Python Implementation of Quantum-Resistant Lattice Cryptography

Liangyu Wang, Hyo Jung Lee

Abstract

In August 2015, the U.S. National Security Agency (NSA) released a major policy statement urging the need to move away from Elliptic Curve-based cryptographic systems to post-quantum cryptography. One such candidate is cryptographic system based on the hardness of solving certain lattice problems. In this paper, we will look at a public key cryptographic system based on the hardness of solving the learning with error problem and demonstrate an implementation in the python programming language.

Contents

1 Why Lattice Cryptography

Since its creation in the 1980s, NSA has been a major proponent of Elliptic Curve cryptographic(ECC) systems, it has recommended ECC to be used to protect the communications of all US government agencies. However in 2015, NSA released a major policy statement on its website on the need to move away from elliptic curve cryptography and develop standards for post-quantum cryptography:

> Unfortunately, the growth of elliptic curve use has bumped up against the fact of continued progress in the research on quantum computing, which has made it clear that elliptic curve cryptography is not the long term solution many once hoped it would be. Thus, we have been obligated to update our strategy.

In 2012, prominent NSA-watcher James Bamford wrote in WIRED magazine:

> NSA made an enormous breakthrough several years ago in its ability to cryptanalyze, or break, unfathomably complex encryption systems employed by not only governments around the world but also many average computer users in the US.

In 2013, Der Spiegel reports that the Snowden leaks indicate that NSA has been able to spy on BlackBerry communications, which has been protected by ECC since blackberry purchased cryptography company Certicom, which owns the world's largest ECC-based patent portfolio.

These developments have prompted widespread speculations that NSA has found a way to break elliptic curve cryptographic systems.

In 2016, NIST(National Institute of Standards and Technology) began the Post-Quantum Cryptography Standardization project in order to "standardize one or more quantum-resistant public-key cryptographic algorithms" that is publicly disclosed worldwide and "capable of protecting sensitive government information well into the foreseeable future" .

Of the 26 submissions that advanced to the second round, most fall into three large families: lattice (12), code-based (7), multivariate polynomial (4). Lattice cryptography is based on hard to solve problems from the mathematical theory of lattices; Code-based cryptography is based on the hardness of decoding linear error-correcting codes; Multivariate polynomial cryptography is based on the hardness of solving multi-variate system of polynomial equations.

Older proposals of multivariate polynomial cryptography has already been broken, and recent proposals have not yet been thoroughly studied; Code-based cryptography is generally considered not very efficient and requires very large key size. This leaves lattice cryptography as the strongest contender for post-quantum cryptography, its encryption and decryption speed is even faster than existing public key systems like RSA.

Lattice cryptography is considered quantum-resistant because at present there is no known quantum algorithm that can solve the hard lattice problems faster than conventional computer. The best known algorithms for solving lattice problems either run in exponential time or have very bad approximation ratios.

2 Mathematics of Lattice

In the study of Algebra, a lattice L in the Euclidean vector space $\mathbb{R}^n$ is defined as:

$$L = \left\{ \sum_{i=1}^{n} x_i * v_i \mid x_i \in \mathbb{Z} \right\}$$

where $\{v_1, v_2, ..., v_n\}$ is a set of basis vectors in $\mathbb{R}^m$.

In plain English, a lattice in n-dimensional vector space $\mathbb{R}^n$ is the set of all integer linear combinations of a set of n linearly independent vectors in $\mathbb{R}^m$. A simple example of a 2-dimensional lattice is one generated by the basis vectors $\{(1,0),(0,1)\}$. The elements of this lattice would be:

$$L = \left\{ \begin{array}{c} 1*(1,0)+1*(0,1)=(1,1), \\ 1*(1,0)+(-1)*(0,1)=(1,-1), \\ -1*(1,0)+1*(0,1)=(-1,1), \\ -1*(1,0)+(-1)*(0,1)=(-1,-1), \\ . \\ . \\ . \end{array} \right\}$$

We can plot L on a graph:

