7-29-1997

Using the Quantum Computer to Break Elliptic Curve Cryptosystems

Jodie Eicher

Yaw Opoku

Using the Quantum Computer to Break Elliptic Curve Cryptosystems

Jodie Eicher and Yaw Opoku*
University of Richmond, VA 23173

July 29, 1997

Abstract

This article gives an introduction to Elliptic Curve Cryptography and Quantum Computing. It includes an analysis of Peter Shor's algorithm for the quantum computer breakdown of Discrete Log Cryptosystems and an analog to Shor's algorithm for Elliptic Curve Cryptosystems. An extended example is included which illustrates how this modified Shor's algorithm will work.

*Thanks to Hewlett-Packard for their generous support

1. Introduction

The goal of this study is to develop an understanding of how a quantum computer, when created, will be able to break the present day elliptic curve cryptosystems. Elliptic curves are difficult to work with and very difficult ot understand. According to Dr. Ronald L. Rivest, founder of RSA Data Security,

> ... the security of crypotosystems based on elliptic curves is not well understood, due in large part to the abstruse nature of elliptic curves. Few cryptographers understand elliptic curves, so there is not the same widespread understanding and consensus concerning the security of elliptic curves that RSA enjoys. Over time, this may change, but for now trying to get an evaluation of the security of an elliptic curve cryptosystem is a bit like trying to get an evaluation of some recently discovered Chaldean poetry. [5]

Thus, elliptic curve cryptosystems are not as widely used as other cryptosystems. Some of the leaders in implementations of elliptic curve cryptosystems, so far, are Matsushita (Japan), Certicom Corporation (Canada), NeXT Computer (USA), Siemens (Germany), Thompson (France), and University at Waterloo (Canada). The Certicom Corporation in Canada is a leading provider of cryptographic technologies and information security products. It is their belief that elliptic curve cryptosystems are the world's most efficient public-key cryptosystem. They are partnered with companies such as, Motorola, Sterling Commerce, Inc., Schlumberger, Verifone, The Toronto-Dominion Bank, and NIST: National Institute of Standards and Technology [1].

To help develop an understanding of elliptic curve cryptosystems, we will begin with our studies of various present day cryptosystems. In Section 2, we will describe and outline the workings of seven cryptosystems, from the simple linear system to the more complex RSA and Discrete Log cryptosystems. In Section 3, we will explain what elliptic curves are and how they function. In Section 4, we will combine the concepts of Sections 2 and 3 to create and explain three types of elliptic curve cryptosystems. The elliptic curve cryptosystems are actually analogs of the Discrete Log cryptosystems described in Section 2 using elliptic curves. In Sections 5 and 6, we will introduce the quantum computer. Section 5 will explain the quantum mechanics that the quantum computer will be using. Section 6 will explain how a quantum computer would work, if it existed. Amazingly, many studies have already been done to figure out its possible capibilities. Next in our research, we studied how quantum computers have already been "used" to break other cryptosystems. Methods have been discovered for breaking down both RSA and Discrete Log cryptosystems. It is our belief that the breakdown of the elliptic curve cryptosystems will be analogous to the breakdown of the Discrete Log cryptosystems, since the elliptic curve cryptosystems are merely analogs of the Discrete Log cryptosystems. Section 7 will explain Peter Shor's algorithm for breaking down the Discrete Log cryptosystems using a quantum computer. Section 8 will explain our analog to Peter Shor's algorithm for elliptic curve cryptosystems. Section 8.1 is an explicit example of how we think think the algorithm will actually work, on a smaller scale. Not all of the details have been worked out. At the end, we will summarize the loose ends that are still open to further research.

2. Cryptography Overview

Cryptography is the study of methods of sending messages in secret code so that only the intended recipients can break the code and read the message. The message we want to send is called the plaintext(P) and the coded message we send is called the ciphertext(C). The process of converting plaintext to ciphertext is called enciphering or encryption. The process of converting ciphertext back to plaintext is called decyphering or decryption. The use of these two processes together forms a cryptosystem: see [2] for more information on cryptosystems.

Plaintext and ciphertext are broken up into message units. A message unit might be a single letter, a pair of letters (digraph), a triple of letters (trigraph), etc. The first step in using a cryptosystem is to encode all possible plaintext for mathematical usage. For example, to use the letters of the 26-letter alphabet we could encode them using their "numerical equivalents," the integers 0-25.

Linear Transformations:

The most basic cryptosystem is a linear shift transformation. To implement this using an N-letter alphabet we would define an enciphering function f by the rule $C = f(P) \equiv P + b \pmod N$. To decrypt this we would use the formula $P \equiv C - b \pmod N$. The parameter b is called a key. Only intended recipients of the message should know the key. Those who do not have the key, but want to figure out the message have to break the encryption. The science of breaking encryptions is called cryptanalysis.

Example 2.1

Suppose we are using the 26-letter alphabet. We define our encryption function f by the rule $f(P) = P + 6 \pmod{26}$. Thus, $P = C - 6 \pmod{26}$ decrypts our message:

$$\text{“YNOLZ”} = 24\ 13\ 14\ 11\ 25 \mapsto 18\ 7\ 8\ 5\ 19 = \text{”SHIFT”}$$

Suppose we do not know b. One way to figure it out is by frequency analysis. This works as follow. We know that "E" is the most frequently occurring letter in the English language. So it is reasonable to assume that it is the most frequenly occurring letter in our ciphertext. "T" is the second most frequently occurring letter. And so on. (This works better for longer strings of ciphertext.)

Unfortunately, this type of cryptosystem is too simple to be much good. And improvement is the more general type of transformation called an affine cryptosystem. In this case, $C \equiv aP + b \pmod N$ and $P \equiv a'C + b' \pmod N$ where $a' = a^{-1} \pmod N$ and $b' = -a^{-1}b$.

Of course, if we do not know a and b, we will have to use frequency analysis again and a system of equations. For example, if H and L are our most frequent letters respectively. Then we can create the system:

$$9a' + b' = 4 \pmod{26}$$
$$13a' + b' = 19 \pmod{26}$$

to find a' and b'.

Digraph Transformations:

Digraph transformations are linear systems where our plaintext and ciphertext are split up into two-letter message units called digraphs. If our plaintext has an odd number of letters, a blank can be added to the end, if our alphabet contains the blank (letter # 26), or an extra letter such as Z or X in the 26-letter alphabet may added to the end to make a whole number of digraphs. Each digraph is then assigned a numerical equivalent using the formula $P = xN + y$ $(0 \leq P \leq N^2 - 1)$. (x and y are the numerical equivalent to the two letters that make up the digraph.) Then we can use the affine transformation, $C = aP + b \pmod{N^2}$ to encrypt our message.

To decipher this cryptosystem, we use the formula $P \equiv a'C + b' \pmod{N^2}$, where $a' \equiv a^{-1} \pmod{N^2}, b' \equiv a^{-1}b \pmod{N^2}$, and $C = x'N + y'$. (x' and y' are the numerical equivelants to the ciphertext digraph.) We then set P equal to $xN + y$ and find x and y.

To break a digraphic encryption system which uses an affine transformation, we need to know the ciphertext corresponding to two different plaintext message units. Using a frequency analysis of two-letter blocks, we can compare the most frequently occuring plaintext and ciphertext digraphs. If we are using the 26-letter alphabet, "TH" and "HE" are the two most frequenly occuring plaintext digraphs respectively. With this information, we can set up a system of equations to find a' and b'. Then we can find P ($P \equiv a'C + b' \pmod{N^2}$), and then we can find x and y ($P = xN + y$).

<u>Matrix Transformations:</u>

An alternative for the linear digraph transformation is a matrix transformation. Matrix transformation uses linear algebra $\pmod{N}$ to encrypt messages. In a matrix transformation each digraph corresponds to a vector. (eg. "NO"$= \left(\begin{smallmatrix} 13 \\ 14 \end{smallmatrix}\right)$) To encipher this matrix we need a matrix $A = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ such that the determinant of A has no common factors with N. Then each plaintext digraph $P = \left(\begin{smallmatrix} x \\ y \end{smallmatrix}\right)$ is taken to a ciphertext digraph $C = \left(\begin{smallmatrix} x' \\ y' \end{smallmatrix}\right)$ by the rule $C = AP$. (ie. $\left(\begin{smallmatrix} x' \\ y' \end{smallmatrix}\right) = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)\left(\begin{smallmatrix} x \\ y \end{smallmatrix}\right)$) to encipher a plaintext sequence of k digraphs P can be a $2 \times k$ matrix with the k vectors as its columns. To decipher the matrix transformation we simply apply the inverse matrix: $P = A^{-1}C$.

