

Performance Improvements and a Baseline Parameter Generation Algorithm for NTRUSign

Jeff Hoffstein, Nicholas Howgrave-Graham, Jill Pipher, Joseph H. Silverman, William Whyte
NTRU Cryptosystems,
5 Burlington Woods, MA 01803.
Submitted to Eurocrypt 2005.

No Institute Given

1 Introduction

The NTRUSign signature scheme was introduced in [8]. The original presentation gave a theoretical description of the scheme and an analysis of its security, along with several parameter choices which gave a claimed security level of 80 bits. The paper did not give a general recipe for generating parameter sets to a specific level of security. In line with recent research on NTRUEncrypt [9], this paper presents an outline of such a recipe for NTRUSign. NTRUSign has many more implementation options than NTRUEncrypt, and research is ongoing to improve the efficiency of NTRUSign operations at a given security level. This paper is therefore not intended to be the last word on parameter generation for NTRUSign, but to provide a specific parameter generation algorithm whose output we believe to have stated security properties. We also present certain technical advances which we intend to build upon in subsequent papers.

In addition to outlining the parameter generation algorithm, this paper makes the following contributions. First, we note that the “transpose lattice” of [8] has greater security against key recovery by lattice reduction attacks than the “standard lattice”, because its shortest vector is longer than the shortest vector in the standard lattice by a factor of, effectively, $N^{\frac{1}{4}}$. This allows a reduction in the size of the public key while maintaining the security of the key against lattice attacks. Along with this increased lattice security, we switch to the use of trinary form for private keys, to increase the combinatorial security possible for a given key size. Second, we note that the structure of signatures in the transpose lattice leads naturally to a slightly different definition of the norm of a signature. Using this norm changes the asymptotic properties of signatures, so that the signature a valid signer can create improves on the expected norm of a forgery not by a constant factor, as was the case for the standard lattice of [8], but by a factor of (also) $N^{\frac{1}{4}}$. Clearly, in the asymptotic case, this is a change of great significance. For the non-asymptotic cases under consideration here, the use of this norm enables us to reduce the lattice dimension necessary to give security against lattice-based signature forgery attacks, validating our ability to reduce bandwidth. Third, we introduce an improved combinatorial method for signature forgery, similar to the methods described in [7]. This attack roughly square-roots the time necessary to forge a signature by combinatorial means. Fourth, we improve the analysis of the number of signatures an attacker must collect to mount the best currently known transcript attack. When there is one perturbation this number is polynomial (with exponent 5) in the security parameter k . Conservative theoretical and experimental evidence specifically puts this lower bound well above 2^{30} for the parameter choices mentioned here.

Our object is to make this paper as self-contained as possible, but we will occasionally refer to [8] for details.

2 Outline of the parameter generation algorithm

2.1 Sketch of NTRUSign: underlying mathematics

We briefly review NTRUSign to establish the parameters that must be output by a parameter generation algorithm.

NTRUSign is defined in terms of operations on the set R of polynomials of degree (strictly) less than N and having integer coefficients. (The parameter N is fixed.) The basic operations on these polynomials are addition and convolution multiplication. Convolution multiplication $*$ of two polynomials f and g is defined by taking the coefficient of X^k in $f * g$ to equal

$$(f * g)_k \equiv \sum_{i+j \equiv k \pmod{N}} f_i \cdot g_j \quad (0 \leq k < N).$$

If one of the polynomials has all coefficients chosen from the set $\{0, \pm 1\}$ we will refer to the convolution as being *ternary* while if coefficients of the polynomials are reduced modulo q for some integer q , we will refer to the convolution as being *modular*.

In more mathematical terms, R is the quotient ring $R = \mathbb{Z}[X]/(X^N - 1)$. Every element of R has a unique representation as a polynomial $r = \sum_{i=0}^{N-1} r_i X^i$. Thus a natural measure of size in R is the centered Euclidean norm of the vector of coefficients

$$\|r\|^2 = \sum_{i=0}^{N-1} r_i^2 - (1/N) \left(\sum_{i=0}^{N-1} r_i \right)^2.$$

If $r \in R$ satisfies $\|r\|^2 = \mathcal{O}(N)$ we will say that r is *short*. This norm possesses the attractive pseudo-multiplicative property $\|r * s\| \approx \|r\| \cdot \|s\|$ for most choices of short $r, s \in R$.

Let us now fix integers N and q , along with some $h \in R$. A lattice can be associated to N, q, h as follows:

$$L_h = L_h(N, q) = \{(r, r') \mid r \in R, r' \equiv r * h \pmod{q}\}.$$

This has a natural embedding in \mathbb{Z}^{2N} and is referred to as a convolution modular lattice. It has dimension equal to $2N$ and determinant equal to q^N .

The norm function $\|\cdot\| : R \rightarrow \mathbb{R}$ can be naturally extended to L_h . For $(r, r') \in L_h(N, q)$ we define $\|(r, r')\|$ to be the minimal value of

$$(\|r + k_1 q\|^2 + \|r' + k_2 q\|^2)^{1/2}$$

for $k_1, k_2 \in R$.

Our parameter generation algorithm must give values for N and q .

2.2 Sketch of NTRUSign: key generation

For a fixed parameter $\delta > 0$, let \mathcal{S}_δ denote the subset of R consisting of polynomials r with the property that

$$\delta N - 1 < \|r\|^2 < \delta N + 1.$$

A particularly useful class of lattices that we will call NTRUSign lattices can now be constructed as follows. Choose $f, g \in \mathcal{S}_\delta$ such that f, g are invertible modulo q , i.e so that $f^{-1}, g^{-1} \in R$ exist with $f * f^{-1} \equiv g * g^{-1} \equiv 1 \pmod{q}$. The process of computing f^{-1}, g^{-1} from f, g is described in [5].

Now construct a corresponding $F, G \in R$ such that $f * G - g * F = q$. A method of constructing such $F, G \in R$ given randomly chosen $f, g \in S_\delta$ is described in [8]. If F, G are constructed by this method then

$$\|F\| \approx \|G\| \approx \|f\| \sqrt{N/12} \approx N \sqrt{\delta/12}. \quad (1)$$

Set

$$h \equiv f^{-1} * F \equiv g^{-1} * G \pmod{q}. \quad (2)$$

By (2) there exist $k_1, k_2 \in R$ such that $f * h = F + k_1 q$ and $g * h = G + k_2 q$.

We will refer to h as the *public key* and to the pair f, g as the *private key*.

