

The NTRU Signature Scheme: Theory and Practice

Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman

NTRU Cryptosystems, Inc., 5 Burlington Woods, Burlington, MA 01803 USA,
jhoffstein@ntru.com, jpipher@ntru.com, jsilverman@ntru.com

Abstract. The NTRU Signature Scheme (NSS) with enhanced document encoding and signature verification is described. Three areas of security are investigated:

- (1) It is proven (under a heuristic assumption) that direct forgery is equivalent to the solution of a closest vector problem, up to constant factor, in an NTRU convolution modular lattice.
- (2) The probability of forgery using partially preselected vectors is calculated, both theoretically and experimentally, for a recommended set of parameters.
- (3) The potential leakage of information from frequency analysis of signature transcripts is analyzed and is shown to be negligible.

Keywords: digital signature, public key authentication, NTRU, NSS, lattice-based cryptography, closest vector problem

Introduction

Secure public key authentication and digital signatures are increasingly important for electronic communications and commerce, and they are required not only on high powered desktop computers, but also on smart cards and wireless devices with severely constrained memory and processing capabilities. The importance of public key authentication and digital signatures is amply demonstrated by the large literature devoted to both theoretical and practical aspects of the problem, see for example [6, 12, 17, 18, 26, 31, 34, 39, 40, 42].

At CRYPTO '96 the authors introduced a highly efficient new public key cryptosystem called NTRU. (See [14] for details.) Underlying NTRU is a hard mathematical problem of finding short vectors in certain lattices. A complementary fast authentication and digital signature scheme called NSS, based on the same underlying hard problem and using keys of the same form, was presented at Eurocrypt 2001 [15]. NSS stands for NTRU Signature Scheme.

The core idea of NSS is as follows. The Signer's private key is a short generating vector for a certain lattice and his public key is a much longer generating vector for the same lattice. The signature on a digital document is a vector in the lattice with two important properties:

- The digital document is encoded into the signature vector.
- The signature vector demonstrates knowledge of a short vector in the lattice.

The NSS encoding method described in the Eurocrypt paper embedded the digital document directly into the signature. Szydło [43] noted that this allowed secret key information to be extracted from a long transcript of signatures by analysis of certain frequency distributions. One purpose of this note is to describe an enhanced encoding method and to prove that it reduces frequency variations to undetectable levels (on the order of 10^{-63}), thereby making Szydło's method or any other similar frequency analysis infeasible.

The Eurocrypt paper also described one way in which the Verifier could check that the signature is tied to the document and shows knowledge of a short lattice vector. Gentry, Jonsson [19], and Stern [41] noted that this single verification test, based on a coarse mod p comparison, is not sufficient to achieve the desired linkage between the signature, the document, and a short vector in the lattice. The second purpose of this note is to describe enhanced verification tests, based on both Euclidean norm and mod p comparisons, and to prove that creation of a valid signature for these tests is (under a heuristic assumption) equivalent to solving a closest vector problem up to a constant factor.

We begin in Section 1 with an overview of the convolution modular lattices and the hard lattice problems that are used by NTRU and NSS. In Section 2 we prove that creation of a valid NSS signature directly from the public key is equivalent (under a standard heuristic assumption) to solving a closest vector problem up to a constant factor. Next we take up the question of forgery in Section 3. We calculate the probability that a forgery constructed by the Gentry-Jonsson-Stern method [19, 41] passes refined norm and mod p tests and show that this probability can be made very small. Finally, in Section 4 we describe Szydło's distribution analysis method [43] and show that with enhanced document encoding, there is virtually no information leakage in even extremely long transcripts of signatures.

1 NTRU Lattices and Associated Vector Problems

The NTRU Public Key Cryptosystem and the NTRU Signature Scheme (NSS) use two public parameters N and q . Typical choices are $(N, q) = (251, 128)$ and $(N, q) = (503, 256)$. Basic operations take place in the ring of convolution polynomials

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

A polynomial $a(X) = a_0 + a_1X + \cdots + a_{N-1}X^{N-1} \in R$ is identified with its vector of coordinates $(a_0, a_1, \dots, a_{N-1}) \in \mathbb{Z}^N$. Note that the product of two polynomials in R is simply the convolution product of their corresponding vectors

The *centered norm* of a polynomial $a(X) \in R$ is defined to be

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

For randomly chosen polynomials a and b , the norm is quasi-multiplicative,

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

The *Convolution Modular Lattice* L_h associated to the polynomial

$$h(X) = h_0 + h_1X + h_2X^2 + \dots + h_{N-1}X^{N-1} \in R$$

is the set of vectors $(u, v) \in R \times R \cong \mathbb{Z}^{2N}$ satisfying

$$v(X) = h(X) * u(X) \pmod{q}.$$

It is easy to see that L_h is generated by the rows of the following matrix.

$$L_h = \text{RowSpan} \left(\begin{array}{cccc|cccc} 1 & 0 & \dots & 0 & h_0 & h_1 & \dots & h_{N-1} \\ 0 & 1 & \dots & 0 & h_{N-1} & h_0 & \dots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & h_1 & h_2 & \dots & h_0 \\ \hline 0 & 0 & \dots & 0 & q & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & q & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & q \end{array} \right)$$

We note that a convolution modular lattice has a rotational invariance property, since if $(u, v) \in L_h$, then

$$(X^i * u, X^i * v) \in L_h \quad \text{for all } 0 \leq i < N.$$

Notice that each of the rotations $(X^i * u, X^i * v)$ has the same (centered) norm as (u, v) . We write $R * (u, v)$ for the sublattice of L_h generated by (u, v) and all of its rotations.

If the polynomial h has a decomposition in the form

$$h \equiv f^{-1} * g \pmod{q}$$

with polynomials f and g having small coefficients (see below for the precise definition of small), then we say that L_h is an *NTRU Lattice* and denote it by L_h^{NT} .

Our goal is to directly relate the NTRU Public Key Cryptosystem and the NTRU Signature Scheme to lattice problems in L_h^{NT} . To do this, we make the following definitions and assumptions:

1. The security parameters N and q are related by

$$q = O(N).$$

In practice, we generally take $\frac{1}{3}N \leq q \leq \frac{2}{3}N$.

2. A *small polynomial* is a polynomial whose coefficients are $O(1)$, that is, a polynomial whose coefficients are bounded independently of N . In practice, small polynomials might have the form $a(X) + p * b(X)$, where $a(X)$ and $b(X)$ are binary or trinary polynomials and $p = 3$. The (centered) norm of a small polynomial $a(X)$ satisfies

$$\|a\| = O(\sqrt{N}).$$

Remark 1. The *Gaussian heuristic* provides a method for predicting properties of a “random” lattice. We make a number of remarks concerning this heuristic which will be useful in our subsequent work.

1. The Gaussian heuristic predicts that the shortest vector in a “random” lattice L (of large dimension) has size approximately

$$\lambda_{\text{Gauss}}(L) = \sqrt{\dim(L)/2\pi e} \cdot \text{Disc}(L)^{1/\dim(L)}.$$

Similarly, most closest vector problems for L have a solution whose size is approximately $\lambda_{\text{Gauss}}(L)$.

2. A general convolution modular lattice L_h has dimension $2N$ and discriminant q^N , so its probable shortest vector and closest vectors have size approximately

$$\lambda_{\text{Gauss}}(L_h) = \sqrt{Nq/\pi e} = O(N).$$

Notice that L_h contains N linearly independent vectors of length $q = O(N)$, namely the bottom N rows of its matrix. Small linear combinations of these “ q -vectors” are the only obvious vectors of length $O(N)$ in L_h .

