

NTRU Cryptosystems Technical Report

Report # 017, Version 2

Title: Enhanced Encoding and Verification Methods for the NTRU Signature Scheme

Author: Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman

Release Date: May 30, 2001

Abstract. This report describes enhanced encoding and verification methods for the NTRU Signature Scheme (NSS).

Section 1. The Hard Problem Underlying NTRU and NSS

The NTRU Signature Scheme (NSS) is a digital signature scheme based on a hard lattice problem. This lattice problem also underlies the NTRU Public Key Cryptosystem described in [1]. In this section we remind the reader of the standard description of the NTRU lattice problem in terms of products of polynomials in convolution rings. For further details, see [1].

Fix three positive integer parameters N , p , and q with $\gcd(p, q) = 1$. Let R be the ring

$$R = \mathbf{Z}[X]/(X^N - 1).$$

Multiplication of two polynomials $a(X)$ and $b(X)$ in the ring R is equivalent to taking the convolution product of the coefficient vectors of a and b . We will often identify a polynomial in R with its N -dimensional vector of coefficients.

The private key for NTRU and NSS consists of two polynomials $f(X)$ and $g(X)$ having small coefficients, possibly with some further structure. The public key is the polynomial

$$h(X) \equiv f(X)^{-1} \cdot g(X) \pmod{q},$$

where $f(X)^{-1}$ denotes the inverse of $f(X)$ modulo q , i.e., the inverse of $f(X)$ in the ring R/qR .

The set of all pairs of polynomials $[a(X), b(X)] \in R \times R$ satisfying

$$a(X) \cdot h(X) \equiv b(X) \pmod{q}$$

forms a $2N$ -dimensional lattice L_h^{NT} . The vector $[f(X), g(X)]$ is a short vector in L_h^{NT} , but if N is large and the other parameters are chosen appropriately, then it is a very difficult problem to either find $[f(X), g(X)]$ or any other vector of similar length in the lattice L_h^{NT} . It is this hard problem that underlies both the NTRU Public Key Cryptosystem and the NTRU Signature Scheme.

Section 2. The NTRU Signature Scheme

The NTRU Signature Scheme (NSS) uses a public key $h(X)$ as described in Section 1. The secret that the signer, call him Bob, possesses is knowledge of two small polynomials $f(X)$ and $g(X)$ (i.e., polynomials with small coefficients) that factor $h(X)$ as a product $h(X) \equiv f(X)^{-1} \cdot g(X) \pmod{q}$. Bob uses his secret to produce a signature s for his digital document D . The verifier, call her Alice, checks that s is tied to both the document D and the public key $h(X)$.

The final requirement for NSS is that it must be difficult for a forger, call him Fred, to produce a valid signature unless he knows a decomposition

$$h(X) \equiv F(X)^{-1} \cdot G(X) \pmod{q}$$

using small polynomials $F(X)$ and $G(X)$. In other words, it should be difficult for Fred to find a valid signature unless he is able to solve the hard underlying lattice problem.

The fundamental structure of NSS may be summarized as follows:

- **Message Encoding**

Bob chooses a small polynomial $w(X)$ that encodes the digital document D .

- **Signing**

Bob publishes the signature $s(X) \equiv f(X) \cdot w(X) \pmod{q}$.

- **Using the Public Key**

Alice uses the public key to compute $t(X) \equiv h(X) \cdot s(X) \pmod{q}$. Notice that $t(X)$ is actually equal to $g(X) \cdot w(X) \pmod{q}$, so both $s(X)$ and $t(X)$ are products of small polynomials. Of course, Alice does not know the value of $f(X)$ or $g(X)$ or $w(X)$.

- **Verification**

Alice performs one or more verification tests to verify that $s(X)$ and $t(X)$ are tied to the digital document D and that they have the appropriate characteristics to be products of two small polynomials.

We will take up the dual questions of message encoding and signature verification in the next section.

Section 3. Enhanced Encoding and Verification Methods for NSS

There are a many ways in which information about the digital document D can be encoded in the polynomial $w(X)$, and there are similarly many ways in which one can verify that $s(X)$ and $t(X)$ are tied to D and look like products of small polynomials. The article on NSS in the Eurocrypt proceedings [2] described one such method. In this section we describe some enhanced encoding/verification techniques that offer additional resistance to forgery and transcript analysis attacks.

The NTRU Public Key Cryptosystem uses a small modulus p that is relatively prime to the larger modulus q as a method for separating the plaintext from random masking material. The NTRU Signature Scheme may similarly use a small modulus p in order to tie the digital document D to the signature. To accomplish this goal, it is convenient to form the private key polynomials in the following way.