The corresponding L_h is referred to as an NTRUSign lattice. Two distinctly different bases can be given for it. First, L_h is generated by all linear combinations of the rows of

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

Here, as in [8], we have abbreviated a $2N$ by $2N$ matrix and $1, h, 0, q$ refer to the N by N circulant matrices corresponding to $1, h, 0, q$. This is the public basis for L_h . However, another basis exists, the rows of the private basis:

$$\begin{pmatrix} f & F \\ g & G \end{pmatrix}.$$

These two bases are related by

$$\begin{pmatrix} f & -k_1 \\ g & -k_2 \end{pmatrix} \begin{pmatrix} 1 & h \\ 0 & q \end{pmatrix} = \begin{pmatrix} f & F \\ g & G \end{pmatrix}.$$

Our parameter generation algorithm must output a description of S_δ . In this paper, we will take all private key polynomials f and g to be drawn from the space $\mathcal{T}(d)$:

Definition 1. Fix $d \in \mathbb{Z}$, $d > 0$. The space $\mathcal{T}(d)$ is defined to be the set of all $r \in R$ such that $d + 1$ coefficients of r are set equal to 1, d coefficients of r are set equal to -1 and the remaining coefficients are set equal to 0.

Definition 2. Define $\delta_d \equiv \|f_d\|^2/N$, for f_d drawn from $\mathcal{T}(d)$:

$$\delta_d = (2d + 1 - 2/N + (1 + 2d)/N^2)/N.$$

Therefore, our parameter generation algorithm must output d .

2.3 Sketch of NTRUSign: Signing

Signing and Verification — The signature algorithm takes as input a digital document D and the private key, and outputs a signature s (for further details, see Appendix A). To verify in [8], the recipient checks that

$$\|(s, s * h - m(D))\| \leq \mathcal{N},$$

where $m(D) \in R$ is a *message representative* derived by hashing D (see [8] for a discussion of hashing requirements), and \mathcal{N} is a *norm bound* output by the parameter generation algorithm.

In the transpose lattice, $\|s\|$ will typically be smaller than $\|s * h - m\|$ by a factor of $\sqrt{12/N}$. We therefore generalize the norm $\|(r, r')\|$ to include a *balancing factor* β :

$$\|(r, r')\|_\beta = \min_{k_1, k_2 \in R} (\|r + k_1 q\|^2 + \beta^2 \|r' + k_2 q\|^2)^{1/2} .$$

Verification then consists of checking that

$$\|(s, s * h - m(D))\|_\beta \leq \mathcal{N} . \quad (3)$$

The norm $\|(r, r')\|_\beta$ can be looked on as the usual norm of $(r, \beta r')$ in the lattice

$$L_h(\beta) = L_h(N, q, \beta) = \{(r, \beta r') \mid r \in R, r' \equiv r * h \pmod{q}\} .$$

Signing Failures — Depending on the parameter sets, it may be possible for the signing algorithm to fail by producing an s such that $\|(s, s * h - m(D))\|_\beta > \mathcal{N}$. To address this, either the signer should use parameter sets such that the chance of a failure is negligible, or she should include some randomness in the signature, perform a trial verification after each signature operation, and re-sign with different randomness if the verification fails. Ideally, the parameter generation algorithm would take as input an acceptable chance of signing failure and use this in selecting \mathcal{N} . In this paper, we denote the expected size of a signature by \mathcal{E} and define the *signing tolerance* ρ by

$$\mathcal{N} = \rho \mathcal{E} .$$

As ρ increases beyond 1, the chance of a signing failure appears to drop off exponentially. In particular, experimental evidence indicates that the probability that a validly generated signature will fail the normbound test with parameter ρ is less than $e^{-C(N)(\rho-1)}$, where $C(N) > 0$ increases with N . In fact, under the assumption that a coefficient of a signature can be treated as a sum of independent identically distributed random variables, a theoretical analysis indicates that $C(N)$ is quadratic in N .

In this paper we take $\rho = 1.1$, which appears to give a vanishingly small probability of valid signature failure for N in the ranges considered here. We also present some sample parameters with $\rho = 1$, where multiple signing may be required.

Transcript Analysis — Signing is not zero-knowledge, and a transcript of signatures leaks information about the private key [4, 8]. The number of signatures that an attacker must see to mount the best currently known attack can be greatly increased by the use of *perturbations*, introduced in [8]. We will argue later that the length of the transcript needed to recover the private key is exponential in the number of perturbations. Ideally, the parameter generation algorithm would take as input the number of signatures that were to be generated with a given key, and output the appropriate number of perturbations. In fact, in this paper we will restrict ourselves to considering parameter sets with one perturbation of a specific form, and argue that for all the parameter sets under consideration the use of a single perturbation makes it safe to produce well over 2^{30} signatures with a single key. Future research will consider alternative, more efficient forms for the perturbations.

Our parameter generation algorithm will output \mathcal{N} and β . This completes the list of parameters that the algorithm must output.

2.4 A proposed parameter generation algorithm for NTRUSign

In this section we will describe an algorithm for determining an NTRUSign parameter set with one perturbation and with a given level of security. The remainder of this paper justifies these choices in more detail.

The inputs to the algorithm are k , the desired security level in bits, the signing tolerance $\rho = \mathcal{N}/\mathcal{E}$ defined in Section 2.3 above, and N_{\max} , the maximum value of N to check in the algorithm below. The parameter sets given below for different values of k , which are the recommended parameter sets for NTRUSign, were generated with $\rho = 1.1$ and $N_{\max} = 2 * \rho * k$.

The algorithm presented here generates parameter sets which are verifiable (anyone can generate them) and secure. They are also general enough to be efficient in a variety of environments. In specific environments with particular requirements (for example, 8-bit processors) various fine tunings and alterations can be made to the algorithm. Future papers will present modified algorithms tailored to these environments.

