this does not lead to a formally negligible probability of decryption failure for $N = 593$ or $N = 743$. Note also that the number of prime ideals lying above (2) is more than recommended for $N = 439$ and $N = 593$. Table 3 presents security estimates for the standardized parameters rather than those that would be output by the algorithm of Sect. 8.

Table 3.

EESS #1 Parameter sets and security estimates

Original security est	N	q	$(d_1, d_2, d_3, d_g, d_m)$	Hybrid attack parameters					Product form search cost	$\log_2$ dec. fail prob
				Dim	β	Rounds	K	Cost		
112	401	2048	(8, 8, 6, 133, 101)	532	197	10	154	**116**	145	-217
128	439	2048	(9, 8, 5, 146, 112)	571	221	10	174	**133**	147	-195
192	593	2048	(10, 10, 8, 197, 158)	732	316	8	261	201	**193**	-139
256	743	2048	(11, 11, 15, 247, 204)	880	407	8	350	272	**256**	-112

9 New Parameters

The parameter derivations above do not take quantum adversaries into consideration. The time/space tradeoff in the hybrid attack can be replaced (trivially) by a Grover search to achieve the same asymptotic time complexity as the hybrid attack with a space complexity that is polynomial in N. One may expect that a quantum time/space tradeoff could do even better, however this seems unlikely given the failure of quantum time/space tradeoffs against collision problems in

Table 4.

Post-quantum parameter sets and security estimates

Classical security est	Quantum security est	N	q	$(d_1, d_2, d_3, d_g, d_m)$	Hybrid attack parameters					Product form search cost	$\log_2$ dec. fail prob
					Dim	β	Rounds	K	Cost		
128	128	443	2048	(9, 8, 5, 148, 115)	575	222	11	177	**133**	147	-196
192	128	587	2048	(10, 10, 8, 196, 157)	723	311	9	258	197	**193**	-139
256	128	743	2048	(11, 11, 15, 247, 204)	880	407	8	350	272	**256**	-112

other domains [3]. Several proposals in this direction have been made, such as [7], however these assume unrealistic models of quantum computation. For now, it seems that the best quantum attack on NTRUEncrypt is the hybrid attack with meet-in-the-middle search replaced by Grover search in the K^{th} projected lattice.

Fluhrer has noted that there are weaknesses in the EESS #1 parameter sets assuming worst-case cost models for quantum computation [7]. In particular, if one Grover iteration is assigned cost equivalent to one classical operation, such as a multiplication in R, then attacks on the hash functions used in key generation and encryption can break the EESS #1 parameter sets.

Developing a realistic quantum cost model is outside the scope of this work. However we can easily provide parameter sets that are secure in Fluhrer's model. Since this model is in some sense a worst-case for quantum computation (it assigns the smallest justifiable cost to quantum operations) the quantum security estimates can be assumed to be quite conservative. In addition to using the parameters in Table 4 one must ensure that pseudorandom polynomial generation functions are instantiated with SHA-256, and that the message is concatenated with a random string b that is at least 256 bits. One should also ensure that any deterministic random bit generators used in key generation or encryption are instantiated with at least 256 bits of entropy from a secure random source.

The parameter sets for $N = 443$ and $N = 587$ in Table 4 are new, $N = 743$ is the same as ees743ep1 from EESS #1.

References

1. NTRU OpenSource Project.online. https://github.com/NTRUOpenSource Project/ntru-crypto
2. 2015. https://www.ntru.com/ntru-challenge/
3. Bernstein, D.J.: Cost analysis of hash collisions: will quantum computers make SHARCS obsolete? (2009). http://cr.yp.to/papers.html#collisioncost
4. Chen, Y., Nguyen, P.Q.: BKZ 2.0: better lattice security estimates. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 1–20. Springer, Heidelberg (2011). doi:10.1007/978-3-642-25385-0_1
5. Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better lattice security estimates (full version) (2011). http://www.di.ens.fr/~ychen/research/Full_BKZ.pdf

6. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal Gaussians. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 40–56. Springer, Heidelberg (2013). doi:10.1007/978-3-642-40041-4_3
7. Fluhrer, S.R.: Quantum cryptanalysis of NTRU. IACR Cryptology ePrint Archive, 2015:676 (2015)
8. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008). doi:10.1007/978-3-540-78967-3_3
9. Hirschhorn, P.S., Hoffstein, J., Howgrave-Graham, N., Whyte, W.: Choosing NTRUEncrypt parameters in light of combined lattice reduction and MITM approaches. In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 437–455. Springer, Heidelberg (2009). doi:10.1007/978-3-642-01957-9_27
10. Hoffstein, J., Pipher, J., Schanck, J.M., Silverman, J.H., Whyte, W., Zhang, Z.: Choosing Parameters for NTRUEncrypt (full version). IACR Cryptology ePrint Archive 2015:708 (2015)
11. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: a ring-based public key cryptosystem. In: Buhler, J.P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 267–288. Springer, Heidelberg (1998). doi:10.1007/BFb0054868
12. Hoffstein, J., Silverman, J.H.: Optimizations for NTRU (2000)
13. Hoffstein, J., Silverman, J.H.: Random small hamming weight products with applications to cryptography. Discrete Appl. Math. **130**(1), 37–49 (2003)
14. Hoffstein, J., Silverman, J.H., Whyte, W.: Provable Probability Bounds for NTRUEncrypt Convolution (2007). http://www.ntru.com
15. Howgrave-Graham, N.: A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 150–169. Springer, Heidelberg (2007). doi:10.1007/978-3-540-74143-5_9
16. Howgrave-Graham, N., Silverman, J.H., Whyte, W.: Choosing parameter sets for NTRUEncrypt with NAEP and SVES-3. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 118–135. Springer, Heidelberg (2005). doi:10.1007/978-3-540-30574-3_10
17. Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 27–47. Springer, Heidelberg (2011). doi:10.1007/978-3-642-20465-4_4