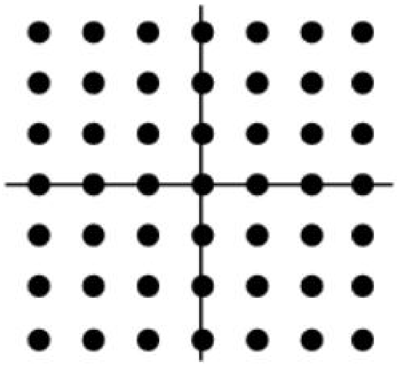
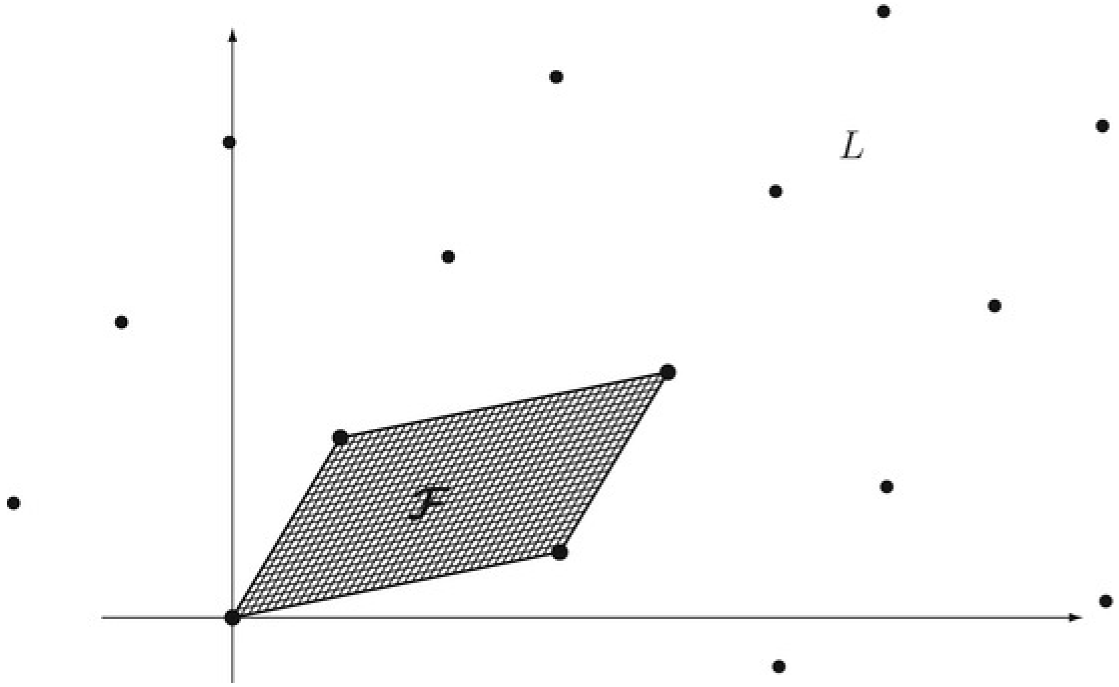


Figure 2 shows a photograph of a gull.

We can think of a lattice as dividing the whole of $\mathbb{R}^m$ into equal copies of an n-dimensional parallelepiped, known as the fundamental region of the lattice. In this instance, $\{x \mid x \in [0,1)^2\}$

is the fundamental parallelogram of the lattice L. In general, $\{\sum_{i=1}^{n} x_i * v_i \mid x_i \in [0,1)^n\}$ is called the fundamental parallelepiped.



From this we can see that lattices have an n-periodic structure with periods given by the norm of its basis vectors.

Suppose we have a set of basis vectors $\{v_1, v_2, ..., v_n\}$ where each vector has m entries, then we can define an $m \times n$ matrix

$$B = [\ v_1 \ \ v_2 \ \ . \ \ . \ \ v_n\ ]$$

The lattice generated by matrix B is

$$L(B) = \{Bx \mid x \in \mathbb{Z}^{\ltimes}\}$$

and the determinant of B is the volume of the fundamental parallelepiped of L.