<u>Example 2.2</u>

$$C = AP = \left(\begin{smallmatrix} 2 & 3 \\ 7 & 8 \end{smallmatrix}\right)\left(\begin{smallmatrix} 12 & 19 & 8 \\ 0 & 17 & 23 \end{smallmatrix}\right) = \left(\begin{smallmatrix} 24 & 89 & 85 \\ 84 & 269 & 240 \end{smallmatrix}\right) = \left(\begin{smallmatrix} 24 & 11 & 7 \\ 6 & 9 & 6 \end{smallmatrix}\right) = \text{“}YGLJHG''$$
$$P = A^{-1}C = \left(\begin{smallmatrix} 14 & 11 \\ 17 & 10 \end{smallmatrix}\right)\left(\begin{smallmatrix} 24 & 11 & 7 \\ 6 & 9 & 6 \end{smallmatrix}\right) = \left(\begin{smallmatrix} 402 & 253 & 164 \\ 468 & 277 & 179 \end{smallmatrix}\right) = \left(\begin{smallmatrix} 12 & 19 & 8 \\ 0 & 17 & 23 \end{smallmatrix}\right) = \text{“}MATRIX''$$

A more general way to encipher a digraph vector P is to use an affine transformation, which would apply a 2×2 matrix A and add a constant vector $B = \left(\begin{smallmatrix} e \\ f \end{smallmatrix}\right)$. The inverse transformation for this is simply $P = A^{-1}C - A^{-1}B$ (aka. $P = A'C + B'$)

To break this type of cryptosystem, we need to use known pairs of letters of plaintext with their corresponding pairs of ciphertext to find A^{-1}. Then we apply that to the entire plaintext message. In the affine case, three digraph pairs must be known, and a system of three equations must be used.

In the above cryptosystems, it would be safer to use larger blocks of k letters which have numerical equivilents modN^k. For $k > 3$ the frequency analysis is harder to use, since the number of k-letter blocks is very large, and because it is very difficult to determine the most frequently occuring k-graph.

<u>Public Key:</u>

The above cryptosystems are known as classical cryptosystems. Classical cryptosystems are cryptosystems in which, once the enciphering information is known, the deciphering transformation can be implemented in approximately the same amount of time. The following cryptosystems will be called public key cryptosystems. A public key cryptosystem has the property that someone who knows only how to encipher cannot use the enciphering key K_E to find the deciphering key K_D without a prohibitively lengthy computation. The reason for the name "public key" is that the information needed to send secret messages, the enciphering key K_E, can be made public information without enabling anyone to read the secret message. With a public key cryptosystem, it is possible for two parties to initiate secret communications without ever having any prior contact or having exchanged any preliminary information, since all of the information needed to send an enciphered message is publicly available.

Note: From this point forth, the sender of the messages will be referred to as Alice, the intended receiver of the message will be named Bob, and the eavesdropper, a member of the public trying to break the cryptosystem, will be named Charlie.

RSA:

The first public key cryptosystem we will examine is known as the "RSA" cryptosystem, named after its inventors Rivest, Shamir, and Adleman. In this system Bob has publicly published the enciphering key (n, s), where n is the product of two very large (approx. 100 digits) random prime numbers p and q that he chooses and keeps private. Bob also publishes s, another number that Bob chooses that is prime to both p and q. He then computes t from the formula $st = 1 \bmod (p-1)(q-1)$ and keeps it private.

If Alice wants to send Bob a message P, she can encrypt it using the formula $C = P^s \pmod{n}$. Bob can then decrypt the message using the formula $P = C^t \pmod{n}$, since $C^t = (P^s)^t = P^{st} = P^1 \pmod{n}$.

Example 2.3 (Using smaller numbers for demonstration purposes.)

Bob chooses $p = 15487903$ and $q = 179939723$, and keeps them private. He publishes $n = pq = 2786888975670869$. He also publishes $s = 1223467907$. Using the formula $1223467907t = 1 \bmod 2786888780243244$, he finds $t = 1086249566652563$, and keeps that private.

Alice wants to send the message $P = 91256$ to Bob. She sends the encrypted message $C = 172615924505195$. Bob finds $P = C^t \bmod n = 17261592450519^{108624956665256} \bmod 278688897567086 = 91256$. Yay!

Charlie does not have access to t, so deciphering this cryptosystem is much more difficult for him. First he must factor n to get p and q. This is practically impossible to do when n is approximately 10000 digits long. Even today's computers can not do it. However, if Charlie were able to find p and q, then he would be able to compute t using the formula $st = 1 \bmod (p-1)(q-1)$, and then compute $P = C^t \bmod n$.

Discrete Log:

When working with real numbers, exponentiation (finding b^x) is not significantly easier than its inverse operation (finding $\log_b x$). However, when working with finite groups, such as $(\boldsymbol{Z}/n\boldsymbol{Z})^*$ or $\boldsymbol{F}_q^*$ (with the group operation of multiplication), one can compute b^x for large x fairly rapidly, but its inverse is significantly more difficult. The problem of computing $x = \log_b y$ (given y) is known as the "discrete logarithm problem." The word "discrete" distinguishes the finite group situation from the classical continuous situation. Even today's computers can not do this.

Diffie-Hellman:

This system is merely a method for exchanging keys; no messages are involved. Suppose that Alice and Bob want to agree upon a key, a random element of $\boldsymbol{F}_q^*$, which they will use later to encrypt messages to one another. q is public knowledge (ie. everyone knows what finite field the key is in). g is a fixed element of $\boldsymbol{F}_q$ and is also public knowledge. (Ideally, g should be a generator of $\boldsymbol{F}_q^*$, but it is not absolutely necessary. This method for generating a key will lead only to elements of $\boldsymbol{F}_q$ that are powers of g; thus, if we want our random element of $\boldsymbol{F}_q^*$ to have a chance of being any element, g must be a generator.) Alice chooses a random integer a between 1 and $q-1$ and keeps it secret. She then computes $g^a \in \boldsymbol{F}_q$ and makes that public. Bob computes and publishes g^b using his own secret random integer b. The secret key that they will use is g^{ab}. Both Alice and Bob can compute this key. For example, Alice knows g^b (public knowledge) and her own secret a. Charlie, on the other hand, only knows g^a and g^b. Without solving the discrete logrithm problem (finding a knowing g and g^a), there is no way for him to compute g^{ab} only knowing g^a and g^b.

Massey-Omura:

In this system the finite field $\boldsymbol{F}_q$ has been made public. Alice and Bob both select a random integer e between 0 and $q-1$ such that $\gcd(e, q-1) = 1$. They also compute their inverses $d = e^{-1} \bmod q-1$ (ie. $de \equiv 1 \bmod q-1$) and keep everything secret. If Alice wants to send message P to Bob, she first sends him the message P^{e_A}. This means nothing to Bob, since he does not know d_A. However, he can raise it to the e_B power and send the message $P^{e_A e_B}$ back to Alice. Then Alice can help unravel the message by raising this new message to the d_A power which sends $P^{e_A e_B d_A} = P^{e_B}$ back to Bob. Then Bob can raise this message to the d_B power to get the original message ($P^{e_B d_B} = P$). During this process Charlie sees P^{e_A}, $P^{e_A e_B}$, and P^{e_B}. Without solving the discrete logarithm problem (eg. finding e_B (and then its inverse) knowing P^{e_A} and $P^{e_A e_B}$), there is no way for him to find P.

ElGamal:

In this system the finite field $\boldsymbol{F}_q$ and an element $g \in \boldsymbol{F}_q^*$ (preferably, but not necessarily, a generator) are public information. Bob randomly chooses an integer b ($0 < b < q-1$) and keeps it secret. However, he does publish the element $g^b \in \boldsymbol{F}_q$. If Alice wants to send message P to Bob, she will choose a secret random integer a and send (g^a, Pg^{ab}) to Bob. Bob will then raise g^a to the b-th power and divide Pg^{ab} by g^{ab} to find P. In the meantime, Charlie has only seen g^b and g^a. Without solving the discrete logarithm problem (eg. finding a knowing g^a and then finding g^{ab}), there is no way for him to find P.

3. Elliptic Curves Overview

Elliptic Curves are not ellipses. So, what are they and where do they come from? Elliptic curves are cubic equations in the general form $y^2 = f(x) = x^3 + ax^2 + bx + c$ with distinct roots. They arose when studying the problem of how to compute the arc length of an ellipse. To compute the arc length of an ellipse, one integrates a function involving $y = \sqrt{f(x)}$, and the answer is given in terms of certain functions on the "elliptic" curve $y^2 = f(x)$.

Different Forms of Elliptic Curves:

Let K be a finite field of characteristic $\neq 2$ or 3. An *elliptic curve over* K, E_K, is the set of points (x, y) with $x, y \in K$ which satisfy the equation $y^2 = x^3 + ax + b$ (where the cubic on the right has no multiple roots), together with a single element denoted $\mathcal{O}$ and called the "point at infinity" (discussed below). If K is a finite field of characteristic 2, then an *elliptic curve over* K, E_K, is the set of points satisfying the equation $y^2 + y = x^3 + ax + b$ (where the cubic may or may not have multiple roots), together with a "point at infinity" $\mathcal{O}$. If K is a finite field of characteristic 3, then an *elliptic curve over* K, E_K, is the set of points satisfying the equation $y^2 = x^3 + ax^2 + bx + c$ (where the cubic on the right has no multiple roots) together with a "point at infinity" $\mathcal{O}$.

