NTRU Cryptosystems Technical Report

Report # 015, Version 2
Title: Reaction Attacks Against the NTRU Public Key Cryptosystem
Author: Jeffrey Hoffstein and Joseph H. Silverman
Release Date: v1. August 1999; v2. June 2000

Abstract. Hall, Goldberg, and Schneier have proposed a *Reaction Attack* against several public key cryptosystems based on lattice problems, including the McEliece, Atjai-Dwork, and Goldreich-Goldwasser-Halevi cryptosystems. In this note we describe a Reaction Attack on the NTRU public key cryptosystem and describe a number of ways in which such attacks may be easily detected and thwarted.

Note for Technical Report #015 Version 2. The countermeasures described in this report are largely superceded by NTRU Technical Report #016, "Protecting NTRU Against Chosen Ciphertext and Reaction Attacks." The report #016 describes a padding technique of Fujisaki and Okamoto that protects against reaction attacks and also against the adaptive chosen ciphertext attacks described in NTRU Technical Report #007 and in the paper "A chosen-ciphertext attack against NTRU," E. Jaulmes and A. Joux, *Proceedings of CRYPTO 2000*, Lecture Notes in Computer Science, Springer-Verlag.

In a recent preprint, Hall, Goldberg, and Schneier [1] have proposed an attack against several public key cryptosystems based on lattice problems. They call their attack a *Reaction Attack*. In their paper they describe how to mount a Reaction Attack on a number of cryptosystems, including those suggested by McEliece, Atjai-Dwork, and Goldreich-Goldwasser-Halevi. Since the NTRU public key cryptosystem is also based on an underlying lattice problem, it is natural that a Reaction Attack should exist for NTRU. In this note we explain how an NTRU Reaction Attack would work and we describe a number of ways in which a user of the NTRU public key cryptosystem can thwart such attacks. Our main point will be that before such an attack can begin to yield useful information, it must deform the encrypted message sufficiently that the attack can be detected by the NTRU user. We also note that such an attack can only hope to succeed if a single private key is used for the decryption of a large number of messages. This is a situation that will rarely arise in an NTRU based system, since the ease of NTRU key generation key means that keys will be changed frequently.

Section 1. A Reaction Attack on the NTRU PKCS — Summary

In this section we will briefly describe the main ideas for a Reaction Attack on the NTRU PKCS. For complete details, see section 3 of this note. The idea underlying a Reaction Attack is for the attacker to produce a sequence of encrypted messages e, each of which has a significant probability of decrypting into a valid message and also a significant probability of decrypting into an invalid message. It is assumed that the attacker is able to distinguish which messages have valid decryptions and which ones do not. This may be possible because an error message is generated when an invalid message is received, or it may be possible via a timing attack in which the attacker measures how long it takes a message to be decrypted.

In the Reaction Attacks described in [1], and in the Reaction Attack on the NTRU PKCS described in section 3 below, one starts with a valid encrypted message e and creates small modifications $e' = e + \epsilon$. The attacker makes the modification ϵ larger and larger until the modified message e' causes a decryption error. By comparing the ϵ's that cause decryption errors to those that do not, the attacker gains information about either the plaintext message m underlying e or about the private key used to encrypt m.

We now briefly describe the Reaction Attack on the NTRU PKCS. For basic information and notation for the NTRU PKCS, see [2] or the NTRU web site ⟨`www.ntru.com`⟩. For details of the Reaction Attack, see section 3. An encrypted NTRU message has the form $e \equiv \phi h + m \pmod q$. The smallest modification that an attacker can make to e and still have it sometimes decrypt correctly is to replace e with $e' = e + npX^i$ for some $0 \le i < N$ and some $n \ge 1$. This will cause a decryption failure (a so-called wrap or gap failure) if some coefficient of the intermediate decryption polynomial $a = p\phi g + mf$ is within np of $q/2$ and if $X^i f$ has a corresponding $+1$ coefficent. The important point to observe is that for the correct choice of n, the i's which cause decryption failure for $e + npX^i$ will reflect (with some shifting and possible duplication) the i's for which the private key f has a term of the form $+X^i$. Thus the attacker potentially gains information about the $+1$ bits in the private key f. Similarly, using negative values for n may give information about the -1 bits of f.

