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.

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 claimed to yield an 80 bit security level. The paper [8] 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 has, we believe, the stated security properties. We also present certain technical advances upon which we intend to build in subsequent papers.

In addition to outlining a parameter generation algorithm for NTRUSign, this paper makes the following four important contributions.

1. We note that the "transpose lattice" of [8] has greater security against lattice reduction key recovery attacks than does the "standard lattice," because shortest vector in the transpose lattice 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. This increased lattice security is combined with the use of trinary form for private keys, which increases the possible combinatorial security for a given key size.
2. 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. More precisely, a valid signer can now create signatures that are a factor of $N^{\frac{1}{4}}$ closer to the expected closest vector in the lattice than was previously possible using the standard lattice of [8]. Clearly, in an asymptotic sense, this is a change of great significance. For the practical cases under consideration in this paper, the use of this norm enables us to reduce the lattice dimension required for security against lattice-based signature forgery attacks, validating our ability to reduce bandwidth.
3. 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.
4. We improve the analysis of the number of signatures an attacker must collect in order to mount the best transcript attack that is currently known. For the parameter sets under analysis here, where there is one perturbation basis of the same size as the private basis of the public key, the transcript length goes as d^6/N, where d is the number of 1s in the private key. We present theoretical and experimental evidence to demonstrate that 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 calculated 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 basic operations on these polynomials are addition and convolution multiplication. Convolution multiplication $*$ of two polynomials f and g is defined by the formula

$$(f * g)(X) = \sum_{k=0}^{N-1} \Bigg(\sum_{i+j \equiv k \pmod N} f_i \cdot g_j \Bigg) X^k.$$

If one of the polynomials has all coefficients chosen from the set $\{0, \pm 1\}$, we will refer to the convolution as being *trinary*, 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 $r = \sum_{i=0}^{N-1} r_i X^i$. A natural measure of size in R is the centered Euclidean norm (essentially the variance) of the vector of coefficients. Thus we write $\bar{r} = \frac{1}{N} \sum_{i=0}^{N-1} r_i$ for the average of the coefficients and define the *centered norm* by the formula

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

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

Given any positive integers N and q and any polynomial $h \in R$, we construct a lattice L_h contained in $R^2 \cong \mathbb{Z}^{2N}$ as follows:

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

This sublattice of $\mathbb{Z}^{2N}$ is called a *convolution modular lattice*. It has dimension equal to $2N$ and determinant equal to q^N.

The centered norm $\| \cdot \| : R \to \mathbb{R}$ can be naturally extended to L_h as follows. For $(r, r') \in L_h(N, q)$, we set

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

Note that 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 $\mathcal{S}_\delta$ denote the following subset of R:

$$\mathcal{S}_\delta = \{r \in R : \delta N - 1 < \|r\|^2 < \delta N + 1\}.$$

We now construct a particularly useful class of lattices. We start by choosing $f, g \in S_\delta$ such that f and g are invertible modulo q, i.e. so that there are polynomials $f^{-1}, g^{-1} \in R$ satisfying $f * f^{-1} \equiv g * g^{-1} \equiv 1 \pmod q$. The process of computing f^{-1} and g^{-1} from f and g is described in [5]. Next we find polynomials $F, G \in R$ satisfying $f * G - g * F = q$. See [8] for an algorithm to

construct $F, G \in R$ for most randomly chosen $f, g \in S_\delta$. We note for future reference that if F and G are constructed using the method from [8], then they will satisfy

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

Having found the 4-tuple (f, g, F, G), we set

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

Again for future reference, we observe that (2) implies that there exist $k_1, k_2 \in R$ such that

$$f * h = F + k_1 q \qquad \text{and} \qquad g * h = G + k_2 q. \tag{3}$$

The corresponding lattice L_h is called an NTRUSign *lattice*, the polynomial h is called the *public key*, and the pair (f, g) is called the *private key*. The importance of the NTRUSign lattice is that it can be described by two distinctly different bases. First, L_h is generated by all linear combinations of the rows of the matrix

$$\begin{pmatrix} 1 & h \\ 0 & q \end{pmatrix}. \tag{4}$$

Here, as in [8], the 2-by-2 matrix in (4) is an abbreviation for the $2N$-by-$2N$ matrix whose four N-by-N square blocks are the N-by-N circulant matrices corresponding to $1, h, 0, q$. The matrix (4) is the *public basis* for L_h, since the integers (N, q) and the polynomial h are public knowledge.