Select three small secret polynomials $u(X)$, $F(X)$, and $G(X)$. The private key consists of the polynomials

$$\begin{aligned} f(X) &= u(X) + pF(X), \\ g(X) &= u(X) + pG(X). \end{aligned}$$

For signing purposes, it is also convenient to compute and store the polynomial

$$U(X) = u(X)^{-1} \pmod{p}.$$

The digital document D is converted, using an appropriate method (e.g., a hash function) for computing a message representative, into a small polynomial $m(X)$. It is encoded into the polynomial $w(X)$ by choosing small masking polynomials $w_1(X)$ and $w_2(X)$, letting

$$w_0(X) = (U(X)m(X) \pmod{p}) + (U(X)w_1(X) \pmod{p}),$$

and setting

$$w(X) = w_0(X) + pw_2(X).$$

(The polynomial w_2 is partially random and partially chosen to equalize coefficient frequency distributions, while w_1 may be chosen to enhance certain masking properties. This will be described more fully below.) Bob's signature is the polynomial

$$s(X) \equiv f(X) \cdot w(X) \pmod{q}.$$

Alice begins the verification process by using Bob's public key $h(X)$ to compute

$$t(X) \equiv h(X) \cdot s(X) \equiv g(X) \cdot w(X) \pmod{q}.$$

She now subjects $s(X)$ and $t(X)$ to a number of tests to determine if $s(X)$ is a valid signature. Among the many types of tests available to Alice, we mention the following three:[†]

Signature Norm Binding Tests. The *centered norm* of a polynomial $a(X) = a_0 + a_1X + \cdots + a_{N-1}X^{N-1}$ is the quantity

$$\|a\| = \sqrt{\sum_{0 \leq i < N} a_i^2 - \frac{1}{N} \left(\sum_{0 \leq i < N} a_i \right)^2}.$$

(Equivalently, $\|a\|/\sqrt{N}$ is the standard deviation of the sequence $[a_0, a_1, \dots, a_{N-1}]$.) The centered norm is roughly multiplicative, that is, for random polynomials $a(X)$ and $b(X)$ it satisfies

$$\|a \cdot b\| \approx \|a\| \cdot \|b\|.$$

In particular, since

$$s(X) \equiv f(X) \cdot w(X) \pmod{q} \quad \text{and} \quad t(X) \equiv g(X) \cdot w(X) \pmod{q}$$

are products of small polynomials, their norms will be small. Thus bounds on $\|s\|$ and $\|t\|$ provide a possible test for a valid signature.

However, it turns out that an even better test is obtained by considering the norms of the polynomials

$$\begin{aligned} s'(X) &\equiv p^{-1} \cdot (s(X) - m(X)) \pmod{q}, \\ t'(X) &\equiv p^{-1} \cdot (t(X) - m(X)) \pmod{q}. \end{aligned}$$

We illustrate why s' has small norm, the argument for t' being similar. If we multiply out the expression for $s(X)$, we find that

$$\begin{aligned} s(X) &\equiv f(X) \cdot w(X) \pmod{q} \\ &\equiv (u(X) + pF(X)) \cdot (w_0(X) + pw_2(X)) \pmod{q} \\ &\equiv m(X) + w_1(X) + p \cdot (\text{medium sized polynomial}) \pmod{q}. \end{aligned}$$

In the last line we have used the fact that

$$u(X)w_0(X) \equiv u(X) \cdot (U(X) \cdot m(X) + U(X) \cdot w_1(X)) \equiv m(X) + w_1(X) \pmod{p},$$

[†] Before performing mod p reduction of a mod q polynomial, the coefficients should be placed in the correct interval of length q . For example, if the coefficients of $f(X)$, $g(X)$, $m(X)$, $w_0(X)$, $w_1(X)$ and $w_2(X)$ are chosen more-or-less symmetrically around 0, then the coefficients of $s(X)$ and $t(X)$ would be chosen in the range from $-q/2$ to $q/2$.

since by definition $u(X)U(X) \equiv 1 \pmod{p}$. Hence

$$s'(X) \equiv p^{-1} \cdot w_1(X) + (\text{medium sized polynomial}) \pmod{q}.$$

Thus aside from the term $p^{-1}w_1(X)$, which does not make a large contribution because $w_1(X)$ has very few nonzero coefficients, we see that $s'(X)$ has small norm.

Thus one test that Alice will perform to verify Bob's signature is to check that the centered norms $\|s'\|$ and $\|t'\|$ are smaller than a specified bound. She may also check that the centered norm of the $2N$ -dimensional vector (s', t') is smaller than a specified bound.

Remark. The condition that $\|(s', t')\|$ be smaller than a specified bound provides a direct link between a valid signature and a hard lattice problem. We briefly sketch; see [3] for details.

Recall that the standard NTRU lattice L_h^{NT} attached to the polynomial h is the $2N$ -dimensional lattice consisting of all vectors $(u, v) \in \mathbf{Z}^{2N}$ satisfying

$$v \equiv h \cdot u \pmod{q}.$$

The private key pair (f, g) (and its rotations $(X^i \cdot f, X^i \cdot g)$) are probably the shortest vectors in L_h^{NT} . Now consider a message m that is to be signed, and let a_m denote the polynomial

$$a_m(X) \equiv p^{-1} \cdot (m(X) - h(X) \cdot m(X)) \pmod{q}.$$

One can show that if $s(X)$ is a valid signature on $m(X)$ and $s'(X)$ and $t'(X)$ are defined as above, then the vector $(s', t' + a_m)$ is in the lattice L_h^{NT} and its distance to the known point $(0, a_m)$ is $\|(s', t')\|$.