1. Set $N = 2$.
2. If we've returned here from a later step, check that $N \leq N_{\max}$. If it is larger, go to step 10.
3. Set $q = 32$.
4. If we've returned here from a later step, check that $q \leq 2048$. If $q > 2048$, increment N to the next prime and go to step 2.
5. Choose minimal d such that $d \leq N/3$ and the combinatorial security $\omega_{\text{cmb}} \equiv \log_2 \binom{N}{d+1} / \sqrt{N} > k$. If no such d exists, replace N by the next largest prime and return to step 2. Otherwise, continue.
6. Calculate δ_d (from definition 2) and

$$c = \sqrt{\frac{2\pi e \delta N}{q}} \sqrt{\frac{N}{3}}. \quad (4)$$

Refer to Table 1 to obtain $A(c), B(c)$, the lattice key security constants. Calculate ω_{lk} , the security against [l]attice [k]ey recovery attacks, as

$$\omega_{\text{lk}}(A(c), B(c), N) = AN - B - \max_r \left(\log_2 \left(1 - \left(1 - \prod_{i=0}^{2d} \left(1 - \frac{r}{N-i} \right) \right)^{2N} \right) + Ar/2 \right).$$

- (a) If $\omega_{\text{lk}} < k$, increment d by 1, check $A(c)$ and $B(c)$ in Table 1, and calculate the new value of ω_{lk} . Repeat until $\omega_{\text{lk}} > k$ or $d \geq N/3$.
 - (b) If $d \geq N/3$, increment N to the next prime and return to step 2.
7. The best known combinatorial forgery attacks require effort

$$\omega_{\text{frg}} = -\log_2 \left(\sqrt{\frac{\pi^{N/2}}{\Gamma(1+N/2)}} \cdot \left(\frac{\rho N \sqrt{\delta/3} (1 + \beta^2 N/12)^{1/2}}{q\beta} \right)^N \right).$$

Choose minimal β , $\sqrt{12/N} \leq \beta \leq 1$ such that $\omega_{\text{frg}} > k$. If there is no β in this range that satisfies this condition, set $q = 2q$ and return to step 4. Otherwise, set

$$\mathcal{N} = \frac{\rho}{6} \sqrt{12\delta_d N^2 + \beta^2 \delta_d N^3} \quad (5)$$

and continue.

8. Forgery attacks based on solving CVP with the public key are characterized by N/q and

$$\gamma(N, q, \beta, \delta, \rho) = \rho \sqrt{\frac{\pi e \delta}{6q} \left(\frac{1}{\beta} + \frac{\beta N}{12} \right)}. \quad (6)$$

Calculate $\gamma(N, q, \beta, \delta, \rho)$. Refer to Table 2 to obtain ω_{f} , the strength against lattice-based forgery attacks (to use a given line of Table 2, both γ and N/q must be less than or equal to the stated values). If $\omega_{\text{f}} < k$, set $q = 2q$ and return to step 4.

9. If we've got to this step, the parameters chosen give the desired level of security against all known attacks for the current value of N . Calculate the time to sign $\sigma_S = 8dN + N^2$, the time to verify $\sigma_V = N^2$, the size in bits of the public key and signature $b_{pk} = N * \log_2 q$, and the transcript length $\tau = 2^9 d^6$. Store the parameters and the other just calculated values. Increment N to the next prime and return to step 2.
10. Check the valid parameter sets that were stored in calls to step 9. Depending on requirements, select the one that gives the lowest σ_S , σ_V , or b_{pk} , or the highest value of τ . Output this parameter set and complete.

$c > \dots$	$A(c)$	$B(c)$
3.7	0.451	-0.218
5.3	0.649	-5.436
6.8	1.539	-102.59

Table 1. Constants used to calculate bit security against lattice key attacks, based on experimental evidence for different values of c .

$\gamma < \dots$	$N/q < \dots$	$\omega_{lf}(N)$
0.1774	1.305	$0.995113N - 82.6612$
0.1413	0.707	$1.16536N - 78.4659$
0.140	0.824	$1.14133N - 76.9158$

Table 2. Bit security against lattice forgery attacks, ω_{lf} , based on experimental evidence for different values of $(\gamma, N/q)$.

To see that the algorithm terminates, first, note that if $c > 5.3$ and $\gamma < 0.1774$, increasing N will lead to sufficient ω_{lk} and ω_{lf} so long as N/q is sufficiently low. Now,

$$\frac{c}{\gamma} = \frac{4}{\rho} \sqrt{\frac{(3N)^{3/2} \beta}{12 + \beta^2 N}}$$

so as N increases so will c/γ . Once $c/\gamma > 5.3/0.1774$, we can clearly find values of N , δ , N that give sufficient ω_{lk} and ω_{lf} . This will be the case only if $N/q = a$ is less than some bound, in other words if $q = N/a$. We will therefore take q to increase as N .

Now we consider ω_{frg} for the two extreme values of β , 1 and $\sqrt{12/N}$. For $\beta = 1$, we have

$$\omega_{frg}(\beta = 1) = -\log_2 \left[\frac{2\pi e \rho^2}{3} \left(\frac{1}{N} + \frac{1}{12} \right) a^2 \delta \right]^{N/4}, \quad (7)$$

and for $\beta = \sqrt{12/N}$, we have

$$\omega_{frg}(\beta = \sqrt{12/N}) = -\log_2 \left[\frac{\pi e \rho^2}{9} a^2 \delta \right]^{N/4}. \quad (8)$$

If δ is any constant > 0 , increasing N will increase ω_{cmb} roughly linearly with N and will increase c as \sqrt{N} . We can therefore choose δ to be arbitrarily small, such that both (7) and (8) take on the form $-N/4 \cdot \log_2(p)$, where $p < 1$. This demonstrates that ω_{frg} increases linearly with N . Since ω_{cmb} , ω_{frg} , ω_{lk} , and ω_{lf} all increase with N , this therefore demonstrates that for sufficiently high N a parameter will exist with security level k . Since this algorithm is essentially a brute-force search through the space of possible parameters, if the set exists the algorithm will find it. This demonstrates that the algorithm will terminate.

2.5 Recommended Parameter Sets

The parameter sets in Table 3 were generated with $\rho = 1.1$ and $N_{\max} = 2 * \rho * N$, and selected to give the shortest possible signing time σ_S . This was found to also give the lowest values for the other performance measures σ_V and b_{pk} . The value for N_{\max} was a heuristic; as N increased beyond the values in the recommended parameter sets, all the performance measures deteriorated noticeably. Table 3 gives the parameters and all relevant measures of security. The transcript length required, τ , is derived by the methods described in Section 6 and will in practice underestimate the required transcript length by a considerable margin. Appendix C gives the performance measures for these parameters, and for comparison the parameter sets obtained by setting $\rho = 1$.