3. If $(u, v) \in L_h$, then the vector obtained by reducing the coordinates of u and v modulo q is in L_h . Thus L_h contains many vectors of length $O(q\sqrt{N}) = O(N^{3/2})$, and it is clearly also trivial to find vectors in L_h whose distance to a given vector is at most $O(N^{3/2})$.
4. In an NTRU lattice L_h^{NT} , the polynomial h has the form $h \equiv f^{-1} * g \pmod{q}$ for small polynomials f and g , so L_h^{NT} contains the short vector (f, g) . More generally, all of the rotations $(X^i * f, X^i * g)$ are short vectors in L_h^{NT} having length $O(\sqrt{N})$. Based on the Gaussian heuristic, these secret short vectors are probably $O(\sqrt{N})$ smaller than any vector not in the subspace $R * (f, g)$ that they span.

Definition 1. Let L_h^{NT} be an NTRU lattice. The NTRU Lattice Key Problem is to find a vector of length $O(\sqrt{N})$ in L_h^{NT} .

Remark 2. The NTRU lattice L_h^{NT} contains the subspace $R * (f, g)$ generated by vectors of length $O(\sqrt{N})$, so the NTRU Lattice Key Problem always has solutions. The Gaussian heuristic predicts that the smallest vector in L_h^{NT} that is linearly independent to the subspace $R * (f, g)$ has length $O(N)$. Hence it is highly probable that all solutions of the NTRU Lattice Key Problem are given by small multiples $(u * f, u * g)$, i.e., with $u \in R$ a polynomial of size $\|u\| = O(1)$.

2 NSS Verification and a Close Vector Problem

In this section we give a heuristic argument to show that a valid NSS signature gives a solution to an approximate closest vector problem in an NTRU lattice, where the target vector depends on the message being signed. In this way we prove that signing is at least as hard as solving the indicated approximate closest vector problem.

Suppose that s is a valid NSS signature for the message m and public key h . Let $t \equiv h * s \pmod{q}$ and set

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

We substitute the formulas for s' and t' into the congruence $t \equiv h * s$ and do some algebra to obtain

$$t' \equiv h * s' - \underbrace{p^{-1} * (m - h * m)}_{A_m} \pmod{q}.$$

Note that the polynomial $A_m \equiv p^{-1} * (h * m - m) \pmod{q}$ is a known quantity that depends on the message polynomial m and the public key h .

The NTRU lattice associated to h is the set of vectors

$$L_h^{\text{NT}} = \{(u, h * u) : u \in \mathbb{Z}^N\}.$$

In particular, L_h^{NT} contains the vector

$$(s', h * s') = (s', t' + A_m) = (s', t') + (0, A_m) \in L_h^{\text{NT}}.$$

The L^2 norm test for a valid signature includes the condition that the vector (s', t') have norm no greater than `L2NormBound`. We now consider the question of how small the bound `L2NormBound` can be set while still allowing valid signatures to be efficiently generated. A typical format for s is

$$s = f * w = (u + p * F) * (w_0 + p * W),$$

where we now explain the various quantities involved:

- The polynomials u , F , and W are small polynomials.
- p is a small integer that is independent of N and q , other than the restriction that $\gcd(p, q) = 1$. For example, one typically takes $p = 3$ and q to be a power of 2. Thus $p = O(1)$.
- The polynomial w_0 has the form

$$w_0 = (u^{-1} * m \pmod{p}) + (u^{-1} * w_1 \pmod{p}),$$

where u^{-1} denotes the inverse of u modulo p . Note that w_0 is a small polynomial, since its coefficients are bounded by p and $p = O(1)$.

- The polynomial w_1 has a very small number of nonzero coefficients. More precisely, $\|w_1\| = O(1)$. Thus w_1 is even smaller than a “small” polynomial.

For our next calculation, it is helpful to write the congruence defining w_0 as an equality. Thus we define a polynomial $b(X) \in R$ by the equation

$$u * w_0 = m + w_1 + p * b.$$

Using this formula and the formula for s , we compute

$$\begin{aligned}
s' &\equiv p^{-1} * (s - m) \pmod{q} \\
&\equiv p^{-1} * (u * w_0 + p * u * W + p * F * w - m) \pmod{q} \\
&\equiv p^{-1} * (m + w_1 + p * b + p * u * W + p * F * w - m) \pmod{q} \\
&\equiv p^{-1} * w_1 + b + u * W + F * w \pmod{q}.
\end{aligned}$$

We will bound the norm of s by using the following estimates:

$$\begin{aligned}
p^{-1} \bmod q &= O(q) = O(N) \\
\|b\| &= \|u * w_0 + m + w_1\| \ll \|u\| \cdot \|w_0\| + \|m\| + \|w_1\| = O(N) \\
\|u * W\| &\ll \|u\| \cdot \|W\| = O(N) \\
\|F * w\| &= \|F * (w_0 + p * W)\| \ll \|F\| \cdot \|w_0\| + p \cdot \|F\| \cdot \|W\| = O(N)
\end{aligned}$$

Hence $\|s'\| = O(N)$, and a similar calculation gives the same bound for $\|t'\|$. This proves that by using the private key (f, g) , it is possible to find a valid signature s on the document m that satisfies the norm bound

$$\|(s', t')\| \leq O(N),$$

so we may take $\text{L2NormBound} = O(N)$.

Thus a signature created using the private key gives a vector in L_h^{NT} whose distance to the target vector A_m is $O(N)$. This may be compared to the Gaussian heuristic, which predicts that most closest vector problems have a solution whose length is $O(\sqrt{qN}) = O(N)$. In other words, knowledge of the private key allows a person to find a vector in L_h^{NT} whose distance to the target vector A_m is a constant multiple of the probable shortest distance (as predicted by the Gaussian heuristic).

We restate this more formally.

Theorem 1. *Let s be a valid NSS signature on a message m for the public key h . Let*

$$s' \equiv p^{-1} * (s - m) \pmod{q} \quad \text{and} \quad A_m = p^{-1} * (m - h * m) \pmod{q}.$$

*Then the distance from the lattice vector $(s', h * s' \bmod q) \in L_h^{\text{NT}}$ to the target vector $(0, A_m)$ is $O(N)$. Using the Gaussian heuristic to estimate the likely distance from $(0, A_m)$ to the closest vector in L_h^{NT} yields the estimate*

$$\frac{\text{distance from } (0, A_m) \text{ to } (s', h * s' \bmod q)}{\text{distance from } (0, A_m) \text{ to closest vector in } L_h^{\text{NT}}} = O(1).$$

In other words, based on the Gaussian heuristic, a valid signature on m solves the closest vector (to A_m) problem in L_h^{NT} to within a constant factor.

Remark 3. It is known that finding the shortest vector in a general lattice is NP-complete for the L^∞ norm [5] and that it is NP-hard for the L^2 norm under

randomized reductions [1], even if one only wants to approximate the shortest vector up to some factor less than $\sqrt{2}$ [28]. It is not unreasonable to expect that this remains true when $\sqrt{2}$ is replaced by a larger constant. The best known algorithms [3] have (theoretical) running time $2^{O(n)}$ to find the shortest vector in a lattice of dimension n and always find vectors at most $2^{n \log \log n / \log n}$ times longer than the shortest vector in polynomial time.

The closest vector problem seems to be even harder [8, 11] than the shortest vector problem, although in practice they are of about the same order of difficulty. (See also [4, 9, 20] for work on related problems.) Of course, in practical applications, the big- O constants are important. However, Theorem 1 shows that if N is chosen sufficiently large, then forging an NSS signature is (heuristically) at least as hard as a probable hard lattice problem.

Remark 4. There is also the issue of whether it is as difficult to find short or close vectors in convolution modular lattices as it is in more general classes of lattices. This is certainly an interesting question. The principal method for approximating shortest or closest vectors in a general lattice is the LLL algorithm [24] and its generalizations and enhancements such as [21–23, 32, 33, 35]. The evidence to date is that there are no special algorithms for convolution modular lattices that work significantly better than standard LLL-type lattice reduction algorithms, and lattice reduction algorithms do not appear to work better on convolution modular lattices than they do on general lattices.