Points on Elliptic Curves:

We will use the geometric principle that a line intersects a cubic at three points to do our compositions. If we have two points (both rational, real, complex, or finite) on the curve, then we can find a third point on the curve of the same kind.

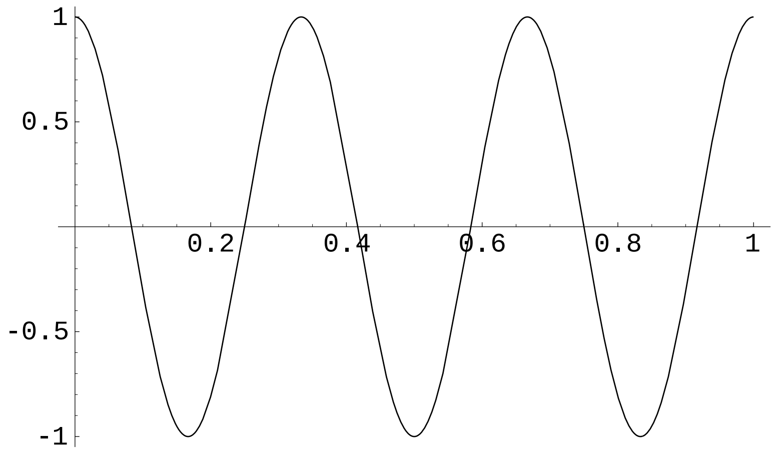

Figure 3.1

Adding: (Figure 3.1)

Starting with two points P and Q, we draw the line through P and Q and let $P * Q$ denote the third point of intersection of the line with the cubic. If we only have one point P, we can find another point by drawing the tanget line to the cubic at P. This is essentially the line through P and P and will result in the point $P * P$. Now we will designate a point $\mathcal{O}$ on the curve as the zero element. There is nothing special about our choice of $\mathcal{O}$. To add P and Q, we join the point $P * Q$ to the point $\mathcal{O}$ and take the third intersection point to be $P+Q$. (ie. $P+Q = \mathcal{O} * (P * Q)$) This is commutative, since clearly $P+Q = Q+P$.

Identity Element: (Figure 3.1)

Now we want to prove that the point $\mathcal{O}$ is the Zero Element. (ie. $P+\mathcal{O} = P$) First we will draw the line through P and $\mathcal{O}$ and find the point $P * \mathcal{O}$. Then we will join the point $P * \mathcal{O}$ to the pint $\mathcal{O}$ and find that the third intersection point is P. Thus, $P + \mathcal{O} = P$.

Inverse: (Figure 3.1)

Now we want to find negatives. First we will draw the tangent line to the cubic at $\mathcal{O}$ and call the third point of intersection S. Then we will join P and S to find the third point $P * S$ which we will call $-P$. To prove this is the inverse we want to make sure $P + (-P) = \mathcal{O}$. To do this we will find the third intersection of the line through P and $-P$, $(P * P)$, which is S. Then we will find the third intersection of line through S and $\mathcal{O}$, $(S * \mathcal{O})$, which is $\mathcal{O}$ since the line is tangent to the curve at $\mathcal{O}$. Thus, $P + (-P) = \mathcal{O}$.

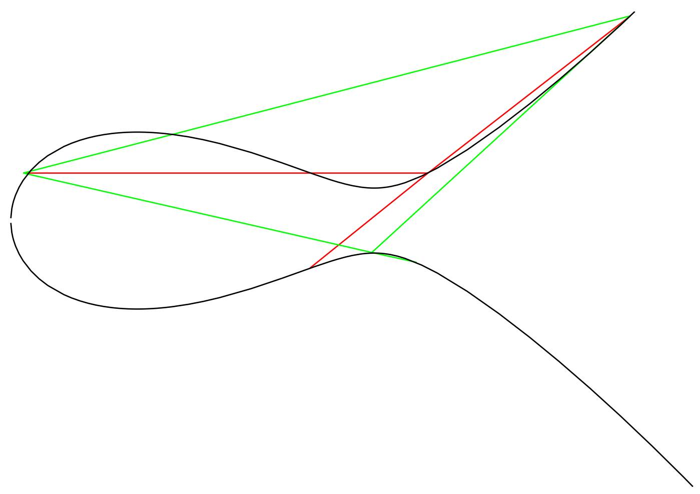

Figure 3.2

Associative Law: (Figure 3.2)

Let P, Q, R be three points on the curve. We want to prove that $(P+Q)+R = P+(Q+R)$. To get $P+Q$, we find $P * Q$ and then find the third point of intersection of the line through $P * Q$ and $\mathcal{O}$. To add $P+Q$ and R, we draw the line through R and $P+Q$ and find the third point of intersection at $(P+Q) * R$. To find $(P+Q)+R$, we would have to find the third point of intersection of the line through $(P+Q) * R$ and $\mathcal{O}$. This does not show up well in the picture, but to prove that $(P+Q)+R = P+(Q+R)$, it will be enough to show that $(P+Q) * R = P * (Q+R)$. (When you connect the same point with $\mathcal{O}$ you will get the same third point of intersection.) To find $P * (Q+R)$, we first have to find $Q * R$, join it to $\mathcal{O}$, and then find the third point of intersection at $Q+R$. Then we draw the line between P and $Q+R$ to find the third intersection point $P * (Q+R)$, which should be at the same location as $(P+Q) * R$.

Note: Now that we have proven that the points on the curve are associative, have an inverse, and have an identity element, we have also proven that the operation + makes the points on the curve form a group. Thus, we can now use group law. Later, we will give explicit formulas for adding points.

Now we are going to move $\mathcal{O}$. You will now have to accept the idea of the existence of a point at infinity. If we draw a line at infinity, it would intersect our graph three times at the point $\mathcal{O}$ (it is an inflection point). To make everything work, we have to make the convention that the points on our cubic consist of the ordinary points in the xy plane together with one other point $\mathcal{O}$ that you cannot see. Every vertical line meets the cubic at two points in the xy plane and also at the point $\mathcal{O}$. Every non-vertical line meets the cubic in three points in the xy plane. To add points on a curve, we now find the point $P * Q$, and then draw the line through $P * Q$ and $\mathcal{O}$, which is just the vertical line through $P * Q$. To find the negative of a point, we simply reflect it about the x axis; if $P = (x, y)$, then $-P = (x, -y)$.

Note: The point at infinity $\mathcal{O}$ and the point $(0, 0)$ are not the same. They are two different points on an elliptic curve.

Order:

When working with elliptic curves the product nP is interpreted to mean the addition of P to itself n times (eg. $3P = P + P + P$). A point P is said to have *order* m if $mP = \underbrace{P + P + ... + P}_{m\ summands} = \mathcal{O}$. In other words, the number of times you have to add P to itself to get $\mathcal{O}$ is the called the *order* of P.

Example 3.1

Suppose P is an inflection point. Then $P + P = (P * P) * \mathcal{O} = P * \mathcal{O} = -P$, which implies that $P + P + P = -P + P = (-P * P) * \mathcal{O} = \mathcal{O} * \mathcal{O} = \mathcal{O}$. Thus, P has order 3.

Note: For the rest of this article we will be dealing with elliptic curves over finite fields $\boldsymbol{F}_q(q = 2^r)$. They will be in the form $y^2 + cxy + dy = x^3 + ax + b$ (characteristic 2, general form).

Adding Formulas:

Remember that the points on E_K form a group with an identity element the point at infinity. And the negative of a point $P \in E_K$ is the second point on E_K having the same x-coordinate as P. Suppose that $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ are two points on E_K; not the point at infinity and not the negatives of one another. There are algebraic formulas that can be applied to find a third point, $P_3 = (x_3, y_3)$. If E_K has the equation for characteristic 2, then

$$x_3 = -x_1 - x_2 + \alpha^2 + c\alpha, \quad y_3 = -cx_3 - d - y_1 + \alpha(x_1 - x_3),$$

where

$$\alpha = \begin{cases} (y_2 - y_1)/(x_2 - x_1), & \text{if } P_1 \neq P_2; \\ (3x_1^2 + a - cy_1)/(2y_1 + cx_1 + d), & \text{if } P_1 = P_2. \end{cases}$$

To make things easier there are duplication formulas to compute the value of $2P$. For large numbers you can simplify the process by reducing it down to a series of doublings and fewer additions using the method of repeated doubling.

Example 3.2

$$100P = 2(2(1 + 2(2(2(1 + 2P)))))$$

N:

Let N be the number of $\boldsymbol{F}_q$-points on an elliptic curve defined over $\boldsymbol{F}_q$ $(q = 2^r)$. To find N, we use the formula:

$$N_r = \begin{cases} 2^r + 1, & \text{if } r \text{ is odd;} \\ 2^r + 1 - 2(-2)^{r/2}, & \text{if } r \text{ is even.} \end{cases}$$

This number N includes the point at infinity $\mathcal{O}$. The order of a point on the elliptic curve will be equal to N or a divisor of N.

The addition of points on an elliptic curve and the number of points on an elliptic curve over a finite field will be used to make elliptic curve cryptosystems in the next section.