Thus a valid signature on m gives a vector in L_h^{NT} that is close to a target vector $(0, a_m)$ that depends on m . A more detailed analysis using the Gaussian heuristic shows that the distance from $(s', t' + a_m)$ to $(0, a_m)$ is a constant multiple of the probable distance from the closest vector in L_h^{NT} to $(0, a_m)$. In other words, creating a valid signature is (heuristically) at least as hard as solving a closest vector problem in L_h^{NT} up to a constant factor. This is significant, since it is known that solving SVP or CVP up to a constant factor in a general lattice is a very hard problem.

Signature Congruence Binding Tests. Alice compares

- (i) $s(X) \pmod{p}$ to $m(X) \pmod{p}$,
- (ii) $t(X) \pmod{p}$ to $m(X) \pmod{p}$.

For appropriately chosen parameters, there will be good agreement in both case (i) and case (ii). We illustrate why this is true for $s(X)$, the argument for $t(X)$ being similar.

We saw earlier that

$$s(X) \equiv m(X) + w_1(X) + p \cdot (\text{medium sized polynomial}) \pmod{q}.$$

Even when multiplied by p , the “medium sized polynomial” will not have too many coefficients outside the chosen interval mod q , so when $s(X)$ is reduced modulo p , most of its coefficients will satisfy

$$s_i \equiv m_i + w_{1,i} \pmod{p}.$$

Thus $s(X) \bmod p$ looks like $m(X)$ up to the errors introduced by $w_1(X)$ and by nontrivial reduction modulo q in the product $f(X)w(X)$. Notice that NSS is using the fact that $f(X)$ and $w(X)$ are small, which ensures that $f(X)w(X)$ does not have a large amount of reduction modulo q .

Thus Alice can check that the differences $s(X) - m(X) \bmod p$ and $t(X) - m(X) \bmod p$ do not have too many nonzero entries, which is the test suggested in [2]. We will call these nonzero entries *deviations*. It turns out that the deviations in a true signature (i.e., one created using the private key) satisfy much more stringent criteria than simply being scarce, and these criteria can be used to craft a much stronger test of validity.

The deviations in s and t come from two sources:

- (1) Nontrivial mod q reduction in $f(X) \cdot w(X)$ and $g(X) \cdot w(X)$.
- (2) Nonzero entries in $w_1(X)$.

The Type 1 deviations will tend to cluster in a particular mod p congruence class depending on whether the corresponding coefficient of $s(X)$ is nearer the top or the bottom of the specified mod q interval. More precisely, it will tend to equal $q \bmod p$ (respectively $-q \bmod p$) if the coefficient of $s(X)$ is near to the top (respectively bottom) of the mod q interval. A similar remark applies to $t(X)$. Thus Alice may check that almost all of the (Type 1) deviations in $s(X)$ and $t(X)$ lie in the appropriate mod p congruence class.

As for the Type 2 deviations, they are quite scarce. Further, Bob has a great deal of freedom in choosing the polynomial $w_1(X)$, and if he chooses it to hide some of the common mod q reduction, then w_1 will tend to cause deviations to disappear entirely, so actually will not add very much to the total.

For concreteness, we will describe a precise Signature Congruence Binding Test, but we emphasize that there are a number of similar tests that one might perform. We let I_1, \dots, I_4 denote the four quartiles modulo q ,

$$I_1 = (-q/2, -q/4], \quad I_2 = (-q/4, 0], \quad I_3 = (0, q/4], \quad I_4 = (q/4, q/2].$$

Now consider the following two quantities, which measure the deviations that occur outside of where we expect them to cluster. In each set, we are counting the number

of indices $0 \leq j < N$ for which the indicated coefficients of s , t , and m have the indicated properties.

$$\begin{aligned}
 \text{Dev}_1 &= \overline{\# \{j : s_j \in I_4 \text{ and } s_j - m_j \not\equiv 0, q \pmod{p}\}} \\
 &\quad + \# \{j : s_j \in I_1 \text{ and } s_j - m_j \not\equiv 0, -q \pmod{p}\} \\
 &\quad + \# \{j : t_j \in I_4 \text{ and } t_j - m_j \not\equiv 0, q \pmod{p}\} \\
 &\quad + \# \{j : t_j \in I_1 \text{ and } s_j - m_j \not\equiv 0, -q \pmod{p}\} \\
 \text{Dev}_2 &= \overline{\# \{j : s_j \in I_3 \text{ and } s_j - m_j \not\equiv 0, q \pmod{p}\}} \\
 &\quad + \# \{j : s_j \in I_2 \text{ and } s_j - m_j \not\equiv 0, -q \pmod{p}\} \\
 &\quad + \# \{j : t_j \in I_3 \text{ and } t_j - m_j \not\equiv 0, q \pmod{p}\} \\
 &\quad + \# \{j : t_j \in I_2 \text{ and } s_j - m_j \not\equiv 0, -q \pmod{p}\}
 \end{aligned}$$

Alice verifies that Dev_1 and Dev_2 are smaller than some specified quantities.