Operations on such matrices form the basis of lattice-based cryptography.

3 Shortest Vector Problem(SVP) and SVP$_\gamma$

Suppose the length of a vector $v \in L$ is defined by the standard Euclidean norm

$$\|v\| = \sqrt{v_1^2 + \cdots + v_n^2}$$

We want to find the shortest non-zero vector in a Lattice L that is an integer linear combination of the basis of L, that is we want to find $v_{shortest} \in L$ that minimizes the Euclidean norm $\|v\|$ and that $\|v\| \neq 0$. This is called the shortest vector problem.

This problem can be solved relatively easily in 2 dimensions but becomes exponentially more difficult if we scale the problem to hundreds or thousands of dimensions. In fact,in higher dimensions, there is no better way to solve the problem than enumerating all possible vectors lesser than a given length which requires exponential time to compute.

Now instead of solving the SVP which is hard, we can try to solve an approximate SVP. That is we want to find $v \in L$ such that $\|v\| \leq \gamma(n)\|v_{shortest}\|$, where n is the dimensions of the lattice L and $\gamma(n)$ is a function of n. In higher dimensions, this approximation of the SVP can be solved in polynomial time using an algorithm called LLL lattice basis reduction algorithm, a vector analogue of the Euclidean algorithm to compute greatest common divisor of two integers.

4 The Bounded Distance Decoding Problem

Suppose we are given a basis B that generates a Lattice L, and a point $x \in \mathbb{R}^n$ not necessarily in L, we want to find the point in L that is closest to x. This is called the Bounded Distance Decoding Problem (BDDP).

For example, suppose we have a lattice L generated by the basis $\{(\frac{1}{2}, 0), (0, \frac{2}{3})\}$, we are then given a point $x = (3, 5)$. We want to find the integer linear combination of the basis that is closest to x. We can solve this problem by solving the optimization problem

$$\begin{cases} 3 + y = \frac{1}{2}k \\ 5 + z = \frac{2}{3}j \end{cases}$$

for $\min y, z \in \mathbb{R}$ and $k, j \in \mathbb{Z}$.

Since these two equations are independent, we can easily solve them one by one

Now, what does one have to do to solve this problem? Letâs get a graphical feeling for it and formalize it.

This problem is hard to solve in general, but can be easy to solve if the basis of the lattice is made up of nearly orthogonal short vectors, that is if every basis vector is of the form $(0, ..., 0, k, 0, ..., 0)$ for some small value $k \in \mathbb{R}$.

Now we can turn the BDDP into a public key cryptographic system. We encode a secret message as a lattice point m, then add a small noise to m, this makes the secret point m easy to calculate for someone who knows the basis of the lattice, but hard for someone who does not know the basis.

5 Learning with Error

Learning with Error is a generalization of the Learning Parity with Noise problem from machine learning.

Suppose we have a function

$$f_x(a) = a_0x_0 + \cdots + a_nx_n$$

and we are given some samples $(a, f(a))$, we want to learn what the function f is.

If given enough samples $(a, f(a))$, we can reduce this problem to that of solving a system of linear equations:

$$\begin{align} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \tag{1}\\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \tag{2}\\ &\vdots \tag{3}\\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m, \tag{4} \end{align}$$

where we can solve for $(x_1, x_2, \cdots, x_n)$ using Gaussian elimination.

Now what if we add some noise ϵ to our function and require our solution to be in $\mathbb{Z}_q$, then our problem becomes learning the form of the function:

$$f_x(a) = a_0x_0 + \cdots + a_nx_n + \epsilon \pmod q$$

This problem has been proven to be extremely difficult to solve, even a quantum computer will not help. The intuition is that at each step in the Gaussian elimination procedure, the error term grows bigger and bigger until it eclipses all useful information about the function we want to learn.

