Security of Algebraic Variations on NTRU

Thomas Lee

December 6, 2018

1 Introduction

The goal of public key cryptography, according to [2], is for two people (we will call them Alice and Bob) who can only communicate by a channel that is monitored by an adversary (we will call her Eve) to exchange secret messages even if they have never met. To meet this goal, a cryptosystem should make it difficult for the adversary to decrypt messages. Hence, the security of the system is improved if the adversary's difficulty of decryption is increased. Quantum computers can decrease the level of security of a cryptosystem by utilizing extremely fast quantum algorithms. When quantum computers become widely available, cryptosystems that are currently used, such as RSA and classical and elliptic curve versions of Elgamal, will become outdated due to quantum algorithms; however, there is currently no polynomial time quantum algorithm to solve the hard lattice based problems relevant to lattice based cryptography [2]. Because of this, lattice based systems like NTRU (a public key cryptosystem created by the authors of [2], Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman) are receiving much attention. In the current study, we studied the security of variations on the NTRU public key cryptosystem.

1.1 NTRU

This section is to remind to reader of some of the relevant details of NTRU. A detailed discussion of the cryptosystem can be found in [2]. NTRU is based around three polynomial rings:

$$R = \frac{\mathbb{Z}[x]}{x^N - 1} \qquad R_p = \frac{(\mathbb{Z}/p\mathbb{Z})[x]}{x^N - 1} \qquad R_q = \frac{(\mathbb{Z}/q\mathbb{Z})[x]}{x^N - 1}$$

where N and p are prime and $gcd(N,q) = gcd(p,q) = 1$. NTRU also makes use of the ternary polynomial, $a(x) \in T(d_1, d_2)$, where $a(x)$ has d_1 coefficients equal to 1, d_2 coefficients equal to -1, and all other coefficients equal to 0. The cryptosystem has public parameters (N, p, q, d) where N, p, q are as stated, and d is used for the set of ternary polynomials described above.

1.1.1 Key Creation

Alice chooses the private key to be two polynomials, f and g, such that

$$f(x) \in T(d+1, d) \qquad g(x) \in T(d, d)$$

and $f(x)$ is invertible in both R_q and R_p. Once Alice finds a suitable $f(x)$, she computes the inverses in each ring to be $F_q(x)$ and $F_p(x)$. Then the public key is

$$h(x) = F_q(x) * g(x) \text{ in } R_q$$

1.1.2 Decryption Process

The following definition of the center-lift of a polynomial from [2] will be used in decryption.

Definition. Let $a'(x) \in R_q$. The *center-lift* of $a'(x)$ to R is the unique polynomial $a(x) \in R$ satisfying $a(x) \pmod{q} = a'(x)$ with coefficients between $-\frac{q}{2}$ and $\frac{q}{2}$.

Bob sends Alice an encrypted message, $e(x)$. Once she receives the encrypted message, she will decrypt as follows:

$$\begin{aligned} a'(x) &= f(x) * e(x) \quad \text{(mod q)} \\ a(x) &= \text{center-lift} \quad a'(x) \\ m(x) &= F_p(x) * a(x) \quad \text{(mod p)} \end{aligned}$$

where $m(x)$ is the message. If Eve wants to decrypt the message, she needs the private key, $f(x)$, and its inverse in R_p, F_p. After she finds the private key, Eve's decryption process is the same as Alice's. The best method is as follows. Once the public key $h(x)$ is published, Eve performs the LLL algorithm (first published in [3]) on the $2N \times 2N$ NTRU Matrix. Candidates for the coefficients of $f(x)$ will be the vectors of length N on the left side of the LLL basis. Let $\hat{f}_i(x)$ be the polynomial with coefficients from the i^{th} vector of length N returned by the LLL algorithm. Then for each i, Eve must compute

$$\begin{aligned} \hat{F}_{ip}(x) &= \hat{f_i}^{-1}(x) \quad \text{in } \mathrm{R_p} \\ a'(x) &= \hat{f}_i(x) * e(x) \quad \text{(mod q)} \\ a(x) &= \text{center-lift} \quad a'(x) \\ \hat{m}(x) &= \hat{F}_{ip}(x) * a(x) \quad \text{(mod p)} \end{aligned}$$

until $\hat{m}(x) = m(x)$.

