Lattice Based Cryptosystems

Andrew Fiori

University of Lethbridge

March 22nd, 2019

Talk Plan

1. Explain the most basic premise of public key cryptography.
2. Briefly hint at why people care about lattice based cryptography.
3. Explain a (very) simple variant (by Regev) of lattice based cryptography by working through an a very small explicit example.
4. Assuming time allows, explain how statistics is involved in understanding the security of the system.

We are going to be using the notation

$$a \,\%\, b \qquad \text{for} \qquad a \bmod b$$

which take the remainder (using long division), so for example

$$9 \,\%\, 5 = 4 \qquad 11 \,\%\, 6 = 5 \qquad 23 \,\%\, 11 = 1$$

Public Key Cryptography

The setup:

- Bob (your bank) is lonely (greedy) and wants to receive messages (money).
- Bob (your bank) doesn't want anyone else to be able to read (intercept) the messages (money) people send him.
- Bob (your bank) writes an instruction manual for how to send Bob (your bank) messages (money) that only he will be able to decode.
- Bob (your bank) posts this manual publicly where anyone can see it.
- Bob (your bank) waits for messages (money).

The idea of public key crytography is that Bob will post **publicly** a complete instruction manual for how to send him messages.
The public *instructions* typically consists of a **public key** which is some numerical data, together with information about which algorithm to use with that data to send messages.
For this to be secure, he must know some underlying

private secret

that he can use to decrypt the messages that are sent to him.
His secret is the reason he can decrypt messages, but no one else can.

That creating a secure system like this is possible at all is on its own perhaps surprising. There are now several popular systems.

Problem
The old popular systems won't be as secure once powerful quantum computers are built.

Solution
New systems, the new system I will tell you about today is generally refered to as

Lattice Based Cryptography

Lattice Crytography – Getting Right Into It

Bob posts the following message:

My public key consists of the following, 9 pieces of data, each of which should be interpreted as a vector $\vec{v}_i$ of length three containing numbers modulo 11 (so 3 numbers in the set $\{0, 1, 2, 3, \ldots, 9, 10\}$), together with a number $\mathbf{b_i}$ modulo 22 that is connected to the list of three numbers.

$$\begin{array}{ccc} ((6,2,4),20) & ((4,9,5),9) & ((3,8,3),21) \\ ((5,10,5),2) & ((9,0,7),0) & ((5,6,1),5) \\ ((1,0,7),7) & ((5,2,6),21) & ((10,2,10),10) \end{array}$$

To send me a 1-bit message $m \in \{0, 1\}$ do the following:

1. Pick a random subset of these vectors, so random $S \subset \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
2. Compute
$$((x, y, z), b) = \left(\sum_{i \in S} \vec{v}_i, \sum_{i \in S} \mathbf{b_i} + 11m \right)$$
3. Reduce the coordinates of the vector modulo 11, and the other coordinate modulo 22.
4. Transmit the result of this computation to me.

For longer messages, repeat as necessary.

Public Key

$$\begin{array}{ccc} ((6,2,4),20) & ((4,9,5),9) & ((3,8,3),21) \\ ((5,10,5),2) & ((9,0,7),0) & ((5,6,1),5) \\ ((1,0,7),7) & ((5,2,6),21) & ((10,2,10),10) \end{array}$$

Example if Alice wants to send the message: 1

Alice picks random vectors, say $((6,2,4),20)$ and $((4,9,5),9)$, and computes

$$((6,2,4)+(4,9,5))\%11 = (10,11,9)\%11 = (10,0,9) \qquad (20+9+1\cdot(11))\%22 = 18$$

then sends

$$((10,0,9),18)$$

So if Alice sends Bob the message:

$$((9,4,6),8),\ ((1,1,5),7),\ ((10,8,6),7)$$

What message does bob recieve?

If you can answer this question, based only on what I told you, then you have broken the cryptosystem!