Note that the Reaction Attack does not compromise the hard mathematical problem underlying the NTRU PKCS, which is the problem of finding the shortest vector in a lattice of high dimension. None-the-less, it is a potentially serious attack for implementations of the NTRU PKCS in hardware or software, so measures such as those described in section 2 should be used to avoid such attacks.

Section 2. Countermeasures to Reaction Attacks on the NTRU PKCS

There are a number of methods that can be used to thwart Reaction Attacks against the NTRU PKCS. Which one is best will depend on the particular implementation environment.

Self-Referential Padding Solution

See NTRU Technical Report #016 "Protecting NTRU Against Chosen Ciphertext and Reaction Attacks" for an easy and efficient method of Fujisaki and Okamoto that simultaneously protects NTRU from reaction attacks and adaptive chosen ciphertext attacks. This method largely supercedes the other methods described in this section, although there may be some situations in which the alternative methods are useful.

Disposible Key Solution

There is no chance for a Reaction Attack to succeed if any given public/private key pair is used for only one message, or at most a few messages. This solution would not be useful for public key cryptosystems such as RSA or ECC, because key creation is quite time consuming. However, NTRU key creation is so fast that use of disposible key pairs is a very reasonable option. Here is how the Disposible Key Solution works in practice, say for transmitting credit card numbers (or exchanging a private key to be used for a symmetric cipher) over the internet.

Suppose a consumer Bob wants to send his credit card number to buy an item from Alice's web site. When Bob makes his initial request, Alice creates a public/private key pair and sends the public key to Bob. She also signs the public key with a digital certificate proving that the public key was sent by Alice. Bob checks the digital certificate, normally by using information from a trusted third party. He then uses the public key to encrypt his credit card information and sends it back to Alice.

Since each public/private key pair is used for only a single message exchange, there is no possibility of mounting a Reaction Attack, or indeed of mounting any sort of attack which depends on monitoring differences in the decryption process for different messages.

Remark. Note that even if Alice were to use the same public/private key for all transactions, Bob still needs to check that the key really comes from Alice. So regardless of whether or not Alice uses disposible keys, Bob must verify some sort of digital certificate or signature using information from a trusted third party.

Decryption Failure Tracking Solution

When NTRU parameters are chosen properly, very few correctly encrypted messages will cause wrap or gap decryption failures, say no more than 1 wrap failure in 10^5 messages and no more than 1 gap failure in 10^{10} messages. Thus it is very easy for Alice, the private key owner, to tell when she is being subjected to a Reaction Attack. If she keeps track of the number of wrap failure messages received, she can simply discard the current key and replace it with a new public/private key

pair as soon as the number of wrap/gap failures exceeds (say) 5. If not under attack, this will require changing keys after approximately half a million messages, not an onerous burden; while an attacker will gain little information from 5 decryption failures.

For even greater security, Alice might change keys after every wrap/gap failure. In any case, given the ease of NTRU key creation, for added security from any type of interactive attack, she might also want to automatically change keys after every (say) 1000 messages.

These solutions suppose that it acceptable to have Alice change keys under certain circumstances. This is not a problem in terms of speed or processor power, because NTRU key creation is extremely fast and easy, but there may be situtations in which Alice wants to use a single public/private key pair for a considerable period of time. In this case the cheapest solution would be for Alice to keep a running total of the number of wrap/gap failures associated to a given user. If this exeeds a plausable number, say 5, Alice can simply stop responding to messages from that user. One may ask why it is probable that the messages will all come from a single user. This is because, as observed below in Section 3, a Reaction Attack depends on repeated deformations of a single message. The simplest approach for an attacker is to send the messages himself, revealing his identity after a short while. The next simplest approach for an attacker is to intercept a message from a third party and then resend deformations of it a multitude of times. This will at least alert Alice that an attack is underway, though the attacker may remain annonymous. Finally, in a rather extreme variant, the attacker could launch deformations of the same message from a variety of forged email addresses. Once again Alice can become aware that an attack is underway by the number of failures. In this last case it is probably most cost effective to have the NTRU program simply notify Alice of the attack. However,it is still possible, if desired to have a completely automated parry to even the last form of reaction attack, as the following two comments will outline.

