NTRUSign: Digital Signatures Using the NTRU Lattice

Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher,
Joseph H. Silverman, William Whyte

NTRU Cryptosystems,
5 Burlington Woods,
Burlington, MA 02144

{jhoffstein,nhowgravegraham,jpipher,jsilverman,wwhyte}@ntru.com

Abstract. In this paper we introduce NTRUSIGN, a new family of signature schemes based on solving the approximate closest vector problem (APPR-CVP) in NTRU-type lattices. We explore the properties of general APPR-CVP based signature schemes (e.g. GGH) and show that they are not immune to transcript attacks even in the random oracle model. We then introduce the idea of using carefully chosen perturbations to limit the information that is obtainable from an analysis of a large signature transcript. In the case of NTRUSIGN this can be achieved while maintaining attractive efficiency properties.

1 Introduction

Lattices have been studied by cryptographers for quite some time, both in the field of cryptanalysis (see for example [18–20]) and as a source of hard problems on which to build encryption schemes (see [1, 8, 10]). Interestingly, though, research about building secure and efficient *signature schemes* using the theory of lattices is extremely sparse in the cryptographic literature. An early scheme is due to Goldreich, Goldwasser and Halevi [8], who proposed that one could sign a message by demonstrating the ability to solve the approximate closest vector problem (APPR-CVP) reasonably well for a point in space (hereafter referred to as the *message digest point*) generated from a hash of the message, and verify by checking that the "close lattice point" returned was indeed a lattice point and that it was close enough to the message digest point to make forgeries impractical. However, this idea was not analyzed in detail by its authors.

Another public-key cryptosystem, NTRUENCRYPT [10], was proposed at roughly the same time as [8]. One of the (several) ways that NTRUENCRYPT can be viewed is as a lattice cryptosystem based on a particularly efficient class of convolution modular lattices, which we will refer to as *NTRU lattices.* In this paper we study signature schemes based on solving APPR-CVP in the special class of NTRU lattices. A private key for the NTRUENCRYPT encryption scheme consists of a good basis for an N-dimensional sublattice of a $2N$-dimensional NTRU lattice, but in order to solve APPR-CVP efficiently for arbitrary message digest points one must know a full good basis for the lattice. A major contribution of this paper is given in section 4.1 where we detail an efficient technique to generate a full basis from knowledge of the small half-basis, thereby validating this approach.

A second major undertaking of this paper concerns the security of signature algorithms based on APPR-CVP. Such algorithms do not give a zero-knowledge scheme, because a transcript of signatures leaks information about the private key. In section 5.2 we therefore introduce *perturbation techniques* as a general way to reduce the effectiveness of transcript analysis in APPR-CVP-based signature schemes. In the case of NTRUSIGN (section 5.3), the use of perturbations guarantees that the number of signatures required to extract useful information far exceeds any practical requirements.

A third idea proposed in this paper is to consider a slightly different class of convolution modular lattices, which we refer to as *transpose NTRU lattices.* These lattices allow for more efficient signing than the standard NTRU lattices, but have slightly different security considerations. See section 4.3 for details.

We now describe the organization of this paper. In section 2 we explain the basic operations behind NTRUSIGN in practical engineering terms. In sections 3 and 4 we describe the mathematics underlying the NTRU technology. We begin by describing the NTRU lattice[1] as a module of rank 2 over certain algebraic rings. We then explain how we can still perform standard lattice tasks, such as completing bases and solving APPR-CVP, by working over these rings.

Next, we turn to the security of NTRUSIGN. Due to page limitations, we have been forced to considerably condense this analysis. A far more detailed presentation can be found at [11]. We first consider an adversary who does not use any signatures generated by the private key. In section 6 we show that the underlying lattice problems are infeasible to attack (using the best known methods) when N is suitably large.

We then turn our attention to an attacker who has access to a transcript of signatures. In section 7 we show that any signature scheme based on solving APPR-CVP using a (secret) short basis is vulnerable to a transcript attack after some finite number of signatures have been obtained, even in the random oracle model. This has implications for both the GGH signature scheme [8] and NTRUSIGN. In section 7.2 we specifically concentrate on the information leaked by NTRUSIGN signatures, including discussion of the effectiveness of the perturbations of sections 5.2–5.3 as a countermeasure.

Further technical details are described in the appendices. Appendix A completes the proof of a theorem in section 4.1 and a lemma in section 4.2. Appendices B and C extend the analyses of sections 6 and 7 respectively. Appendix D details the requirement on the hash function used with NTRUSIGN. Appendix E gives preliminary performance figures.

It should be noted that there have been several other attempts to create a signature scheme using the mathematics underlying the NTRUENCRYPT cryptosystem [12], but they succumbed to successful cryptanalysis in [6] and most

[1] Strictly speaking, the "NTRU module," but we will continue to call it a lattice.

recently in [7]. The downfall of these approaches was that without knowledge of a full basis of an NTRU lattice, additional structure needed to be added to the signature scheme. An attacker could exploit this structure, leading to both forgery of individual signatures and key recovery. These attacks on previous signature schemes either do not apply to NTRUSign or require an infeasible number of signed messages to be effective; see section 7 for details.

2 NTRUSign: An Engineering Specification

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

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

In more mathematical terms, R is the quotient ring $R = \mathbb{Z}[X]/(X^N - 1)$. If one of the polynomials has all coefficients chosen from the set $\{0, 1\}$ we will refer to the convolution as being *binary*. If coefficients of the polynomials are reduced modulo q for some q, we will refer to the convolution as being *modular*.

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

The basic operations of NTRUSign are as follows:

Key Generation — requires a source of (pseudo)random bits.

1. Input: Integers $N, q, d_f, d_g, B \ge 0$, and the string $t =$ “standard” or “transpose”.
2. Generate B private lattice bases and one public lattice basis: Set $i = B$. While $i \ge 0$:
   (a) Randomly choose $f, g \in R$ to be binary with d_f, d_g ones, respectively.
   (b) Find small $F, G \in R$ such that $f * G - F * g = q$. Sections 3 to 4.2 give more detail on this process.
   (c) If $t =$ “standard”, set $f_i = f$ and $f'_i = F$. If $t =$ “transpose”, set $f_i = f$ and $f'_i = g$. Set $h_i = f_i^{-1} * f'_i \bmod q$. Set $i = i - 1$.
3. Public output: The input parameters and $h = h_0 \equiv f_0^{-1} * f'_0 \bmod q$.
4. Private output: The public output and the set $\{f_i, f'_i, h_i\}$ for $i = 0 \ldots B$.