4. Elliptic Curve Cryptosystems

Imbedding Plain Text:

Before we can use elliptic curves to encrypt our messages, our messages must be encoded to make mathematical sense. In this case, we must imbed our plaintext message, m, as a point P_m on our elliptic curve, E, over the finite field, $\boldsymbol{F}_q$. The x-coordinate of the point P_m is a simple relationship to the integer equivilant of m. The corresponding y-coordinate on the elliptic curve will then be used. For all of the cryptosystems in this article, the message, m, will already be imbedded into the point P_m.

Elliptic Curve Discrete Log:

As we mentioned in the cryptography overview, the term "discrete" distinguishes the finite group situation from the classical continuous situation. In the cryptography overview, we discussed public key cryptosystems based on the discrete logarithm problem in the multiplicative group of a finite field. Now we will do the same in the group (under addition of points) of an elliptic curve E defined over a finite field $\boldsymbol{F}_q$. If E is an elliptic curve over $\boldsymbol{F}_q$ and B is a point on E, then the discrete log problem on E (to the base B) is the problem: given a point $P_m \in E$, find an integer $x \in Z$ such that $xB = P$. Special methods for solving the discrete log problem in $\boldsymbol{F}_{2^r}$ make it relatively easy to break the discrete log cryptosystems in finite fields, unless r is rather large. The analogous systems using elliptic curves defined over $\boldsymbol{F}_{2^r}$ is secure wit! h significantly smaller values of r. We would like to remind you that even today's computers can not solve the discrete logarithm problem.

Analog of Diffie-Hellman:

Once again, this is merely a system for exchanging keys; no messages are involved. Alice and Bob first publicly choose a finite field $\boldsymbol{F}_q$ and an elliptic curve E defined over it. Then they publicly choose a point $B \in E$ to serve as their "base." (B is preferably, but not necessarily the generator of the group of points on E. It is a generator of the key.) To generate a key, Alice chooses a random integer a of order of magnitude q (which is approximately the same as N) and keeps it secret. She then computes $aB \in E$ and makes that public. Bob chooses his own secret random integer b and makes public $bB \in E$. The secret key is then $abB \in E$. Both Alice and Bob can compute this key. For example, Alice knows bB (public knowledge) and her own secret a. Charlie, on the other hand, only knows aB and bB. Without solving the discrete logarithm problem (finding a knowing B and aB), there is no way for him! to compute abB only knowing aB and bB.

Analog of Massey-Omura:

In this system the finite field $\boldsymbol{F}_q$ and the elliptic curve E have been made publicly known. The number N of points on E has been computed and is also publicly known. Alice and Bob both select a random integer e betweeen 1 and N such that $\gcd(1, N) = 1$. They also compute their inverses $d = e^{-1} \bmod N$ (ie. $de \equiv 1 \bmod N$) and keep everything secret. If Alice wants to send the message P_m to Bob, she first sends him the message $e_A P_m$. This means nothing to Bob, since he does not know d_A. However, he can multiply it by his e_B and send the message $e_A e_B P_m$ back to Alice. Then Alice can help unravel the message by multiplying this new message by d_A which sends $e_A e_B d_A P_m = e_B P_m$ back to Bob. Then Bob can multiply this message by d_B to get the original message

$(e_Bd_BP_m = P_m)$. During this process Charlie sees $e_AP_m, e_Ae_BP_m$, and e_BP_m. Without solving the discrete logarithm problem (eg. f! inding e_B (and then its inverse) knowing e_AP_m and $e_Ae_BP_m$), there is no way for him to find P_m.

Example 4.1

We will use the elliptic curve (characteristic 2): $y^2 + y = x^3$
We will use the finite field: $\boldsymbol{F}_8 = \boldsymbol{F}_{2^3} = \{0, 1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6\}$
where $\alpha^3 = \alpha + 1$ and α is the generator.
$N = 2^r + 1 = 2^3 + 1 = 9$
The nine points that satisfy this equation are:
$(0, 0), (0, 1), (\alpha^2, \alpha^6), (\alpha^3, \alpha^4), (\alpha^5, \alpha^2), (\alpha^5, \alpha^3), (\alpha^6, \alpha), (\alpha^6, \alpha^3), \mathcal{O}$
To add two different points, P_1 and P_2, on this curve, we will use the equations that we've seen before, modified to work in the finite field $\boldsymbol{F}_8$ with α as the generator:

$$P_1 + P_2 = P_3 = (x_3, y_3)$$
$$x_3 = -x_1 - x_2 + A^2 + cA\ (c = 0) \Rightarrow x_1 + x_2 + A^2$$
$$y_3 = cx_3 - d - y_1 + A(x_1 - x_3)\ (c = 0, d = 1) \Rightarrow 1 + y_1 + A(x_1 - x_3)$$
$$A = (y_2 - y_1)/(x_2 - x_1) \Rightarrow (y_2 + y_1)(x_2 + x_1)^6$$

This equation has a special duplication formula: $(2P) = (x^4, y^4 + 1)$ found by substituting $x_1 = x_2 = x, a = b = c = 0$, and $d = 1$ into the formulas for adding two points, and by using the equation $y^2 + y = x^3$ to simplify.

Alice's random point and its inverse: $e_A = 2,\ d_A = 5$
Bob's random point and its inverse: $e_B = 4,\ d_B = 7$

Message Point: $P_m = (\alpha^3, \alpha^4)$

$e_AP_m = 2(\alpha^3, \alpha^4) = (\alpha^5, \alpha^6)$ (Alice's encrypted message)

$e_Ae_BP_m = 4(\alpha^5, \alpha^6) = 2(2(\alpha^5, \alpha^6)) = 2(\alpha^6, \alpha) = (\alpha^3, \alpha^5)$

$$e_Ae_Bd_AP_m = 5(\alpha^3, \alpha^5) = 1 + 2(2(\alpha^3, \alpha^5)) = 1 + 2(\alpha^5, \alpha^2) = 1 + (\alpha^6, \alpha^3) = (\alpha^6, \alpha)$$

which equals $e_BP_m = 4(\alpha^3, \alpha^4) = 2(2(\alpha^3, \alpha^4)) = 2(\alpha^5, \alpha^6) = (\alpha^6, \alpha)$

$$e_Bd_BP_m = 7(\alpha^6, \alpha) = 1 + 2(1 + 2(\alpha^6, \alpha)) = 1 + 2(1 + (\alpha^3, \alpha^5)) = 1 + 2(0, 1) = 1 + (0, 0) = (\alpha^3, \alpha^4)$$ (Bob's encrypted message)

which equals P_m. Yay!

Analog of ElGamal:

In this system the finite field $\boldsymbol{F}_q$, the elliptic curve E, and the base point $B \in E$ (preferably, but not necessarily a generator of the curve) are public information. Bob randomly chooses an secret integer b $(1 < b < N)$ and publishes the point bB. If Alice wants to send the message P_m to Bob, she will choose a secret random integer a $(1 < a < N)$ and send $(aB, P_m + abB)$ to Bob. Bob will then multiply the first point in the pair by b and subtract abB from $P_m + abB$ to find P_m. In the meantime, Charlie has only seen aB and bB. Without solving the discrete logarithm problem (eg. finding a knowing aB and then finding abB), there is no way for him to find P_m.

5. Quantum Mechanics Overview

Our general information on quantum mechanics was taken from the textbook [4].

Its Origin

At the turn of the nineteenth century, physics was in a state of turmoil; and that was due to the numerous experimental observations which were totally inexplicable on the grounds of the firmly established classical physics. One of such observations was the photoelectric effect (light hitting a metal surface and ejecting electrons). In terms of classical physics, light of a higher intensity will eject more electrons; but this could not be supported by experiment. And to make things more confusing, it was discovered experimentally that only certain frequencies of light could cause the photoelectric effect; and below a certain threshold no electrons were ejected no matter how high the intensity of light.

Another crucial experimental observation was the emmision spectrum of hydrogen (that is, the glowing of hydrogen gas in a tube when a potential difference is applied across the ends of the tube). According to classical physics, there should be an infinite number of colors in the emmision spectrum of the hydrogen gas. But experiments consistently revealed that the spectrum of the hydrogen gas is discrete.

These two observations, as well as several others, led to the conclusion that when light is isolated, it behaves as a wave; but on the other hand, when light interacts with matter, it behaves as particles (called photons) and its energy can exist only in discrete chunks. Each chunk of energy is called a quantum of energy and its value is $h\nu$ where h is the Planck's constant and ν is the frequency of the light. This extraordinary discovery called for a new type of physics to describe matter at the microscopic level and this is what gave birth to *Quantum Mechanics*

Basic Postulates of Quantum Mechanics

There are four postulates that serve to formalize the rules of quantum mechanics. These are:

Postulate I

To any well-defined observable in physics (call it A), there corresponds an operator (call it $\hat{A}$) such that measurement of A yield values (call the values a) which are are eigenvalues of $\hat{A}$. In other words, measurement of the observable A can be expressed mathematically as

$$\hat{A}\phi = a\phi$$

where ϕ (called the eigenfunction of $\hat{A}$) is a function representing the state of the system at the time of the measurement. Examples of physical observables are momentum, velocity and energy. Two examples of operators are:

Momentum: $-i\hbar\Delta = -i\hbar\left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right) = \hat{\mathbf{p}}$

Energy: $\hat{H} = \dfrac{\hat{p}^2}{2m} + V(\mathbf{r})$

Postulate II

Measurement of an observable A which yields the value a leaves the system in the state ϕ_a (where ϕ_a is the eigenfuction of A which corresponds to a).

