LEARNING WITH ERRORS ENCRYPTION

-By Aditya Koranga

INTRODUCTION

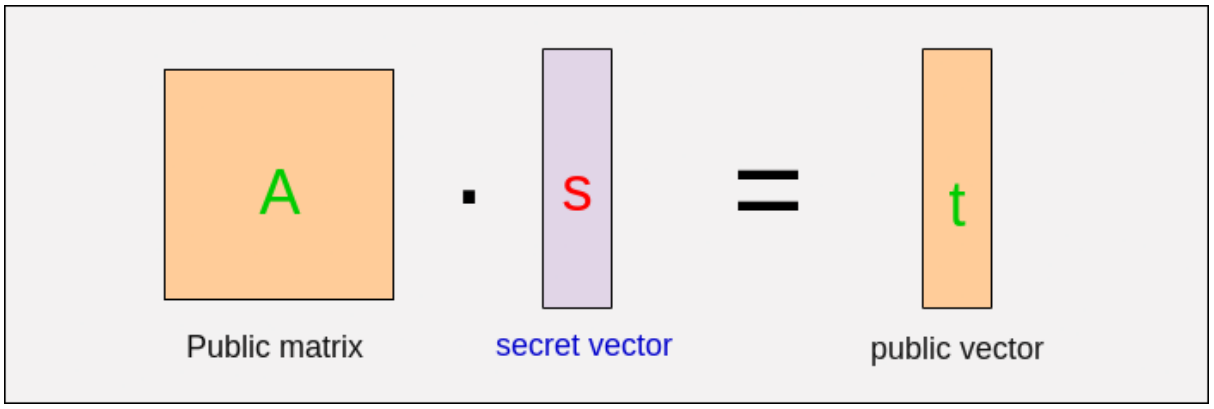
The first thing that comes to mind while talking about lattice-based cryptography is the Learning with errors method. So, learning with errors is a new public key encryption method that is considered to be robust and is used by lattice-based algorithms such as Kyber and Frodokem. As the name suggests, we will be playing with some errors in this method. So, let's see how it works.

HOW LWE WORKS

Firstly let's start with a normal equation :

A . s = t

Where 'A' is a public matrix, 's' is a secret vector and 't' is a public vector. We can even say that: (A,t) is the public key pair, and (s) is our secret or private key.

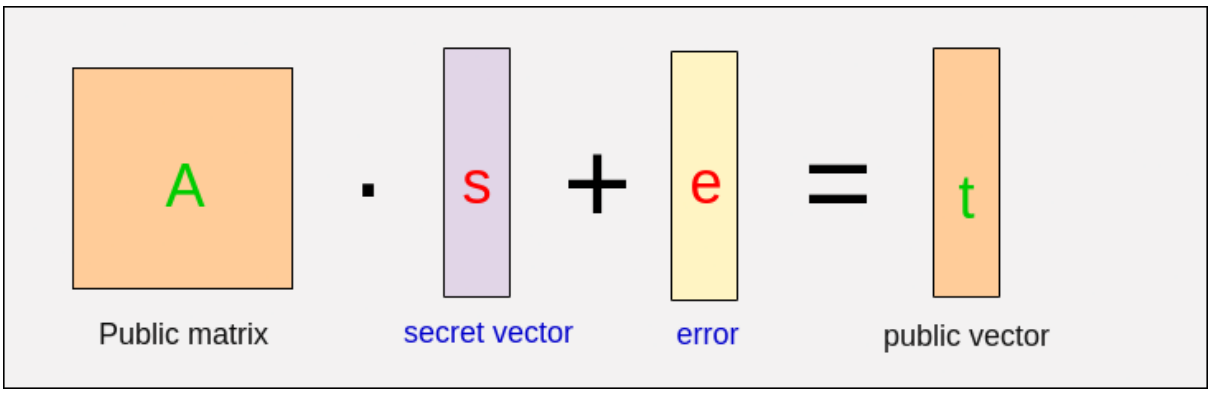


Since A and t are public, using the Gaussian elimination method it is very easy to find the secret vector 's'. So, now we need to do some changes and this is when errors come into the picture.

So, now let's see how LWE actually works. Now, there will be a slight difference in the equation. The equation will now look like this:

A . s + e = t

Where 'e' is a small 'error' vector, also known as 'noise'. Everything remained the same but we just added some errors/noise in the equation and this changes the game as now it is hard to recalculate the secret vector 's'.



Now, by just the knowledge of ‘A’ and ‘t’, we cannot find the secret key ‘s’. Once again (A,t) is the public key pair, and (s) is the secret key. Now’s the time for encryption and decryption.