Signing — requires a hash function $H : \mathcal{D} \to R$ on a digital document space $\mathcal{D}$. The properties required of this hash function are explored in appendix D. Signing also requires a norm function $\|.\| : R^2 \to \mathbb{R}$ and a “norm bound” $\mathcal{N} \in \mathbb{R}$. For $(s, t) \in R^2$ we define $\|(s \bmod q, r \bmod q)\|$ to be the minimal value of $\|(s + k_1 q, r + k_2 q)\|$ for $k_1, k_2 \in R$.

1. INPUT: A digital document $D \in \mathcal{D}$ and the private key $\{f_i, f_i', h_i\}$ for $i = 0 \ldots B$.
2. Set $r = 0$.
3. Set $s = 0$. Set $i = B$. Encode r as a bit string. Set $m_0 = H(D||r)$, where "$||$" denotes concatenation. Set $m = m_0$.
4. PERTURB THE POINT USING THE PRIVATE LATTICES: While $i \geq 1$:
   (a) Set $x = \lfloor -(1/q)m * f_i' \rceil$, $y = \lfloor (1/q)m * f_i \rceil$, $s_i = x * f_i + y * f_i'$.
   (b) Set $m = s_i * (h_i - h_{i-1}) \bmod q$.
   (c) Set $s = s + s_i$. Set $i = i - 1$.
5. SIGN THE PERTURBED POINT USING THE PUBLIC LATTICE:
   Set $x = \lfloor -(1/q)m * f_0' \rceil$, $y = \lfloor (1/q)m * f_0 \rceil$, $s_0 = x * f_0 + y * f_0'$, $s = s + s_0$.
6. CHECK THE SIGNATURE:
   (a) Set $b = ||(s, s * h - m_0 \bmod q))||$.
   (b) If $b \geq \mathcal{N}$, set $r = r + 1$ and go to step 3.
7. OUTPUT: The triplet (D, r, s).

Verification — requires the same hash function H, norm function $||.||$ and "norm bound" $\mathcal{N} \in \mathbb{R}$.

1. INPUT: A signed document(D, r, s) and the public key h.
2. Encode r as a bit string. Set $m = H(D||r)$.
3. Set $b = ||(s, s * h - m \bmod q))||$.
4. OUTPUT: `valid` if $b < \mathcal{N}$, `invalid` otherwise.

Remark 1. The recommended parameters

$$(N, q, d_f, d_g, B, t, \mathcal{N}) = (251, 128, 73, 71, 1, \text{"transpose"}, 310),$$

where $||.||$ is the centered Euclidean norm (section 3) appear to give a practical and efficient signature scheme with 2^{80} security. We henceforth use $d = 72$ to denote the case where $d_f \approx d_g \approx 72$.

3 A View of NTRU: Background Mathematics

Underlying NTRU is the ring $R = \mathbb{Z}[X]/(X^N - 1)$. There is a natural lattice of dimension N associated with any element $r = \sum_{i=0}^{N} r_i X^i \in R$, namely the one generated by the rows of the following matrix:

$$\begin{pmatrix} r_0 & r_1 & \ldots & r_{N-1} \\ r_{N-1} & r_0 & \ldots & r_{N-2} \\ \vdots & & \ddots & \vdots \\ r_1 & r_2 & \ldots & r_0 \end{pmatrix}.$$

It is easily checked that if M_r and M_s are the matrices corresponding to $r, s \in R$ then the matrix corresponding to $r * s \in R$ is given by $M_r M_s$; i.e. this matrix mapping is a ring isomorphism since it respects both addition and multiplication.

For each $q \in \mathbb{Z}$ and $h \in R$, the set $M_{h,q} = \{(u,v) \in R^2 | v \equiv u * h \pmod{q}\}$ is an R-module of rank 2. (Notice $M_{h,q}$ is also a lattice of dimension $2N$.) Every element of R has a unique representation as a polynomial $r = \sum_{i=0}^{N-1} r_i X^i$. Then a natural measure of size in R is the centered Euclidean norm of the vector of coefficients $\|r\|^2 = \sum_{i=0}^{N-1} r_i^2 - (1/N)\left(\sum_{i=0}^{N-1} r_i\right)^2$. The norm imposed on elements of $M_{h,q}$, or more generally on $(u,v) \in R^2$, is the component-wise Euclidean norm: $\|(u,v)\|^2 = \|u\|^2 + \|v\|^2$.

The element $h \in R$ is chosen so that $(f,g) \in M_{h,q}$ for some f and g of a special form. Assuming that f is invertible in R/qR, the lattice $M_{h,q}$ will contain (f,g) if we set $h = f^{-1} * g \bmod q$. The specified form of f and g is that they be binary elements of R in the sense that d_f of the coefficients of f (respectively d_g of the coefficients of g) are set to 1 and the rest are 0. Thus the norms of $f, g \in R$ and the norm of $(f,g) \in M_{h,q} \subset R^2$ are given by

$$\|f\| = \sqrt{d_f\,(1 - d_f/N)}, \quad \|g\| = \sqrt{d_g\,(1 - d_g/N)}, \quad \|(f,g)\| = \sqrt{\|f\|^2 + \|g\|^2}.$$

There is an obvious R-basis for $M_{h,q}$, namely $\{(1,h),(0,q)\}$. Since we know that $(f,g) \in M_{h,q}$, it is natural to ask if we can find another vector $(F,G) \in M_{h,q}$ so that the pair $\{(f,g),(F,G)\}$ is also an R-basis for $M_{h,q}$. This is possible if (and only if) $\gcd(\text{resultant}(f, X^N-1), \text{resultant}(g, X^N-1)) = 1$ – see section 4.1. A supplemental paper to this one will explain how this condition can be relaxed.