Postulate III

The state of a system at any time can be represented by a state function ψ which is finite, single valued, continous and differentiable. Moreover, every information regarding this syetem can be obtained from ψ. In particular, the average value of any physical observable A is

$$\langle A\rangle = \int \psi^* \hat{A}\psi \, d\vec{r}$$

where ψ^* denotes the complex conjugate of ψ

Postulate IV

The state function ψ of a system evolves in time according to the equation

$$i\hbar\frac{\partial}{\partial t}\psi(\vec{r},t) = \hat{H}\psi(\vec{r},t)$$

This equation is known as the *time-independent Schrödinger's equation.*

The Dirac Notation

In this notation, the state function ψ is represented as $|\psi\rangle$ (called the "ket" vector) and ψ^* is represented by $\langle\psi|$ (called the "bra" vector). Also the inner product of two functions ψ and ϕ is represented by

$$\langle\psi|\phi\rangle = \int \psi^*\phi \, d\vec{r}$$

and the average value of the observable A is represented by

$$\langle\psi|\hat{A}|\psi\rangle = \int \psi^* \hat{A}\psi \, d\vec{r}$$

The Superposition Principle

Consider a system in a state ψ. Let A be an observable and $\hat{A}$ the operator corresponding to A. Then the set of all possible eigenfunctions of $\hat{A}$ form a vector space known as the *Hilbert space.* Suppose that the basis of this vector space is $\{\phi_1, \phi_2, \phi_3, \dots, \phi_n\}$, then every possible state ψ of the system can be written expressed in the form

$$\begin{aligned}\psi &= a_1\phi_1 + a_2\phi_2 + \cdots + a_n\phi_n \\ &= \sum_{i=1}^{n} a_i\phi_i\end{aligned}$$

This is known as the superposition principle. When a reversible operation is applied to the state ψ, all the terms of the superposition are preserved. But when a measurement is made on ψ, it projects the state to one of the elements of the superpostion, say ϕ_k and the probabilty of obtaining ϕ_k is given by

$$P_k = \frac{b_k^* b_k}{\sum\limits_{i=1}^{n} b_i^* b_i}$$

and all the other eigenfunctions in the superposition are irreverisbly destroyed. If the basis form an orthonormal set, then the probability is simply given by

$$P_k = b_k^* b_k = |b_k|^2$$

This interpretation of the meaning of the state function is known as the Copenhagen interpretation because it was first advocated by Neils Bohr who is a native of Copenhagen.

6. Quantum Computing

Each unit of the memory of a classical computer is made up of a two-state device the state of which is denoted by the binary digits 0 and 1 (called bits). The *Quantum Computer* maintains this binary feature by using a device whose state is represented by the quantum states $|0\rangle$ and $|1\rangle$. These "quantum bits" are called *qubits*. But the quantum computer has the additional capability of expressing the registers of its memory as a complex superpostion of *all possible classical inputs*.

In order to illustrate the idea of quamtum computation we will use a specific example. This example involves the calculation of a function of a variable x. Suppose that x ranges from 0 to $2^m - 1$ and that $f(x)$ ranges from 0 to $2^n - 1$, then the computer (classical or quantum) will need a register of size $(m + n)$ for the computation. As far as computation is concern we can let m bits of the register represent the input part of the register (denoted by the subscript (i)) and n bits represent the output part of the register (denoted by the subscript (o). Let $\hat{F}$ denote the operation corresponding to the calculation of f. Then the function calculation can be represented in quantum mechanics notation as:

$$\hat{F}|x\rangle_i|0\rangle_o = |x\rangle_i|f(x)\rangle_o$$

The calculation is done as follows:

The output register is initialized to 0. The logic gate corresponding to $\hat{F}$ goes to the input part of the register, calculates $f(x)$ and stores it in the output register, leaving x in the input part of the register to ensure reversibility of the computation. The classical computer will compute $f(x)$ for each of the 2^m possible value of x.

On the other hand the quantum computer expresses the initial state of the input part of the register as a *superposition of all the* 2^m *clasical inputs* . Therefore, prior to the computation of f, the register of the quantum computer is in the state:

$$|\psi\rangle_i|0\rangle_o = \frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle_i|0\rangle_o$$

The computation of f is done by applying the operator $\hat{F}$ to the above state and this is represented in quantum mechanics notation as

$$\hat{F}|\psi\rangle_i|0\rangle_o = \frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle_i|f(x)\rangle_o \tag{1}$$

Thus all the 2^m possible values of $f(x)$ have been calculated simultaneously!

It must be noted that even though all possible values of $f(x)$ have been obtained in one run, measurement on the final state will yield only one of the superposition state (1) at random with a probabilty of $1/2^m$ and all the others are irreversibly destroyed. This is because, measurement (which is the only way of obtaining information from a quantum computer) is an irreversible operation.

Therefore we cannot have an access to all the possible values of $f(x)$ in one measurement. Nevertheless, this idea can be used to attack certain problems which are deemed impracticable to solve with classical computer. The strategy here is that sometime the problem we want to solve may involve only one or a few values of f. Therefore if we can manipulate the amplitudes of various terms in the state (1) to increase our chances of obtaining the right answer, then obviously the quantum computer can solve the problem in fewer number of steps than a classical computer.

7. Shor's algorithm for solving the *Discrete Logarithm* problem

For every prime p, the multiplicative group (mod p) is cyclic, that is, there exists an element g such that every element x, of the group can be expressed as $x = g^i$, where $0 \leq i \leq p-2$. As an example take the prime $p = 7$, then the multiplicative group can be chosen as $U(7) = \{1, 2, 3, 4, 5, 6\}$. It can be observed that $\{3^0, 3^1, 3^2, 3^3, 3^4, 3^5\} = \{1, 3, 2, 6, 4, 5\} = U(7)$. The *discrete logarithm* of a number x with respect to p and g is the integer r with $0 \leq r \leq p-2$ such that $g^r \equiv x \bmod(p)$. The following algorithm, discovered by Peter Shor, shows how a quantum computer finds discrete logarithms by the use of two modular exponentiations coupled with two discrete Fourier transforms.

This algorithm requires the use of three registers. We first find an integer q such that q is a power of 2 and $p < q < 2p$. We prepare the first two registers as a *uniform superposition of all possible classical imputs* $|a\rangle$ and $|a\rangle$ (mod $(p-1)$), and initialize the third register to 0. This leaves the quantum computer in the state:

$$\frac{1}{p-1}\sum_{a=0}^{p-2}\sum_{b=0}^{p-2}|a\rangle|b\rangle|0\rangle$$

We then create a gate or a series of gates that receives the contents of the first two registers as inputs and computes $g^a x^{-b}$ and puts it in the third register. Let $\hat{\mathrm{F}}$ be the operator corresponding to the computation of $g^a x^{-b}$, then the process can be represented by:

$$\hat{F}\frac{1}{p-1}\sum_{a=0}^{p-2}\sum_{b=0}^{p-2}|a\rangle|b\rangle|0\rangle = \frac{1}{p-1}\sum_{a=0}^{p-2}\sum_{b=0}^{p-2}|a\rangle|b\rangle|g^a x^{-b} \bmod p\rangle$$

We then use the reversible quantum Fourier transforms

$$|a\rangle \longrightarrow \frac{1}{q^{1/2}}\sum_{c=0}^{q-1}\exp(\frac{2\pi iac}{q})|c\rangle$$

and

$$|b\rangle \longrightarrow \frac{1}{q^{1/2}}\sum_{d=0}^{q-1}\exp(\frac{2\pi ibd}{q})|d\rangle$$

to effect the transformation

$$|a\rangle|b\rangle \longrightarrow \frac{1}{q}\sum_{c=0}^{q-1}\sum_{d=0}^{q-1}\exp(\frac{2\pi i}{q}(ac+bd))|c\rangle|d\rangle$$

This leaves the quantum computer in the entangled state

$$|\psi\rangle_i = \frac{1}{(p-1)q}\sum_{a=0}^{p-2}\sum_{b=0}^{p-2}\sum_{c=0}^{q-1}\sum_{d=0}^{q-1}\exp(\frac{2\pi i}{q}(ac+bd))|c\rangle|d\rangle|g^a x^{-b} \bmod p\rangle \qquad (2)$$