Remark 5. We might ask how close to A_m can we easily find a vector in L_h^{NT} . As noted above, in any convolution modular lattice it is easy to find vectors whose distance to any given vector is $O(N^{3/2})$. We can do a bit better by selecting N of the coordinates of (u, v) to be anything that we want and then using the relation $v \equiv h * u \pmod{q}$ to solve for the other N coordinates, taking them in the range $(-q/2, q/2]$. However, the coordinates that we do not preselect will be more-or-less randomly distributed modulo q , so on average they will be $O(q)$. Thus this procedure again yields a vector whose norm is $O(q\sqrt{N}) = O(N^{3/2})$, although the big- O constant will be somewhat improved. The conclusion is that the obvious elementary methods of creating vectors in a convolution modular lattice give solutions of length $O(N^{3/2})$.

Example 1. Let $(N, p, q) = (251, 3, 128)$ and $\text{L2NormBound} = 485$. Then we find that a valid signature probably satisfies

$$\frac{\text{distance from } (0, A_m) \text{ to } (s', h * s')}{\text{distance from } (0, A_m) \text{ to closest vector in } L^{\text{NT}}} \leq \frac{485}{\sqrt{251 \cdot 128 / \pi e}} \approx 7.91.$$

Extensive experimentation with NTRU lattices has shown that solving a close vector problem of this type is very difficult. And even reducing the bound to

$$\text{L2NormBound} = 420$$

still allows creation of signatures while reducing the multiplier from 7.91 to 6.85.

Remark 6. We have shown that signing is at least as hard as solving the indicated approximate closest vector problem. Conversely, suppose that we have constructed a vector $(u, v) \in L_h^{\text{NT}}$ satisfying

$$\|(u, v) - (0, A_m)\| \leq O(N).$$

We set

$$s = m + p * u$$

and compute

$$t \equiv h * s \equiv h * (m + p * u) \equiv h * m + p * h * u \equiv (m - p * A_m) + p * v \pmod{q}.$$

Then

$$s' = u \quad \text{and} \quad t' = v - A_m,$$

and hence

$$\|(s', t')\| = \|(u, v - A_m)\| \leq O(N).$$

Thus we have constructed an s that appears to pass the L^2 norm test.

However, there is one point that we have neglected. It turns out not to matter asymptotically, but can make a significant difference in practical situations. The issue is that s is not exactly equal to $m + p * u$ and t is not exactly equal to $m + p * (v - A_m)$; they are only congruent to these quantities modulo q . Thus if any coefficients of u or $v - A_m$ are larger than $q/2p$ in magnitude, then nontrivial reduction modulo q probably occurs. This does not matter if there are only a few such coefficients, but can have an effect if there are very many. So we need to estimate the size of the coefficients of u and $v - A_m$. We know that $\|u\|$ and $\|v - A_m\|$ are $O(N)$, so most of their coefficients are probably $O(\sqrt{N})$. This is asymptotically smaller than $q/2p = O(N)$, so not too much wrapping occurs if N is large. Hence based on the Gaussian heuristic, we see that finding a signature s that passes the L^2 norm test

$$\|(s', t')\| \leq \text{L2NormBound} = O(N)$$

is equivalent to solving a closest vector problem to within a constant factor.

3 Probability Estimates for NSS Forgery

One method for attempting to forge an NSS signature, noted independently by Craig Gentry, Jakob Jonsson [19] and Jacques Stern [41], is to preselect half of the coefficients of s and t to have any desired values and then to solve

$$t \equiv h * s \pmod{q}$$

for the remaining coefficients. In this section we estimate the probability that this procedure yields a valid signature. For ease of exposition, we will only subject the signatures to the following two tests. In practice there will be other tests performed that make it even less likely that this forgery method succeeds.

L2 NORM TEST Form $s' \equiv p^{-1}(s - m) \pmod q$ and $t' = p^{-1}(t - m) \pmod q$ and verify that

$$\|(s', t')\| \leq \text{L2NormBound}. \quad (1)$$

MOD 3 DISTRIBUTION TEST Compute the following quantities:

$$\begin{aligned} \text{Dev}_1 &= \overline{\#\{j : -q/2 < s_j \leq -q/4 \text{ and } s_j \equiv m_j - 1 \pmod 3\}} \\ &\quad + \#\{j : q/4 < s_j \leq q/2 \text{ and } s_j \equiv m_j + 1 \pmod 3\} \\ &\quad + \#\{j : -q/2 < t_j \leq -q/4 \text{ and } t_j \equiv m_j - 1 \pmod 3\} \\ &\quad + \#\{j : q/4 < t_j \leq q/2 \text{ and } t_j \equiv m_j + 1 \pmod 3\} \\ \text{Dev}_2 &= \overline{\#\{j : -q/4 < s_j \leq 0 \text{ and } s_j \equiv m_j - 1 \pmod 3\}} \\ &\quad + \#\{j : 0 < s_j \leq q/4 \text{ and } s_j \equiv m_j + 1 \pmod 3\} \\ &\quad + \#\{j : -q/4 < t_j \leq 0 \text{ and } t_j \equiv m_j - 1 \pmod 3\} \\ &\quad + \#\{j : 0 < t_j \leq q/4 \text{ and } t_j \equiv m_j + 1 \pmod 3\} \end{aligned}$$

Verify that

$$\text{Dev}_1 \leq \text{DevBound}_1 \quad \text{and} \quad \text{Dev}_2 \leq \text{DevBound}_2. \quad (2)$$

Remark 7. In addition to the norm test 1 on the pair (s', t') , we can also impose conditions on $\|s'\|$ and $\|t'\|$ individually. Of course, the triangle inequality

$$\|(s', t')\| \leq \|s'\| + \|t'\|$$

shows that these conditions are not independent, but it may be worthwhile imposing conditions somewhat weaker than

$$\|s'\|, \|t'\| \leq \text{L2NormBound}/\sqrt{2}$$

in order to prevent an attacker from heavily weighting a potential forgery onto either s' or t' .

Remark 8. Rather than using a separate bounds on Dev_1 and Dev_2 , it may be advantageous to use a cumulative bound. Thus the bounds (2) would be replaced by

$$\text{Dev}_1 \leq \text{DevBound}_1 \quad \text{and} \quad \text{Dev}_1 + \text{Dev}_2 \leq \text{DevBound}_1 + \text{DevBound}_2. \quad (3)$$

It turns out that this can make it considerably easier to create valid signatures using the private key, while making only a marginal difference in the difficulty of forging. (See Section 3.2 for specific numbers.) The reason is that it makes it easier to sign using the private key is that much of the time, the signature is so good that its Dev_1 value is much smaller than the required bound DevBound_1 . When this happens, it is sometimes the case that those “missing” deviations in Dev_1 have been shifted to Dev_2 . So if Dev_1 is very small, it is reasonable to allow Dev_2 to be somewhat larger, hence the use of a cumulative bound.

The only coefficients of s and t that have a negative impact on the verification tests are coefficients that do not agree modulo p with the corresponding coefficients of m . Thus it is in the forger's best interest to make sure that the N coefficients that he chooses in s and t agree modulo p with the corresponding coefficients of m . Once he does this, the remaining N coefficients of s and t will be unpredictably and uniformly randomly distributed in the set $(-q/2, q/2]$. This leads to the following two questions.

Let V_q^N be the set of vectors of dimension N with integer coordinates in the range $(-q/2, q/2]$. Let I_1, I_2, I_3, I_4 denote the four quartiles

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

and fix integers $0 \leq \epsilon_{i,j} < p$ for $1 \leq i \leq 4$ and $0 \leq j < N$.