4 NTRUSign Key Generation

4.1 Completing the Basis

The general strategy for completing the basis of $M_{h,q}$ is to project f and g down to $\mathbb{Z}$ via the resultant mapping, which respects multiplication. The definition of the resultant of f with $X^N - 1$, R_f, is the product of f evaluated at all the complex roots of $X^N - 1$. For basic properties of resultants, see for example [3, Chapter 3]. We here use the fact that

$$R_f \equiv \prod_{i=1}^{N-1} f(x^i) \bmod \Phi \in \mathbb{Z},$$

where $\Phi(X) = \sum_{i=0}^{N-1} X^i \in R$. Therefore we define $\rho_f \equiv \prod_{i=2}^{N-1} f(x^i) \bmod \Phi$, and ρ_g similarly. We then know that for some $k_f, k_g \in \mathbb{Z}[X]$,

$$\rho_f f + k_f(X^N - 1) = R_f = \text{resultant}(f, X^N - 1) \bmod \Phi,$$
$$\rho_g g + k_g(X^N - 1) = R_g = \text{resultant}(g, X^N - 1) \bmod \Phi.$$

Assuming that R_f and R_g are coprime over the integers, we can now use the extended Euclidean algorithm to find $\alpha, \beta \in \mathbb{Z}$ such that $\alpha R_f + \beta R_g = 1$, in which case we have

$$(\alpha\rho_f)f + (\beta\rho_g)g = 1 + k(X^N - 1).$$

Thus if we set $F = -q\beta\rho_g$ and $G = q\alpha\rho_f$, then

$$f * G - g * F = q. \tag{1}$$

Theorem 1. *Let $f, g, F, G \in R$ satisfy (1), let $h = f^{-1} * g \pmod q$, and let $M_{h,q}$ be the NTRU R-module generated by $\{(1,h),(0,q)\}$.*

(a) *$\{(f,g),(F,G)\}$ form a basis for $M_{h,q}$.*
(b) *If $F', G' \in R$ also satisfy $f * G' - g * F' = q$, then there is an element $c \in R$ so that $F' = F + c * f$ and $G' = G + c * g$.*

Proof. Elementary linear algebra. For details, see Theorem 2 in Appendix A.

If we view $M_{h,q}$ is as a matrix of generating rows, the above theorem can be seen to be a unimodular change of basis.

$$\begin{pmatrix} f & g \\ F & G \end{pmatrix} = \begin{pmatrix} f & (g - f * h)/q \\ F & (G - F * h)/q \end{pmatrix} \begin{pmatrix} 1 & h \\ 0 & q \end{pmatrix}$$

With this notation we can define the *discriminant* of the R-module to be the determinant of the matrix, which can be seen to be an invariant modulo multiplication by a unit of R.

Although the F and G generated as in Theorem 1 complete a basis for $M_{h,q}$, they typically have very large coefficients. However, we can clearly remove any R-multiple of the vector (f,g) from (F,G) and still have an R-basis. In the next section we discuss how to find a good R-multiple by which to reduce.

4.2 Finding a Good Second Half for the Basis

We first discuss what we mean by APPR-CVP in the context of an R-module.

Definition 1. *Let $\mathcal{L}$ be a R-module of rank r, and let $\mathbf{m}$ be an arbitrary element $\mathbf{m} \in (\mathbb{R}[X]/(X^N - 1))^r$. A vector $\mathbf{v} \in \mathcal{L}$ is said to be a solution of* APPR-CVP *if $\|\mathbf{v} - \mathbf{x}\| < \mathcal{N}$ for some suitably small $\mathcal{N} \in \mathbb{R}$.*

In this paper the most common instance of APPR-CVP that will be considered is the case when the rank is $r = 2$, and the point $\mathbf{m}$ will actually be restricted to lie in $\mathbf{m} \in R^2 \subset (\mathbb{R}[X]/(X^N - 1))^2$. It is important to note that if $\mathbf{v}$ is a close vector to $\mathbf{m}$ then $\mathbf{v} + \mathbf{w}$ is equally close to the vector $\mathbf{m} + \mathbf{w}$ for any $\mathbf{w} \in M_{h,q}$. Thus if one is given an arbitrary point $(m_1, m_2) \in R^2$ then it suffices to find a close vector $(w_1, w_2) \in M_{h,q}$ to the point $(0, m_2 - m_1 * h) \in R^2$. For this reason we will only consider solving APPR-CVP on points of the form $(0, m)$ where $m \in R$.

The approach we use to solve APPR-CVP in R-modules is equivalent to Babai's "inverting and rounding" approach [2] in lattices, i.e. we find the exact rational coordinates to solve the problem, and then round the coefficients to integers to obtain an R-module point.

For example, we can use this technique to reduce the (F,G) vector obtained in section 4.1 from completing the basis of $M_{h,q}$. In Appendix A we show that reducing the first coordinate F will also usefully reduce the second coordinate G. Therefore it suffices to find a $k \in R$ such that $F_{red} = ||F - k * f||$ is small. We know $f^{-1} = (1/R_f)\rho_f \in \mathbb{Q}[X]/(X^N - 1)$, and if we could take k to be equal to $l = F * f^{-1} \in \mathbb{Q}[X]/(X^N - 1)$, F_{red} would be 0. To obtain a $k \in R$, we take $k = \lfloor l \rceil$, the rounding of l to the nearest integer.[2]

Since we know $F - l * f = F - (\lfloor l \rceil + \{l\}) * f = 0$, i.e. $F - k * f = \{l\} * f$ we can use the following lemma to estimate the norm of the resulting vector.

Lemma 1. *Let $X_1, \ldots, X_N$ be independent continuous random variables that are uniformly distributed in the range $[-B/2, B/2]$, and let $Y = X_1^2 + \cdots + X_N^2$. Then the mean and standard deviation of Y are given by the formulas*

$$\mu(Y) = \frac{NB^2}{12} \qquad \textit{and} \qquad \sigma(Y) = \frac{B^2}{6}\sqrt{\frac{N}{5}}.$$

The proof of this lemma is left to appendix A, but we see that with $B = 1$ this lemma implies that the expected Euclidean norm of l is $\sqrt{N/12}$ (the centered norm will be very marginally smaller), and so using the pseudo-multiplicative property $||F - k * f|| \approx \sqrt{N/12}||f|| \approx \sqrt{Nd_f/18}$. Once we have reduced F in this fashion we can simply set $G = h * F \bmod q$ in the range $(-q/2, q/2]$, since we know G will also be small. One can further reduce the vector (F, G) by performing weak reduction (i.e. taking dot products) with rotations of the (f, g) vector, but in practice this appears not to be necessary. This therefore completes the description of "good" basis completion in the NTRU lattice.