We then make a measurement on the computer and this removes the entanglement in the state (1). Suppose that measurement gives the result

$$|c\rangle|d\rangle|g^k(\text{mod}p)\rangle$$

then the quantum computer is left in the state

$$|\psi\rangle_f = \frac{1}{(p-1)q}|c\rangle|d\rangle \sum_{\substack{(a,b)\\ a-rb\equiv k}} \exp(\frac{2\pi i}{q}(ac+bd))|g^a x^{-b}(\text{mod}p)\rangle \tag{3}$$

where the summation is done over all (a, b) such that

$$g^a x^{-b} \equiv g^k(\text{mod}p) \tag{4}$$

By combining condition (4) with $g^r \equiv x(\text{mod}p)$, we see that the constraint on (a, b) is that
$g^a g^{-br} \equiv g^k(\text{mod}p)$ which simplifies to

$$g^{a-br} \equiv g^k(\text{mod}p) \tag{5}$$

By using the fact that the powers of g are elements of Z_{p-1}, the constraint (4) implies that
$a - rb \equiv k(\text{mod}(p-1))$; therefore, we can express a as

$$a = br + k - (p-1)\lfloor\frac{br+k}{p-1}\rfloor \tag{6}$$

Substituting for x in equation (3), the final state of the quantum computer after the measurement is:

$$|\psi\rangle_f = \frac{1}{(p-1)q}|c\rangle|d\rangle \sum_{b=0}^{p-2} \exp(\frac{2\pi i}{q}(bd+brc+kc-c(p-1)\lfloor\frac{br+k}{p-1}\rfloor))|g^a x^{-b}(\bmod p)\rangle \tag{7}$$

We will now show that if c and d satisfy certain conditions, then there is a reasonable probabilty of deducing the value of r from (c, d) and the known parameters p and q. From equation (7), the probability,$P(c, d, k)$, of observing the state $|c\rangle|d\rangle|g^k \bmod p\rangle$ is:

$$P(c,d,k) = \left| \frac{1}{(p-1)q} \sum_{b=0}^{p-2} \exp(\frac{2\pi i}{q}(bd+brc+kc-c(p-1)\lfloor\frac{br+k}{p-1}\rfloor)) \right|^2 \tag{8}$$

By splitting the argument of the exponential in equation (8), the probability can be written as

$$P(c,d,k) = \left| \frac{1}{(p-1)q} \sum_{b=0}^{p-2} \exp(\frac{2\pi i}{q}bT)\exp(\frac{2\pi i}{q}V) \right|^2 \tag{9}$$

where

$$T = rc + d - \frac{r}{p-1}\{c(p-1)\}_q \tag{10}$$

and

$$V = \left(\tfrac{br}{p-1} - \left\lfloor \tfrac{br+k}{p-1} \right\rfloor\right)\{c(p-1)\}_q \tag{11}$$

where $\{c(p-1)\}_q = c(p-1) \bmod q$ and $-q/2 < \{c(p-1)\}_q \leq q/2$. We then classify the output of the measurement as "good" if c and d satisfy the following condition:

$$|\{T\}_q| = |rc + d - \frac{r}{p-1}\{c(p-1)\}_q - q\lfloor T/q \rfloor| \leq \frac{1}{2} \tag{12}$$

$$\{c(p-1)\}_q \leq q/12 \tag{13}$$

If condition (12) holds, then the phase of $\exp(\frac{2\pi i}{q}bT)$ in equation (9) is at most π. Furthermore if the condition (13) holds, then the variation of the phase of $\exp(\frac{2\pi i}{q}bT)$ due to the factor $\exp(\frac{2\pi i}{q}V)$ is at most $\pi/6$. Under these conditions, as b ranges from 0 to $p-2$, the phase of $\exp(\frac{2\pi i}{q}bT)$ ranges from 0 to $2\pi i W$ where

$$W = \frac{p-2}{q}\left(\ rc + d - \tfrac{r}{p-1}\{c(p-1)\}_q - \lfloor \tfrac{T}{q} \rfloor\ \right) \tag{14}$$

The component of $\exp(\frac{2\pi i}{q}bT)$ in the direction of $\exp(\pi i W)$ is given by:

$$(\exp(\pi i W))^* \exp(\frac{2\pi i}{q}bT) = \exp(2\pi i W \frac{b}{p-2} - \pi i W) \tag{15}$$

Therefore the least value of this component is $\cos(2\pi|W/2 - Wb/(p-2)|)$. Also, by using the condition (13) the phase of this component can vary by at most $\pi/6$; therefore, the least value of the summand in equation (9) is

$$\cos(2\pi|W/2 - Wb/(p-2)| + \tfrac{\pi}{6}) \tag{16}$$

Thus,

$$P(c,d,k) \geq \left|\ \frac{1}{(p-1)q}\sum_{b=0}^{p-2}\cos(2\pi|W/2 - Wb/(p-2)| + \frac{\pi}{6})\ \right|^2 \tag{17}$$

Since p is usually very large, we can replace the discrete sum with an integral, and this gives

$$P(c,d,k) \geq \left|\ \frac{1}{(p-1)q}\int_0^{p-2}\cos(2\pi|W/2 - Wb/(p-2)| + \frac{\pi}{6})db\ \right|^2 + O\left(\ \tfrac{W}{pq}\ \right) \tag{18}$$

where

$$O\left(\ \tfrac{W}{pq}\ \right)$$

is an error term due to the approximation. By using the substitution

$$u = \left|\ \tfrac{W}{2} - \tfrac{Wb}{p-2}\ \right| + \frac{\pi}{6}$$

we have

$$P(c,d,k) \geq \left(\ \frac{1}{q}\frac{2}{\pi}\int_{\pi/6}^{2\pi/3}\cos u\, du\ \right)^2 \tag{19}$$

Evaluating the integral gives

$$P(c,d,k) \geq 0.054/q^2 > 1/(20q^2)$$

In order to find the number of good pairs (c,d) available, we first note from condition (12) that for each value of c there exists only one d. Therefore, the number of good pairs (c,d) is the same as the number of possible c's. Since there are q c's, it follows that there are q pairs (c,d) which satisfy the condition (12). Also, there are at least $q/12$ pairs (c,d) satisfying the condition (13); therefore, there are at least $q/12$ pairs satisfying both conditions (12) and (13). So, there are at least $q/12$ good pairs (c,d). For each of these pairs there are $p-1$ possible g^k; therefore, there are at least $(p-1)q/12$ good states $|c\rangle|d\rangle|g^k(\text{mod}p)\rangle$. The probability of observing some good state is at least $P(c,d,k) \times (p-1)q/12$ ie. at least

$$1/(20q^2) \times pq/12 = p/240q$$

Thus if we choose q to be close to p, then there is a realistic chance of finding some good states from which r can be deduced.

Recovering r from the good pair (c,d):

The condition (12) can be rewritten as

$$-\frac{1}{2} \leq rc + d - \frac{r}{p-1}\{c(p-1)\}_q - qj \leq \frac{1}{2} \tag{20}$$

where j is the closest integer to T/q. Dividing through (20) by q and rearranging we obtain

$$-\frac{1}{2q} \leq \frac{d}{q} + r\left(\frac{c(p-1) - \{c(p-1)\}_q}{q(p-1)} \right) - j \leq \frac{1}{2q} \tag{21}$$

This further reduces to

$$-\frac{1}{2q} \leq \frac{d}{q} + r\left(\frac{c(p-1) - \{c(p-1)\}_q}{q(p-1)} \right) \leq \frac{1}{2q} \qquad (\text{mod } 1) \tag{22}$$

It should be noted that the mod 1 reduces the number between the inequality signs to a proper fraction.

It should also be noted that q divides $c(p-1) - \{c(p-1)\}_q$; therefore, the coefficient of r is a fraction with denominator $p-1$. A candidate r is recovered by approximating d/q to the nearest multiple of $1/(p-1)$ and dividing the result $(\text{mod}p-1)$ by the number

$$c' = \frac{c(p-1) - \{c(p-1)\}_q}{q}$$

After finding a candidate r, we then plug in the values (r,c,d) into the relations (12) and (13). If both conditions are satisfied, then there is a reasonable chance that the result is accurate. If the conditions are not satisfied, then we will run the quantum computer again. We will continue until we obtain enough good states.

8. Application to Elliptic Curves

Attacking the Elliptic Curve Analog of the Massey-Omurra Cryptosystem:

In this cryptosystem, if Alice wants to send a message P_m(ie, a point on E with the message m hidden in it) to Bob, she first sends $e_A P_m$ to Bob. Bob then performs the operation $e_B e_A P_m(\text{ mod } N)$ and sends the result back to Alice. Alice then performs the operation $d_A e_B e_A P_m = e_B P_m(\text{mod}N)$. She then sends the result to Bob who finally recovers the message by performing the operation $d_B e_B P_m = P_m(\text{mod}N)$. The evesdropper Charlie,only sees $e_A P_m$, $e_B e_A P_m$ and $e_B P_m$ when he attempts to intercept the communication between Alice and Bob. In order to recover the point P containing the message, he needs to know either e_A or e_B. If he takes the x in Shor's algorithm to be $X = e_B e_A P_m$, g to be $G = e_B P_m$, and the operation to be addition of points on the elliptic curve, then he needs to solve the discrete logarithm problem

$$r(\text{mod}N)G = X \tag{23}$$

for r. We can show that $r = e_A$ by sustituting the vaules of G and X in equation (23). This gives

$$re_B P_m = e_B e_A P_m \tag{24}$$

From equation (24) we see that $r = e_A(\text{mod}N)$.