However, the construction of L_h reveals another basis, namely the rows of the following matrix, which form a *private basis* for L_h:

$$\begin{pmatrix} f & F \\ g & G \end{pmatrix}. \tag{5}$$

The two bases (4) and (5) are related by the following formula, where k_1 and k_2 are given by (3):

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

Our parameter generation algorithm must output a description of $\mathcal{S}_\delta$. In this paper, we will take all private key polynomials f and g to be drawn from a space of trinay polynomials of the following form.

Definition 1. *For a given positive integer d, the space $\mathcal{T}(d)$ is defined to be the set of all $r \in R$ such that $d+1$ coefficients of r are equal to 1, d coefficients of r are equal to -1, and the remaining coefficients are equal to 0*

Definition 2. *Define δ_d to be the quantity*

$$\delta_d = \frac{2d+1}{N} - \frac{1}{N^2}.$$

We observe that

$$\frac{\|f_d\|^2}{N} = \delta_d \qquad \text{for all } f_d \in \mathcal{T}(d).$$

Note that our parameter generation algorithm must determine a value for d. This is then used to compute δ_d, which is the value of δ used to define the space $\mathcal{S}$ from which f and g are chosen.

2.3 Sketch of NTRUSign: Signing

Signing and Verification — The signature algorithm takes as input a digital document D and the private key (f, g, F, G) and outputs a signature s. (For further details, see Appendix A). In [8], the recipient verifies the signature by checking 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* specified by the parameter generation algorithm.

In the transpose lattice, the norm of s is typically smaller than the norm of $s * h - m$ by a factor of $\sqrt{12/N}$. We therefore generalize the norm $\|(r, r')\|$ to include a *balancing factor* $\beta > 0$, which leads to the definition

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

Verification then consists of checking that

$$\|(s, s * h - m(D))\|_\beta \leq \mathcal{N} \ . \tag{6}$$

One way to interpret the β-norm $\|(r, r')\|_\beta$ is as the usual norm of the point $(r, \beta r')$ in the lattice

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

Of course, $L_h(\beta)$ may no longer live in $\mathbb{Z}^{2N}$, but it is a lattice in $\mathbb{R}^{2N}$.

Signing Failures — Depending on the choice of parameter set, it may be possible for the signing algorithm to fail because it produces an s with $\|(s, s * h - m(D))\|_\beta > \mathcal{N}$. To address this potential difficulty, either the signer should use a parameter set for which 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 resign 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 the formula

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

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 smaller than $e^{-C(N)(\rho-1)}$, where $C(N) > 0$ increases with N. In fact, under the assumption that each coefficient of a signature can be treated as a sum of independent identically distributed random variables, a theoretical analysis indicates that $C(N)$ grows quadratically in N.

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

Transcript Analysis — Signing is not zero-knowledge, since a transcript of signatures leaks information about the private key [4, 8]. The number of signatures that an attacker must acquire in order to mount the best currently known attack can be greatly increased by the use of *perturbations* as 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 of 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 and more efficient forms for the perturbations.

Our parameter generation algorithm must 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.

Before giving the algorithm, we remind the reader of the quantities that the algorithm takes as input and the quantities that it provides as output.

Input to the NTRUSign Parameter Generation Algorithm

k	the desired security level in bits
ρ	the signing tolerance $\rho = \mathcal{N}/\mathcal{E}$ defined in Section 2.3
$N_{\max}$	try all values of N up to this $N_{\max}$

Output from the NTRUSign Parameter Generation Algorithm

N	polynomials have degree $< N$
q	coefficients of polynomials are reduced modulo q
d	polynomials in $\mathcal{T}(d)$ have $d+1$ coefficients equal to 1, have d coefficients equal to -1, and the other coefficients are 0.
$\mathcal{N}$	the norm bound used to verify a signature.
β	the balancing factor for the norm $\| \cdot \|_\beta$.

The parameter sets given below for various values of the bit security parameter k were generated with $\rho = 1.1$ and $N_{\max} = 2 \cdot \rho \cdot k$. These parameter sets are recommended parameter sets for NTRUSign at the k bit security level.