Remark. Although lattice methods are not the focus of this note, we mention that it is important for $f(X)w(X)$ and $g(X)w(X)$ to have enough nontrivial reduction modulo q so as to prevent Fred from recovering their values exactly in R , since otherwise he can work with an N -dimensional lattice and use it to try to recover $f(X)$ and $g(X)$. The purpose of w_1 is to hide enough of the nontrivial reduction to make it infeasible to lift $s(X)$ to $f(X)w(X)$, or to lift $t(X)$ to $g(X)w(X)$. For a similar reason, it is important the the polynomial $(F(X) - G(X))w(X)$ have enough nontrivial reduction modulo q to prevent Fred from recovering its exact value from its value modulo q , since he can certainly find its value modulo q from the congruence

$$(F(X) - G(X))w(X) \equiv p^{-1}(s(X) - t(X)) \pmod{q}.$$

It is thus advisable that when Bob creates a signature, he check that there is sufficient mod q reduction and start with a new w_2 if there is not.

Signature Coefficient Magnitude Tests. The signature norm binding test described earlier checks that the polynomials

$$s' \equiv p^{-1}(s - m) \pmod{q} \quad \text{and} \quad t' \equiv p^{-1}(t - m) \pmod{q}$$

have small (centered) norm. The fact that these polynomials are actually sums of products of small polynomials (if s has been formed using the private key) implies further that the actual coefficients of s' and t' tend to be fairly small. On the other hand, they will not have a huge number of small coefficients and a few large coefficients, with nothing in between. Thus Alice can perform a further validity check by verifying that s' and t' have a suitable number of coefficients in certain ranges.

To formulate a precise test, let J_1, J_2, J_3, J_4 be the quartiles

$$J_1 = [0, q/8), \quad J_2 = [q/8, q/4), \quad J_3 = [q/4, 3q/8), \quad J_4 = [3q/8, q/2],$$

and count the number of coefficients of s' and t' whose absolute value lies in each interval.

$$\begin{aligned} \text{Coef}_{s,1} &= \#\{j : |s'_j| \in J_1\} & \text{Coef}_{t,1} &= \#\{j : |t'_j| \in J_1\} \\ \text{Coef}_{s,2} &= \#\{j : |s'_j| \in J_2\} & \text{Coef}_{t,2} &= \#\{j : |t'_j| \in J_2\} \\ \text{Coef}_{s,3} &= \#\{j : |s'_j| \in J_3\} & \text{Coef}_{t,3} &= \#\{j : |t'_j| \in J_3\} \\ \text{Coef}_{s,4} &= \#\{j : |s'_j| \in J_4\} & \text{Coef}_{t,4} &= \#\{j : |t'_j| \in J_4\} \end{aligned}$$

Then Alice may check upper and lower bounds on $\text{Coef}_{s,i}$ and $\text{Coef}_{t,i}$ for each $1 \leq i \leq 4$.

Section 4. Frequency Analysis of Signature Transcripts

The NSS message encoding method described in [2] used $u(X) = 1$ and concealed the message $m(X)$ in the masking polynomial $w(X)$ by choosing $w_2(X)$ to equalize the (first) moment of $m(X) + pw_2(X)$. Szydlo [5] has described a very interesting method for analyzing the frequency distribution of s_k or t_k (i.e., the k^{th} coefficient of s and t , respectively) for a fixed k on a subtranscript of signatures in order to extract information about f_k or g_k . The subtranscripts are selected based on using message polynomials $m(X)$ that have a particular coefficient taking on a specific value. On sufficiently long transcripts (a few tens of thousands), Szydlo's method is effective on signatures generated using $u(X) = 1$ and first moment equalization. We now describe Szydlo's method in more detail and explain why the enhanced message encoding methods detailed in this document prevent it from gaining useful information on even very long transcripts.

Before beginning, we note that if s is a valid signature on m , then $X^i \cdot s$ is a valid signature on $X^i \cdot m$. For this reason, we will restrict attention to the constant coefficient m_0 of m , since each coefficient of m will become the constant coefficient of one of the rotations $X^i \cdot m$.