Induced Randomness Solution

It is possible to guard against Reaction Attacks by introducing some extra randomness into the decryption process. This is similar to the sort of the methods that are used to guard against timing attacks on RSA. We also note that induced randomness may also defeat, or at least impede, the Reaction Attacks on the other systems described in [1]. This form of defense can ultimately be averaged out by a determined attacker, but the number of message decryptions required would be considerably larger.

Recall that the attacker in a Reaction Attack takes a valid NTRU encrypted message e and introduces a small variation $e' = e + \epsilon$, where the change ϵ might have the form $\epsilon = npX^i$. The solution proposed here is that before beginning the decryption process, Alice replaces the encrypted message e that she receives by the altered message

$$E = e \pm pX^{j_1} \pm \cdots \pm pX^{j_r}.$$

Here r is a fixed value (say $r = 5$ or $r = 10$), and Alice chooses the exponents $0 \le j_1 < \cdots < j_r < N$ and the $\pm$-signs randomly. She then proceeds to decrypt E. If e is the encryption of a valid message m, then even with the alterations introduced by Alice, there is a very high probability that E will decrypt properly to yield the message m.

Now consider the situation from the point of view of the attacker. He sends Alice the altered message $e' = e + npX^i$, so she will decrypt the message $E + npX^i$. The intermediate stage in the decryption process at which wrap/gap failures occur is then given by the polynomial

$$p\phi g + fm + pf(nX^i \pm X^{j_1} \pm \cdots \pm X^{j_r}).$$

(See section 3 for more details.) Each time the attacker sends an altered message e', the exponents $j_1, \ldots, j_r$ and the $\pm$-signs will change. Since they are each multiplied by the polynomial f, this means that r random shifts of the polynomial f are being added or subtracted from the total. The occurance of wrap/gap failure on the edge of where it occurs is thus tremendously randomized. It is, of course, conceivable that after a very large number of trials the attacker might be able to gather some statistical information about f, but we observe that even for a minimal value of r, say for $r = 5$, the number of possiblities for the induced randomness is 2^{35}, 2^{38}, and 2^{43} for NTRU with parameters $N = 167$, $N = 263$, and $N = 503$ respectively.

Coefficient distribution analysis

We will close this section with a description of one final method that Alice can use to passively ignore reaction attacks. As described in detail below, a reaction attack requires the "inflating" of the partial decryption of a message. This means that the number of coefficients of the polynomial

$$p\phi g + fm + pf(nX^i \pm X^{j_1} \pm \cdots \pm X^{j_r}).$$

falling into ranges close to $q/2$ and $-q/2$ will be larger than expected. This will *always* be the case when a reaction attack takes place. Alice can test for this inflating of the above intermediate polynomial by counting the number of coefficients in the ranges $[-q/2, -q/2 + D]$ and $[q/2 - D, q/2]$ (for an appropriate value of D) and simply not responding to any inflated polynomials. It is easy to give a precise formulation of this test in the unlikely event that Alice requires it.

Section 3. A Reaction Attack on the NTRU PKCS — Detail

In this section we will give a detailed description of a Reaction Attack on the NTRU PKCS. We begin with a quote from section 4 of [1] which describes the idea underlying such attacks.

> The success of our attacks rely on a common weakness of the PKCS we examined: given a ciphertext C, an attacker can produce a second ciphertext C' which has non-negligible probabilities of decoding to the same plaintext and to a different plaintext. ... Each of these systems (McEliece, Atjai-Dwork, Goldreich-Goldwasser-Halevi) could be considered a closest point system. ... In all of these systems one could consider the class of ciphertexts corresponding to a particular plaintext to be a sphere surrounding a point in the respective space. By examining ciphertexts which are close to each other in the space (but possibly in different classes) we can determine the boundaries of this sphere and hence the center of the sphere.