The algorithm presented here generates parameter sets that are verifiable (i.e. 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, such as 8-bit processors, various fine tunings and alterations can be made to the algorithm. Future papers will present modified algorithms tailored to these environments.

A Parameter Generation Algorithm For NTRUSign

1. Set $N = 1$.
2. Increment N to the next prime strictly larger than N.
3. If $N > N_{\max}$, then go to step 18.
4. Set $q = 32$.
5. Set $q_{\max} = 2^{\lfloor \log_2(\frac{2\pi e(2/3+1/N-1/N^2) {N_{\max}}^{3/2}}{\sqrt{3}(3.8)^2} \rfloor}$
6. If $q > q_{\max}$, then go to step 2.

7. Let d be the smallest positive integer satisfying

$$d \le N/3 \qquad \text{and} \qquad \omega_{\text{cmb}} \stackrel{\text{def}}{=} \log_2 \left(\frac{\binom{N}{d+1}}{\sqrt{N}} \right) > k.$$

If no such d exists, go to step 2; otherwise proceed to the next step. (The quantity ω_{cmb} is the *combinatorial security*.)

8. Compute $\delta_d = (2d+1)/N - 1/N^2$ (cf. definition 2) and set

$$c = \sqrt{\frac{2\pi e \delta_d N^{3/2}}{q\sqrt{3}}}\ . \tag{7}$$

9. Refer to Table 1(a) to obtain the lattice key security constants $A(c)$ and $B(c)$.
10. Set $A = A(c)$ and $B = B(c)$ and compute

$$\omega_{\text{lk}}(A, B, N) = AN - B - \max_{0 \le r < N-2d} \left(\log_2 \left(1 - \left(1 - \prod_{i=0}^{2d} \left(1 - \frac{r}{N-i} \right) \right)^{2N} \right) + \frac{Ar}{2} \right). \tag{8}$$

The maximum is taken over integers r, so we note that there will be a maximum value. The "lk" subscript indicates that ω_{lk} measures the security against *lattice key recovery attacks*.

11. If $\omega_{\text{lk}} < k$ and $d < \frac{1}{3}N - 1$, increment d by 1 and go to step 8.
12. If $d \ge \frac{1}{3}N$, go to step 2.
13. Compute smallest value of β satisfying

$$\sqrt{\frac{12}{N}} \le \beta \le 1 \quad \text{and} \quad \omega_{\text{frg}} \stackrel{\text{def}}{=} -\frac{1}{2} \log_2 \left(\frac{\pi^{N/2}}{\Gamma(1 + N/2)} \cdot \left(\frac{\rho N}{q\beta} \sqrt{\frac{\delta_d}{3} \left(1 + \frac{\beta^2 N}{12} \right)} \right)^N \right) > k. \tag{9}$$

(The quantity ω_{frg} is the effort required by the best known combinatorial forgery attacks.) If there is no β satisfying these two conditions, then set $q = 2q$ and return to step 6. Otherwise, set

$$\mathcal{N} = \frac{\rho N}{6} \sqrt{\delta_d (12 + \beta^2 N)} \tag{10}$$

and continue.

14. The effectiveness of forgery attacks based on solving CVP using the public key h and the lattice L_h are characterized by the two quantities N/q and

$$\gamma(N, q, \beta, \delta, \rho) \stackrel{\text{def}}{=} \rho \sqrt{\frac{\pi e \delta}{6q} \left(\frac{1}{\beta} + \frac{\beta N}{12} \right)}\ . \tag{11}$$

Calculate $\gamma(N, q, \beta, \delta_d, \rho)$. Refer to Table 1(b) to obtain ω_{lf}, the strength against *lattice-based forgery attacks*. Note that in order to use a particular line of Table 1(b), both γ and N/q must be less than the listed values.