2 Security of Variations on NTRU

The most common attack on NTRU is to find the private key using the LLL algorithm. Hence, the security of NTRU is improved if the algorithm takes a longer time to terminate or if the algorithm does not return the private key. We applied encryption to messages in NTRU to compare Eve's decryption process in the following four rings to compare security.

$$R_1 = \frac{\mathbb{Z}[x]}{x^N - 1} \qquad R_2 = \frac{\mathbb{Z}[x]}{x^N + 1} \qquad R_3 = \frac{\mathbb{Z}[x]}{x^{2^n} + 1} \qquad R_4 = \frac{\mathbb{Z}[x]}{x^N + x + 1}$$

where N is prime. Security was measured in each ring by measuring the average time to termination of the LLL algorithm, and the percent of trials that LLL returned the private key. The goal was to determine if some rings yield better security.

2.1 Method

We generated random keys and messages at a variety of public parameter combinations for each trial, with in depth analysis at $N = 31$ and $2^n = 32$. In each trial, the variation of LLL called LLL with deep insertions was used. Details can be found in [4]. Although this variant takes longer to terminate, it returns much better bases, which gives Eve a better chance at successfully decrypting a message. Trials were run as follows:

1. Choose public parameters and generate random private key and compute the public key
2. Generate a random message and encrypt
3. Create the NTRU matrix with the public key
4. Perform LLL with deep insertions on the NTRU matrix and record running time
5. Attempt decryption with every candidate for the private key returned by LLL with deep insertions
6. Determine if LLL with deep insertions returned the private key
7. Average the LLL with deep insertions running times for each ring
8. Calculate percent of time that LLL with deep insertions returned the private key

2.2 Percent LLL with Deep Insertions Returned Private Key

At $N = 31$ and $2^n = 32$, Figure 1 compares the percent of successful returns of the private key by LLL with deep insertions in each ring. Let Q_i be the percent of private keys returned in R_i. We performed statistical analysis at the 95% confidence level ($\alpha = 0.05$) with sample size = 263. To determine statistically significant differences between the rings, we used the P-value criteria, with P-value being the smallest significance level, α, at which the null hypothesis can be rejected [1]. We concluded the following: $Q_4 < Q_3$ (P = 0.01), $Q_4 < Q_1$ (P = 2.0 $\cdot 10^{-8}$), $Q_3 < Q_1$ (P = 0.02), $Q_2 < Q_1$ (P = 7.4 $\cdot 10^{-4}$). Although it is unclear which ring LLL with deep insertions is least likely to return the private key, it is less likely to return the private key in rings R_2, R_3, R_4 than in the original R_1. Hence, each of the variant rings could yield improved security for NTRU since Eve is less likely to find the private key. However, the message was successfully decrypted in every trial in every ring whether or not the vector used for decryption was equal to the coefficients of $x^i * f(x)$ for some i. We will discuss this in more detail later. Since messages can always be decrypted in each ring, there does not appear to be an advantage to using R_2, R_3, or R_4 despite the lower return rates of the private key.