The Reaction Attack on the NTRU PKCS uses this idea of producing a modified ciphertext C' which has non-negligible probabilities of decoding to the same plaintext and to a different plaintext. However, the notion of finding a sphere is not quite appropriate, so instead a shift-and-compare method is used to reconstruct the key.

The basic idea is as follows. Let p, q, N be the usual NTRU parameters, let h, f be a public/private NTRU key pair, and let

$$e = \phi h + m \pmod{q}$$

be an encrypted NTRU message. Further let

$$a = p\phi g + mf$$

be the usual intermediate step in the decryption process. (See [2] for a description of NTRU parameters and the encryption and decryption processes.) The NTRU decryption process proceeds smoothly provided that the largest coefficient of a, say a_μ, satisfies $a_\mu \leq q/2$ and the smallest coefficient of a, say a_ν, satisfies $a_\nu > -q/2$. We say that *wrap failure* has occurred if either $a_\mu > q/2$ or $a_\nu \leq -q/2$. This does not mean that the message cannot be decrypted, but it does mean that some additional computation is necessary, and it might be possible for an attacker to detect this additional time and deduce that wrapping failure has occurred. (There is also a less common occurance called *gap failure* in which the difference $a_\mu - a_\nu > q$. If gap failure occurs, the decryption process takes even longer.) We are going to describe a Reaction Attack on the NTRU PKCS based on the assumption that an attacker is able to detect when a wrapping failure occurs. A similar, but somewhat more complicated, attack is possible if the attacker is able to detect when gap failures occur.

Let e be a valid message that decodes correctly, so $a_\mu \le q/2$ and $a_\nu > -q/2$. We will also assume that the polynomial a associated to the encrypted message e has the following two properties:

(i) Aside from a_μ, every coefficient of a satisfies $a_i \le a_\mu - p$.

(ii) $a_\mu + a_\nu > p$. Notice this inequality can be rewritten as $q/2 - a_\mu + p < a_\nu + q/2$, so it says that a_μ is p closer to $q/2$ than a_ν is to $-q/2$.

Since generally $p = 2$ or 3, there is a significant probability that a randomly chosen message will have these properties. Further, even if they are not true, repetition of the attack for several messages e will still give a significant amount of information about f, although it will be somewhat harder to piece that information together.

Let np be the smallest multiple of p such that $np + a_\mu > q/2$. In other words, n is chosen to satisfy

(iii) $a_\mu + (n-1)p \le q/2 < a_\mu + np$.

Of course, the attacker doesn't actually know the value of n, so he will execute the algorithm with $n = 1, 2, \ldots$ until he finds an n that works. Or, if he is allowed only a limited number of decryptions, he can guess a likely value for n.

The attacker transmits the encrypted messages

$$e + npX^i \qquad \text{for } i = 0, 1, 2, \ldots, N-1$$

and monitors which ones cause wrap failure. Suppose that for $i = i_1, \ldots, i_s$ the messages $e + npX^i$ cause wrap failure, but that they are decrypted without wrap failure for the other values of i. The attacker forms the polynomial

$$F = X^{N-i_1} + X^{N-i_2} + \cdots + X^{N-i_s}.$$

(If some $i_j = 0$, note that $X^N = 1$.) Recall that an NTRU private key f has a certain number of coefficients equal to 1, a certain number of coefficients equal to -1, and the rest equal to 0. I claim that the polynomial F constructed by the attacker is a shifted version of the "1-coefficients" in f. More precisely,

Claim: The polynomial $f - X^\mu F$ has all of its coefficients equal to 0 and -1.