Now consider a transcript \mathcal{S} consisting of some large number of valid signature-message pairs (s, m) , where we also include all the rotations $(X^i \cdot s, X^i \cdot m)$ in \mathcal{S} . For each value of $0 \leq k < N$ and each value of $\epsilon \in \{-1, 0, 1\}$, we consider the subtranscript consisting of messages with $m_0 = \epsilon$ and assemble a probability function that gives the probability that the k^{th} coefficient of s takes on each particular mod q value,

$$P_{k,\epsilon}(b) = \text{Prob}(s_k = b \mid m_0 = \epsilon) = \frac{\#\{(s, m) \in \mathcal{S} \mid s_k = b \text{ and } m_0 = \epsilon\}}{\#\{(s, m) \in \mathcal{S} \mid m_0 = \epsilon\}}.$$

In order to see why these probability distributions might be useful, we need to write out the expression for s_k . If $u(X) = 1$, then

$$s_k = f_k(m_0 + w_{1,0} + pw_{2,0}) + \sum_{i=1}^{N-1} f_{k-i}(m_i + w_{1,i} + pw_{2,i}).$$

If we restrict attention to signatures on messages with m_0 taking a fixed value, then the distribution of values of s_k will look somewhat different depending on the value of f_k . In other words, by comparing and contrasting the probability distributions $P_{k,\epsilon}(b)$ for different pairs (k, ϵ) , one may gain some information about f_k .

The message encoding method described in [2] includes choosing $w_{2,0}$ in such a way that for every index i , the mean of $m_i + pw_{2,i}$ over a large transcript will be 0. This first moment equalization is accomplished by the simple rule:

Subtract m_i from $w_{2,i}$ with probability $1/p$.

First moment equalization already makes the probability distributions $P_{k,\epsilon}$ look more alike. It is not difficult to extend this idea by equalizing higher moments, thereby making the different $P_{k,\epsilon}$'s even more difficult to distinguish. For example, if $p = 3$, then the following rule makes both the first and second moments of $m_i + pw_{2,i}$ the same, regardless of the value of m_i :

If $m_i = \pm 1$, then subtract m_i from $w_{2,i}$ with probability $1/3$.

If $m_i = 0$, then $\begin{cases} \text{add } 1 \text{ to } w_{2,i} \text{ with probability } 1/9, \\ \text{subtract } 1 \text{ to } w_{2,i} \text{ with probability } 1/9, \\ \text{leave } w_{2,i} \text{ unchanged with probability } 7/9. \end{cases}$

Similar, but slightly more complicated, rules can be used to equalize moments of even higher order; and when higher moments are equalized, it requires even longer transcripts to extract useful information.

Another effective method to inhibit the leakage of information from long transcripts is provided by the encoding method described above in which the private key includes a nontrivial (secret) polynomial $u(X)$. The effect of $u(X)$ is to replace the known message polynomial $m(X)$ in the masking polynomial $w(X)$ with the quantity $U(X) \cdot m(X) \bmod p$, where $U(X) \equiv u(X)^{-1} \bmod p$. The coefficients of $U(X)$ are unknown to an observer, and the coefficients of $U(X) \cdot m(X) \bmod p$ take their values in the set $\{1, 0, -1\}$.

This greatly limits the amount of information available from even a long transcript of signatures. Subtranscripts selected on the basis of coefficients of $m(X)$ will display only tiny frequency differences, which will be drowned out by random noise until the number of signatures collected is immense. Further, moment equalization can also be applied in this situation (i.e., when $u(X) \neq 1$) to further mask frequency

differences. In the following we will expand upon this and quantify the increase in the length of transcript required to extract even partial information.

The analysis of frequency distributions when $u(X)$ is nontrivial proceeds as follows. In this situation, the k^{th} coefficient of s has the form

$$s_k = \sum_{i=0}^{N-1} f_{k-i}(w_{0,i} + pw_{2,i}),$$

where

$$\begin{aligned} w_{0,i} &= (U \cdot m \bmod p)_i + (U \cdot w_1 \bmod p)_i \\ &= \left(\left(\sum_{j=0}^{N-1} U_{i-j} m_j \right) \bmod p \right) + \left(\left(\sum_{j=0}^{N-1} U_{i-j} w_{1,j} \right) \bmod p \right). \end{aligned}$$

Suppose now that we restrict attention to subtranscripts whose messages differ in (say) their constant terms m_0 . Then the quantity

$$\left(\sum_{j=0}^{N-1} U_{i-j} m_j \right) \bmod p = \left(U_i m_0 + \sum_{j \neq 0} U_{i-j} m_j \right) \bmod p \quad (*)$$

will exhibit a slightly different distribution of values depending on the value of U_i , while the other terms in s , those involving w_1 and w_2 , appear as random noise and will eventually average out to zero.

For concreteness, take $p = 3$ and consider the three sets of indices

$$I_\epsilon = \{i : U_i = \epsilon\}, \quad \epsilon \in \{-1, 0, 1\}.$$

Then the quantity $(*)$ will have a certain frequency distribution depending on which I_ϵ contains i . This means that we can view s_k as having the form

$$s_k = \sum_{i \in I_{-1}} f_{k-i} A + \sum_{i \in I_0} f_{k-i} B + \sum_{i \in I_1} f_{k-i} C, \quad (**)$$

where A , B , and C are random variables with slightly different distribution functions. If it is possible to detect the differences in the distributions A , B , and C , then it might be possible to recover the individual sums $\sum_{i \in I_\epsilon} f_{k-i}$.

Note that the quantity $(*)$ that leads to the differences in A , B , and C only takes on three values, so the differences between the the distribution functions of A , B and C will be very small. Thus it will require an enormous number of signatures to distinguish the differences. We can quantify this by the following experiment.