Parameters						Security Measures						
k	N	d	q	β	\mathcal{N}	ω_{cmb}	c	ω_{lk}	ω_{frg}	γ	ω_{if}	$\log_2(\tau)$
80	157	29	256	0.38407	150.02	104.43	5.34	93.319	80	0.139	102.27	31.9
112	197	28	256	0.51492	206.91	112.71	5.55	117.71	112	0.142	113.38	31.2
128	223	32	256	0.65515	277.52	128.63	6.11	134.5	128	0.164	139.25	32.2
160	263	45	512	0.31583	276.53	169.2	5.33	161.31	160	0.108	228.02	34.9
192	313	50	512	0.40600	384.41	193.87	5.86	193.22	192	0.119	280.32	35.6
256	349	75	512	0.18543	368.62	256.48	7.37	426.19	744	0.125	328.24	38.9

Table 3. Parameters and relevant security measures for ternary keys, one perturbation, $\rho = 1.1$, $q = \text{power of } 2$

3 Security considerations

In this section we review the security considerations that have led to the algorithm proposed above.

The standard definition of a CCA2 secure signature scheme is given in [10]. However the premise underlying NTRUSign is that considerable efficiency gains can be made by weakening this requirement by only allowing the adversary access to a bounded (constant) number of signatures. In this model to be an effective signature scheme the following security issues must be addressed:

NTRUSign security

1. Given only the public parameters $N, q, \delta, \beta, \mathcal{N}$ and the public key h , it should be very hard to recover f, g .
2. Given only the public parameters, h , and D it should be very hard to create an s satisfying (3).
3. Given the additional information f, g it should be computationally efficient to create a signature s on D satisfying (3).
4. Given only the public parameters and a long transcript of valid signatures $(D_1, s_1), (D_2, s_2), \dots, (D_\tau, s_\tau)$, it should be computationally unfeasible to create a valid pair of D and s a signature on D .

Item (1) has been well studied already and is essentially the NTRU key recovery problem. We will discuss this first in Section 4. Item (2) is really a subset of item (4), but for conceptual reasons it will be considered separately in Section 5. Item (3), the method of computing a signature given the private key will be briefly reviewed in Appendix A and item (4) will be covered in Section 6.

4 Security of the private key

Given the public key h and the parameters N, q, δ an adversary is faced with the problem of determining some $r, r' \in R$ such that r, r' are reasonably small and $r' \equiv r * h \pmod{q}$. These can be the original keys f, F or g, G or some other pair of similar size. Experiments [5] indicate that finding useful imitations is not significantly easier than finding the original keys, and so we will concentrate here on recovery of f, F or g, G . As is expounded in [9] there are *two* primary methods of approaching the key recovery problem, both of which must be considered when selecting a parameter set to give a specific security level. These are lattice-based attacks [6] and combinatorial attacks [7]. Other methods of attack exist, but are less efficient than these two.

4.1 Combinatorial Security

We refer to the security of a polynomial against combinatorial attack as its *combinatorial security*, and denote the combinatorial security of polynomials drawn from \mathcal{S} by $\text{Comb}[\mathcal{S}]$. A combinatorial attack can be accomplished via a meet in the middle technique on the known space $\mathcal{T}(d)$ that f, g are chosen from. Then

$$\text{Comb}[\mathcal{T}(d)] > \binom{N}{d+1} / \sqrt{N}. \quad (9)$$

4.2 Lattice Security

The point (f, F) will be contained in the lattice L_h , and The point $(f, \lambda F)$ will be contained in the lattice $L_h(\lambda)$ for any $\lambda \in \mathbb{R}$. As noted in [5, 6], an attacker minimizes the running time for a lattice-based attack by selecting

$$\lambda = \|f\| / \|F\| .$$

We define the lattice constant c as

$$c = \sqrt{2N} \cdot \frac{\|(f, \lambda F)\|}{\sigma, \text{ length of expected shortest vector in } L_h(\lambda)} .$$

The length of the expected shortest vector, σ , is given (approximately) by [6]:

$$\sigma(N, q, \delta, \lambda) = \sqrt{\frac{Nq\lambda}{\pi e}}. \quad (10)$$

In the transpose lattice, $\lambda = \|f\| / \|F\| = \sqrt{12/N}$, and so

$$c = \sqrt{\frac{2\pi e \delta N^{3/2}}{3^{1/2} q}}. \quad (11)$$

Experimentally, for fixed c and N/q , the running times for lattice reduction behave roughly as

$$\log(T) = AN + B ,$$

for some experimentally-determined constants A and B . Thus for constant c and N/q , increasing N increases the breaking time exponentially. As c increases, with N and N/q held constant, the coefficient A appears to increase. The relevant experiments are summarized in Table 1.

For NTRUSign in the “standard” lattice, the small vector in the lattice is of the form (f, g) , where $\|f\| \sim \|g\|$, and so the balancing constant $\lambda = 1$. Since the definition of c involves $\sqrt{\lambda^{-1}}$, the

effect of moving from the “standard” NTRUEncrypt lattice to the “transpose” NTRUSign lattice is to increase c by a factor of $(N/12)^{1/4}$ for free. This allows for a given level of lattice security at lower dimensions for the transpose lattice than for the standard lattice. Note that NTRUEncrypt uses the standard lattice, which is why the key sizes given in [9] are greater than the equivalent NTRUSign key sizes at the same level of security.

4.3 Zero-forcing

Zero-forcing [11] allows an attacker to reduce the dimension of the lattice they must attack to recover the key. The formula of [11], corrected in [9], applies here with two changes. First, because the private key is trinary, there are $2d + 1$ nonzero entries rather than d as in the binary case. Second, because a pattern of zeroes can be found in either f or g , there are $2N$ rotations rather than N rotations of the pattern that might be of use – hence the “. 2N ”. We obtain:

$$\text{Gain} \sim \left(1 - \left(1 - \prod_{i=0}^{2d} \left(1 - \frac{r}{N-i} \right) \right)^{2N} \right) 2^{\alpha r/2},$$

where α is the slope of the lattice strength.

5 Security against forgery

Next, we quantify the probability that an adversary, without knowledge of f, g , can compute a signature s on a given document D . The constants $N, q, \delta, \beta, \mathcal{N}$ must be chosen to ensure that this probability is less than 2^{-k} , where k is the desired bit level of security. To investigate this some additional notation will be useful:

1. EXPECTED LENGTH OF s : \mathcal{E}_s
2. EXPECTED LENGTH OF $t - m$: \mathcal{E}_t

By $\mathcal{E}_s, \mathcal{E}_t$ we mean respectively the expected values of $\|s\|$ and $\|t - m\|$ (appropriately reduced mod q) when generated by the signing procedure described in Appendix A. These will be independent of m but dependent on N, q, δ . A genuine signature will then have expected length

$$\mathcal{E} = \sqrt{\mathcal{E}_s^2 + \beta^2 \mathcal{E}_t^2}$$

and we will set

$$\mathcal{N} = \rho \sqrt{\mathcal{E}_s^2 + \beta^2 \mathcal{E}_t^2}. \tag{12}$$

As in the case of recovering the private key, an attack can be made by combinatorial means, by lattice reduction methods or by some mixing of the two. By balancing these approaches we will determine the optimal choice of β , the public scaling factor for the second coordinate.