This is a very small example, if you stare at it long enough, maybe you can...

but can you do it without trying all 512 **subsets of those** 9 **vectors?**

With the size of this example we are maybe getting something like 5 or 6-bit security, but if there were more and larger vectors...

But Wait!!!

But Bob is using a generic algorithms, so anyone who sees a message also knows most of what Bob plans to do to decrypt the message!

Bob roughly copied this system from a proposed system by Ogov Regev (2005) which is described on wikipedia (https://en.wikipedia.org/wiki/Learning_with_errors),

So although we may not know the outcome of any randomization steps he took while making his keys, we do know the basic steps he followed and plans to follow.

Maybe that will help us see how we are supposed to decrypt the message.

It is not the obscurity of what Bob is doing that protects the message.

What is Bob Going to Do

Bob has a secret key, a vector:

$$(r, s, t)$$

Bob decodes the message $\big((x, y, z), b\big)$ using the following steps:

1. He computes $e = ((r, s, t) \cdot (x, y, z))\%11$.
2. He multiplies that number by 2, and subtracts it from b, so
$$m' = (b - 2e)\%22.$$
3. If the result, m', is closer to 11 than it is to 0 (or 22) he interprets the message as a 1, otherwise it is a zero.

We will do an example in a moment, but it is worth asking:

If I know:

Public Key:

$$\begin{array}{ccc} \big((6,2,4),20\big) & \big((4,9,5),9\big) & \big((3,8,3),21\big) \\ \big((5,10,5),2\big) & \big((9,0,7),0\big) & \big((5,6,1),5\big) \\ \big((1,0,7),7\big) & \big((5,2,6),21\big) & \big((10,2,10),10\big) \end{array}$$

Message:

$$\big((9,4,6),8\big), \quad \big((1,1,5),7\big), \quad \big((10,8,6),7\big)$$

What was the message and/or what was the secret key?

If you could easily figure out the secret key from the public key or the message, this would be a serious problem.

Bob's Secret, and the message

Bob's Secret Key:

$$(9, 9, 4)$$

Message:

$$((9, 4, 6), 8),\ ((1, 1, 5), 7)\ ,\ ((10, 8, 6), 7)$$

We need to take the dot product $(9, 4, 6) \cdot (9, 9, 4)$, then subtract twice that result from 8.

$$\begin{aligned}
((9, 4, 6), 8) \Rightarrow &\quad (9 * 9 + 4 * 9 + 6 * 4)\%11 = 9 &\quad (8 - 9 * 2)\%22 = 12 \\
((1, 1, 5), 7) \Rightarrow &\quad (1 * 9 + 1 * 9 + 5 * 4)\%11 = 5 &\quad (7 - 5 * 2)\%22 = 19 \\
((10, 8, 6), 7) \Rightarrow &\quad (10 * 9 + 8 * 9 + 6 * 4)\%11 = 3 &\quad (7 - 3 * 2)\%22 = 9
\end{aligned}$$

So the message is

$$1, 0, 1$$

because 12 and 9 are closer to 11, and 19 is closer to $22 = 0 \pmod{22}$.

Both encoding the message, and decoding the message is just taking dot products, adding/subtracting, and taking remainders. None of those things are all that complicated... so...

- Why does it work?
- Why (do we think) it is secure?

Why does this work?

Bob's Secret Key:

$$(9, 9, 4)$$

Public Key:

$$\begin{array}{ccc} ((6,2,4),20) & ((4,9,5),9) & ((3,8,3),21) \\ ((5,10,5),2) & ((9,0,7),0) & ((5,6,1),5) \\ ((1,0,7),7) & ((5,2,6),21) & ((10,2,10),10) \end{array}$$