The proof of this claim is easy. When attempting to decrypt $e + npX^i$, wrap failure will occur if the polynomial

$$a' := a + npX^i f$$

has a coefficient greater than $q/2$ or less than or equal to $-q/2$. The j^{th} coefficient of a' is given by

$$a'_j = a_j + npf_{j-i} = \begin{cases} a_j - np & \text{if } f_{j-i} = -1 \\ a_j & \text{if } f_{j-i} = 0 \\ a_j + np & \text{if } f_{j-i} = 1 \end{cases}$$

where the subscripts on f are taken modulo N. We first observe that

$$\begin{aligned} a'_j &\geq a_j - np \\ &\geq a_\nu - np \\ &> p - a_\mu - np \quad \text{from (ii)} \\ &\geq -q/2 \quad \text{from (iii)}. \end{aligned}$$

So a' does not have wrapping failure from below. Next we note that if $j \neq \mu$, then

$$\begin{aligned} a'_j &\leq a_j + np \\ &\leq a_\mu - p + np \quad \text{from (i)} \\ &\leq q/2 \quad \text{from (iii)}. \end{aligned}$$

So the only coefficient of a' which might possibly cause wrapping failure is a'_μ. Indeed, since $a_\mu \leq q/2$ and $a_\mu + np > q/2$, we see from its definition that a'_μ causes wrapping failure if and only if $f_{\mu-i} = 1$. Thus the indices $i_1, \ldots, i_s$ determined above are exactly those indices i for which $f_{\mu-i} = 1$. It is immediate from this that the polynomial F defined above has the property that $X^\mu F$ is equal to the terms of the polynomial f that have coefficient equal to 1. This proves the Claim.

At this point the attacker has two options. He can use the information already gained to aid him in his search for f via other means (brute-force, lattice reduction, etc.). Or he can repeat the attack until he finds a message e whose related polynomial a has the following two properties: (i′) Aside from a_ν, every coefficient of a satisfies $a_i > a_\nu + p$; and (ii′) $a_\mu + a_\nu < -p$. Then the same algorithm will give him a polynomial G such that $-X^\nu G$ is equal to the terms of the polynomial f that have coefficient equal to -1. It is then a simple matter to check the N different polynomials $F - X^k G$, $0 \leq k < N$, until finding a shifted version of f which will function as a decryption key. (Note that the attacker doesn't know the value of μ or ν.)

We mention again that our assumption that e and a satisfy properties (i) and (ii) are designed to simplify our description of the Reaction Attack. In general, an arbitrary $e+npX^i$ will yield an a' which may have both upper and lower wrapping failures at one or more coefficients. But even in this general situation, by observing which i's lead to wrapping failure, one can gain significant statistical information concerning the distribution of 1's and -1's in the private key f. Thus a small multiple of N messages (e.g., between $3N$ and $10N$ messages) may well be enough to compromise the security of the key, and it is even possible that some information about the key could be gained in just a few messages.

The following pseudo-code algorithm describes how the Reaction Attack recovers the 1-bit part of the private key f assuming properties (i) and (ii). We leave for the reader the necessary alterations to recover the -1-bit part and to deal with the case that (i) or (ii) is not true.

Reaction Attack Against NTRU PKCS

Input: encrypted message e and public key h
Output: possible 1-bit part of private key f
Step 1: $n := 1$
Step 2: Loop:
Step 3: transmit $e + npX^i$ for $i = 0, 1, \ldots, N-1$
Step 4: if every $e + npX^i$, $0 \le i < N$, decrypts correctly
Step 5: $n := n + 1$
Step 6: goto Loop
Step 7: let $i_1, \ldots, i_s$ be the i's for which $e + npX^i$ causes decryption failure
Step 8: return $X^{i_1} + \cdots + X^{i_s}$

References

[1] C. Hall, I. Goldberg, B. Schneier, Reaction attacks against several public-key cryptosystems, preprint April 1999, available at www.counterpane.com

[2] J. Hoffstein, J. Pipher, J.H. Silverman, NTRU: A new high speed public key cryptosystem, Algorithmic Number Theory (ANTS III), Portland, OR, June 1998, Lecture Notes in Computer Science 1423, J.P. Buhler (ed.), Springer-Verlag, Berlin, 1998, 267–288

Comments and questions concerning this technical report should be addressed to

techsupport@ntru.com

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

www.ntru.com