It must be noted that this discrete logarithm problem is analogous to the one in the multiplicative group mod$p$. The main difference is:

In the the case of the elliptic curve cryptosystem, we are working with an additive group as opposed to the multiplicative group in Shor's algorithm. The operation of this additive group is the addition of points on the elliptic curve which was defined in the previous sections of this paper.

The Shor's algorithm can be modified to solve this problem. In order to show how it works we first prepare the first two registers in the uniform superposition of *all possible classical imputs* $|a\rangle$ and $|b\rangle(\text{ mod } N)$. We then create a gate that receives the contents of the first two registers as inputs and computes $aG-bX$ and puts it in the third register. It must be noted that the third register could be more than one in this case since we are dealing with points with two coordinates instead of single numbers. Let $\hat{H}$ be the operator corresponding to the computation of $aG - bX$, then the process can be represented by:

$$\hat{H}\frac{1}{N}\sum_{a=0}^{N-1}\sum_{b=0}^{N-1}|a\rangle|b\rangle|0\rangle = \frac{1}{N}\sum_{a=0}^{N-1}\sum_{b=0}^{N-1}|a\rangle|b\rangle|aG - bX\rangle$$

We then use the reversible quantum Fourier transforms

$$|a\rangle \longrightarrow \frac{1}{q^{1/2}}\sum_{c=0}^{q-1}\exp(\frac{2\pi iac}{q})|c\rangle$$

and

$$|b\rangle \longrightarrow \frac{1}{q^{1/2}}\sum_{d=0}^{q-1}\exp(\frac{2\pi ibd}{q})|d\rangle$$

as before, to effect the transformation

$$|a\rangle|b\rangle \longrightarrow \frac{1}{q}\sum_{c=0}^{q-1}\sum_{d=0}^{q-1}\exp(\frac{2\pi i}{q}(ac+bd))|c\rangle|d\rangle$$

This leaves the quantum computer in the entangled state

$$|\psi\rangle_i = \frac{1}{Nq}\sum_{a=0}^{N-1}\sum_{b=0}^{N-1}\sum_{c=0}^{q-1}\sum_{d=0}^{q-1}\exp(\frac{2\pi i}{q}(ac+bd))|c\rangle|d\rangle|aG-bX\rangle \tag{25}$$

Suppose we observe computer and find the result

$$|c\rangle|d\rangle|Y\rangle$$

where $Y = kG$ then the quantum computer is left in the state

$$|\psi\rangle_f = \frac{1}{Nq}|c\rangle|d\rangle \sum_{\substack{(a,b)\\ a-rb\equiv k}} \exp(\frac{2\pi i}{q}(ac+bd))|aG-bX\rangle \tag{26}$$

where the summation is done over all (a,b) such that

$$aG - bX \equiv kG \tag{27}$$

In order to deduce the value of r from the pair (c,d) we need to establish the condition under which (c,d) constitutes a "good" pair. These conditions are easily obtained by making the transformation $(p-1) \longrightarrow N$ in the equations (10), (12) and (13) in the algorithm for the multiplicative group (mod$p$) above. This transformation gives the conditions:

$$|\{T\}_q| = |rc + d - \frac{r}{N}\{cN\}_q - q\lfloor T/q\rfloor| \le \frac{1}{2} \tag{28}$$

and

$$\{cN\}_q \le q/12 \tag{29}$$

Where

$$T = rc + d - \frac{r}{N}\{cN\}_q \tag{30}$$

and $\{cN\}_q = cN \bmod q$ and $-q/2 < \{cN\}_q \le q/2$

Using the same transformation as the one above, the condition (22) in the previous algorithm becomes

$$-\frac{1}{2q} \le \frac{d}{q} + r\left(\frac{cN - \{cN\}_q}{qN}\right) \le \frac{1}{2q} \qquad (\text{mod } 1) \tag{31}$$

A candidate r is recovered by approximating d/q to the nearest multiple of $1/N$ and dividing the result by the number

$$c' = \frac{cN - \{cN\}_q}{q}$$

After finding a candiadte r, we then plug in the values (r,c,d) into the relations (28) and (29). If both conditions are satisfied, then there is a reasonable chance that the result is accurate. If the conditions are not satisfied, then we will run the quantum computer again. We will continue until we obtain enough good states.

8.1 Example of Elliptic Curve Breakdown

We will again use the elliptic curve (characteristic 2): $y^2 + y = x^3$. We will now use the finite field: $\boldsymbol{F}_{32} = \boldsymbol{F}_{2^5} = \{0, 1, \alpha, \alpha^2, \alpha^3, \alpha^4 ... \alpha^{30}\}$ where $\alpha^5 = \alpha^2 + 1$ and α is the generator.

$N = 2^r + 1 = 2^5 + 1 = 33$

The thirty-three points that satisfy this equation are:

$(0,0), (0,1), (\alpha^2, \alpha^{14}), (\alpha^4, \alpha^{26}), (\alpha^5, \alpha^{25}), (\alpha^5, \alpha^{21}), (\alpha^9, \alpha^{13}), (\alpha^{10}, \alpha^{11}), (\alpha^{10}, \alpha^{19}), (\alpha^{11}, \alpha^6), (\alpha^{11}, \alpha^{27}), (\alpha^{13}, \alpha^{15}), (\alpha^{13}, \alpha^{24}), (\alpha^{15}, \alpha^4), (\alpha^{15}, \alpha^{10}), (\alpha^{18}, \alpha^{28}), (\alpha^{20}, \alpha^7), (\alpha^{20}, \alpha^{22}), (\alpha^{21}, \alpha^3), (\alpha^{21}, \alpha^{29}), (\alpha^{22}, \alpha^{12}), (\alpha^{22}, \alpha^{23}), (\alpha^{23}, \alpha^2), (\alpha^{23}, \alpha^5), (\alpha^{26}, \alpha^{17}), (\alpha^{26}, \alpha^{30}), (\alpha^{27}, \alpha), (\alpha^{27}, \alpha^{18}), (\alpha^{29}, \alpha^9), (\alpha^{29}, \alpha^{16}), (\alpha^{30}, \alpha^8), (\alpha^{30}, \alpha^{20}), \mathcal{O}$

To add two different points, P_1 and P_2, on this curve, we will use the equations that we have seen before, modified to work in the finite field $\boldsymbol{F}_{32}$ with α as the generator:

$P_1 + P_2 = P_3 = (x_3, y_3)$

$x_3 = -x_1 - x_2 + A^2 + cA \ (c = 0) \Rightarrow x_1 + x_2 + A^2$

$y_3 = cx_3 - d - y_1 + A(x_1 - x_3) \ (c = 0, d = 1) \Rightarrow 1 + y_1 + A(x_1 - x_3)$

$A = (y_2 - y_1)/(x_2 - x_1) \Rightarrow (y_2 + y_1)(x_2 + x_1)^{30}$

We will again use the special duplication formula: $(2P) = (x^4, y^4 + 1)$

Alice's random integer and its inverse: $e_A = 5, \ d_A = 20$
Bob's random integer and its inverse: $e_B = 7, \ d_B = 19$

Message Point: $P_m = (\alpha^{15}, \alpha^{10})$

$e_A P_m = 5(\alpha^3, \alpha^4) = (\alpha^9, \alpha^{14})$ (Alice's encrypted message)

$e_A e_B P_m = 7(\alpha^9, \alpha^{14}) = (\alpha^{29}, \alpha^{16})$

$e_A e_B d_A P_m = 20(\alpha^{29}, \alpha^{16}) = (\alpha^{18}, \alpha^{26})$

which equals $e_B P_m = 7(\alpha^{15}, \alpha^{10}) = (\alpha^{18}, \alpha^{26})$

$e_B d_B P_m = 19(\alpha^{18}, \alpha^{26}) = (\alpha^{15}, \alpha^{10})$ (Bob's decrypted message)

which equals P_m.