L^2 Norm Test What is the probability that a randomly chosen vector $\mathbf{v} \in V_q^N$ satisfies

$$\|\mathbf{v}\| \leq A?$$

Quartile Congruence Test What is the probability that a randomly chosen vector $\mathbf{v} \in V_q^N$ simultaneously satisfies the following two inequalities?

$$\begin{aligned} & \#\{j : v_j \in I_1 \text{ and } v_j \equiv \epsilon_{1,j} \pmod{p}\} \\ & \quad + \#\{j : v_j \in I_4 \text{ and } v_j \equiv \epsilon_{4,j} \pmod{p}\} \leq B \\ & \#\{j : v_j \in I_2 \text{ and } v_j \equiv \epsilon_{2,j} \pmod{p}\} \\ & \quad + \#\{j : v_j \in I_3 \text{ and } v_j \equiv \epsilon_{3,j} \pmod{p}\} \leq C \end{aligned}$$

3.1 The L^2 Norm Test

The coordinates v_i are chosen randomly and independently to be integers in the interval $(-q/2, q/2]$. Since q is reasonably large and we are considering centered norms, there is not a large error if we assume that the v_i are in fact chosen uniformly and independently as real numbers in the interval

$$\left(-\frac{q-1}{2}, \frac{q-1}{2}\right).$$

To ease notation, we set $Q = (q-1)/2$.

The centered norm of a vector $\mathbf{v} \in V_q^N$ is given by

$$\|\mathbf{v}\|^2 = \sum_{i=1}^N v_i^2 - \frac{1}{N} \left(\sum_{i=1}^N v_i \right)^2.$$

The second term is generally very small compared to the first term, since most vectors have both positive and negative coordinates that cancel out. So to simplify our analysis, we ignore the second term and simply look at the usual L^2 norm $\|\mathbf{v}\|^2 = \sum v_i^2$.

The coordinates v_1, v_2, \dots, v_N are independent random variables with the same distribution functions, so the same is true of their squares $v_1^2, v_2^2, \dots, v_N^2$. It follows from the central limit theorem that their sum $\|\mathbf{v}\|^2$ can be approximated by a normal distribution. Thus if we let v be a random variable that is uniformly distributed in the interval $(-Q, Q)$ and if we let μ and σ denote the mean and standard deviation of the random variable v^2 , then $\|\mathbf{v}\|^2$ is approximately normally distributed with mean $N\mu$ and standard deviation $\sqrt{N}\sigma$.

It is easy to compute μ and σ .

$$\begin{aligned} \mu &= \mu(v^2) = \frac{1}{2Q} \int_{-Q}^Q v^2 dv = \frac{Q^2}{3} \\ \sigma^2 &= \mu(v^4) - \mu(v^2)^2 = \frac{Q^4}{5} - \left(\frac{Q^2}{3}\right)^2 = \frac{4Q^4}{45} \end{aligned}$$

Hence $\|\mathbf{v}\|^2$ is approximately normally distributed with mean and standard deviation

$$\frac{NQ^2}{3} \quad \text{and} \quad \frac{2Q^2}{3} \sqrt{\frac{N}{5}}.$$

For example, if we take

$$N = 251, \quad q = 128, \quad Q = \frac{q-1}{2} = \frac{127}{2},$$

then the theoretical mean and standard deviation are 337365 and 19046, respectively. These compare well with the experimental values of 342600 and 19400 obtained from a large random sample of vectors in V_{128}^{251} .

We can now approximate the probability that $\|\mathbf{v}\|$ is smaller than a given bound by using the error function erf that describes a normal distribution of mean 0 and standard deviation 1,

$$\begin{aligned} \text{Prob}(\|\mathbf{v}\| \leq A) &= \text{Prob}(\|\mathbf{v}\|^2 \leq A^2) \approx \text{erf}\left(\frac{A^2 - N\mu}{\sqrt{N}\sigma}\right) \\ &= \text{erf}\left(\frac{A^2 - NQ^2/3}{(2Q^2/3)\sqrt{N/5}}\right). \end{aligned}$$

Table 1 uses this approximation with

$$N = 251, \quad q = 128, \quad Q = \frac{q-1}{2} = \frac{127}{2},$$

to estimate the probability that a forgery constructed as above passes the L^2 norm test for various values of `L2NormBound`.

Remark 9. Experimental evidence shows that the distribution of $\|v\|^2$ is close to a normal distribution, but it is not clear if the tail probabilities follow exactly a normal distribution. If they do not, then this would have only a small effect on the above probability estimates, but it is an interesting question that deserves further study.

L2NormBound	Prob(Successful Forgery)
485	$2^{-24.54}$
450	$2^{-37.91}$
420	$2^{-53.24}$

Table 1. Probability of Passing the L^2 Norm Test

3.2 The Quartile Congruence Test

We make the simplifying assumption that each quartile contains an equal number of intervals of length p . This is not exactly correct, but is a reasonable approximation if q is considerably larger than p ; for example, if $q = 128$ and $p = 3$.

We may view the choice of a vector $\mathbf{v} \in V_q^N$ as randomly placing N numbers into $4p$ boxes, where the $(i, \epsilon)^{\text{th}}$ box is described by the condition

$$\text{Box}_{i,\epsilon} = \{v \in I_i : v \equiv \epsilon \pmod{p}\}, \quad 1 \leq i \leq 4, 0 \leq \epsilon < p.$$

Each number has an equal probability of landing in any particular box. We have picked out four of these boxes for special consideration and are asking for the probability that the first two special boxes contain no more than B numbers and that the second two special boxes contain no more than C numbers.

We may as well consider the first two special boxes to form a single special container, and similarly for the second two special boxes. So finally we are reduced to the following problem, where we let $m = 2p$.

Choose N objects and randomly place them into m containers X_1, \dots, X_m , where the probability of being placed into any particular container is $1/m$. What is the probability

$$\text{Prob}(|X_1| \leq B \text{ and } |X_2| \leq C),$$

where $|X_i|$ denotes the number of objects in X_i ?

This is an elementary exercise in combinatorics. We first compute

$$\text{Prob}(|X_1| = b \text{ and } |X_2| = c).$$

If no restrictions are imposed, then the number of ways to place the N objects, one by one, into the m containers is equal to m^N . Now suppose that we require that $|X_1| = b$ and $|X_2| = c$. This means that we need to choose b of the objects to go into X_1 , which can be done in $\binom{N}{b}$ ways, then we need to choose c of the remaining $N - b$ objects to go into X_2 , which can be done in $\binom{N-b}{c}$ ways. Finally, we need to distribute the remaining $N - b - c$ objects into the remaining $m - 2$

boxes, which can be done in $(m - 2)^{N-b-c}$ ways. Hence the probability that the first container gets b objects and the second container gets c objects is

$$\begin{aligned} \text{Prob}(|X_1| = b \text{ and } |X_2| = c) &= \frac{\binom{N}{b} \binom{N-b}{c} (m-2)^{N-b-c}}{m^N} \\ &= \left(1 - \frac{2}{m}\right)^N \frac{N!}{b! \cdot c! \cdot (N-b-c)!} (m-2)^{-b-c}. \end{aligned}$$

Now we can answer the original question by adding up the individual probabilities.

$$\begin{aligned} \text{Prob}(|X_1| \leq B \text{ and } |X_2| \leq C) &= \sum_{0 \leq b \leq B} \sum_{0 \leq c \leq C} \text{Prob}(|X_1| = b \text{ and } |X_2| = c) \\ &= \left(1 - \frac{2}{m}\right)^N \sum_{0 \leq b \leq B} \sum_{0 \leq c \leq C} \frac{N!}{b! \cdot c! \cdot (N-b-c)!} (m-2)^{-b-c}. \end{aligned}$$

Table 2 gives the probability of passing the quartile congruence test (2) for the global NSS parameters

$$N = 251, \quad q = 128, \quad \text{and} \quad p = 3,$$

(so $m = 2p = 6$) and for various values of `DevBound1` and `DevBound2`

		DevBound ₂				
		9	12	15	18	21
DevBound ₁	4	$2^{-92.38}$	$2^{-84.99}$	$2^{-78.69}$	$2^{-73.28}$	$2^{-68.63}$
	6	$2^{-85.45}$	$2^{-78.09}$	$2^{-71.83}$	$2^{-66.46}$	$2^{-61.84}$
	8	$2^{-79.44}$	$2^{-72.12}$	$2^{-65.89}$	$2^{-60.56}$	$2^{-55.97}$
	10	$2^{-74.13}$	$2^{-66.85}$	$2^{-60.66}$	$2^{-55.36}$	$2^{-50.81}$
	12	$2^{-69.40}$	$2^{-62.15}$	$2^{-56.00}$	$2^{-50.74}$	$2^{-46.23}$