Let U be a (random) polynomial with coefficients taken from the set $\{-1, 0, 1\}$ and let I_ϵ be the sets defined above. Generate a large number of polynomials m

with coefficients taken in $\{-1, 0, 1\}$, but with m_0 fixed to be equal to 1. For each m , compute the quantity (*),

$$(U \cdot m)_i \bmod 3 = \left(\sum_{j=0}^{N-1} U_{i-j} m_j \right) \bmod 3,$$

and use the data to determine experimental probabilities

$$\text{Prob}(\epsilon, b) = \text{Prob}((U \cdot m)_k \equiv b \pmod{3} \mid k \in I_\epsilon).$$

We performed this experiment with a fixed U and a set of 100,000,000 randomly chosen polynomials m . The results are given in the following table.

$\epsilon \setminus b$	-1	0	1
-1	0.33332732	0.33333292	0.33333975
0	0.33332719	0.33333274	0.33334007
1	0.33332735	0.33333231	0.33334033

$$\text{Prob}(\epsilon, b) = \text{Prob}((U \cdot m)_k \equiv b \pmod{3} : k \in I_\epsilon)$$

It is clear that the differences in the three distributions are virtually non-existent after 100 million messages. More precisely, if the distributions were actually identical, then one would expect to find random fluctuations on the order of $1/\sqrt{K}$ in different samples of size K . So the fact that the probabilities in the table show variations smaller than $1/10^4$ indicates that a sample of size 10^8 is insufficient to distinguish the different distributions.

In the interests of completeness, we also consider the security implications if one were able to distinguish the three distributions A , B , and C sufficiently to untangle the three sums in (**). More precisely, suppose that an attacker could recover, either partially or entirely, the $3N$ sums

$$\sum_{i \in I_\epsilon} f_{k-i}, \quad \epsilon \in \{-1, 0, 1\}, \quad 0 \leq k < N. \quad (***)$$

This information can be summarized by writing U as a difference of binary polynomials,

$$U = U_1 - U_2,$$

and noting that

$$\begin{aligned} f \cdot U_1 &= \sum_{k=0}^{N-1} \left(\sum_{i=0}^{N-1} f_{k-i} U_{1,i} \right) X^k = \sum_{k=0}^{N-1} \left(\sum_{i \in I_1} f_{k-i} \right) X^k \\ f \cdot U_2 &= \sum_{k=0}^{N-1} \left(\sum_{i=0}^{N-1} f_{k-i} U_{2,i} \right) X^k = \sum_{k=0}^{N-1} \left(\sum_{i \in I_{-1}} f_{k-i} \right) X^k. \end{aligned}$$

Thus knowledge of all of the sums (***) is equivalent to knowing the products $f \cdot U_1$ and $f \cdot U_2$. And similarly, by working with $t \equiv h \cdot s \pmod{q}$ one might be able to approximate or compute the values of $g \cdot U_1$ and $g \cdot U_2$.

There are several issues that must now be addressed.

- [1] How long a transcript is needed in order to reliably find the values of

$$f \cdot U_1, \quad f \cdot U_2, \quad g \cdot U_1, \quad g \cdot U_2?$$

More precisely, how long a transcript is needed in order to determine (say) two of them closely enough that the remaining coefficients can be found by an exhaustive search?

- [2] Can the quantities $f \cdot U_1$, $f \cdot U_2$, $g \cdot U_1$, and $g \cdot U_2$ be used to directly forge a signature?
- [3] Can the quantities $f \cdot U_1$, $f \cdot U_2$, $g \cdot U_1$, and $g \cdot U_2$ be used to recover the values of f , g , U_1 and/or U_2 .

We consider each of these questions in turn.

[1] **Length of Transcript**

An attacker would need enough signatures to decompose the distribution of s_k in (**) into a linear combination of A , B , and C with coefficients in the range $(-q/2, q/2]$. This is considerably more difficult than simply distinguishing between A , B , and C , and even for this latter task we have had no success using transcripts consisting of 100 million signatures.

[2] **Direct Forgery**

The only natural way to directly forge a signature using $f \cdot U_1$ and $f \cdot U_2$ appears to be to set

$$\begin{aligned} s &\equiv (f \cdot U_1 - f \cdot U_2) \cdot m \pmod{q} \\ &\equiv f \cdot (U_1 - U_2) \cdot m \pmod{q} \\ &\equiv f \cdot U \cdot m \pmod{q}. \end{aligned}$$

Then

$$t \equiv h \cdot s \equiv (f^{-1} \cdot g) \cdot s \equiv g \cdot U \cdot m \pmod{q},$$

so s and t appear to have the correct form to be valid signatures. However, it is easy to check experimentally that since s and t constructed in this way are products of three trinary polynomials, the norms of s' and t' and the deviations of s and t will be far too large to pass the verification tests using the suggested parameters. The crucial point is that someone who knows the private key f can construct a signature

$$s \equiv f \cdot ((U \cdot m \bmod p) + (\text{a small polynomial})) \pmod{q}$$

consisting of f multiplied by a single small polynomial; but an erstwhile forger who knows $f \cdot U$ can only construct $f \cdot U \cdot m$, which is a product of three small polynomials.