The main thing that makes the secret key special is the following:

$$\begin{array}{rl} (6,2,4)\cdot(9,9,4)\%11 = (6*9+2*9+4*4)\%11 = 0 & (2*0-20)\%22 = 2 \sim 0 \\ (5,10,5)\cdot(9,9,4)\%11 = (5*9+10*9+5*4)\%11 = 1 & (2*1-2)\%22 = 0 \sim 0 \\ (1,0,7)\cdot(9,9,4)\%11 = (1*9+0*9+7*4)\%11 = 4 & (2*4-7)\%22 = 1 \sim 0 \\ (4,9,5)\cdot(9,9,4)\%11 = (4*9+9*9+5*4)\%11 = 5 & (2*5-9)\%22 = 1 \sim 0 \\ (9,0,7)\cdot(9,9,4)\%11 = (9*9+0*9+7*4)\%11 = 10 & (2*10-0)\%22 = -2 \sim 0 \\ (5,2,6)\cdot(9,9,4)\%11 = (5*9+2*9+6*4)\%11 = 10 & (2*10-21)\%22 = -1 \sim 0 \end{array}$$

So for each of those 9 vectors, $(\vec{v_i}, \mathbf{b_i})$, in the public key we have that the extra piece satisfies

$$\mathbf{b_i} \sim 2(\vec{v_i}\cdot(9,9,4)\%11) = (2\vec{v_i}\cdot(9,9,4))\%22$$

The secret key, $(9, 9, 4)$, is a vector which will make all the differences small.

So Alice sent the message m as

$$((x,y,z),b) = \left(\sum_{i\in S}\vec{v_i}, \sum_{i\in S}\mathbf{b}_i + 11m\right)$$

when Bob computes $e = 2(((x,y,z)\cdot(9,9,4))\%11)$ what he gets is

$$\begin{aligned}
e &= 2(((x,y,z)\cdot(9,9,4))\%11) && \text{definition}\\
&= 2\left(\sum_{i\in S}(\vec{v_i}\cdot(9,9,4))\%11\right) && \text{As } (x,y,z) = \sum_{i\in S}\vec{v_i} \text{ and Linearity of dot produt}\\
&= \left(\sum_{i\in S}2(\vec{v_i}\cdot(9,9,4))\%22\right) && \text{property of remainders}\\
&\sim \left(\sum_{i\in S}\mathbf{b_i}\right)\%22 && \text{because } (2(\vec{v_i}\cdot(9,9,4)))\%22 \sim \mathbf{b_i}
\end{aligned}$$

and so when Bob then computes

$$b - e = \sum_{i\in S}\mathbf{b}_i + 11m - e \sim \sum_{i\in S}\mathbf{b}_i + 11m - \sum_{i\in S}\mathbf{b_i} = 11m$$

he can see approximately the message.

How did Bob get his public/private key so this would work?

1. Pick a random vector with entries modulo 11. **Keep it a secret**
   So bob picked $(9, 9, 4)$
2. Pick 9 more random vectors modulo 11, and compute the dot product with your secret.
   So bob picked

$$\begin{array}{ccc} (6,2,4) & (4,9,5) & (3,8,3) \\ (5,10,5) & (9,0,7) & (5,6,1) \\ (1,0,7) & (5,2,6) & (10,2,10) \end{array}$$

   and computed the dot products. (so 0, 1, 4, 5, 10, . . . *)*
3. Randomly add small errors, *Bobs errors were:* -2, 0, -1, -1, 2, . . . , after doubling to each of the dot products so got

$$0 * 2 - 2, \quad 1 * 2 + 0, \quad 4 * 2 - 1, \quad 5 * 2 - 1, \quad 10 * 2 + 2, \quad \ldots$$

To make things more secure you generally replace 11 by a larger prime, q, increase the number of entries in your vectors from 3 to roughly $\sqrt{q}$, the number of vectors in the public key to roughly $\sqrt{q}\ln(q)$ and the error distribution should have standard deviation less than $\frac{\sqrt{q}}{\ln(q)}$. ($q \sim 2^{20}$ *gives roughly 128 bit security.*)

Why might breaking this crypto system be hard?