Table 2. Probability of Passing the Quartile Congruence Test

Remark 10. If we instead use the cumulative quartile congruence test (3), then the probability formula derived above must be altered in the obvious way:

$$\begin{aligned} & \text{Prob}(|X_1| \leq B \text{ and } |X_1| + |X_2| \leq B + C) \\ &= \sum_{0 \leq b \leq B} \sum_{0 \leq c \leq B+C-b} \text{Prob}(|X_1| = b \text{ and } |X_2| = c) \\ &= \left(1 - \frac{2}{m}\right)^N \sum_{0 \leq b \leq B} \sum_{0 \leq c \leq B+C-b} \frac{N!}{b! \cdot c! \cdot (N - b - c)!} (m - 2)^{-b-c}. \end{aligned}$$

Table 3 uses this formula to compute the probability that a forgery passes the cumulative quartile congruence test. Comparison of the two tables shows that the cumulative test is almost as good at detecting forgeries as the noncumulative test, and as explained in Remark 8, use of the cumulative test can make it easier to generate valid signatures using the private key.

		DevBound ₂				
		9	12	15	18	21
DevBound ₁	4	$2^{-91.85}$	$2^{-84.61}$	$2^{-78.40}$	$2^{-73.06}$	$2^{-68.45}$
	6	$2^{-84.58}$	$2^{-77.48}$	$2^{-71.37}$	$2^{-66.10}$	$2^{-61.56}$
	8	$2^{-78.18}$	$2^{-71.23}$	$2^{-65.23}$	$2^{-60.05}$	$2^{-55.58}$
	10	$2^{-72.41}$	$2^{-65.65}$	$2^{-59.78}$	$2^{-54.69}$	$2^{-50.29}$
	12	$2^{-67.13}$	$2^{-60.58}$	$2^{-54.86}$	$2^{-49.87}$	$2^{-45.56}$

Table 3. Probability of Passing the Cumulative Quartile Congruence Test

3.3 The Probability of a Successful Forgery

As is clear from the tables, the probability of a random forgery attempt passing either the L^2 norm test or the quartile congruence test individually is small, but a determined search might be successful. However, an attempted forgery is required to simultaneously pass both the L^2 norm test and the quartile congruence test, and it is easy to see that these tests are essentially independent of one another, since the quantities

$$v^2 \quad \text{and} \quad v \pmod{p}$$

are essentially independent when v is randomly chosen as an integer in the interval $(-q/2, q/2]$. (As usual, we assume that q is considerably larger than p .)

Therefore the probability of an attempted forgery passing both tests is approximately the product of the individual probabilities. We thus obtain the probabilities, given in Table 4, that a forgery attempt passes both tests for various choices of L^2 norm and quartile congruence bounds.

DevBound ₁	DevBound ₂	L2NormBound	Prob(Successful Forgery)
10	18	485	$2^{-79.90}$
8	15	450	$2^{-103.80}$
6	12	420	$2^{-131.34}$

Table 4. Probability of Simultaneously Passing the L^2 Norm and Quartile Congruence Tests

4 Coefficient Frequency Analysis of Signature Transcripts

As we have seen in Section 2, the ability to forge an NSS signature using only the public key is equivalent to solving a difficult lattice problem. A second line of attack on NSS, and indeed on any digital signature scheme, is to use a long transcript of valid signatures to either reconstruct the private key or to directly forge a signature on a new document. In this section we discuss what information is available from a transcript of signatures.

In order to illustrate how transcript information might be used, we begin with a description of Szydło’s coefficient frequency analysis method [43]. Szydło’s method is reasonably effective on moderately long transcripts (a few tens of thousands) of signatures generated using keys in which the private key is weakened by setting $u(X)$ equal to 1. We next describe a moment equalization method that greatly reduces the effectiveness of frequency analysis. Finally, we explain why the use of a nontrivial secret polynomial $u(X)$ reduces frequency differences to a level that is indistinguishable even on extremely long transcripts consisting of hundreds of millions of signatures. Thus as long as $u(X)$ is a nontrivial (secret) polynomial, then signatures generated using a private key of the form

$$f = u + p * F, \quad g = u + p * G,$$

are not susceptible to coefficient frequency analysis attacks.

Remark 11. Before beginning our discussion of transcript analysis, we observe that a valid signature s on a document m may be viewed as N signatures $X^i * s$ on the N documents $X^i * m$. Hence if we are interested, for example, in studying pairs (s, m) in which a particular coefficient m_j of m takes on a particular value, it suffices to study the pairs for which m_0 takes on that value, since each coefficient of m is the leading coefficient of some rotation $(X^i * s, X^i * m)$ of (s, m) .

Hence whenever we talk about a transcript \mathcal{S} consisting of some large number of signatures (s, m) , we always assume that all of the rotations $(X^i * s, X^i * m)$ are also in \mathcal{S} .

4.1 Szydło's Coefficient Frequency Analysis Method: $u = 1$

In this section we describe Szydło's method [43]. This method is effective if $u(X) = 1$,

$$f = 1 + p * F \quad \text{and} \quad g = 1 + p * G. \quad (4)$$

More generally, it is effective if $f = f_0 + p * F$ or $g = g_0 + p * G$ for any fixed *public* polynomials f_0 or g_0 .

Let \mathcal{S} be a long transcript of signature pairs (s, m) . (See Remark 11.) For each index $0 \leq k < N$ and each value $\epsilon \in \{-1, 0, 1\}$, consider the subtranscript consisting of messages with $m_0 = \epsilon$ and count how many times the k^{th} coefficients of s takes on each possible mod q value. Using this data, we can approximate the probability functions

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

In order to see why these probability distributions might be useful, we need to write out the expression for s_k . Remembering our assumption (4) that $u(X) = 1$, we find that

$$s_k = f_k(m_0 + w_{1,0} + pw_{2,0}) + \sum_{i=1}^{N-1} f_{k-i}(m_i + w_{1,i} + pw_{2,i}). \quad (5)$$

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 . (Similar remarks apply to t and g , of course.)

More formally, consider the $3N$ random variables

$$X_{k,\epsilon} : \{\text{signatures } (s, m) \text{ with } m_0 = \epsilon\} \longrightarrow [0, q - 1]$$

defined by $X_{k,\epsilon}(s, m) = s_k$. From the formula (5), we see that each of the $3N$ random variables $X_{k,\epsilon}$ is actually equal to one of nine possible random variables $Y_{\delta,\epsilon}$, where $\delta \in \{-1, 0, 1\}$ is equal to the value of f_k . Thus the goal of frequency analysis is to identify which of the nine possible random variables $Y_{\delta,\epsilon}$ is equal to the experimentally determined random variable $X_{k,\epsilon}$, since if the attacker can determine that $X_{k,\epsilon} = Y_{\delta,\epsilon}$, then he has determined that $f_k = \delta$.

There are a number of ways in which an attacker might try to identify $X_{k,\epsilon}$ with the correct $Y_{\delta,\epsilon}$. We briefly indicate two possibilities.

- Determine the nine probability functions $\text{Prob}(Y_{\delta,\epsilon} = b)$ very accurately, either theoretically using (5) or experimentally by generating a large number of signatures for a large number of keys. Then match the approximate probability functions $\text{Prob}(X_{k,\epsilon} = b)$ derived from the transcript \mathcal{S} against the exactly known functions $\text{Prob}(Y_{\delta,\epsilon} = b)$ and assign f_k to the value of δ that gives the best match.
- Fix a value of ϵ , say $\epsilon = 0$, and consider the N approximate probability functions $\text{Prob}(X_{k,0} = b)$ derived from the transcript \mathcal{S} . Divide this set of N functions into three subsets of functions that most closely resemble one another. Then the k values in each subset should correspond to coefficients of f that have the same value.