[3] Lattice Attacks

If $f \cdot U_1$, $f \cdot U_2$, $g \cdot U_1$, and $g \cdot U_2$ could be recovered modulo q , then there is a reasonable chance that there would be little enough wrapping that they could be lifted to \mathbf{Z} . There are a then a number of different lattice attacks available:

- (a) Choose a (large) integer M and look at the NTRU lattice corresponding to either

$$(fU_1) \cdot (gU_1)^{-1} \pmod{M} \quad \text{or} \quad (fU_2) \cdot (fU_1)^{-1} \pmod{M}.$$

These are $2N$ -dimensional lattices that contain the short vectors (f, g) and (U_1, U_2) respectively.

- (b) Another possibility is to look at the N -dimensional lattice generated by the $2N$ vectors

$$\begin{aligned} fU_1, XfU_1, X^2fU_1, \dots, X^{N-1}fU_1, \\ fU_2, XfU_2, X^2fU_2, \dots, X^{N-1}fU_2. \end{aligned}$$

There is a good chance that the shortest vector in this lattice will be f , and if not, the shortest vector will be some small multiple of f .

However, there are three important points to note when considering the possibility of lattice attacks. First, they can only succeed if two products such as $f \cdot U_1$ and $f \cdot U_2$ are guessed *exactly*; if even a single coefficient is incorrect, then the lattice provides no useful information. Second, even these easier lattice problems are nontrivial. Third, and most importantly, there does not appear to be any practical way of even beginning an improved lattice search using a transcript of fewer than 100 million signatures.

In summary, with the enhanced encoding methods described in this note, there appears to be no practical way to recover any useful information about the distribution of the coefficients of the private key.

Section 5. Sample Parameter Choices

In this section we give specific parameter choices for NSS that appear to yield a security level comparable to that of 1024 bit RSA. We begin with the basic parameters

$$N = 251, \quad p = 3, \quad q = 128.$$

For notational convenience, we define a *ternary polynomial* to be a polynomial whose coefficients are all -1 , 0 , and 1 , and we let

$$\mathcal{T}_N(d_1, d_2) = \{\text{ternary polynomials with } d_1 \text{ } -1\text{'s and } d_2 \text{ } 1\text{'s}\}$$

The private keys have the form

$$f = u + pF \quad \text{and} \quad g = u + pG \quad \text{with} \quad u \in \mathcal{T}_{251}(83, 82), \quad F, G \in \mathcal{T}_{251}(65, 65).$$

We also (as above) let $U \equiv u^{-1} \pmod{p}$.

The message digest m , which is formed as a hash of the digital document D being signed, is a ternary polynomial. In other words, we use an agreed upon hash function

$$\text{Hash} : \{\text{documents}\} \longrightarrow \{\text{ternary polynomials}\},$$

and we set $m = \text{Hash}(D)$.

Remark. In practice, one might take u , F , G and/or m to be products, as described in [4]. For the parameters $(N, p, q) = (251, 3, 128)$ as above, we take

$$u = u_1 \cdot u_2, \quad F = F_1 \cdot F_2, \quad G = G_1 \cdot G_2, \quad m = m_1 \cdot m_2 \cdot m_3$$

with

$$\begin{aligned} u_1 &\in \mathcal{T}_{251}(7, 6), & u_2 &\in \mathcal{T}_{251}(7, 8), \\ F_1, G_1 &\in \mathcal{T}_{251}(6, 6), & F_2, G_2 &\in \mathcal{T}_{251}(6, 6), \\ m_1, m_2 &\in \mathcal{T}_{251}(4, 4), & m_3 &\in \mathcal{T}_{251}(5, 5). \end{aligned}$$

The masking polynomial is formed using

$$w_1 \in \mathcal{T}(d_1, d_2) \quad \text{and} \quad w_2 \in \mathcal{T}(58, 58).$$

The values of d_1 and d_2 will vary, since w_1 is chosen to hide common deviations in s and t , but for the indicated parameters one will tend to have $12 \leq d_1 + d_2 \leq 20$. Also, one may choose some of the coefficients of w_2 to moment balance the coefficients of $U \cdot m \pmod{3}$. (There are no practical attacks known even without moment balancing.)

The signature s on the digital document D is then computed as

$$\begin{aligned} w_0(X) &= (U(X)m(X) \pmod{p}) + (U(X)w_1(X) \pmod{p}), \\ w &= w_0 + pw_2, \\ s &\equiv f \cdot w \pmod{q}. \end{aligned}$$

In order to verify that s is a valid signature on D for the public key h , Alice performs various tests. We will give explicit versions of three such tests from among the many that are possible. Alice first regenerates the message digest $m = \text{Hash}(D)$ and then computes the following polynomials:

$$\begin{aligned} t &\equiv h \cdot s \pmod{q} \\ s' &= p^{-1}(s - m) \pmod{q} \\ t' &= p^{-1}(t - m) \pmod{q} \end{aligned}$$

She next computes the centered norms $\|s'\|$, $\|t'\|$, $\|(s', t')\|$, the deviation numbers Dev_1 and Dev_2 , and the coefficient numbers $\text{Coef}_{s,i}$ and $\text{Coef}_{t,i}$ as described earlier. Finally she checks the following conditions and she rejects the signature as invalid if it fails any of them.