5.1 Combinatorial forgery

Let us suppose that $N, q, \delta, \beta, \mathcal{N}, h$ are fixed. An adversary is given m , the image of a digital document D under the hash function H . His problem is to locate an s such that

$$\|(s \bmod q, \beta(h * s - m) \bmod q)\| < \mathcal{N}.$$

In particular, this means that for an appropriate choice of $k_1, k_2 \in R$

$$(\|s + k_1q\|^2 + \beta^2\|h * s - m + k_2q\|^2)^{1/2} < \mathcal{N}.$$

A purely combinatorial attack that the adversary can take is to choose s at random to be quite small, and then to hope that the point $h * s - m$ lies inside of a sphere of radius \mathcal{N}/β about the origin after its coordinates are reduced mod q . The attacker can also attempt to combine guesses, in a way similar to the meet-in-the-middle attacks on private NTRUEncrypt keys originally due to Odlyzko [7]. Here, the attacker would calculate a series of random s_i and the corresponding t_i and $t_i - m$, and file the t_i and the $t_i - m$ for future reference. If a future s_j produces a t_j that is sufficiently close to $t_i - m$, then $(s_i + s_j)$ will be a valid signature on m . As with the previous meet-in-the-middle attack, the core insight is that filing the t_i and looking for collisions allows us to check l^2 t -values while generating only l s -values.

An important element in the running time of attacks of this type is the time that it takes to file a t value. We are interested not in exact collisions, but in two t_i that lie close enough to allow forgery. In a sense, we are looking for a way to file the t_i in a spherical box, rather than in a cube as is the case for the similar attacks on private keys. It is not clear that this can be done efficiently. However, for safety, we will assume that the process of filing and looking up can be done in constant time, and that the running time of the algorithm is dominated by the process of searching the s -space. Under this assumption, the attacker's expected work before being able to forge a signature is:

$$p(N, q, \beta, \mathcal{N}) < \sqrt{\frac{\pi^{N/2}}{\Gamma(1 + N/2)} \cdot \left(\frac{\mathcal{N}}{q\beta}\right)^N}. \quad (13)$$

If k is the desired bit security level it will suffice to choose parameters so that the right hand side of (13) is less than 2^{-k} .

5.2 Signature forgery through lattice attacks

On the other hand the adversary can also launch a lattice attack by attempting to solve a closest vector problem. In particular, he can attempt to use lattice reduction methods to locate a point $(s, \beta t) \in L_h(\beta)$ sufficiently close to $(0, \beta m)$ that $\|(s, \beta(t - m))\| < \mathcal{N}$. We'll refer to $\|(s, \beta(t - m))\|$ as the norm of the intended forgery.

The difficulty of using lattice reduction methods to accomplish this can be tied to another important lattice constant:

$$\gamma(N, q, \beta) = \frac{\mathcal{N}}{\sigma(N, q, \delta, \beta)\sqrt{2N}}. \quad (14)$$

This is the ratio of the required norm of the intended forgery over the norm of the expected smallest vector of $L_h(\beta)$, scaled by $\sqrt{2N}$. For usual NTRUSign parameters the ratio, $\gamma(N, q, \beta)\sqrt{2N}$, will be larger than 1. Thus with high probability there will exist many points of $L_h(\beta)$ that will work as forgeries. The task of an adversary is to find one of these without the advantage that knowledge of the private key gives. As $\gamma(N, q, \beta)$ decreases and the ratio approaches 1 this becomes measurably harder.

Experiments have shown that for fixed $\gamma(N, q, \beta)$ and fixed N/q the running times for lattice reduction to find a point $(s, t) \in L_h(\beta)$ satisfying

$$\|(s, t - m)\| < \gamma(N, q, \beta)\sqrt{2N}\sigma(N, q, \delta, \beta)$$

behave roughly as

$$\log(T) = AN + B$$

as N increases. Here A is fixed when $\gamma(N, q, \beta), N/q$ are fixed, increases as $\gamma(N, q, \beta)$ decreases and increases as N/q decreases. Experimental results are summarized in Table 2.

Our analysis shows that lattice strength against forgery is maximized, for a fixed N/q , when $\gamma(N, q, \beta)$ is as small as possible. By (10),(12),(14) we have

$$\gamma(N, q, \beta) = \rho \sqrt{\frac{\pi e}{2N^2 q} \cdot (\mathcal{E}_s^2/\beta + \beta \mathcal{E}_t^2)} \quad (15)$$

and so clearly the value for β which minimizes γ is $\beta = \mathcal{E}_s/\mathcal{E}_t$. This optimal choice yields

$$\gamma(N, q, \beta) = \rho \sqrt{\frac{\pi e \mathcal{E}_s \mathcal{E}_t}{N^2 q}}. \quad (16)$$

Referring to (13) we see that increasing β has the effect of improving combinatorial forgery security. Thus the optimal choice will be the minimal $\beta \geq \mathcal{E}_s/\mathcal{E}_t$ such that $p(N, q, \beta, \mathcal{N})$ defined by (13) is sufficiently small.

An adversary could attempt a mixture of combinatorial and lattice techniques, fixing some coefficients and locating the others via lattice reduction. However, as explained in [8], the lattice dimension can only be reduced a small amount before a solution becomes very unlikely. Also, as the dimension is reduced, γ decreases, which sharply increases the lattice strength at a given dimension.

6 Transcript security

In this section we will assume that signatures are generated by a private basis $\{f, g, F, G\}$ together with T private perturbation bases $\{f_i, g_i, F_i, G_i\}, i = 1, \dots, T$. We will assume that $f * G - g * F = f_i * G_i - g_i * F_i = q$ for each i , and that $\|F_i\| = \sqrt{N/12}\|f_i\|$.

