Enhancing the NTRU cryptosystem

Awnon K. Bhowmik
Department of Mathematics and Computer Science
CUNY York College
awnon.bhowmik@yorkmail.cuny.edu

Unnikrishnan R. Menon
Department of Electrical and Electronics
Vellore Institute of Technology
unnikrishnanr.menon2017@vitstudent.ac.in

Abstract

NRTU is an open-source public key cryptosystem that uses lattice-based cryptography to encrypt and decrypt data. It consists of two algorithms, NTRUEncrypt, which is used for encryption, and NTRUSign which is used for digital signatures. Unlike other popular public-key cryptosystems, it is resistant to attacks using Shor's Algorithm and its performance has been shown to be significantly greater. This paper talks about how Koblitz encoding from Elliptic Curve Cryptography (ECC) can be used to convert each character in a dataset to a point on a elliptic curve. A Pythagoras Theorem analogy is used to turn the point to a single number, which is converted to a sequence of coefficients in $\mathbb{Z}$. A polynomial is then generated for each of these characters. Then the polynomial is reduced, and we show that choosing appropriate parameters for the cryptosystem can make it highly secure and that the decryption algorithm turns out taking exponential time. Since each character is represented by its own polynomial, it increases obscurity thereby increasing the time for decryption and thus the security level. We also implement a form of data compression and test whether data compression $\rightarrow$ encryption $\rightarrow$ decryption $\rightarrow$ data expansion results in original data with no or minimal loss.

1 Why Lattice Cryptography

The real reason[1]

- In 1994, Shor's Algorithm break RSA and ECC with quantum computers
- 2015, NSA announcement: prepare for the quantum apocalypse
- 2017, NIST call for competition/standardization
- 2030, predicted general purpose quantum computers

Further usefulness[1]

- Good understanding of underlying hard problem
- Fast, parallelable, hardware friendly
- Numerous applications: FHE, ABE, MMap, Obfuscation...

Data vaulting attack[1]

- A.k.a. harvest-then-decrypt attack
- Data needs to be secret for, lets say, 30 years
- Quantum computer arrives in, lets say, 15 years
- Perhaps the most practical attack in cryptography

Encrypt Schemes[1]

- NTRUEncrypt - standardized by IEEE and ASC X9

Signature Schemes[1]

- BLISS (NTRU)
- pqNTRUSign (NTRU)

This paper focuses on the NTRUEncrypt and pqNTRUSign algorithm also known as NTRUSign algorithm.

2 Koblitz Encoding Algorithm for ECC

The encoding algorithm for ECC is as follows...

- Given a message M, convert each character m_k into a number a_k using Unicode, where $b = 2^{16}$ and $0 < a_k < 2^{16}$
- Convert the message M into an integer using

$$m = \sum_{k=1}^{n} a_k b^{k-1}$$

  . In practice we choose an n to be less than or equal to 160 such that m satisfies $m \leq 2^{16 \cdot 160} < p$.
- Fix a number d such that $d \leq \frac{p}{m}$. In practice we choose the prime p large enough so that we can allow $d = 100$.
- For integers $j = 0, 1, 2, \cdots d-1$ we do the following loop
  - Compute the x coordinate of a point on the elliptic curve as $x_j = (dm + j) \mod p$ where $m = \left\lfloor \frac{x_j}{d} \right\rfloor$
  - Compute $s_j = (x_j^3 + Ax + B) \mod p$
  - If $(s_j)^{\frac{p+1}{2}} \equiv s_j \mod p$, then define y coordinate of a point on the elliptic curve as $y_j = (s_j)^{\frac{p+1}{4}} \mod p$. Return the point (x_j, y_j).

Thus we are able to encode our message M, as an element of the abelian group $G = E(\mathbb{F}_p)$ [9]

3 NTRUEncrypt

NTRU operations are based on objects in a truncated polynomial ring $R = \mathbb{Z}[X]/(X^N - 1)$ with convolution multiplication and all polynomials in the ring have integer coefficients and degree at most $N - 1$:

$$\mathbf{c}(X) = c_0 + c_1 X + c_2 X^2 + \cdots + c_{N-1} X^{N-1} \tag{1}$$

4 Equivalence class modulo 3

Any number divided by 3 can result in remainders $\{0, 1, 2\}$. So any number divided by 3 is one of $3r, 3r + 1, 3r + 2$. This is useful in constructing the trinary basis $\{-1, 0, 1\}^{\dim}$. The elements in the basis forms the coefficients of the polynomial (1).