- (A) $\text{Dev}_1 \leq 10$ and $\text{Dev}_2 \leq 18$
- (B) $\|(s', t')\| \leq 485$
- (C) $\|s'\| \leq 360$ and $\|t'\| \leq 360$
- (D) $95 \leq \text{Coef}_{s,1}, \text{Coef}_{t,1} \leq 153$
 $50 \leq \text{Coef}_{s,2}, \text{Coef}_{t,2} \leq 100$
 $7 \leq \text{Coef}_{s,3}, \text{Coef}_{t,3} \leq 42$
 $\text{Coef}_{s,4}, \text{Coef}_{t,4} \leq 14$

Remark. Based on the Gaussian heuristic, the norm bound $\|(s', t')\| \leq 485$ given in (B) corresponds to (s', t') solving a closest vector problem in L_h^{NT} up to a factor of 7.91. See [3] for details.

Remark. The signature verification method described in [2] lacked the norm test that had appeared in the original version. This led Jonsson [6], Stern [7] (and others) to observe that if half of the coefficients of s and t are preselected and the other half are determined from the required congruence $t \equiv h \cdot s \pmod{q}$, then the resulting s and t have a significant chance of passing a crude congruence test based only on counting the total number of deviations. One can estimate the probability that an s and t constructed in this way will pass the tests described in this section. Of course, tests (B), (C) and (D) are not fully independent, but tests (A) and (B) are essentially independent. One finds that the probability that s passes even these two tests is approximately 2^{-80} . See [3] for further details.

Section 6. Conclusions

In this note we have presented an enhanced signature encoding method for NSS and described various verification methods. These methods are meant to indicate by example some of the many ways in which a document can be encoded into a signature $s(X)$ and ways in which $s(X) \equiv f(X) \cdot w(X) \pmod{q}$ and its corresponding $t(X) \equiv h(X) \cdot w(X) \pmod{q}$ can be distinguished from a forgery due to the structure of s and t as products of two small polynomials.

We conclude by reiterating that the idea underlying NSS is Bob's knowledge of the secret decomposition $h(X) = f(X)^{-1} \cdot g(X) \pmod{q}$. Bob demonstrates his knowledge by publishing a product of small polynomials

$$s(X) \equiv f(X) \cdot w(X) \pmod{q}.$$

This allows Alice to compute the quantity

$$t(X) \equiv h(X) \cdot s(X) \equiv g(X) \cdot w(X) \pmod{q},$$

which is also a product of two small polynomials. Thus Bob's secret knowledge allows him to give to Alice two small products that are related by $h(X)$, but without having to tell Alice his secret decomposition of $h(X)$. We have also seen that a norm verification test proves to Alice that Bob is heuristically able to solve a certain closest vector problem, up to a constant factor, in a lattice associated to his public key. We have also seen that with proper choices of encoding schemes even an extremely long transcript of signatures reveals no useful information. These are the fundamental ideas of NSS, and the remaining details involve choosing one of the many ways of encoding the digital document into $w(X)$ and verifying that the signature has the right form (i.e., is bound to the digital document and consists of products of small polynomials).

References

- [1] J. Hoffstein, J. Pipher, J.H. Silverman, NTRU: A new high speed public key cryptosystem, 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
- [2] J. Hoffstein, J. Pipher, J.H. Silverman, NSS: An NTRU Lattice-Based Signature Scheme, Eurocrypt 2001, Lecture Notes in Computer Science, Springer-Verlag, to appear.
- [3] J. Hoffstein, J. Pipher, J.H. Silverman, The NTRU Signature Scheme: Theory and Practice, preprint.
- [4] J. Hoffstein, J.H. Silverman, Small Hamming Weight Products in Cryptography, preprint, September 2000.
- [5] M. Szydło, A frequency analysis attack on the NTRU signature scheme, in preparation, May 2001.
- [6] J. Jonsson, A forgery method for the NTRU signature scheme, in preparation, May 2001.
- [7] J. Stern, Cryptanalysis of the NTRU signature scheme, preprint, May 2001.

Comments and questions concerning this technical report should be addressed to

`info@ntru.com`

Additional information concerning NTRU Cryptosystems and the NTRU Public Key Cryptosystem are available at

`www.ntru.com`

NTRU is a trademark of NTRU Cryptosystems, Inc.

The NTRU Public Key Cryptosystem is patented, and the NTRU Signature Scheme is patent pending. The contents of this technical report are copyright May 30, 2001 by NTRU Cryptosystems, Inc.