Finally, we note that the APPR-CVP technique extends naturally to R-modules of higher dimension. For example in the case of $M_{h,q}$ we are trying to find $x, y \in R$ such that $x * (f, g) + y * (F, G) \approx (0, m)$. Over $\mathbb{Q}[X]/(X^N - 1)$ the exact solutions are given by

$$(x', y') = (0, m)\begin{pmatrix} f & g \\ F & G \end{pmatrix}^{-1} = \frac{1}{q}(0, m)\begin{pmatrix} G & -g \\ -F & f \end{pmatrix} = \left(\frac{-m * F}{q}, \frac{m * f}{q}\right),$$

thus we take $x = \lfloor x' \rceil$ and $y = \lfloor y' \rceil$ in which case

$$||(0, m) - x * (f, g) - y * (F, G)|| \; = \; ||\{x\} * (f, g) + \{y\} * (F, G)|| \; \approx \; \frac{N\sqrt{2d_f}}{18},$$

assuming (F, G) is as above (i.e. after reduction by (f, g)).

4.3 The Transpose NTRU Lattice

The NTRU R-module is characterized by $h = f^{-1} * g \bmod q$, where f and g are binary elements of R. However, once we have obtained a collection of vectors

[2] By the mapping between elements of R and N-dimensional lattices described in section 3, we see that this is exactly equivalent to solving a simple N-dimensional APPR-CVP lattice problem using Babai's inverting and rounding approach. The reason that the lattice corresponding to the matrix M_f is well reduced (which is essential for Babai's technique) is that f is a relatively sparse binary element of R and thus highly orthogonal to its rotations.

f, g, F, G such that $f * G - F * g = q$ we can exchange the roles of F and g, and consider the R-module generated by the rows of the following matrix:

$$\begin{pmatrix} f & F \\ g & G \end{pmatrix} = \begin{pmatrix} f & (F - f * h')/q \\ F & (G - g * h')/q \end{pmatrix} \begin{pmatrix} 1 & h' \\ 0 & q \end{pmatrix},$$

where $h' = f^{-1} * F \bmod q$. We call the R-module generated by these rows the *transpose NTRU lattice*. In the transpose lattice, the shortest known vector is of the form (f, F) (i.e. with very small first coordinate, and reasonably small second coordinate) rather than (f, g) (i.e. both coordinates very small), so the lattice is significantly more efficient for signing operations, which now involve only multiplications of m with *binary* elements of R.

For verification, we could define any norm we like in this R-module, but there appears to be no advantage to changing from the standard centered Euclidean norm. However, the fact that one coordinate of the signature will be much smaller than the other affects transcript analysis; this is analyzed in section 7.

5 Signing and Verification Reviewed

5.1 The Case of No Perturbations ($B = 0$)

With the above mathematical background, we can see that, when $B = 0$, the specification in section 2 is simply a description of solving APPR-CVP within the norm bound $\mathcal{N}$ for the R-module $M_{h,q}$. The only "trick" is that in the NTRU lattice it is sufficient to just give the first module coordinate s as the signature, since all such module points are of the form $(s, h * s + kq)$ and the k which makes this as close to $(0, m)$ as possible can be trivially obtained by a modulo q reduction.

For NTRUSIGN with the parameters $(N, q, d, \mathcal{N}, B) = (251, 128, 72, 310, 0)$, the typical experimental size of a signature is about 210.

5.2 Perturbation Techniques

In section 7 it is shown that general lattice based APPR-CVP signing algorithms are not zero knowledge, and that an attacker can average a finite transcript to obtain the private information. Section 7 shows that the averages involved converge quite fast for a signing algorithm based on APPR-CVP alone. To make these schemes practical, we must increase the length of transcript required.

For general APPR-CVP signing algorithms, we propose that the signer first hashes the digital document D to obtain a point $\mathbf{m}$, and then "perturbs" this point to $\mathbf{m}' = \mathbf{m} + \epsilon$ by adding a point ϵ. ϵ is different for each message and not known to the verifier. The signer then signs $\mathbf{m}'$; if ϵ is small enough, a signature on $\mathbf{m}'$ will also be a valid signature for $\mathbf{m}$.

At a minimum, this will slow down the convergence of any transcript averages. However, for full effectiveness, we would like it to be impossible for an attacker to distinguish between those parts of the averages due to the perturbations and those parts due to the "true" signatures.

Therefore, in APPR-CVP signing algorithms, we propose that the perturbations be generated using a signing process in one or more secret lattices of similar geometry to the public lattice. In the case of GGH, each signer would know a good basis $\mathbf{b}_1$ for an entirely secret lattice. To sign, the document would be hashed to a point $\mathbf{m}$, then "signed" using $\mathbf{b}_1$ to obtain $\mathbf{m}_1$, a point in the secret lattice. Then $\mathbf{m}_1$ would be signed with the good basis $\mathbf{b}$ of the public lattice. If the signer knew B secret lattices, the good basis of each would be applied in turn to the result of signing in the previous lattice.

This process will clearly multiply the average norm of signatures by $\sqrt{B}$, and there is a limit to the number of perturbations that can be applied before a validly-generated signature will in general exceed $\mathcal{N}$. Designers of APPR-CVP-based systems should ensure that $\mathcal{N}$ is sufficiently large, if possible, to allow the use of at least one secret lattice.

5.3 NTRUSign with Perturbations

In the case of NTRUSIGN the use of perturbations is already specified in section 2. Here, we start from a digest point $(0, m)$ and find a close point (s_B, t_B) to this in the lattice generated by $\{(1, h_B), (0, q)\}$. We only need to find the first coordinate, since $t_B = s_B * h_B \bmod q$.

For subsequent bases B_i we now want to find the closest point (s_i, t_i) to (s_{i+1}, t_{i+1}) in the lattice generated by $\{(1, h_i), (0, q)\}$. Again, we can transform (s_B, t_B) by a lattice vector to a point $(0, m')$, with

$$\begin{aligned} m' &= t_B - s_B * h_{B-1} \bmod q \\ &= s_B(h_B - h_{B-1}) \bmod q, \end{aligned}$$

as in steps 4a and 4b of the signing process in section 2.