15. If $\omega_{\text{lf}} < k$, set $q = 2q$ and return to step 6.
16. When we arrive at this step, the computed parameters $(N, q, d, \mathcal{N}, \beta)$ give the desired k bit security level against all known attacks. Store the parameters $(N, q, d, \mathcal{N}, \beta)$ and the following additional quantities:

$$\begin{aligned} \sigma_S &= 8dN + N^2 && = \text{time to sign}, \\ \sigma_V &= N^2 && = \text{time to verify}, \\ b_{\text{pk}} &= N \cdot \log_2 q && = \text{size of public key and signature (in bits)}, \\ \tau &= 2^9 d^6 && = \text{transcript length}. \end{aligned} \tag{12}$$

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

(a) Constants used to calculated bit security against lattice key attacks, based on experimental evidence for different values of c

bound for γ and N/q	$\omega_{\rm lf}(N)$
$\gamma < 0.1774$ and $N/q < 1.305$	$0.995113N - 82.6612$
$\gamma < 0.1413$ and $N/q < 0.707$	$1.16536N - 78.4659$
$\gamma < 0.1400$ and $N/q < 0.824$	$1.14133N - 76.9158$

(b) Bit security against lattice forgery attacks, $\omega_{\rm lf}$, based on experimental evidence for different values of $(\gamma, N/q)$

Table 1. Experimentally determined quantities

17. Go to step 2.
18. Check the valid parameter sets $(N, q, d, \mathcal{N}, \beta)$ and associated quantities (12) that were stored in calls to step 16. Depending on requirements, select the one that gives the lowest value of σ_S, or of σ_V, or of $b_{\rm pk}$, or the one that gives the highest value of τ. Output this parameter set and terminate.

2.5 Asymptotic aspects of the NTRUSign parameter generation algorithm

We will demonstrate in this section that given an input ρ and k there exists a constant $\alpha_1 > 0$ such that if $N_{\max} = \alpha_1 k$ then there exist parameters satisfying the requirements of the NTRUSign parameter generation algorithm. In fact, we will show that there exist constants $\alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6$, depending only upon ρ, such that for any $N_{\max}/2 < N < N_{\max}$, the corresponding $d, q, \beta, \mathcal{N}$ can be chosen to satisfy

1. $\alpha_2 N < d < N/3$,
2. $\alpha_3 N < q < \alpha_4 N$,
3. $\beta = \alpha_5/\sqrt{N}$.
4. $\mathcal{N} = \alpha_6 N$.

Begin by fixing any $1/3 > \alpha_2 > 0$. Then if d is chosen in the above range it is simple to verify using Stirling's formula that there exists a constant α_7 such that for any $N > \alpha_7 k$, the combinatorial security bound $\omega_{\rm cmb}$ defined in step 7 will be larger than k.

Next choose, α_3, α_4 such that $\alpha_3 > (0.707)^{-1}$ and a power of 2 lies in the interval $(\alpha_3 N, \alpha_4 N)$. Set q equal to this power of 2. With these choices, there exists a constant α_8 such that c, defined in (7), satisfies $c = \alpha_8 N^{1/4}$. Thus there exists a constant α_9 such that for any $N > \alpha_9 k$, $c > 3.7$. The security lower bound of Table 1(a) corresponding to $c > 3.7$ increases linearly with N. The effect of zero forcing is simply to decrease the slope by at most a constant factor. Thus there exists a constant α_{10} such that for any $N > \alpha_{10} k$, the lattice security bound $\omega_{\rm lk}$ defined in (8) will be larger than k.

Referring now to (26), (27) and (29) we see that the expected signature sizes will in general satisfy $\mathcal{E}_s = \alpha_{11} N$ and $\mathcal{E}_t = \alpha_{12} N^{3/2}$ for constants α_{11}, α_{12} that depend on the size of the bases used. Setting, $\beta = \mathcal{E}_s/\mathcal{E}_t$ we then have $\beta = \alpha_5/\sqrt{N}$, where $\alpha_5 = \alpha_{11}/\alpha_{12}$. Note that in this asymptotic analysis we are describing the parameters such as β in slightly greater generality than in the preceding algorithm. Referring to (16) we set $\mathcal{N} = \alpha_6 N$, where $\alpha_6 = \rho \alpha_{11} \sqrt{2}$. We also note that an application of Sterling's formula shows that with this definition of β there must exist a constant α_{13} such that $\omega_{\rm frg} > k$ for $N > \alpha_{13} k$. Here $\omega_{\rm frg}$ is defined in (9).