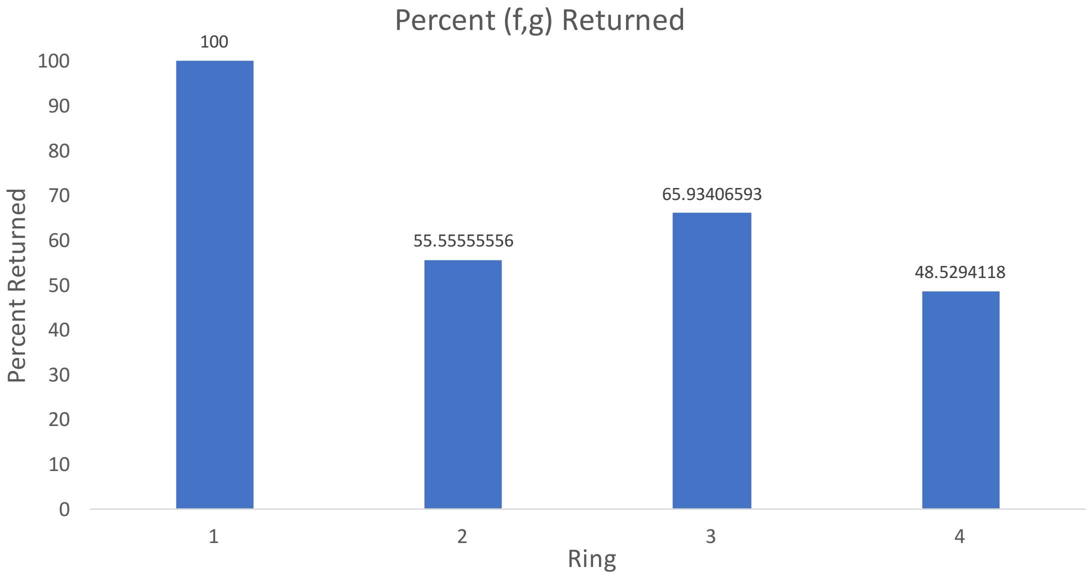

Figure 1: percent of time that LLL with deep insertions returned the private key, (f,g)

2.3 LLL with Deep Insertions Time to Termination

At $N = 31$ and $2^n = 32$, Figure 2 compares the average time that LLL with deep insertions took to terminate in each of the rings. Let T_i be the average time that LLL with deep insertions took to terminate in R_i. After performing statistical analysis at the 95% confidence level ($\alpha = 0.05$) with sample size = 263, we concluded the following: $T_4 < T_1$ (P = $1.7{\cdot}10^{-8}$), $T_4 < T_2$ (P = $1.83 \cdot 10^{-33}$), $T_4 < T_3$ (P = $2.23 \cdot 10^{-70}$), $T_1 < T_2$ (P = $5.5 \cdot 10^{-6}$), $T_1 < T_3$ (P = $1.5{\cdot}10^{-15}$), $T_2 < T_3$ (P = $1.7 \cdot 10^{-4}$). From these results, it is clear that rings R_2 and R_3 yield better security due to the algorithm taking a longer time to terminate. Alice and Bob have increased security in either of these rings compared the the original R_1 because Eve's best attack would take a longer time to complete. To show how much better the security is, we measured the average time to termination at several dimensions and fitted an exponential curve for each ring, which is displayed in Figure 3. At dimension 200, the average time to termination in each ring is as follows:

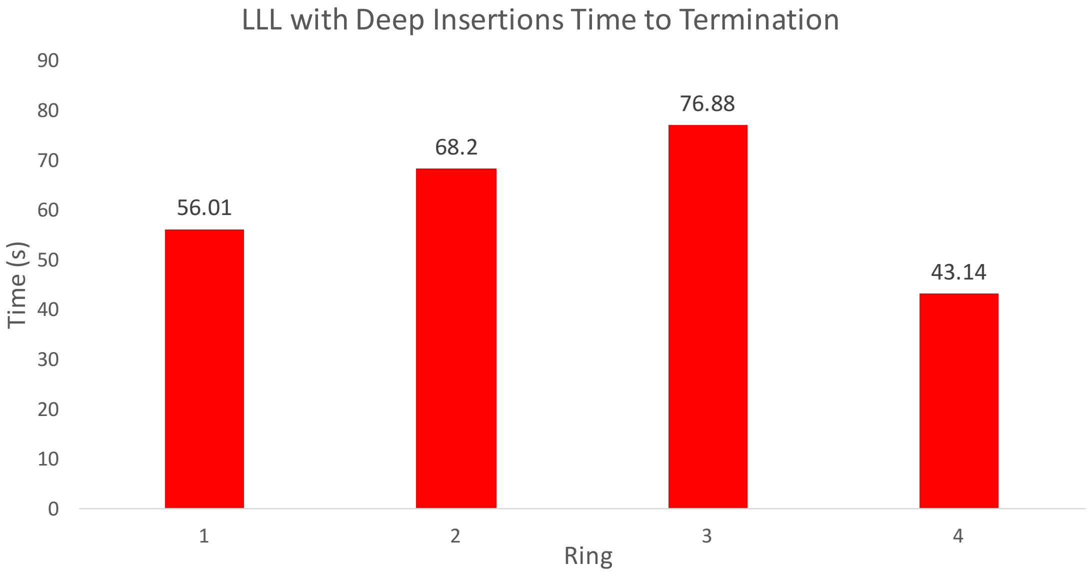

Figure 2: compares how long LLL with deep insertions took to terminate on average in each ring

$$\begin{aligned} R_1 &: 61 \quad \text{days} \\ R_2 &: 90 \quad \text{days} \\ R_3 &: 109 \quad \text{days} \\ R_4 &: 46 \quad \text{days} \end{aligned}$$

This makes the advantages of using R_2 or R_3 over the original R_1 more clear. Alternatively, one could set a security level (e.g. set the average time to termination of the algorithm to 100 years) and find the dimension needed in each ring. Lower dimension implies fewer problems with storage space of keys, etc. For the algorithm to terminate at about 100 years, the following dimensions are needed for each ring:

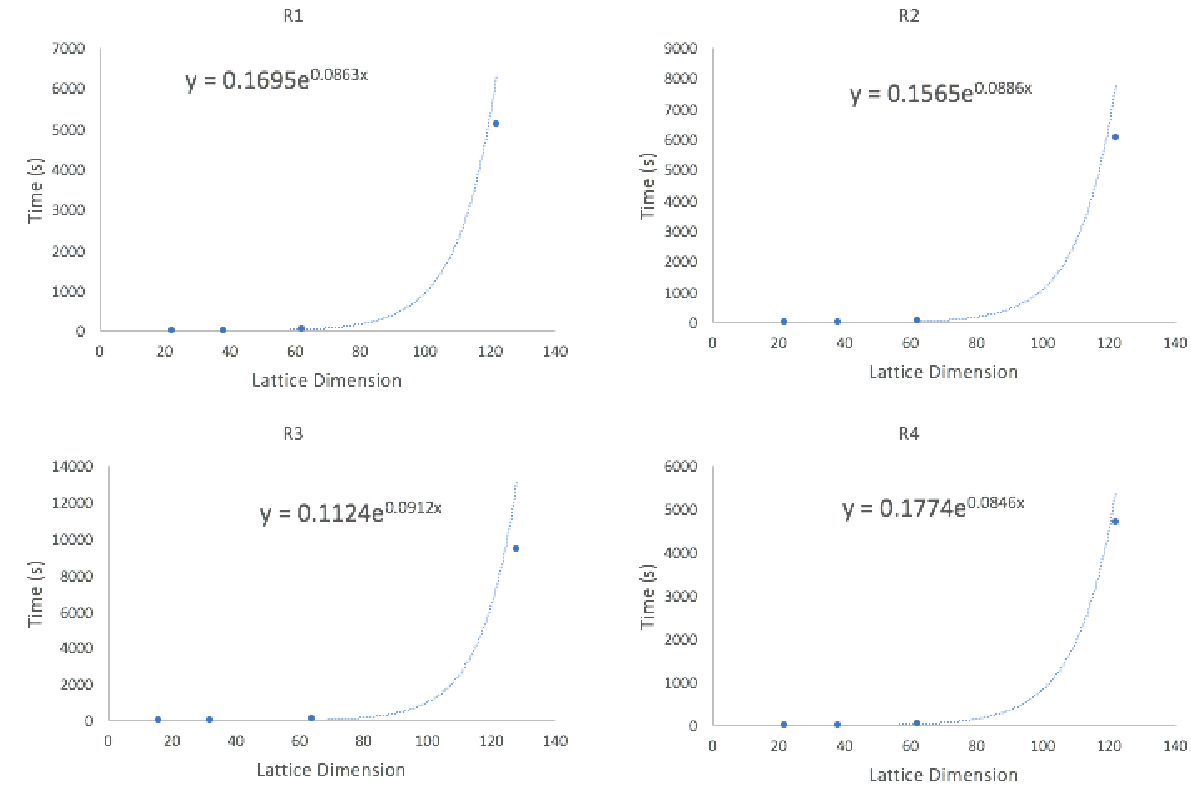

Figure 3: time to termination of LLL with deep insertions curves for each ring

$$R_1 : 274$$
$$R_2 : 268$$
$$R_3 : 264$$
$$R_4 : 279$$

Although these differences are small, if one wanted greater than 100 years security, the storage saved would be much greater.