If we consider the first coordinate only, then the difference between the given point and the lattice point is s_i for each $i = B, \ldots, 0$. Thus we define $s = \sum_{i=0}^{B} s_i$, and the final signature point is $(s, h_0 * s \bmod q)$.

Perturbations increase signing time in two ways: first, by requiring the solution of APPR-CVP in several lattices; second, because they increase the average norm of signatures and therefore the chance that the norm of a validly-generated signature will exceed $\mathcal{N}$. The variable r used in signing allows us to recover when a validly generated signature is too large. So long as the average norm of a validly generated signature is less than $\mathcal{N}$, no more than k attempts are needed to give a chance of $1 - 2^{-k}$ of successfully signing a message. We can therefore bound r above by 256 for practical purposes, and encode it in a single byte.

6 Security against a Transcriptless Adversary

6.1 Security of Private Keys

If the adversary is forced to work without the knowledge of any signed messages, key recovery from the public information is equivalent to finding small vectors in the NTRU lattice. This is the long standing hard problem that NTRUEncrypt is based on. In the transpose lattice the situation is even harder for the adversary since the target small vectors (even after weighting) are considerably nearer the Gaussian heuristic of the lattice (and hence harder to find in practice by lattice reduction techniques). Of course any well reduced basis could be used for signing, but finding such a basis is a very hard problem for large N.

The measurements of [10, 11, 17] imply that for the parameters $(N, q, d) = (251, 128, 72)$, a lattice attack on the private key requires about $6.6 * 10^{12}$ MIPS-years. This corresponds to the strength of an 80-bit symmetric cipher [15].

6.2 Security against Forgery

We now consider the difficulty of forging a specific signature without knowledge of the private key. The typical way to attack this [6] is to use a combination of setting some coordinates and using lattice techniques to find others.

Consider a forger who picks a small s, with the hope that $m - sh \bmod q$ will have all small coefficients too. On average these coefficients will be more-or-less random modulo q, so using lemma 1 the average norm of an attempted forgery will be $q\sqrt{N/12}$. Since the asymptotic value of q is $O(N)$, the forgery will have norm $O(N^{3/2})$, which is the same order of magnitude as the (good) signature norm given at the end of section 4.2. For NTRUSign with no perturbations and the parameters $(N, q, d) = (251, 128, 72)$, experimentally generated signatures have an average norm of about 210. The average forgery norm can be calculated to be 585.40. Thus the security of NTRUSign appears to lie in the relative constants involved.

A more refined measure of the difficulty of forging a signature via this method is reflected by the number of standard deviations between an average forgery norm and the average norm of a true signature. Again using lemma 1 we find

$$\frac{\mu(\|\text{Forgery}\|^2) - \mu(\|\text{Good Sig}\|^2)}{\sigma(\|\text{Forgery}\|^2)} \approx \sqrt{\frac{5N}{4}}\left(1 - \frac{Nd}{6q^2}\right) \approx 0.87\sqrt{N}$$

for the parameters in question. So for a fixed ratio of $Nd/(6q^2) < 1$, the number of standard deviations from an average forgery to a true signature grows like $O(\sqrt{N})$.

Further information on the security against forgery, e.g. when the above techniques are used in conjunction with lattice reduction, is given in appendix B.

7 Transcript Leakage

7.1 The Security of appr-CVP Based Signature Schemes

In this section we consider the information available to an attacker who can obtain a large number of signatures generated by the same key.

First, consider general signature schemes based on solving APPR-CVP, such as GGH [8] and NTRUSign. Let us denote the private basis $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$, and let B be the matrix with rows corresponding to these vectors. If H is the matrix with rows corresponding to the public basis, then we know $B = UH$ for some unimodular matrix $U \in GL_n(\mathbb{Z})$.

Any APPR-CVP-type signature constructed using $\mathcal{B}$ has the form $\mathbf{s} - \mathbf{m} = \epsilon_1 \mathbf{b}_1 + \ldots + \epsilon_n \mathbf{b}_n$, where $\epsilon_1, \ldots, \epsilon_n$ are (essentially) uniformly distributed in the interval $(-1/2, 1/2)$. (This follows from the fact that $\mathbf{s}$ is is obtained from $\mathbf{m}$ by a process of rounding. The distribution of the ϵs is constrained by the fact that the coordinates of $(\mathbf{s}-\mathbf{m})$ must be integers; however, experimentally, this distribution proves to be indistinguishable from the uniform distribution.) Therefore, a transcript is a random collection of points in a (centered) fundamental domain for the lattice spanned by the basis vectors $\mathcal{B}$.

Thus we can view $\mathbf{s}$ as a vector valued random variable, and each signature in the transcript is a sample value of that random variable. The question becomes how to extract information about the basis from a transcript of signatures. Take an arbitrary bilinear form $F : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$. We can compute the expectation of $F(\mathbf{s}, \mathbf{s})$ as

$$E(F(\mathbf{s},\mathbf{s})) = \sum_{i,j=1}^{n} E(\epsilon_i \epsilon_j) F(\mathbf{b}_i, \mathbf{b}_j) = \frac{1}{12} \sum_{k=1}^{n} F(\mathbf{b}_k, \mathbf{b}_k)$$

since $E(\epsilon_i \epsilon_j) = 1/12$ if $i = j$ and 0 otherwise.

The values $E(F(\mathbf{s},\mathbf{s}))$ are second moments, since F is a form of degree 2. Thus an attacker can use a sufficiently long transcript to recover the second moment values $E(F(\mathbf{s},\mathbf{s}))$ for every bilinear form.

However, the space of bilinear forms is itself a vector space of dimension n^2, i.e. every bilinear form F is a linear combination of the bilinear forms E_{ij} defined by $E_{ij}(\mathbf{x}, \mathbf{y}) = x_i y_i$, so the maximum amount of information that an attacker can obtain from second moments is the set of n^2 values $E(F_{ij}(\mathbf{s},\mathbf{s}))$. We have

$$E(F_{ij}(\mathbf{s},\mathbf{s})) = \frac{1}{12} \sum_{k=1}^{n} F_{ij}(\mathbf{b}_k, \mathbf{b}_k) = \frac{1}{12} \sum_{k=1}^{n} b_{ki} b_{kj},$$

and these sums give $1/12$ times the coordinates of the Gram matrix BB^t.