An adversary studying a long transcript of valid signatures will by (24) have at his disposal a long list of pairs of polynomials of the form

$$s = \epsilon f + \epsilon' g + \epsilon_1 f_1 + \epsilon'_1 g_1 + \dots + \epsilon_T f_T + \epsilon'_T g_T \quad (17)$$

and

$$t - m = \epsilon F + \epsilon' G + \epsilon_1 F_1 + \epsilon'_1 G_1 + \dots + \epsilon_T F_T + \epsilon'_T G_T. \quad (18)$$

Let $f_0 = f, F_0 = F, \epsilon_0 = \epsilon, \dots$ for the purpose of having a more uniform notation. Then by (19), for $i = 0, \dots, T$

$$\epsilon_i = \left\{ \frac{m * g_i}{q} \right\}, \quad \epsilon'_i = -\left\{ \frac{m * f_i}{q} \right\}$$

Let $a(X) = \sum a_i X^i \in R$ be a polynomial. The *reversal* of a is the polynomial

$$\bar{a}(X) = a(X^{-1}) = a_0 + \sum_{i=1}^{N-1} a_{N-i} X^i.$$

We then set

$$\hat{a}(X) = a(X) * \bar{a}(X).$$

Notice that \hat{a} has the form

$$\hat{a} = \sum_{k=0}^{N-1} \left(\sum_{i=0}^{N-1} a_i a_{i+k} \right) X^k.$$

From [8], the expectation of \hat{s} and $\hat{t} - \hat{m}$, given by (17),(18) is (up to lower order terms)

$$E(\hat{s}) = (N/12)(\hat{f}_0 + \hat{g}_0 + \dots + \hat{f}_T + \hat{g}_T)$$

and

$$E(\hat{t} - \hat{m}) = (N/12)(\hat{F}_0 + \hat{G}_0 + \dots + \hat{F}_T + \hat{G}_T).$$

We refer to these as the second moments. If these second moments could be recovered and if the $\hat{f}_i, \hat{g}_i, \hat{F}_i, \hat{G}_i$ could be removed for $i \geq 1$ then the problem of recovering the private key would reduce to the problem of factoring a Gram matrix $U^T U$, where U is an unknown orthonormal lattice basis (see [4]). For safety we will assume that a reduction to this problem reveals the key, although at this moment the problem of efficient Gram factorization has not been solved. If one perturbation is added, i.e if $T = 1$, then the best known attack is to eliminate the perturbation and the $\hat{f}_1, \hat{g}_1, \hat{F}_1, \hat{G}_1$ by first recovering $E(\hat{s}^2)$, $E((\hat{t} - \hat{m})^2)$, $E(\hat{s}^3)$ and $E((\hat{t} - \hat{m})^3)$ (known as fourth and sixth moments respectively) and then using simple algebra to reduce to the Gram factorization problem. Even this involves some unexplored territory, such as the taking of square roots in this context, but we will again assume that this causes the attacker has no significant problems.

Let us suppose now that $T = 1$. If τ is sufficiently large, then an attacker has a reasonable chance of determining $(12/N)E(\hat{s}) = \hat{f}_0 + \hat{g}_0 + \hat{f}_1 + \hat{g}_1$ by averaging over τ signatures and rounding to the nearest integer. This will give a reasonably correct answer when the error in many coefficients (say at least half) is less than $1/2$. To compute the probability that an individual coefficient has an error less than $1/2$, write $(12/N)\hat{s}$ as a main term plus an error, where the main term converges to $\hat{f}_0 + \hat{g}_0 + \hat{f}_1 + \hat{g}_1$. The error will converge to 0 at about the same rate as the main term converges to its expected value. If the probability that a given coefficient is further than $1/2$ from its expected value is less than $1/(2N)$ then we can expect at least half of the coefficients to round to their correct values. (Note that this convergence cannot be speeded up using lattice reduction in, for example, the lattice \hat{h} , because the terms \hat{f}, \hat{g} are unknown and are larger than the expected shortest vector in that lattice).

The rate of convergence of the error and its dependence on τ can be estimated by an application of Chernoff-Hoeffding techniques, using an assumption of a reasonable amount of independence and uniform distribution of random variables within the signature transcript. This assumption appears to be justified by experimental evidence, and in fact benefits the attacker by ensuring that the cross-terms converge to zero. Details of the calculation are given in Appendix B.

Using this technique, we estimate that to have a single coefficient in the $2k$ -th moment with error less than $\frac{1}{2}$, the attacker must analyze a signature transcript of length $\tau > 2^{2k+4} d^{2k} / N$. Here d is the number of 1's in the trinary key. Experimental evidence for the second moment indicates that the required transcript length will in fact be much longer than this. For one perturbation, the attacker needs to recover the sixth moment accurately, leading to required transcript lengths $\tau > 2^{30}$ for all the recommended parameter sets in this paper.

7 Conclusion and Alternative algorithms

This paper has outlined an algorithm that produces a set of NTRUSign signatures that allow signing of 2^{30} messages at a security level of k bits. Future refinements might include:

1. Taking q to be a prime, rather than a power of 2.
2. Different values of ρ , allowing a tradeoff between reduction of \mathcal{N} and increased probability of having to re-sign.
3. Closer consideration of the requirements for perturbation bases, to establish whether they have to be generated with exactly the same properties as the public basis.

References

1. M. Brown, D. Hankerson, J. López, and A. Menezes, Software Implementation of the NIST Elliptic Curves Over Prime Fields, *CT-RSA 2001*, D. Naccache (Ed.), LNCS 2020, 250–265, Springer-Verlag, 2001.
2. Kirill Levchenko, Chernoff Bound, available at <http://www.cs.ucsd.edu/~klevchen/techniques/chernoff.pdf>
3. D. Coppersmith and A. Shamir, *Lattice Attack on NTRU*, Advances in Cryptology - Eurocrypt'97, Springer-Verlag
4. C. Gentry, M Szydło, *Cryptanalysis of the Revised NTRU SignatureScheme*, Advances in Cryptology—Eurocrypt '02, Lecture Notes in Computer Science, Springer-Verlag, 2002.
5. J. Hoffstein, J. Pipher, J.H. Silverman, *NTRU: A new high speed public key cryptosystem*, in Algorithmic Number Theory (ANTS III), Portland, OR, June 1998, Lecture Notes in Computer Science 1423 (J.P. Buhler, ed.), Springer-Verlag, Berlin, 1998, 267–288.
6. J. Hoffstein, J. H. Silverman, W. Whyte, Estimated Breaking Times for NTRU Lattices, Technical report, NTRU Cryptosystems, June 2003 Report #012, version 2, available at <http://www.ntru.com>.
7. N. A. Howgrave-Graham, J. H. Silverman, W. Whyte, A Meet-in-the-Middle Attack on an NTRU Private key, Technical report, NTRU Cryptosystems, June 2003. Report #004, version 2, available at <http://www.ntru.com>.
8. J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. Silverman, W. Whyte, NTRUSign: Digital Signatures Using the NTRU Lattice, CT-RSA 2003,
9. N. A. Howgrave-Graham, J. H. Silverman, W. Whyte, Choosing Parameter Sets for NTRUEncrypt with NAEP and SVES-3, CT-RSA 2005, to appear.
10. E. Kiltz, J. Malone-Lee, *A General Construction of IND-CCA2 Secure Public Key Encryption*, In: Cryptography and Coding, pages 152–166. Springer-Verlag, December 2003.
11. A. May, J.H. Silverman, *Dimension reduction methods for convolution modular lattices*, in Cryptography and Lattices Conference (CaLC 2001), J.H. Silverman (ed.), Lecture Notes in Computer Science 2146, Springer-Verlag, 2001