ENCRYPTION

Let’s take a message ‘m’ and now we need to encrypt this message (we will have to convert this message into a polynomial form).
So now we have plain text ‘m’ and we know that for encryption we use a public key. So in this case we will use our (A,t) key pair for encryption which will then create ciphertext.
The ciphertext will be two values ‘u’: cipher polynomial & ‘v’: vector of the polynomial. Here also we will use some errors along with the public key pair.

$$u = t \,.\, e1 + e2 + m$$
$$v = A \,.\, e1 + e3$$

Where e1,e2, and e3 are random error values. Note that we used error ‘e1’ on both values. Since the ciphertext is ready, now’s the time for decryption.

DECRYPTION

Time for converting the ciphertext into its original form and we know for decryption we use a private key, hence we will use (s) in this step. The decrypted message ‘d’ will be:

$$d = v - s \,.\, u$$

After putting the value of v and u as calculated above.

$$d = t \,.\, e1 + e3 + m - s \,.\, (A \,.\, e1 + e2)$$
$$d = t \,.\, e1 + e3 + m - A \,.\, s \,.\, e1 - s \,.e2$$

Now again we will perform some mathematics. We will add and subtract ‘e.e1’ on the right side.

$$d = e \,.\, e1 + e3 + m + (\, t \,.\, e1 - A \,.\, s \,.\, e1 - e \,.\, e1) - s \,.\, e2$$
$$d = e \,.\, e1 + e3 + m + e1\,[\, t - (\, A \,.\, s + e\,)\,] - s \,.\, e2$$

And we know A . s + e = t . Therefore they can cancel each other.

$$d = m + e \,.\, e1 + e3 - s \,.\, e2$$

Now as compared to ‘m’ the rest ones left (e.e1 + e3 + s.e2) are very small noises that can be easily removed by performing some mathematics which basically rounds off these errors/ noises (for which there is a particular ‘q/2’ method).
And hence **d is equal to m’**. Therefore we can say that decryption is completed.

OTHER TYPES OF LWE

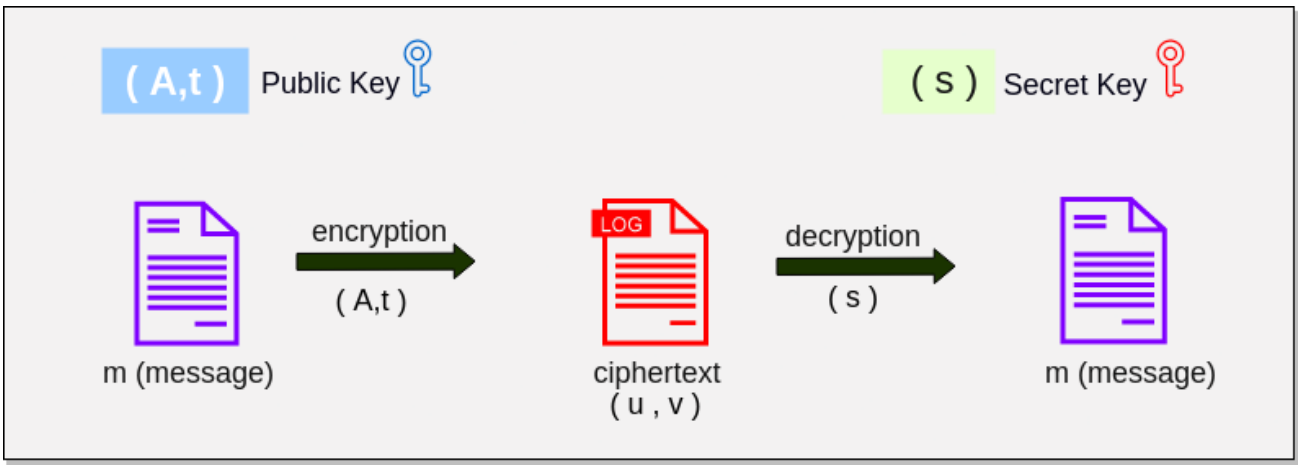
Yes, there are some other types of LWE methods such as Ring-LWE and M-LWE and Kyber particularly uses M-LWE. The concept of such LWE methods is completely the same as discussed above. It is just that there is a slight difference in the creation of public matrix ‘A’, the elements of one column are repeated in the other columns. Like this:

4	1	11	10
3	4	1	11
2	3	4	1
12	2	3	4
9	12	2	3
10	9	12	2
11	10	9	12

Public matrix in M-LWE

COMPLETE FLOW DIAGRAM

The complete flow of the Learning with errors method can be seen below.


(A,t)
Public Key
(s)
Secret Key
encryption
(A,t)
decryption
(s)
LOG
m (message)
ciphertext
(u , v)
m (message)