As shown in [7] if the bases B and H have integral entries, then this leaks the information UU^t where U is the unimodular transformation matrix. This contains essentially all the information about the basis U of $\mathbb{Z}^n$; indeed this is effectively the internal information that the LLL lattice reduction algorithm uses when reducing. So if lattice reduction techniques could reduce this basis fully to the *orthogonal* basis $\mathbb{Z}^n$ then the implied transformation matrix would leak U itself, and hence B. It is a fascinating open problem to find any results about reducing lattices for which there is known to exist a completely orthogonal basis. Such lattices have been conjectured to be easier, but no results are known. However, we do not wish the security of NTRUSIGN to rest on the conjectured hardness of this problem. This is discussed in the next section, as is the generalization of the above computations to moments of higher order.

7.2 Transcript Leakage by NTRUSign

For NTRUSIGN without perturbations ($B = 0$), second moment information reveals the entries of an associated $2N$ by $2N$ Gram matrix. As noted, the problem of using this information to an attacker's advantage remains unsolved. Fourth moment information reveals the private key in polynomial time by a combination of simple algebra and a method of Gentry and Szydlo [7].

Experiments have determined that for the usual NTRU lattice second moment information can be obtained with transcripts on the order of 10,000 signatures. At least 100 million signatures are required to obtain the fourth moment, if a certain associated square root problem can be solved. If the square root problem is not solved, then the number of signatures required is increased considerably beyond 100 million. In the transpose lattice, the signatures are smaller and the moments converge faster to the appropriate integer values.

However, the use of correctly chosen perturbations adds two additional unknown basis vectors which must be eliminated by an attacker, postponing private key leakage until the next *two* even moments have converged. For example, with NTRUSIGN parameters $(N, q, d, B) = (251, 128, 72, 1)$ (one private basis), we find that obtaining the Gram matrix requires 6th moment convergence. The transcript length needed for this appears to be at least 10^{18} signatures.

We would like to thank Craig Gentry and Mike Szydlo for pointing out that the fourth moment analysis needed to be extended and included in the present discussion. See appendix C for more details.

8 Conclusions and Open Problems

A fascinating open problem is to build an (efficient) lattice based signature scheme that leaks no useful transcript information to an adversary (e.g. provably zero knowledge). Note that such a scheme need not necessarily be based on the closest vector problem.

Another fundamental research question to do with NTRU lattices, is to try to utilize the factorization of $N - 1$. At present there is no evidence to suggest that this has an influence on security. The fact that none of the subrings of $\mathbb{Z}[\zeta]$, other than $\mathbb{Z}$, are Euclidean domains seems to be a serious obstacle.

A third fascinating problem is to devise a lattice reduction algorithm that efficiently finds an orthogonal basis of a lattice when one exists, even in high dimensions. This too seems a hard problem, since the ordering of the vectors is critical to such an algorithm. Note however that with the use of perturbation techniques the security of NTRUSIGN does not need to rest on the hardness of this problem.

See appendix E for practical timing results of NTRUSIGN.

Acknowledgments

The authors would like to acknowledge much useful input from the cryptographic community while NTRUSIGN was in its draft stages. Useful conversations were had with Craig Gentry, Shai Halevi, Jyrki Lahtonen, Tommi Meskanen, Ari Renvall and Mike Szydlo. Particular thanks go to Craig and Mike who have had a large input to this area of research.

References

1. 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.
2. L. Babai *On Lovász lattice reduction and the nearest lattice point problem*, Combinatorica, vol. 6, 1986, 1–13
3. H. Cohen, *A course in computational algebraic number theory*, GTM 138, Springer-Verlag, 1993.
4. Wei Dai, Crypto++ 4.0 Benchmarks, `http://www.eskimo.com/ weidai/benchmarks.html`.
5. Consortium for Efficient Embedded Security, *Efficient Embedded Security Standard #1*, available from `http://www.ceesstandards.org`.
6. Craig Gentry, Jakob Jonsson, Jacques Stern, Michael Szydlo *Cryptanalysis of the NTRU Signature Scheme (NSS) from Eurocrypt '01*, Advances in Cryptology—Asiacrypt '01, Lecture Notes in Computer Science, Springer-Verlag, 2001.
7. C. Gentry, M Szydlo, *Cryptanalysis of the Revised NTRU Signature Scheme*, Advances in Cryptology—Eurocrypt '02, Lecture Notes in Computer Science, Springer-Verlag, 2002.
8. O. Goldreich, S. Goldwasser, S. Halevi, *Public-key cryptography from lattice reduction problems.* In Proc. CRYPTO '97, Lect. Notes in Computer Science 1294, Springer-Verlag, 1997, 112–131.
9. D. Hankerson, J. L. Hernandez, A. Menezes, *Software Implementation of Elliptic Curve Cryptography over Binary Fields*, Cryptographic Hardware and Embedded Systems - CHES 2000, LNCS 1965, C. K. Koc and C. Paar (eds), Springer-Verlag, 2000, 1-19.
10. 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.
11. J. Hoffstein, N. Howgrave-Graham, J. Pipher, J.H. Silverman, W. Whyte *NTRUSign: Digital signatures using the NTRU lattice. Preliminary draft 2* `http://www.ntru.com/NTRUFTPDocsFolder/NTRUSign_v2.pdf`
12. 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.
13. 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.
14. 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.

15. A.K. Lenstra, E.R. Verheul, *Selecting Cryptographic Key Sizes*, Journal of Cryptology vol. 14, no. 4, 2001, 255-293.

16. T. Meskanen and A. Renvall, University of Turku, private communication.

17. A. May, J.H. Silverman, *Dimension reduction methods for convolution modular lattices*, in Cryptography and Lattices Conference (CaLC 2001), J.H. Silverman (ed.), Lecture Notes in Computer Science 2146, Springer-Verlag, 2001

18. 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.

19. P. Nguyen and J. Stern, *Lattice Reduction in Cryptology: An Update*, ANTS 2000, pp 85-112.

20. 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.

A Further Analysis of NTRU Algebra

In this appendix we cover the pieces of mathematics that were a little too detailed for the main report. We start by giving a a more detailed version of Theorem 1.