- Given only the public key, finding the private key requires solving the *learning with errors* problem. (More to come on what this is assuming if I have time).
- Even determining that a thing which looks like a message:

$$((x, y, z), b)$$

  actually was constructed as a message, requires solving the same *learning with errors* problem.
- If you have a system which can figure out how Bob would translate

$$((x, y, z), b) \Rightarrow m$$

  without knowing if $((x, y, z), b)$ was actually a well formed message, then you can find Bob's secret key... which solves the same *learning with errors* problem.

Learning With Errors (Intuitive Model)

Imagine your statistics professor tries to give your class a multiple choice exam, 25 questions, 5 options each, each option A-E earns between -2 and 2 points, but how it works for each question isn't on the exam.
But there are some problems:

- The test covers material that no one knows, so everyone just answers randomly for most questions.
- When the professor grades the exams, after figuring out the grade of a student, he makes a mistake and adjusts each grade by ± 3.
- Before noticeing these errors he gives the exams back to you, but does not provide solutions.

When everyone complains, he agrees to give you another attempt at an **identical** test, so the correct answers will be the same!!

Question: If all students pool their graded exams, is there enough information to figure out a set of answers so that everyone gets an A?
Can you learn from data with errors?
If you had enough students, then you can use standard statistics techniques to solve the problem, but is 50, 100, or 500, enough for that?
If you don't actually have enough data, you can still check if a proposed set of solutions is plausible...

The actual learning with errors problem is similar to what I just described, only **harder.**

- On their own the vectors $\vec{v_i}$ in the public key are samples from a uniformly distributed random variables.
- On their own the numbers $\mathbf{b_i}$ in the public key are samples from a uniformly distributed random variables.
- Taken together, $(\vec{v_i}, \mathbf{b_i})$ are samples from a random variable... that is **not** uniform.
- For each $\vec{s}$ the numbers $\vec{v_i} \cdot \vec{s} - \mathbf{b_i}$ are samples from a random variable.
- If $\vec{s}$ is a random vector, then with high probability they will appear to be uniform, if $\vec{s}$ is the secret key, they will appear to come from the error distribution (typically Gaussian).

Moreover, the parameters used in the crypto-system

- The size of the number q.
- The size of vectors.
- The number of vectors.
- The error distribution.

are all tuned to try to make the statistics problems basically intractable. (The only known solutions involve an exponential time search)

But this isn't to say there aren't weaknesses. This is all very new.

Problems with the system:

- The exact system I proposed is vulnerable to something called a
  chosen cyphertext attack
  This doesn't undermine the security of the system completely, it just makes means you need to regenerate new public keys very often.
  Variants of the system exist to address the current known attacks. (Chris Peikert-2009).
- This system is very data innefficient.
  To send even a single bit with 128 bit security (which is not that much) could easily require public keys to be over 1 megabyte, and the message to send that bit could be several kilobytes. Even with fast networks, you will notice this level of latency.
  Variants of the system exist to improve this, but they are still no where near as data efficient as our older systems
- These are very new systems, and not enough people have actually looked at them to be necissarily be confident that there are no security holes or easy ways to work around them.

The Future

Linear algebra problems are normally *easy*, polynomial time, this one doesn't seem to be, but finding other vulnerabilities, or improving the efficiency of these algorithms really requires thinking about the underlying linear algebra that is happening.

The main augmentations people currently are pursuing in this fall under the title

Ring Learning With Errors

(Jintai Ding 2011) though the ideas really still are mostly just linear algebra... the terminology, notation, computational tools are very much ring theory. (for math people: they replace the $\mathbb{F}_q$ vector space by the ring of integers in the 2^nth cyclotomic field, mod q, chosen so that q splits... to obtain, an $\mathbb{F}_q$ vector space.)

Right now this is a field where Computer Scientists and Mathematicians often use incompatible terminology, so won't understand what the other is saying. It is very valuable to have people who understand both sides of this, the linear algebra, and the CS implementation and crypto side, looking at these problems.

The End.