A Computation of the signature

A.1 The basic signature process

We will need to round numbers to the nearest integer and to take their fractional parts. For any $a \in \mathbb{Q}$, let $\lfloor a \rfloor$ denote the integer closest to a , and define $\{a\} = a - \lfloor a \rfloor$. If A is a polynomial with rational (or real) coefficients, let $\lfloor A \rfloor$ and $\{A\}$ be A with the indicated operation applied to each coefficient.

Suppose we are given a point $(0, m)$, where m is the image of some digital document \mathcal{D} under the hash function H . Our object is to find a point $(s, t) \in L_h(\beta)$ such that $\|s\|^2 + \beta^2\|t - m\|^2$ is as small as possible. As is described in [8] this is accomplished by the following process. Solve for real (x, y) satisfying

$$(0, m) = (x, y) \begin{pmatrix} f & F \\ g & G \end{pmatrix}$$

by writing

$$(x, y) = (0, m) \begin{pmatrix} G & -F \\ -g & f \end{pmatrix} / q = \left(\frac{-m * g}{q}, \frac{m * f}{q} \right).$$

Define ϵ, ϵ' with rational coefficients varying uniformly between $-1/2$ and $1/2$ by

$$\lfloor x \rfloor = x + \epsilon, \quad \lfloor y \rfloor = y + \epsilon'.$$

Thus

$$\epsilon = -\{x\} = \left\{ \frac{m * g}{q} \right\}, \quad \epsilon' = -\{y\} = -\left\{ \frac{m * f}{q} \right\} \quad (19)$$

Note that \mathcal{E}_ϵ , the expected size of $\|\epsilon\|$ equals the expected size of $\|\epsilon'\|$ and

$$\mathcal{E}_\epsilon = \sqrt{N/12}. \quad (20)$$

(This is approximately but not exactly correct, as the fractional part must make a choice between $-1/2$ and $1/2$ if q is even, but the correction is simple.)

Letting

$$(s, t) = (\lfloor x \rfloor, \lfloor y \rfloor) \begin{pmatrix} f & F \\ g & G \end{pmatrix}$$

we then obtain

$$(s, t - m) = (\epsilon f + \epsilon' g, \epsilon F + \epsilon' G). \quad (21)$$

For computational convenience we have only described how to sign points of the form $(0, m)$. However it is useful to note that a signature on a general point (m_1, m_2) can be reduced to this special case. Simply sign $(0, m_2 - h * m_1)$, obtaining a signature (s_1, t_1) satisfying

$$(s_1, t_1 - (m_2 - h * m_1)) = (\epsilon f + \epsilon' g, \epsilon F + \epsilon' G).$$

Then setting $(s, t) = (s_1, t_1) + (m_1, h * m_1)$ we obtain a new lattice point satisfying

$$(s - m_1, t - m_2) = (\epsilon_1 f + \epsilon_2 g, \epsilon_1 F + \epsilon_2 G).$$

We are now in a position to measure the expected size of a signature. By (21)

$$\|s\| \approx \sqrt{\|\epsilon_1\|^2 \|f\|^2 + \|\epsilon_2\|^2 \|g\|^2}.$$

By (20) and our assumption that $f, g \in \mathcal{S}_\delta$ it follows that

$$\mathcal{E}_s = \sqrt{\frac{N^2 \delta}{6}}. \quad (22)$$

Similarly, if our signature is derived from the key generation process described in [8] then F, G will satisfy (1) and

$$\mathcal{E}_t = \sqrt{\frac{N^3 \delta}{72}}. \quad (23)$$

A.2 The addition of perturbations

NTRUSign is not zero-knowledge, but the rate of information leakage can be reduced considerably by the use of *perturbations*. In this section we explain this concept, and discuss how perturbations are constructed and their implications for the size of the final signatures.

As before, consider a point $(0, m)$, where m is the image of some digital document \mathcal{D} under the hash function H . Suppose the signer has in his possession another private basis $\{f_1, g_1, F_1, G_1\}$. Using this basis to sign $(0, m)$ he obtains (s_1, t_1) satisfying

$$(s_1, t_1 - m) = (\epsilon_1 f_1 + \epsilon'_1 g_1, \epsilon_1 F_1 + \epsilon'_1 G_1),$$

where ϵ_1, ϵ'_1 are defined analogously to (19). He may then use the generalized signing procedure described in the previous section to sign (s_1, t_1) and obtain (s, t) satisfying

$$(s - s_1, t - t_1) = (\epsilon f + \epsilon' g, \epsilon F + \epsilon' G).$$

Thus

$$(s, t - m) = (\epsilon f + \epsilon' g + \epsilon_1 f_1 + \epsilon'_1 g_1, \epsilon F + \epsilon' G + \epsilon_1 F_1 + \epsilon'_1 G_1). \quad (24)$$

The important observation from the point of view of security is that the distribution of the ϵ s is the same as the distribution of the ϵ_1 s. An attacker who averages functions of signatures will therefore not be able to pick a method of averaging that removes the effect of the perturbations. As discussed in [8], the attacker must instead obtain sufficient linearly independent averaged values to allow them to eliminate the perturbations by linear algebra.