Theorem 1. *Let $R \subseteq S \times S$ be an equivalence class on a set S. Then the set of $\mathcal{R}$-classes constitutes the whole of S*

Proof.

$$\begin{aligned}
\forall x \in S : x \in [x]_{\mathcal{R}} & \quad \text{Definition of equivalence class} \\
\neg\left(\exists x \in S : x \notin [x]_{\mathcal{R}}\right) & \quad \text{Assertion of universality} \\
\neg\left(\exists x \in S : x \notin \bigcup [x]_{\mathcal{R}}\right) & \quad \text{Definition of set union} \\
\forall x \in S : x \in \bigcup S/\mathcal{R} & \quad \text{Assertion of universality} \\
S \subseteq \bigcup S/\mathcal{R} & \quad \text{Definition of subset}
\end{aligned}$$

Also:

$$\begin{aligned}
\forall X \in S/\mathcal{R} : X \subseteq S & \quad \text{Definition of equivalence class} \\
\bigcup S/\mathcal{R} \subseteq S & \quad \text{Union is smallest superset: general result}
\end{aligned}$$

By definition of set equality

$$\bigcup S/\mathcal{R} = S$$

and so the set of all $\mathcal{R}$-classes constitutes the whole of S[2]. □

This theorem tells us that in our case, we are using mod 3 to generate coefficients from the set $\{-1, 0, 1\}$ which is basically the numbers of the form $3r - 1, 3r, 3r + 1$. It means that the union of all these equivalence classes spans the entire set of integers before modulo reduction.

5 Algorithm Flow Diagram

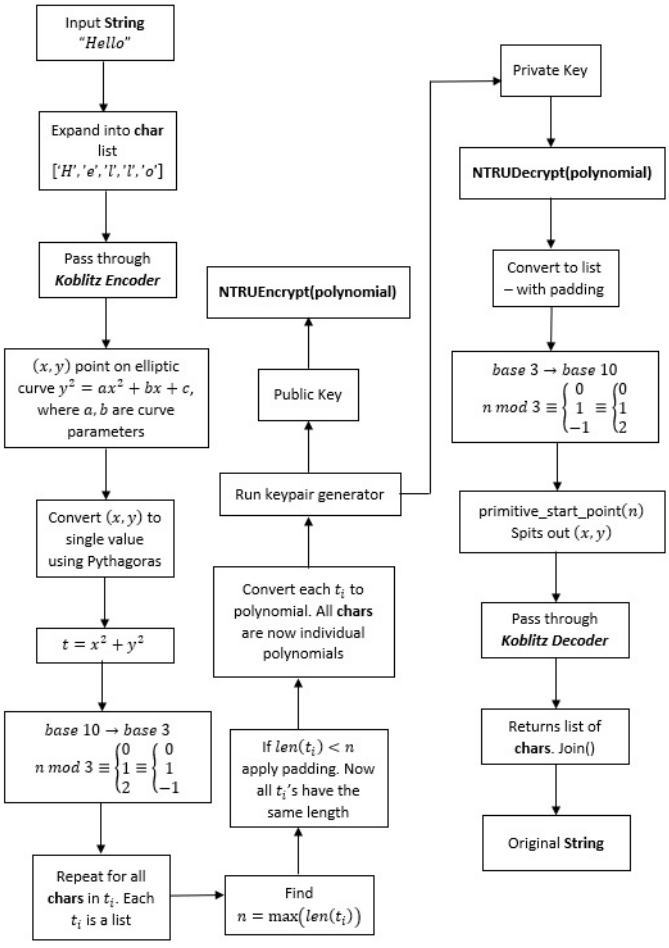

Fig 1. **Algorithm**

6 An attempt on loss less encoding

Messages encrypted by the Koblitz method results in a long polynomial. It is safe to assume that the length of this polynomial increases exponentially with the length of message. Hence, a modification to the aforementioned algorithm in Fig 1. was attempted by converting the polynomials into a gray scale map to obtain some sort of data compression. Following sample results were obtained.

```
entropy@hyperspace:~/Desktop/NTRU_cryptography$ python3 test.py
Convert this to image:
[[0.21428571428571427, 0.10714285714285714, 0.29464285714285715], [0.22321428571428573, 0.24107142857142858, 0.25], [0.39285714285714285
, 0.11607142857142858, 0.3125], [0.33035714285714285, 0.0, 1.0]]
Generating Image...
crap.py:43: DeprecationWarning: `imsave` is deprecated!
`imsave` is deprecated in SciPy 1.0.0, and will be removed in 1.2.0.
Use ``imageio.imwrite`` instead.
  scipy.misc.imsave('encrypted_image.png', to_img)
Reading from image
Data Type: uint8
Min: 0.000, Max: 255.000
[[0.21568627 0.10588235 0.29411765]
 [0.22352941 0.23921569 0.25098039]
 [0.39215686 0.11764706 0.31372549]
 [0.32941176 0.         1.        ]]
entropy@hyperspace:~/Desktop/NTRU_cryptography$
```