With these constraints, the value of $\gamma(N, q, \beta, \delta, \rho)$ given by (11) or by (20) must satisfy $\gamma(N, q, \beta, \delta, \rho) = \rho \alpha_{13} N^{-1/4}$ for some $\alpha_{13} > 0$. Because of the constraint $\alpha_3 > (0.707)^{-1}$ we satisfy the N/q requirements of Table 1(b), and as $\gamma(N, q, \beta, \delta, \rho) = \rho \alpha_{14}/N^{1/4}$, the linear lower bounds

Parameters						Security Measures						
k	N	d	q	β	$\mathcal{N}$	ω_{cmb}	c	ω_{lk}	ω_{frg}	γ	ω_{lf}	$\log_2(\tau)$
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 2. Parameters and relevant security measures for trinary keys, one perturbation, $\rho = 1.1$, q = power of 2

for ω_{frg} guarantee that there exists a constant α_{15} such that for $N > \alpha_{15}k$ we will have $\omega_{\text{lf}}(N) > k$. Choosing α_1 to be the maximum of $\alpha_7, \alpha_{10}, \alpha_{13}, \alpha_{15}$ we achieve our goal.

Note that as $\gamma(N, q, \beta, \delta, \rho) = \rho\alpha_{13}N^{-1/4}$, by (18) the algorithm will terminate with a signature norm on the order of less than a constant times $N^{1/4}$ times the size of the expected smallest vector. This is better, by a factor of $N^{1/4}$ than the $N^{1/2}$ times the expected smallest achieved without the use of β. Because of this, the resistance to lattice reduction based forgery is asymptotically higher with this approach. It should be remarked though, that the norm of a random forgery remains within a constant factor of the norm of a valid signature. However, as the constant is worse, the chance of success of a random forgery decays exponentially with N, as quantified by (9).

2.6 Recommended Parameter Sets

The parameter sets in Table 2 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 2 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 $\rho = 1$.

3 Security considerations

In this section we review the security considerations that have led to the algorithm proposed in Section 2.4. The standard definition of "unforgeability against adaptively chsoen message attacks" is given in [10]. However, NTRUSignrequires us to weaken our notion of unforgeability and allow the adversary access to only a bounded, but large, number of signatures. In order for this model to be a secure and effective signature scheme, the following security issues must be addressed:

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 (6).
3. Given the additional information consisting of f and g, there should be a computationally efficient method to create a signature s on D satisfying (6).
4. Given only the public parameters, public key h, and a long transcript of valid signatures

$$(D_1, s_1), (D_2, s_2), \ldots, (D_\tau, s_\tau),$$

it should be computationally infeasible to create a valid signature pair (D, s) for any message digest D not already in the list.

Item (1) has been well studied already and is essentially the NTRU key recovery problem. We discuss this in Section 4. Item (2) is really a subset of item (4), but for conceptual reasons it is considered separately in Section 5. Item (3), the method of computing a signature given the private key, is briefly reviewed in Appendix A, and item (4) is 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 exposited 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 $\mathrm{Comb}[\mathcal{S}]$. 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

$$\mathrm{Comb}[\mathcal{T}(d)] > \binom{N}{d+1} / \sqrt{N}. \tag{13}$$

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]:

$$\sigma(N, q, \delta, \lambda) = \sqrt{\frac{Nq\lambda}{\pi e}}. \tag{14}$$

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

$$c = \sqrt{\frac{2\pi e \delta N^{3/2}}{3^{1/2} q}}. \tag{15}$$

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(a).

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 $2N$ that appears in the exponent. 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

$$\mathcal{N} = \rho\sqrt{\mathcal{E}_s^2 + \beta^2 \mathcal{E}_t^2}. \tag{16}$$

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 $k_1, k_2 \in R$

$$(\|(s + k_1 q\|^2 + \beta^2 \|h * s - m + k_2 q)\|^2)^{1/2} < \mathcal{N}.$$

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 $\mathcal{N}/\beta$ 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:

$$p(N, q, \beta, \mathcal{N}) < \sqrt{\frac{\pi^{N/2}}{\Gamma(1 + N/2)} \cdot \left(\frac{\mathcal{N}}{q\beta}\right)^N}. \tag{17}$$

If k is the desired bit security level it will suffice to choose parameters so that the right hand side of (17) 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:

$$\gamma(N, q, \beta) = \frac{\mathcal{N}}{\sigma(N, q, \delta, \beta)\sqrt{2N}}. \tag{18}$$

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 1(b).