Theorem 2. *Let $f, g \in R$, let $h \equiv f^{-1} * g \pmod q$, and let $M_{h,q}$ be the NTRU R-module generated by $\{(1,h),(0,q)\}$.*
(a) *Suppose that $F, G \in R$ satisfy $f * G - g * F = q$. Then $\{(f,g),(F,G)\}$ also form a basis for $M_{h,q}$.*
(b) *Suppose that $F', G' \in R$ also satisfy $f * G' - g * F' = q$. Then there is an element $c \in R$ so that $F' = F + c * f$ and $G' = G + c * g$.*

Proof. (a) It suffices to check that the following change-of-basis matrix B has coefficients in R and has determinant equal to a unit in R:

$$B = \begin{pmatrix} 1 & h \\ 0 & q \end{pmatrix} \begin{pmatrix} f & g \\ F & G \end{pmatrix}^{-1} = \frac{1}{q} \begin{pmatrix} 1 & h \\ 0 & q \end{pmatrix} \begin{pmatrix} G & -g \\ -F & f \end{pmatrix} = \begin{pmatrix} (G - F * h)/q & (-g + f * h)/q \\ -F & f \end{pmatrix}$$

The congruence $h \equiv f^{-1} * g \pmod q$ ensures that $(-g + f * h)/q \in R$, and the equation $f * G - g * F = q$ implies that $F^{-1} * G \equiv f^{-1} * g \pmod q$, so we also get that $(G - F * h)/q \in R$. This proves that B has coefficients in R, and one easily calculates that $\det(B) = (f * G - F * g)/q = 1$, so B^{-1} also has coefficients in R. Therefore $\{(1,h),(0,q)\}$ and $\{(f,g),(F,G)\}$ generate the same R-module.
(b) First observe that $F' * G \equiv F * G' \pmod q$, since $F^{-1} * G = F'^{-1} * G' \equiv f^{-1} * g \equiv h \pmod q$. Set $c = (F' * G - G' * F)/q$. Then

$$F + c * f = F + \frac{F' * G * f - G' * F * f}{q} = F + \frac{F' * (q + g * F) - F * (q + g * F')}{q} = F'.$$

Similarly, $G + c * g = G'$, which completes the proof of the theorem.

We next give the proof of Lemma 1.

Proof. Let X be a random variable uniformly distributed in $[-B/2, B/2]$. Then

$$\mu(Y) = N\mu(X^2) = \frac{N}{B}\int_{-B/2}^{B/2} x^2\,dx = \frac{NB^2}{12}.$$

Similarly,

$$\begin{aligned}\sigma^2(Y) &= N\sigma^2(X^2) = N\bigl(\mu(X^4) - \mu(X^2)^2\bigr)\\ &= \frac{N}{B}\int_{-B/2}^{B/2} x^4\,dx - N\left(\frac{1}{B}\int_{-B/2}^{B/2} x^2\,dx\right)^2 = \frac{NB^4}{180}.\end{aligned}$$

To see why it is sufficient to just reduce the first coordinate of (F, G) by (f, g) in key generation, one should observe that $f * G - g * F = q$, so

$$\left\|G * g^{-1} - F * f^{-1}\right\| = \left\|q(fg)^{-1}\right\| \approx \frac{q}{\|f\| \cdot \|g\|}$$

For the parameter set $(N, q, d) = (251, 128, 72)$ this quantity is equal to 2.49, which means that on average, corresponding coefficients of $G * g^{-1}$ and $F * f^{-1}$ differ by only 0.157. Therefore, one obtains almost the same result whether one rounds $G * g^{-1}$ or $F * f^{-1}$.

In practice however one obtains a smaller result by treating the two components together. This corresponds to the standard lattice paradigm of multiplying a non-square basis by its transpose in order to be able to perform Babai's inverting and rounding technique. In this context the optimal k to reduce $\|(F, G) - k * (f, g)\|$ is

$$k = \left\lfloor \frac{\bar{f} * F + \bar{g} * G}{f * \bar{f} + g * \bar{g}} \right\rceil,$$

where $\bar{f}(x) = f(1/x) \bmod X^N - 1$ and similarly for $\bar{g}$.

B Security against Forgeries

As mentioned in section 6 a forger could use lattice reduction to aid a preselection process. Specifically, he could preselect somewhat fewer than N coordinates and use lattice reduction techniques to find the remaining coordinates. Thus suppose that a forger preselects αN coordinates of s and t for some choice of $0 \le \alpha \le 1$. He then reduces a lattice of dimension $(2 - \alpha)N$ and determinant $q^{N(1+\alpha)}$ to make the remaining $(1 - \alpha)N$ coordinates as small as possible.[3] As α increases, the fundamental ratio

$$\frac{\mathcal{N}}{\sqrt{\frac{(2-\alpha)N}{2\pi e}} \cdot q^{(1+\alpha)/(2-\alpha)}}$$

[3] Notice that $\alpha = 0$ corresponds to pure lattice reduction and $\alpha = 1$ corresponds to pure exhaustive search.

decreases, and when it passes below 1, the Gaussian heuristic says that it is very unlikely for any solutions to exist.

The case $\mathcal{N} = 310$ gives a cut off point of $\alpha = 0.3835$, which corresponds to a lattice of dimension 407. Thus a lattice reduction attack cannot hope to be reduced below dimension 407 (down from 502). At dimension 502, with $\alpha = 0$, experiments suggest a breaking time of greater than 10^{80} MIPS years. Further, as the dimension is reduced towards 407, the advantage gained from the reduction in dimension is offset by the decrease in the Gaussian ratio, causing predicted breaking time to increase.

C Further Transcript Analysis Experiments

In the particular case of NTRUSIGN without perturbations ($B = 0$), the signature description given in Section 7.1 translates into a collection of pairs of polynomials

$$\epsilon_1 * f + \epsilon_2 * F, \quad \epsilon_1 * g + \epsilon_2 * G$$

where the coefficients of ϵ_1 and ϵ_2 are more-or-less randomly distributed in the interval $[-1/2, 1/2]$.

Taking a simple average of a collection of signatures is not useful, since the average will be zero (or some other simple expression that reveals no useful information). However, there are other ways to take averages that introduce higher moments. The subject of moment averaging attacks was discussed in [12–14], and briefly mentioned in Section 7.1. Here we give more details.