We now consider the size of perturbed signatures. For generality, we first consider an arbitrary private basis $\{f_1, g_1, F_1, G_1\}$ where $\|f_1\| = \sqrt{\delta_1 N}$ and $\|F_1\| = \sqrt{\omega_1} \|f_1\|$. Following the analysis in [8] it is easily checked that

$$\mathcal{E}_s = \sqrt{\frac{N^2(\delta + \delta_1)}{6}}$$

and that

$$\mathcal{E}_t = \sqrt{\frac{N^2(\delta\omega + \delta_1\omega_1)}{6}}.$$

Similarly, if two private bases are used then

$$\mathcal{E}_s = \sqrt{\frac{N^2(\delta + \delta_1 + \delta_2)}{6}}, \quad \mathcal{E}_t = \sqrt{\frac{N^2(\delta\omega + \delta_1\omega_1 + \delta_2\omega_2)}{6}}, \quad (25)$$

and so on.

B Transcript bounds

An application of the Chernoff-Hoeffding technique for bounding sums of uniformly bounded discrete and independent random variables [2] leads to the following

Proposition 1. *Let Y_i be a collection of T discrete random variables satisfying $|Y_i| < B$, for a fixed constant B , for every i and satisfying $E(Y_i) = 0$, $E(Y_i^2) = \sigma$ for every i . Then for any $K > 0$,*

$$P\left(\sum_{i=1}^T Y_i > K\right) \leq 2e^{-\frac{K^2}{4\sigma^2 T}}.$$

When calculating the $2k^{\text{th}}$ moment of s the main term will be composed of a number of pieces, one of which is

$$(N/12)^k (f * \bar{f})^k.$$

Similarly, a piece of the error, after averaging a transcript of length R will be

$$\frac{1}{R} \sum_{j=1}^R (\epsilon_f^{(j)} * \bar{\epsilon}_g^{(j)} * f * \bar{g})^k$$

A conservative lower bound for the size of R necessary to achieve an accurate estimate of the main term is a lower bound for the size of R necessary for the l^{th} coefficient of

$$\left(\frac{12}{N}\right)^k \left(\frac{1}{R}\right) \sum_{j=1}^R (\epsilon_f^{(j)} * \bar{\epsilon}_g^{(j)} * f * \bar{g})^k$$

to have an error of absolute value less than $1/2$. This coefficient is a sum of RN^{2k} variables, which we denote as

$$Y_i = \left(\frac{12}{N}\right)^k \left(\frac{1}{R}\right) \epsilon_f^{(j)}(l_1) \cdots \epsilon_f^{(j)}(l_k) \bar{\epsilon}_g^{(j)}(m_1) \cdots \bar{\epsilon}_g^{(j)}(m_k) A(r),$$

where $l_1 + \dots + l_k + m_1 + \dots + m_k + r \equiv l \pmod{N}$. By the quasi-multiplicativity property, the average value of $A(r)^2$ will be approximately given by

$$\|f * \bar{g}\|^{2k}/N \approx \|f\|^{2k} \|g\|^{2k}/N \approx (2d)^{2k}/N.$$

The $\epsilon^{(j)}(l_i)$ coefficients will be very close to independent, with $E(\epsilon^{(j)}(l_i)^2) = 1/12$. We thus have

$$E(Y_i^2) \approx \left(\frac{2d}{N}\right)^{2k} \frac{1}{R^2 N}.$$

Taking $K = 1/2$ and substituting into the Proposition we obtain

$$P\left(\sum_{i=1}^{R^2 N} Y_i > 1/2\right) \leq 2e^{-\frac{RN}{16(2d)^{2k}}}.$$

A necessary requirement for the right hand side to be less than $1/2$ is that

$$R > 2^{2k+4} d^{2k} / N$$

and this is the source of our estimate.

C Performance

Table 4 gives the estimated number of Add-With-Carries necessary for signing and verification with each of the given parameter sets. These are compared to figures for ECDSA which were generalized from [1] as described in [9].

The figures were obtained assuming that the operations can use an instruction that carries out a 32×32 -bit multiply in a single cycle. This is consistent with the assumption made in [1]. They ignore the time necessary to hash the incoming message.

The formulae used to obtain the performance figures are:

- trinary convolution = $(2d + 1)N$ adds-with-carry
- full convolution = N^2 adds-with carry
- sign with no perturbations = 4 trinary convolutions
- sign with one perturbation and no validity check = 8 trinary + 1 full convolution
- verify = 1 full convolution
- ECDSA signing = 1 point multiplication
- ECDSA verification = 1.17 point multiplications.

Table 4 gives the performance measures for each of the recommended parameter sets.

Table 5 investigates the effects on the parameter set of setting $\rho = 1$, increasing the danger that a message will have to be signed twice, but allowing a decreased \mathcal{N} and hence possibly security against forgery at lower values of N . As is shown, this is useful at lower security levels, but at higher security levels the $\rho = 1.1$ parameters are identical to the $\rho = 1$ parameters. This is a result of the increased lattice security at higher dimensions (c going as $N^{\frac{1}{4}}$, γ going as $N^{-\frac{1}{4}}$), which results in the value of d that first gives combinatorial security also giving the desired level of lattice security.

Parameters				b_{pk}			σ_s			σ_v			other	
k	N	d	q	NTRU	ECC	RSA	NTRU	ECDSA	Gain	NTRU	ECDSA	Gain	d/N	N/k
80	157	29	256	1256	192	1024	61073	112210	1.84	24649	130912	5.31	0.185	1.963
112	197	28	256	1576	224	~ 2048	82937	170356	2.05	38809	198749	5.12	0.142	1.759
128	223	32	256	1784	256	3072	106817	277280	2.60	49729	323493	6.51	0.143	1.742
160	263	45	512	2367	320	4096	163849	—	—	69169	—	—	0.131	1.644
192	313	50	512	2817	384	7680	233169	936618	4.20	97969	1092721	11.15	0.159	1.630
256	349	75	512	3141	512	15360	331201	1595434	4.82	121801	1861340	15.28	0.215	1.363

Table 4. Performance measures for the recommended parameter sets

k	N	d	q	β	\mathcal{N}	c	γ
80	127	31	256	0.37264	122.94	5.33	0.133
112	191	29	256	0.45615	176.14	5.60	0.127
128	223	32	256	0.65515	277.52	6.11	0.164
160	263	45	512	0.31583	276.53	5.33	0.108
192	313	50	512	0.40600	384.41	5.86	0.119
256	349	75	512	0.18543	368.62	7.37	0.125

Table 5. trinary keys, one perturbation, $\rho = 1$, $q =$ power of 2