Remark 12. In order to apply the frequency analysis method, it is necessary to determine when two (probability) functions are similar or different. There are many methods for measuring the “distance” between two functions, but in practice for our situation, they all seem to yield similar results. An easy comparison method is to define the distance between two random variables X and Y (that take mod q values) to be

$$\text{Distance}(X, Y) = \left(\sum_{b=0}^{q-1} (\text{Prob}(X = b) - \text{Prob}(Y = b))^2 \right)^{1/2}. \quad (6)$$

For a given value of k and ϵ , we then select the value of $\delta \in \{-1, 0, 1\}$ that minimizes $\text{Distance}(X_{k,\epsilon}, Y_{\delta,\epsilon})$ and guess that $f_k = \delta$.

The principal obstacle faced by an attacker in attempting a coefficient frequency analysis is that he does not know the functions $\text{Prob}(X_{k,\epsilon} = b)$ exactly. He can only approximate them using the signatures in his transcript \mathcal{S} . Each signature (5) contains a considerable amount of random material in w_1 and w_2 . This random material introduces errors, although the randomness will in theory cancel out in a sufficiently long transcript. A rough estimate from probability theory says that a transcript of length T has random fluctuations on the order of $O(\sqrt{T})$. This means that if the nine probability functions $\text{Prob}(Y_{\delta,\epsilon} = b)$ differ by η , then a transcript of length $O(1/\eta^2)$ will probably yield useful information about the private key f , but a shorter transcript will probably not be helpful.

Example 2. We illustrate with a slightly simplified situation. Let $N = 251$, and suppose that f and m are known to have 83 coefficients equal to each of $+1$ and -1 , with the remaining 85 coefficients equal to 0. Consider a transcript \mathcal{S} consisting entirely of messages with $m_0 = 0$. Then an elementary calculation (described in detail in Section 4.3) gives the probability distributions of $Y_{0,0}$ and $Y_{\pm 1,0}$. (In this example, the random variables $Y_{1,0}$ and $Y_{-1,0}$ are the same by symmetry.) We list some of the values in Table 5. The distance between these two distributions, using formula (6), is

$$\text{Distance}(Y_{0,0}, Y_{\pm 1,0}) = 0.00043.$$

This number is comparatively large, so it is not surprising that Szydło's method is effective on transcripts of practical length. Of course, this is all under the assumption that the private key is weakened by choosing $u = 1$.

b	$\text{Prob}(Y_{0,0} = \pm b)$	$\text{Prob}(Y_{\pm 1,0} = \pm b)$
0	$3.7966 \cdot 10^{-2}$	$3.7851 \cdot 10^{-2}$
1	$3.7795 \cdot 10^{-2}$	$3.7682 \cdot 10^{-2}$
2	$3.7287 \cdot 10^{-2}$	$3.7178 \cdot 10^{-2}$
3	$3.6455 \cdot 10^{-2}$	$3.6354 \cdot 10^{-2}$
4	$3.5322 \cdot 10^{-2}$	$3.5230 \cdot 10^{-2}$
5	$3.3916 \cdot 10^{-2}$	$3.3836 \cdot 10^{-2}$
10	$2.4169 \cdot 10^{-2}$	$2.4161 \cdot 10^{-2}$
15	$1.3723 \cdot 10^{-2}$	$1.3766 \cdot 10^{-2}$
20	$6.1961 \cdot 10^{-3}$	$6.2457 \cdot 10^{-3}$
25	$2.2185 \cdot 10^{-3}$	$2.2504 \cdot 10^{-3}$
30	$6.2760 \cdot 10^{-4}$	$6.4162 \cdot 10^{-4}$
40	$2.4288 \cdot 10^{-5}$	$2.5345 \cdot 10^{-5}$
50	$3.4133 \cdot 10^{-7}$	$3.6622 \cdot 10^{-7}$
60	$1.6198 \cdot 10^{-9}$	$1.8023 \cdot 10^{-9}$
70	$2.3430 \cdot 10^{-12}$	$2.7323 \cdot 10^{-12}$

Table 5. Different Probability Distributions for s_k

4.2 Coefficient Frequency Analysis and Moment Equalization

The coefficient frequency analysis method described in Section 4.1 depends on the fact that if we restrict to signatures (s, m) in which (say) m_0 takes a fixed value ϵ , then the function

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

behaves slightly differently depending on the value δ of f_k . This suggests that the method will work less efficiently if we generate signatures so as to make the functions $X_{k,\epsilon}$ for different values of ϵ more closely resemble one another.

For example, we can choose w_2 so that for every index i , the i^{th} coefficient of $m + w_1 + pw_2$ averages to 0. This is easily accomplished by the simple rule (see [15]):

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

The effect is to average value of $X_{k,\epsilon}$ equal zero. We say in this case that we have equalized the first moments, since letting

$$\mathcal{S}_\epsilon = \{\text{signatures } (s, m) \text{ with } m_0 = \epsilon\}$$

be the set of allowable signatures, we have

$$\text{Average}(X_{k,\epsilon}) = \frac{1}{\#\mathcal{S}_\epsilon} \sum_{(s,m) \in \mathcal{S}_\epsilon} s_k = 0$$

is the same for every value of ϵ .

This first moment equalization method can be extended to higher moments, thereby making the distributions look more and more alike. We will say that the set of allowable signatures has been R^{th} moment equalized if for each $0 \leq r \leq R$, the r^{th} moment

$$\frac{1}{\#\mathcal{S}_\epsilon} \sum_{(s,m) \in \mathcal{S}_\epsilon} s_k^r$$

is the same for every value of k and ϵ . It is easy to see that this will be achieved if we can choose w_2 so that for each index i and for each $0 \leq r \leq R$, the r^{th} moment of

$$m_i + pw_{2,i}$$

does not depend on the value of m_i .

For example, suppose that $p = 3$. As noted above, we can achieve first moment equalization by setting

$$w_{2,i} = \begin{cases} -m_i & \text{with probability } 1/3, \\ 0 & \text{with probability } 2/3. \end{cases}$$

In a similar way, we obtain second moment equalization by the slightly more complicated rules given in Table 6, and fourth moment equalization is provided using the rules in Table 7. We leave it as an exercise for the interested reader to derive rules to equalize higher moments.

m_i	$w_{2,i}$
-1	1 with probability 1/3
	0 with probability 2/3
0	-1 with probability 1/9
	0 with probability 7/9
	1 with probability 1/9
1	-1 with probability 1/3
	0 with probability 2/3

Table 6. Rules for Second Moment Equalization

4.3 Coefficient Frequency Analysis When $u \neq 1$

Szydło's coefficient frequency analysis described in Section 4.1 makes use of the assumption $u(X) = 1$ to form a direct relationship between m_0 , s_k , and f_k as

m_i	$w_{2,i}$
-1	1 with probability 56/162
	0 with probability 105/162
	-2 with probability 1/162
0	-1 with probability 7/54
	0 with probability 40/54
	1 with probability 7/54
1	-1 with probability 56/162
	0 with probability 105/162
	2 with probability 1/162

Table 7. Rules for Fourth Moment Equalization

described by equation (5). If $u(X)$ is nontrivial, then the relationship is much more complicated, especially since it is only the mod p reduction of $U(X) * m(X)$ that actually appears in $s(X)$, where $U(X) \equiv u(X)^{-1} \pmod{p}$. As we will see, this means that an attacker who attempts to use coefficient frequency analysis will need to distinguish between random variables that only take on p different values, rather than the q different values that were used in Section 4.1. The effect is striking. Experimentally, no useful information is exposed by 100 million signatures; and a theoretical computation suggests that even 10^{100} signatures is insufficient to mount a credible attack.

Remark 13. It is possible to use the moment equalization method of Section 4.2 in the case that $u(X) \neq 1$. In this case one chooses w_2 to equalize the moments of the coefficient distributions of the quantity

$$(u^{-1} * m \pmod{p}) + pw_2.$$

However, as we will see, the coefficient distributions are already so similar that it does not appear that moment equalization is necessary in practice if $u(X) \neq 1$.