6 Public Key Cryptographic System Based on LWE

In Public Key Cryptographic System, we have two parties Alice and Bobby that want to communicate with each other. Alice and Bobby both select a secret private key and calculate a public key from the private key, then they both publishes their public key If Alice wants to send a message to Bobby, she will encrypt her message using Bobby's public key, once Bobby obtains message, he will decrypt the message using his own private key. If Bobby wants to send a message to Alice, he will encrypt his message using Alice's public key, and once Alice obtains the message, she will decrypt his message using her secret private key.

In the LWE public key cryptographic system, Alice chooses a modulus q, an $m \times n$ matrix A over $\mathbb{Z}_q^{m\times n}$, a secret key $s \in \mathbb{Z}_q^n$ and a "small" error vector $e \in \mathbb{Z}^n$, sampled from a Gaussian distribution. Alice then calculate

$$b^t = s^t A + e^t \tag{5}$$

Alice publishes (A, b, q) as her public key. Here, the dimension of the matrix A is the security parameter that determines the strength of the cryptographic system.

Bobby wants to send a message to Alice. He chooses a secret binary vector $x \in \{0, 1\}^m$, then calculate

$$\begin{aligned} u &= Ax \\ v &= b^t x + bit \cdot \frac{q}{2} \end{aligned} \tag{6}$$

where bit is the binary representation of the message. Bobby then send the encrypted message (u, v) to Alice.

When Alice receives (u, v) she computes

$$v - s^t u \tag{7}$$

The key insight here is that the term $e^t x$ is very small compared to $\dfrac{q}{2}$, therefore

$$v - s^t u \approx bit \cdot \frac{q}{2} \tag{8}$$

Thus if Alice's calculation evaluates to a value close to zero, she will know Bobby sent her a bit of 0; if her calculation evaluates to a value close to $\frac{q}{2}$, she will know Bobby sent her a bit of 1.

7 Python Implementation of LWE

Tools used:

- Python 3
- Numpy: numerical and scientific computation library for Python

u only need to be computed once per session, so we call u "plaintext preamble". We set $q = 256$, this way we can encode our ciphertext into base64 format.

8 Ring Learning With Error

One disadvantage of public key system based on LWE is that it requires very large key size which is an $n \times n$ matrix.

With RLWE we can quadratically reduce the key size, since instead of using $n \times n$ matrix as public key, we can just use an nth degree polynomial as the public key.

Now instead of vectors in a vector space over finite field $\mathbb{Z}/p\mathbb{Z}$, we consider polynomials $f(x)$ with coefficients from $\mathbb{Z}/p\mathbb{Z}$.

A polynomial

$$a_n x^n + a_{n-1} x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0$$

can be viewed as a vector $(a_n, a_{n-1}, ..., a_1, a_0)$ that is a linear combination of the basis $x^n, x^{n-1}, ..., x, 1$.

Therefore polynomials with coefficients from $\mathbb{Z}/p\mathbb{Z}$ is a vector space over $\mathbb{Z}/p\mathbb{Z}$, the advantage over the Euclidean vector space is that polynomials can be multiplied in a natural way that is not true of vectors in a Euclidean vector space. The product of two polynomials is another polynomial, while the dot product of two vectors in the euclidean vector space is a scalar.

In RLWE, instead of using the Euclidean norm, we use another norm called the infinity norm

$$\|f\|_\infty = \max\{|a_1|, \ldots, |a_n|\}$$

where $a^n, a^{n-1}, ..., a_1$ are the coefficients of the polynomial f. In other words, the length of a polynomial f is defined as the largest coefficient of f. Having defined the norm, we can talk about the notion of "small" polynomials.

Now instead of working with an infinite number of polynomials, we want to work with only a finite number of polynomials, so just like we can do arithmetic with only integers modulus some prime number p, we can work with only polynomials modulus some irreducible polynomial. Just like a prime number is a positive integer not divisible by any number other than 1 and itself, an irreducible polynomial is only the product of 1 and itself. In RLWE, we generally work with polynomials modulus irreducible polynomial $x^n + 1$, where n is a power of 2 . It can be shown that the set of polynomials with coefficients from $\mathbb{Z}/p\mathbb{Z}$ modulus $x^n + 1$ denoted by $\mathbf{F}_p[x]/(x^n + 1)$, contains p^n polynomials with degree no greater than n.