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 (14),(16),(18) we have

$$\gamma(N,q,\beta) = \rho\sqrt{\frac{\pi e}{2N^2q}\cdot(\mathcal{E}_s^2/\beta + \beta\mathcal{E}_t^2)} \tag{19}$$

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

$$\gamma(N,q,\beta) = \rho\sqrt{\frac{\pi e\mathcal{E}_s\mathcal{E}_t}{N^2q}}. \tag{20}$$

Referring to (17) 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 (17) 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 $\{f_i,g_i,F_i,G_i\}, i = 1,\ldots,T$. 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 (28) have at his disposal a long list of pairs of polynomials of the form

$$s = \epsilon f + \epsilon' g + \epsilon_1 f_1 + \epsilon_1' g_1 + \ldots + \epsilon_T f_T + \epsilon_T' g_T \tag{21}$$

and

$$t - m = \epsilon F + \epsilon' G + \epsilon_1 F_1 + \epsilon_1' G_1 + \ldots + \epsilon_T F_T + \epsilon_T' G_T. \tag{22}$$

Let $f_0 = f, F_0 = F, \epsilon_0 = \epsilon, \ldots$ for the purpose of having a more uniform notation. Then by (23), for $i = 0,\ldots T$

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

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 (21) and (22) is (up to lower order terms)

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

and

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

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 calcalation 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 Szydlo, *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 $\lfloor a \rceil$ denote the integer closest to a, and define $\{a\} = a - \lfloor a \rceil$. (For numbers a that are midway between two integers, we specify that $\{a\} = +\frac{1}{2}$, rather than $-\frac{1}{2}$.) If A is a polynomial with rational (or real) coefficients, let $\lfloor A \rceil$ 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 and ϵ' with rational coefficients varying uniformly between $-1/2$ and $1/2$ by the formulas

$$\lfloor x \rceil = x + \epsilon \qquad \text{and} \qquad \lfloor y \rceil = y + \epsilon'.$$

Thus

$$\epsilon = -\{x\} = \left\{ \frac{m * g}{q} i \right\} \qquad \text{and} \qquad \epsilon' = -\{y\} = -\left\{ \frac{m * f}{q} \right\} \tag{23}$$

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

$$\mathcal{E}_\epsilon = \sqrt{N/12}. \tag{24}$$

(This is approximately, but not exactly, correct if q is even, due to our arbitrary choice that $\{\frac{1}{2}\} = \frac{1}{2}$, rather than $-\frac{1}{2}$. However, it is easy to compute the necessary correction.)

Letting

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

we then obtain

$$(s, t - m) = (\epsilon f + \epsilon' g, \epsilon F + \epsilon' G). \tag{25}$$

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 (25)

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

By (24) and our assumption that $f, g \in \mathcal{S}_\delta$ it follows that

$$\mathcal{E}_s = \sqrt{\frac{N^2 \delta}{6}}. \tag{26}$$

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

$$\mathcal{E}_t = \sqrt{\frac{N^3 \delta}{72}}. \tag{27}$$

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 (23) 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

$$(s, t - m) = (\epsilon f + \epsilon' g + \epsilon_1 f_1 + \epsilon_1' g_1, \epsilon F + \epsilon' G + \epsilon_1 F_1 + \epsilon_1' G_1). \tag{28}$$

The important observation from the point of view of security is that the distribution of the ϵs is the same as the distribution of the ϵ_1s. 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

$$\mathcal{E}_s = \sqrt{\frac{N^2(\delta + \delta_1 + \delta_2)}{6}}, \quad \mathcal{E}_t = \sqrt{\frac{N^2(\delta\omega + \delta_1\omega_1 + \delta_2\omega_2)}{6}}, \tag{29}$$

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 result.

Proposition 1. *Let $Y_1, \ldots, Y_T$ be a collection discrete random variables such that there are constants B and σ with the property that for every $1 \le i \le T$,*

$$|Y_i| < B, \qquad E(Y_i) = 0, \qquad \textit{and} \qquad E(Y_i^2) = \sigma.$$

Then for any $K > 0$,

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

When calculating the $2k^{th}$ 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 $l_1 + \ldots + l_k + m_1 + \ldots m_k + r \equiv l \pmod{N}$. 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 Proposition 1 yields

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

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 3 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 3 gives the performance measures for each of the recommended parameter sets.

Table 4 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 3. 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 4. Trinary keys, one perturbation, $\rho = 1$, q = power of 2

signature sizes