Attempt 1

```
entropy@hyperspace:~/Desktop/NTRU_cryptography$ python3 test.py
Convert this to image:
[[0.21428571428571427, 0.10714285714285714, 0.29464285714285715], [0.22321428571428573, 0.24107142857142858, 0.25], [0.39285714285714285
, 0.11607142857142858, 0.3125], [0.33035714285714285, 0.0, 1.0]]
Generating Image...
Reading from image
Data Type: float64
Min: 0.000, Max: 1.000
[[0.21428571 0.10714286 0.29464286]
 [0.22321429 0.24107143 0.25      ]
 [0.39285714 0.11607143 0.3125    ]
 [0.33035714 0.         1.        ]]
entropy@hyperspace:~/Desktop/NTRU_cryptography$
```

Attempt 2

The first attempt involved PNG image format and produced results that were correct up to 2 decimal places only. This means that this format produced a large percentage error. Another thing to notice is that this involved only 8 bit precision unsigned integers. However, using TIFF image format with 64 bit precision unsigned integers resulted in a much better match between the encrypted and the decrypted matrix. Following is a sample gray scale map that was generated.

Fig 2. **Gray scale map of data compression**

Remark: It is to be noted that every character in our input string is treated differently, i.e. if we write the message "banana", then each of the characters have a different random polynomial. The polynomials for all 3 a's and both n's are distinct. This is what makes this cryptosystem more secure.

7 Finite fields in NTRU

Field theory is an essential topic that forms the mathematical foundation of many cryptosystems. The NTRU cryptosystem is no different. The operations involved in our procedure are[3]

- p, q primes, where $p < q$
- addition - $f(x) + g(x)$
- multiplication - $f(x) \cdot g(x)$
- cyclic convolution - $f(x) \cdot g(x) \mod (x^n - 1)$
- modular reduction - $\left(f(x_i) + \frac{q}{2}\right) \mod q - \frac{q}{2} \quad \forall i \leq n.\ n$ is the $\deg(f(x))$

A **field** F, sometimes denoted $\{F, +, \times\}$ is a set of elements with two binary operation, called *addition* and *multiplication*, such as for all a, b, c in F the following axioms are obeyed.[4]

7.1 Properties of finite fields

Consider a finite field $\{F, +, \times\}$. Let x, y be any two arbitrary elements in the field. Then the following properties hold true.

1. **Commutativity** $\begin{cases} x + y &= y + x \\ x \times y &= y \times x \end{cases}$
2. **Associativity** $\begin{cases} x + (y + z) &= (x + y) + z \\ x \times (y \times z) &= (x \times y) \times z \end{cases}$
3. **Identity** $\begin{cases} 0 \in F & x + 0 = 0 + x = x \\ 1 \in F & 1 \times x = x \times 1 = x \end{cases}$
4. **Additive Inverse** For any $x \in F$, there exists $-x \in F$ such that $x + (-x) = 0$
5. **Distributivity** $(x + y) \times z = x \times z + y \times z$
6. **Multiplicative Inverse** For any $x \in F, x \neq 0, \exists x^{-1} \in F$ such that $x \times x^{-1} = 1$

It can be shown that the order of a finite field (number of elements in the field) must be a power of a prime p^n, where n is a positive integer. The finite field of order p^n is generally written $GF(p^n)$; GF stands for Galois field, in honor of the mathematician who first studied finite fields. Two special cases are of interest for our purposes. For $n = 1$, we have the finite field $GF(p)$; this finite field has a different structure than that for finite fields with $n > 1$ and is studied in this section. For finite fields of the form $GF(p^n), GF(2^n)$ fields are of particular cryptographic interest.[4]

8 Future Work

We are planning to wrap up the entire program into a pip install package which would be available for general public use. We would also like to incorporate this crypto system as a security layer for a web socket based client server chat app, built with Tkinter.

References

[1] O. Security, "A short review of the ntru cryptosystem," Jul 2017. [Online]. Available: https://www.slideshare.net/OnBoardSecurity/a-short-review-of-the-ntru-cryptosystem

[2] "Union of equivalence classes is whole set." [Online]. Available: https://proofwiki.org/wiki/Union_of_Equivalence_Classes_is_Whole_Set

[3] "Ntru." [Online]. Available: https://latticehacks.cr.yp.to/ntru.html

[4] W. Stallings, "The principles and practice of cryptography and network security 7th edition, isbn-10: 0134444280," *Pearson Education*, vol. 20, 2017.