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

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

Suppose now that we restrict attention to subtranscripts whose messages have (say) a fixed value for their constant terms m_0 . Then the quantity

$$\left(\sum_{j=0}^{N-1} U_{i-j} m_j \right) \bmod p = \left(U_i m_0 + \sum_{j \neq 0} U_{i-j} m_j \right) \bmod p \quad (7)$$

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

$$s_k = \sum_{i \in I_{-1}} f_{k-i} A + \sum_{i \in I_0} f_{k-i} B + \sum_{i \in I_1} f_{k-i} C, \quad (8)$$

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 (7) that leads to the differences in A , B , and C only takes on three values (since $p = 3$), 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 quantify this observation both theoretically and experimentally.

4.3.1 Coefficient Frequency Analysis—Theoretical Results

We take $p = 3$. 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 distribution of the coefficients of the product of two (random) ternary polynomials is very close to a generalized hypergeometric distribution, since the 1 's and the -1 's in the first polynomial can be viewed as selecting coefficients, without replacement, from the second polynomial and then adding or subtracting them to form a total. In order to state a precise result, we define

$$\mathcal{HG}(N; a, b, c, d; K) = \sum_{\substack{a_1, a_2, b_1, b_2 \geq 0 \\ a_1 + a_2 \leq a \\ b_1 + b_2 \leq b \\ a_1 - a_2 - b_1 + b_2 = K}} \frac{\binom{c}{a_1; b_1} \binom{d}{a_2; b_2} \binom{N - c - d}{a - a_1 - a_2; b - b_1 - b_2}}{\binom{N}{a; b}},$$

where

$$\binom{n}{a; b} = \frac{n!}{a! \cdot b! \cdot (n - a - b)!} = \binom{n}{a} \binom{n - a}{b}$$

is a trinomial coefficient, i.e., $\binom{n}{a; b}$ is the coefficient of $x^a y^b z^{n-a-b}$ in the expansion of $(x + y + z)^n$.

Lemma 1. *Let $0 \leq i < N$ be a fixed index. Then for randomly chosen $f \in \mathcal{T}_N(a, b)$ and $g \in \mathcal{T}_N(c, d)$,*

$$\text{Prob}((f * g)_i = K) = \mathcal{HG}(N; a, b; c, d; K).$$

Consider a transcript of signatures m with $m_0 = 0$. To give the attacker every possible advantage, we will assume that every message has the same coefficient distribution, say $m \in \mathcal{T}_N(dm_1, dm_2)$. We will further assume that the attacker knows the coefficient distribution of $U \equiv u^{-1} \pmod{3}$, say $U \in \mathcal{T}_N(dU_1, dU_2)$. Then

$$(U * m)_0 = U_0 m_0 + \sum_{j=1}^{N-1} U_j m_{N-j} = \sum_{j=1}^{N-1} U_j m_{N-j},$$

since by assumption $m_0 = 0$. The value of this sum follows a generalized hypergeometric distribution, but the specific distribution depends on the unknown, but fixed, value of U_0 . More precisely,

$$\text{Prob}((U * m)_0 = K) = \begin{cases} \mathcal{HG}(N - 1; dU_1 - 1, dU_2; dm_1, dm_2; K) & \text{if } U_0 = -1, \\ \mathcal{HG}(N - 1; dU_1, dU_2; dm_1, dm_2; K) & \text{if } U_0 = 0, \\ \mathcal{HG}(N - 1; dU_1, dU_2 - 1; dm_1, dm_2; K) & \text{if } U_0 = 1. \end{cases}$$

If the attacker could see the exact values of $(U * m)_0$, then it is possible he would be able to distinguish between these three different probability distributions. However, and this is the vital point in our security analysis, the exact value of $(U * m)_0$ does not appear in the signature. The only quantity appearing in the signature, and thus the only quantity that the attacker can hope to exploit, is the value of $(U * m)_0 \pmod{3}$. Thus the best that he can do is try to distinguish between the probability distributions of the mod 3 random variables A , B , and C defined by the following formulas:

$$\begin{aligned} \text{Prob}(A = \epsilon) &= \sum_{K \equiv \epsilon \pmod{3}} \mathcal{HG}(N - 1; dU_1 - 1, dU_2; dm_1, dm_2; K) \\ \text{Prob}(B = \epsilon) &= \sum_{K \equiv \epsilon \pmod{3}} \mathcal{HG}(N - 1; dU_1, dU_2; dm_1, dm_2; K) \\ \text{Prob}(C = \epsilon) &= \sum_{K \equiv \epsilon \pmod{3}} \mathcal{HG}(N - 1; dU_1, dU_2 - 1; dm_1, dm_2; K) \end{aligned}$$

These distributions are virtually indistinguishable if the parameters N , dU_1 , dU_2 , dm_1 and dm_2 are of moderate size. To illustrate, we consider the typical values

$$N = 251, \quad dU_1 = 83, \quad dU_2 = 83, \quad dm_1 = 83, \quad dm_2 = 83.$$

By symmetry in this situation, it turns out that A and C have the same distributions, and that for all of A , B , and C , the probability for $\epsilon = 1$ is the same as the probability for $\epsilon = -1$. Thus the interesting comparisons are

$$\text{Prob}(A = 0) \text{ versus } \text{Prob}(B = 0) \text{ and } \text{Prob}(A = \pm 1) \text{ versus } \text{Prob}(B = \pm 1).$$

Using the summation formula for \mathcal{HG} and doing all computations to 100 digits of accuracy, we find that

$$\begin{aligned} \text{Prob}(A = 0) &= \text{Prob}(B = 0) - 10^{-63.36}, \\ \text{Prob}(A = \pm 1) &= \text{Prob}(B = \pm 1) + 10^{-63.66}. \end{aligned}$$

Thus in order to distinguish between A , B , and C , it is necessary to distinguish between probabilities that differ by approximately 10^{-63} . Using the standard estimate that in a sample of size M , the error from random fluctuations will be on the order of $O(\sqrt{M})$, we see that 10^{100} or more signatures are needed before frequency distribution methods have a chance of succeeding.

Remark 14. We also note that the attacker must correctly distinguish these distributions N times, once for each coefficient of s . (Or at least for enough coefficients to perform an exhaustive search for the rest.)

4.3.2 Coefficient Frequency Analysis—Experimental Results

In order to check our theoretical results, we performed 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 (7),

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

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 Table 8. It is clear that the differences in the three distributions are virtually non-existent after 100 million messages, since the differences of approximately 10^{-5} are what one would expect to find from random fluctuations in a sample of size 10^7 .

4.3.3 Coefficient Frequency Analysis—Implications If Successful

In the interests of completeness, we will consider the security implications if one were able to distinguish the three distributions A , B , and C sufficiently to untangle the three sums in (8), although we do not believe that it is actually

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

Table 8. $\text{Prob}(\epsilon, b) = \text{Prob}((U \cdot m)_k \equiv b \pmod{3}) : k \in I_\epsilon$

possible to do so. More precisely, suppose that an attacker could recover, either partially or entirely, the $3N$ sums

$$\sum_{i \in I_\epsilon} f_{k-i}, \quad \epsilon \in \{-1, 0, 1\}, \quad 0 \leq k < N. \quad (9)$$

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 (9) 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 (8) 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 \mathbb{Z} . There are 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, as long as the private key is chosen in the form

$$f(X) = u(X) + p * F(X) \quad \text{and} \quad g(X) = u(X) + p * G(X)$$

with a nontrivial (secret) polynomial $u(X)$, there appears to be no practical way to use coefficient frequency analysis of a transcript of signatures to recover any useful information about the distribution of the coefficients of the private key.