2.4 Conclusion: Which Ring Should be Used?

Answering the question of which ring to use comes down to answering which ring improves security the most. Although the fact that LLL with deep insertions returns the private key less frequently in some rings than in the original R_1, the fact that the message can still be decrypted implies that security is not improved in R_2, R_3, or R_4 with respect to return rate of the private key. Hence, we base improved security solely on increasing the time to termination of LLL with deep insertions. At first glance, it appears that R_3 would improve security the most due to having the longest average time to termination of LLL with deep insertions. However, due to a comment by Hoffstein et al., there may be security issues in R_3 due to $x^{2^n} + 1$ being a polynomial of nonprime degree [2]. Due to R_2 increasing the time to termination of LLL with deep insertions and not having problems with the degree of the polynomial as in R_3, we conclude that using R_2 increases the security of NTRU. It should be noted that if there are security advantages to a lower rate of return of the private key, there was no statistically significant difference between R_2 and R_4, which had the lowest rate of return of the private key. Thus, if there are unknown security advantages to a lower return rate of the private key, there is not enough evidence to claim that R_4 would yield better security than R_2. Therefore, in either case, R_2 should be used to increase the security of the NTRU cryptosystem.

3 Limitations

In depth analysis was performed at $N = 31$ and $2^n = 32$. Other values were used only to obtain average times for the exponential plots in Figure 3. Statistical significance testing was only performed at $N = 31$ and $2^n = 32$, so it is possible that one might find different time or return rate differences between the polynomial rings at different values for N and 2^n.

All experiments were performed using the LLL with deep insertions variant of the LLL algorithm. Using better variants could yield better return rates for each ring. More importantly, if the classical LLL algorithm was used, time differences at very large dimensions may seem insignificant between the rings since the classical algorithm runs more quickly than its variants. Future studies should use different variants of the LLL algorithm since the adversary will be able to use all of them.

4 Future Research

Although there were no clear security advantages to a lower return rate of the private key in the four polynomial rings studied, this observation raises questions. Future work should study why some rings yield lower return rate of the private key from the LLL algorithm.

Let $\hat{f}$ be the closest vector to f that is returned by LLL (i.e. the vector that Eve will use for decryption). How far does $\hat{f}$ have to be from f for decryption to fail? That is, what is the minimum value for y where

$$\left\| f - \hat{f} \right\| = y$$

Is there a polynomial ring that can guarantee this value of y at least some of the time? The largest value for y observed in this study was $\sqrt{6}$ in R_4 when $N = 31$. When the largest difference between any pairs of coefficients was 1, this means that 6/31 coefficients of $\hat{f}(x)$ were different from $f(x)$, but the message could still be decrypted using $\hat{f}(x)$. This raises several questions. If vectors not equivalent to $x^i * f(x)$ for some i can be used for decryption purposes, is a brute force search for the private key as impractical as previously thought? How many possibilities are there for polynomials that can be used for successful decryption? Future work on NTRU should study the relationship between $\left\| f - \hat{f} \right\|$ and exactly when $\hat{f}$ can be used to successfully decrypt a message.

References

[1] Jay L Devore. *Probability and Statistics for Engineers and Scientists.* Cengage Learning, 2010.

[2] Jeffrey Hoffstein, Jill Pipher, and Joseph H Silverman. *An Introduction to Mathematical Cryptography.* Springer, 2014.

[3] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[4] Claus-Peter Schnorr and Martin Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical programming*, 66(1-3):181–199, 1994.