Our tool here is the *reversal* $\bar{c}(X) := c(X^{-1})$ of a polynomial $c(X)$. The product $\hat{c}(X) = c(X) * c(X^{-1})$ of a polynomial with its reversal will often have a nontrivial average. The coefficients of $\hat{c}(X)$ involve products $c_i c_{i+k}$, so the polynomial $\hat{c}(X)$ is known as a *second moment polynomial.* Similarly, its square $\hat{c}(X)^2$ is a *fourth moment polynomial.*

First, we look at the average of the second moment polynomials $\widehat{s}$ from a transcript of signatures s. These are equal to

$$\begin{aligned}\widehat{s} &= (\epsilon_1 * f + \epsilon_2 * F) * (\bar{\epsilon}_1 * \bar{f} + \bar{\epsilon}_2 * \bar{F}) \\ &= \hat{\epsilon_1} * \hat{f} + \hat{\epsilon_2} * \hat{F} + \epsilon_1 * \bar{\epsilon_2} * f * \bar{F} + \epsilon_2 * \bar{\epsilon_1} * \bar{f} * F.\end{aligned}$$

As the number of signatures in the transcript goes to infinity, $\hat{\epsilon_1}$ and $\hat{\epsilon_2}$ (essentially) approach constants, and the cross terms (essentially) average out to zero. Hence by averaging the second moment polynomials over a sufficiently long transcript, an attacker may be able to recover the quantity

$$\hat{f} + \hat{F} = f * \bar{f} + F * \bar{F},$$

along with the remaining entries in the Gram matrix described in Section 7.1. Experiments indicate that in order to reconstruct this value, it is necessary to average a transcript consisting of on the order of 10,000 signatures.

The security of NTRUSIGN with no perturbations and transcripts of moderate length thus depends on the Gram matrix problem being hard. If an attacker cannot solve the Gram matrix problem efficiently, they need a longer signature transcript – as observed by [7] (and earlier, in a different context, [13]), a transcript long enough to yield accurate limiting averages of the fourth power moments should contain enough information to compromise the private key. In fact, an average of $\hat{s}^2$ eventually converges to a linear combination of the three quantities $\hat{f}^2$, $\hat{F}^2$, and $\hat{f} * \hat{F}$. If this limiting value can be determined sufficiently accurately, then it can be combined with the second moment information to recover $\hat{f}$. Finally, the attacker would apply a method of Gentry and Szydlo [7] to $\hat{f}$ and $\hat{f} * h$ to recover f in polynomial time.

Note that the averages required for this attack to proceed will converge quite slowly. It has previously been noted [13], and extensive experiments in the present case have confirmed, that a practical attack would require more than 100 million signatures – possibly much more, if an attacker cannot solve an N-dimensional square-rooting problem described in [11].

We would like to thank Craig Gentry and Mike Szydlo for emphasising the importance of the fourth moment analysis. They also suggested some possible variations on second moment attacks by restricting to subtranscripts where the norms (or coefficients) of signatures meet certain boundary conditions. We have investigated this approach; it appears to require transcripts of at least similar length to fourth moment based attacks.

D Hash Function Considerations

When signing, we map a digital document D to a message representative $(0, m)$. This mapping is actually a two stage process. First a standard secure hash function H_1 is applied to D to give β-bit output $H_1(D)$. Next a (public) function

$$H_2 : (\mathbb{Z}/2\mathbb{Z})^\beta \longrightarrow (\mathbb{Z}/q\mathbb{Z})^N$$

is applied to $H_1(D)$ to yield the message digest $m = H_2(H_1(D))$. We require that the mapping H_2 is "reasonably" uniform into the the set of q^{2N} possible ms. For example, one possible instantiation of H_1, H_2 for $N = 251$, $q = 128$ is given in [5]. Here H_1 is SHA-1. H_2 is defined by taking

$$D' \equiv \text{SHA-1}(H_1(D) \| 0) \,\|\, \text{SHA-1}(H_1(D) \| 1) \,\|\, \ldots,$$

where D' is at least N bytes long. Each coefficient of m is then generated by taking the low-order $\log_2(q)$ bits of the corresponding byte of D'.

We identify two potential attacks related to this mapping. First, if two digital documents D and D' map to two message representatives m and m' which are very close together, and a signer can be induced to sign both of them, then there the difference of the signatures might be a small element of the underlying lattice, and could reveal the private key [16]. Such a pair of documents would endanger all NTRUSign implementations using a common mapping H.

Alternatively, an attacker can attempt direct forgery by generating lattice points of the form $(u, uh \bmod q)$ for arbitrary (small) u. In this case, the attacker generates a large set $\mathcal{L}$ of lattice points and a large set $\mathcal{M}$ of message representatives, and checks to see if one of the lattice points in $\mathcal{L}$ signs one of the message representatives in $\mathcal{M}$.

Both of these problems are analyzed in detail in [11]. For the key recovery attack, we find that for $(N, q) = (251, 128)$, an attacker will have to generate 2^{205} distinct messages before there is a 50% chance of finding two message representatives within $B_{\text{coll}} = 10$ of each other.

For the collision attack, we note that each lattice point generated signs all points within a radius $\mathcal{N}$. An attacker who generates n lattice points can sign at most a fraction $n \cdot C$ of all potential messages, where $C = \frac{\pi^{N/2}}{\Gamma(1+N/2)} \left(\frac{\mathcal{N}}{q}\right)^N$. A standard birthday paradox type argument shows that if the attacker generates n points and k messages, her chance of getting a collision is 50% when $kn \approx 1/C$. Thus, for the parameters $(N, q, \mathcal{N}) = (251, 128, 310)$, an attacker will have to generate 2^{80} lattice points and 2^{80} message representatives to have a 50% chance of forging a signature by this method.

E Performance

Table 1 compares the performance of NTRUSign, ECDSA and RSA on an 800 MHz Pentium machine. RSA times are from [4] and were obtained on the same machine as the NTRUSign times. ECDSA times are from [9] and were scaled relative to the clock speed of the machine used in that paper. NTRUSign figures are obtained in the standard lattice with no perturbations; the figures for transpose lattice with one perturbation will, however, be comparable.

	NTRUSign-251	ECDSA-163	RSA-1024
Keygen (μs)	180,000	1424	500,000
Sign (μs)	500	1424	9090
Verify (μs)	303	2183	781

Table 1. Comparison of NTRUSign, ECDSA, RSA