References

1. M. Ajtai, *The shortest vector problem in L_2 is NP-hard for randomized reductions*. In Proc. 30th ACM Symposium on Theory of Computing, 1998, 10–19.
2. M. Ajtai, C. Dwork, *A public-key cryptosystem with worst case/average case equivalence*. In Proc. 29th ACM Symposium on Theory of Computing, 1997, 284–293.
3. M. Ajtai, R. Kumar, and D. Sivakumar, *A sieve algorithm for the shortest lattice vector problem*. Proc. 33rd ACM Symposium on Theory of Computing, 2001. To appear.
4. J. Blömer, J.-P. Seifert, *On the Complexity of Computing Short Linearly Independent Vectors and Short Bases in a Lattice*, STOC '99
5. P. van Emde Boas. *Another NP-complete partition problem and the complexity of computing short vectors in lattices*. Mathematics Department, University of Amsterdam, TR 81-04, 1981.
6. E.F. Brickell and K.S. McCurley. *Interactive Identification and Digital Signatures*, AT&T Technical Journal, November/December, 1991, 73–86.
7. J.Y. Cai, A.P. Nerukar, *An improved worst-case to average-case reduction for lattice problems*, Proc. 38th Symposium on Foundations of Computer Science, 1997, 468–477
8. I. Dinur, G. Kindler, S. Safra, *Approximating CVP to within almost-polynomial factors is NP-hard*, Proc. 39th Symposium on Foundations of Computer Science, 1998, 99–109
9. O. Goldreich, S. Goldwasser, *On the limits of non-approximability of lattice problems*, Proc. 39th Symposium on Foundations of Computer Science, 1998, 1–9
10. O. Goldreich, S. Goldwasser, S. Halvei, *Public-key cryptography from lattice reduction problems*. In Proc. CRYPTO'97, Lect. Notes in Computer Science 1294, Springer-Verlag, 1997, 112–131.
11. O. Goldreich, D. Micciancio, S. Safra, J.-P. Seifert, *Approximating shortest lattice vectors is not harder than approximating closest lattice vectors*, Electronic Colloquium on Computational Complexity, TR99–002, 1999

12. L.C. Guillou and J.-J. Quisquater. *A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory*, Advances in Cryptology—Eurocrypt '88, Lecture Notes in Computer Science 330 (C.G. Günther, ed.), Springer-Verlag, 1988, 123–128.
13. J. Hoffstein, B.S. Kaliski, D. Lieman, M.J.B. Robshaw, Y.L. Yin, *Secure user identification based on constrained polynomials*, US Patent 6,076,163, June 13, 2000.
14. 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.
15. J. Hoffstein, J. Pipher, J.H. Silverman, *NSS: An NTRU Lattice-Based Signature Scheme*, Advances in Cryptology—Eurocrypt '01, Lecture Notes in Computer Science, Springer-Verlag, 2001.
16. J. Hoffstein, J. Pipher, J.H. Silverman, *Enhanced Encoding and Verification Methods for the NTRU Signature Scheme*, NTRU Technical Note #017, May 2001, <www.ntru.com>.
17. J. Hoffstein, D. Lieman, J.H. Silverman, *Polynomial Rings and Efficient Public Key Authentication*, in Proceeding of the International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC '99), Hong Kong, (M. Blum and C.H. Lee, eds.), City University of Hong Kong Press.
18. J. Hoffstein, J.H. Silverman, *Polynomial Rings and Efficient Public Key Authentication II*, in Proceedings of a Conference on Cryptography and Number Theory (CCNT '99), (I. Shparlinski, ed.), Birkhauser.
19. J. Jonsson, *A forgery method for the NTRU signature scheme*, in preparation, May 2001.
20. P. Klein, *Finding the closest lattice vector when it's unusually close*, Symposium on Discrete Algorithms 2000.
21. H. Koy, C.-P. Schnorr, *Segment LLL-Reduction of Lattice Bases*, Cryptography and Lattices Conference—Proceedings of CaLC '01, (March 2001, Providence, RI), J. Silverman (ed.), Lecture Notes in Computer Science, Springer-Verlag.
22. H. Koy, C.-P. Schnorr, *Segment LLL-Reduction with Floating Point Orthogonalization*, Cryptography and Lattices Conference—Proceedings of CaLC '01, (March 2001, Providence, RI), J. Silverman (ed.), Lecture Notes in Computer Science, Springer-Verlag.
23. J. Lagarias, H.W. Lenstra, Jr., C.P. Schnorr, *Korkin-Zolotarev bases and successive minima of a lattice and its reciprocal lattice*, Combinatorica 10 (1990), 333–348.
24. A.K. Lenstra, H.W. Lenstra Jr., L. Lovász, *Factoring polynomials with rational coefficients*, Mathematische Ann. 261 (1982), 513–534.
25. A.J. Menezes, *Software Implementation of Elliptic Curve Cryptosystems Over Binary Fields*, presentation at CHES 2000, August 17, 2000.
26. A.J. Menezes and P.C. van Oorschot and S.A. Vanstone. *Handbook of Applied Cryptography*, CRC Press, 1996.
27. R. Merkle, M. Hellman, *Hiding information and signatures in trapdoor knapsacks*, IEEE Trans. Inform. Theory, IT-24:525–530, September 1978.
28. D. Micciancio, *The shortest vector in a lattice is hard to approximate to within some constant*, Proc. 39th Symposium on Foundations of Computer Science, 1998, 92–98.

29. I. Mironov, *A note on cryptanalysis of the preliminary version of the NTRU signature scheme*, IACR preprint server, <<http://eprint.iacr.org/2001/005/>>
30. P. Nguyen, *Cryptanalysis of the Goldreich-Goldwasser-Halevi Cryptosystem from Crypto '97*, Advances in Cryptology—Proceedings of CRYPTO '99, (August 15–19, 1999, Santa Barbara, California), M. Wiener (ed.), Lecture Notes in Computer Science, Springer-Verlag.
31. T. Okamoto. *Provably secure and practical identification schemes and corresponding signature schemes*, Advances in Cryptology—Crypto '92, Lecture Notes in Computer Science 740 (E.F. Brickell, ed.) Springer-Verlag, 1993, 31–53.
32. C.-P. Schnorr, *A hierarchy of polynomial time lattice basis reduction algorithms*, Theoretical Computer Science 53 (1987), 201–224.
33. C.-P. Schnorr, *A more efficient algorithm for lattice basis reduction*, J. Algorithms 9 (1988), 47–62.
34. C.-P. Schnorr. *Efficient identification and signatures for smart cards*, Advances in Cryptology—Crypto '89, Lecture Notes in Computer Science 435 (G. Brassard, ed), Springer-Verlag, 1990, 239–251.
35. C.-P. Schnorr, M. Euchner, *Lattice basis reduction: improved practical algorithms and solving subset sum problems*, Math. Programming 66 (1994), no. 2, Ser. A, 181–199.
36. A. Shamir, *A polynomial-time algorithm for breaking the basic Merkel-Hellman cryptosystem*. In Proceedings of the 23rd IEEE Symposium on Foundations of Computer Science, IEEE, 1982, 145–152.
37. J.H. Silverman. *Estimated Breaking Times for NTRU Lattices*, NTRU Technical Note #012, March 1999, <www.ntru.com>.
38. J.H. Silverman. *Almost Inverses and Fast NTRU Key Creation*, NTRU Technical Note #014, March 1999, <www.ntru.com>.
39. J. Stern. *A new identification scheme based on syndrome decoding*, Advances in Cryptology—Crypto '93, Lecture Notes in Computer Science 773 (D. Stinson, ed.), Springer-Verlag, 1994, 13–21.
40. J. Stern. *Designing identification schemes with keys of short size*, Advances in Cryptology—Crypto '94, Lecture Notes in Computer Science 839 (Y.G. Desmedt, ed), Springer-Verlag, 1994, 164–173.
41. J. Stern, *Cryptanalysis of the NTRU signature scheme*, preprint, May 2001.
42. D. Stinson, *Cryptography: Theory and Practice*. CRC Press, 1997.
43. M. Szydło, *A frequency analysis attack on the NTRU signature scheme*, in preparation, May 2001.