To implement ring learning with error, instead of a matrix, we select a random polynomial

9 Public Key Cryptographic System Based On Ring Learning With Error

Key Generation Alice Selects the ring $R = \mathbf{F}_p[x]/(x^n + 1)$such that n is a power of 2. Choose a polynomial $a \in R_p$ such that the coefficients of a are uniformly random in $\mathbb{Z}_p$, as well as two random "small" polynomial $s, e \in R$ where the coefficients of s, e are sampled from a discrete Gaussian error distribution χ. Here , s is Alice's the secret private key. Alice then publishes

$$(a, b = a \cdot s + e) \in R_p^2$$

as her public key.

Encryption In order to send an n-bit message $m \in \{0, 1\}^n$ to Alice, Bobby encodes the message m into the 0-1 coefficients of a polynomial $z \in R_p$. Bobby then chooses 3 random "small" polynomials $r, e_1, e_2 \in R$ from χ.

Bobby computes

$$u = a \cdot r + e_1 \mod p$$

and

$$v = b \cdot r + e_2 + \frac{p}{2} \cdot z \mod p$$

then sends (u, v) as the ciphertext.

Decryption Upon receiving the ciphertext (u, v), Alice computes

$$v - u \cdot s = (r \cdot e - s \cdot e_1 + e_2) + \frac{p}{2} \cdot z \mod p$$

The coefficients of $r \cdot e - s \cdot e_1 + e_2 \in R$ should have magnitude less than $\frac{q}{4}$, therefore each bit of m can be recovered by rounding each coefficient of $v - u \cdot s$ to either 0 or $\frac{q}{2}$, whichever is closest modulo p.

10 Short Integer Solution Problem

The LWE problem has a "dual" problem called short integer solution (SIS) problem

11 Diffie-Hellman Key Exchange Based on RLWE

12 Python Implementation of RLWE

References

[1] National Security Agency *Commercial National Security Algorithm Suite*, `https://apps.nsa.gov/iaarchive/programs/iad-initiatives/cnsa-suite.cfm`, 2015.

[2] National Institute of Standards and Technology *Post-Quantum Cryptography Standardization*, `https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Post-Quantum-Cryptography-Standardization`, 2017

[3] National Institute of Standards and Technology *Post-Quantum Cryptography, Round 2 Submissions*, `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`, 2020

[4] Neal Koblitz and Alfred J. Meneze *A Riddle Wrapped In An Enigma*, IEEE Security Privacy, vol. 14, no. 6, pp. 34-42, Dec. 2016.

[5] Jeremy Hsu *How the United States Is Developing Post-Quantum Cryptography*, IEEE Spectrum, `https://spectrum.ieee.org/tech-talk/telecom/security/how-the-us-is-preparing-for-quantum-computings-threat-to-end-secrecy`, 2019.

[6] Jeffery Hoffstein, Jill Pipher and Joseph H. Silverman *An Introduction To Mathematical Cryptography*, Springer, 2014.

[7] Stefano Ottolenghi *Homomorphic Signatures over Lattices*, Universit‘a di Genova, 2019

[8] Lyubashevsky, Vadim *Towards practical lattice-based cryptography*, 2008

[9] Dong Pyo Chi, Jeong Woon Choi, Jeong San Kim and Taewan Kim *Lattice Based Cryptography for Beginners*, 2008

[10] Douglas Stebila *Introduction to post-quantum cryptography and learning with errors*, 2018

[11] Vadim Lyubashevsky, Chris Peikert, Oded Regev *On Ideal Lattices and Learning with Errors Over Rings*, 2013

[12] Vikram Singh *A Practical Key Exchange for the Internet using Lattice Cryptography*, 2008