Charlie, the evesdropper sees only $e_A P_m = (\alpha^9, \alpha^{14})$, $e_A e_B P_m = (\alpha^{29}, \alpha^{16})$ and $e_B P_m = (\alpha^{18}, \alpha^{26})$. In order to capture the message P_m he only needs to know e_A; and he can do this by solving the discrete logarithm problem $r (\text{mod} N) G = X$where $N = 33$ is the number of points on the elliptic curve and $G = e_B P_m = (\alpha^{18}, \alpha^{26})$ and $X = e_A e_B P_m = (\alpha^{29}, \alpha^{16})$. To use the quantum computer to solve this problem Charlie prepares the registers of the computer as

$$|\psi\rangle_i = \frac{1}{33} \sum_{a=0}^{32} \sum_{b=0}^{32} |a\rangle |b\rangle |aG - bX\rangle$$

A few examples of some of the terms in the summation sign are listed below:
$|1\rangle |31\rangle |(0,0)\rangle$,
$|2\rangle |32\rangle |(\alpha^{11}, \alpha^6)\rangle$,
$|2\rangle |31\rangle |(\alpha^5, \alpha^{25})\rangle$,
$|1\rangle |32\rangle |(\alpha^{13}, \alpha^{15})\rangle$. He then chooses the q in Shor's algorithm as $q = 64$ The quantum computer then uses the Quantum Fourier transform

$$|a\rangle |b\rangle \longrightarrow \frac{1}{33(64)} \sum_{c=0}^{63} \sum_{d=0}^{63} \exp(\frac{2\pi i}{64}(ac + bd)) |c\rangle |d\rangle$$

to transform the state $|\psi\rangle_i$ to

$$|\psi\rangle_f = \frac{1}{33(64)} \sum_{a=0}^{32}\sum_{b=0}^{32}\sum_{c=0}^{63}\sum_{d=0}^{63} \exp(\frac{2\pi i}{64}(ac+bd))|c\rangle|d\rangle|aG - bX\rangle$$

The quantum computer evaluates each of the 2112 terms in the summation and stores them as a single entangled state; this is something a classical computer cannot do.

Suppose Charlie observes the computer's memory and finds the result corresponding to the (c,d) pair $(4,45)$ and the (a,b) pair $(2,32)$, ie, suppose that he finds the state

$$|\phi\rangle = \frac{1}{2112}\exp[\frac{2\pi i}{64}(2\times 4) + (32\times 45)]|4\rangle|45\rangle|(\alpha^{11},\alpha^6)\rangle$$

Then he can deduce a candidate r by using the relations

$$-\frac{1}{2q} \leq \frac{d}{q} + r\left(\frac{cN - \{cN\}_q}{qN}\right) \leq \frac{1}{2q} \qquad (32)$$

This can be written in a more compact form as

$$-\frac{1}{2q} \leq \frac{d}{q} + \frac{rc'}{N} \leq \frac{1}{2q}$$

where

$$c' = \frac{cN - \{cN\}_q}{q} \qquad (33)$$

He first rounds $\frac{d}{q} = \frac{45}{64}$ to the nearest multiple of $\frac{1}{N} = \frac{1}{33}$. This gives $\frac{45}{64} \approx \frac{23}{33}$ By plugging the values of c, N and q in equation (2), he obtains

$$c' = \frac{4(33) - \{4(33)\}_{64}}{64} = 2$$

He then divides $\frac{23}{33}$ by 2 in the multiplicative group mod 33. He notes that $\frac{23}{33} = 23 \bmod 33$; therefore he finds a r by dividing 23 by 2 in the multiplicative group mod 33, ie.

$$\begin{aligned} r &= 23\times 2^{-1} \\ &= 23\times 17 \\ &= 28 \\ &= -5 \bmod 33 \end{aligned}$$

It should be noted that Charlie obtains the negative of the result he is looking for. Assuming that he ignores the negative sign and proceeds with the absolute value of the result, ie. 5, then the next step is to find out whether his values for c and d constitute a "good" pair. This can be done by checking whether the following conditions are satisfied:

$$|\{T\}_q| = |rc + d - \frac{r}{N}\{cN\}_q - jq| \leq \frac{1}{2} \qquad (34)$$

and

$$\{cN\} \leq q/12 \tag{35}$$

where $\{cN\}_q = cN(\mathrm{mod} q)$, $-q/2 < \{cN\}_q \leq q/2$ and j is the closest integer to T/q

$$T = rc + d - \frac{r}{N}\{cN\}_q \tag{36}$$

When Charlie plugs in the values $r = 5$, $c = 4$, $N = 33$, $d = 45$, and $q = 64$ in the equation (36), he obtains $T = 64.4$ which implies that $j = 1$. The relation (35) is satisfied since $\{4(33)\}_{64} = 4 < 64/12$. When he plugs in the values $r = 5$, $c = 4$, $N = 33$, $d = 45$, $q = 64$, and $j = 1$ in relation (34) he gets:

$$\begin{aligned} |\{T\}_{64}| &= |5(4) + 45 - \frac{5}{33}\{4(33)\}_{64} - 1(64)| \\ &= 0.40 \\ &\leq \frac{1}{2} \end{aligned}$$

Since both conditions (34) and (35) are satisfied, Charlie concludes that $(4, 45)$ constitutes a good (c, d) pair. Therefore, he records $r = 5$ and runs the quantum computer a few more times to find more values for r.

Second Run

Suppose Charlie runs the computer the second time and obtains the (c, d) pair $(2, 54)$. Then he can deduce r from the relation (32) by rounding $\frac{d}{q} = \frac{2}{64}$ to the nearest multiple of $\frac{1}{33}$. This gives $\frac{54}{64} \approx \frac{28}{33}$. Also,

$$c' = \frac{2(33) - \{2(33)\}_{64}}{64} = 1$$

So he obtains a candidate r as $r = \frac{28}{33}$ in the multiplicative group (mod 33) and this gives $r = -5$

To check whether this (c, d) pair is good or not, he plugs in the values $c = 2$, $d = 54$, $r = 5$, $N = 33$ and $q = 64$ into equation (36) and obtains $T = 63.7$ which implies $j = 1$. The relation (35) is satisfied since $\{2(33)\}_{64} = 2 < 64/12$. When he plugs in the values $r = 5$, $c = 2$, $N = 33$, $d = 54$, $q = 64$, and $j = 1$ in relation (34) he gets:

$$\begin{aligned} |\{T\}_{64}| &= |5(2) + 54 - \frac{5}{33}\{2(33)\}_{64} - 1(64)| \\ &= 0.30 \\ &\leq \frac{1}{2} \end{aligned}$$

Since both conditions (34) and (35) are satisfied, Charlie concludes that $(2, 54)$ constitutes a good (c, d) pair. Therefore, he records $r = 5$.

Third Run

Suppose Charlie runs the computer the third time and obtains the (c, d) pair $(37, 8)$. Then he can deduce r from the relation (32) by rounding $\frac{d}{q} = \frac{8}{64}$ to the nearest multiple of $\frac{1}{33}$. This gives $\frac{8}{64} \approx \frac{4}{33}$. Also,

$$c' = \frac{37(33) - \{37(33)\}_{64}}{64} = 19$$

So he obtains a candidate r by dividing $r = \frac{4}{33}$ by 19 in the multiplicative group (mod 33) and this gives

$$\begin{aligned} r &= 4 \times 19^{-1} \\ &= 4 \times 7 \\ &= 28 \\ &= -5 \bmod 33 \end{aligned}$$

To check whether this (c, d) pair is good or not, he plugs in the values $c = 37$, $d = 8$, $r = 5$, $N = 33$ and $q = 64$ into equation (36) and obtains $T = 192.2$ which implies $j = 3$. The relation (35) is satisfied since $\{37(33)\}_{64} = 5 < 64/12$. When he plugs in the values $r = 5$, $c = 37$, $N = 33$, $d = 54$, $q = 64$, and $j = 3$ in relation (34) he gets:

$$\begin{aligned} |\{T\}_{64}| &= |5(37) + 8 - \frac{5}{33}\{37(33)\}_{64} - 3(64)| \\ &= 0.24 \\ &\leq \frac{1}{2} \end{aligned}$$

Since both conditions (34) and (35) are satisfied, Charlie concludes that $(37, 8)$ constitutes a good (c, d) pair. Therefore, he records $r = 5$.

Fourth Run

Suppose Charlie runs the computer the fourth time and obtains the (c, d) pair $(35, 17)$. Then he can deduce r from the relation (32) by rounding $\frac{d}{q} = \frac{17}{64}$ to the nearest multiple of $\frac{1}{33}$. This gives $\frac{17}{64} \approx \frac{9}{33}$. Also,

$$c' = \frac{35(33) - \{35(33)\}_{64}}{64} = 18$$

Since 18 is not in the multiplicative group mod 33, we cannot deduce the value of r from the (c, d) pair $(35, 17)$.

But Charlie will run the quantum computer a few more times till he has enough evidence to make a conclusion about the value of r. In this example, if we assume that running the computer four times is enough, then Charlie will conclude that $r = 5$ and so $e_A = 5$. Once he knows e_A, he can calculate $d_A = e^{-1}(mod 33)$ and use it to decrypt the message from $e_A P_m$.

9. Conclusion and comments

In the above article we have given a brief introduction to elliptic curves and some elliptic curve cryptosystems. We have also shown how a quantum computer will break down the security of the elliptic curve cryptosystem, that is, by solving an elliptic curve discrete logarithm problem in an amount of time which scales as a polynomial function of the size of the input. But we must say that the practical quantum computer is still something in the future and at the moment researchers continue to work on possible ways of making it a reality. Once an actual quantum computer is built, RSA and the elliptic curve cryptosystems will lose their security, a security which have been presumed to depend on the difficulty of factorization and the discrete logarithms respectively. This will create a great demand for the *Quantum Cryptography* which has been proved to be more secured than the two cryptosystems mentioned above.

In describing how a quantum computer will solve the discrete logarithm problem we adopted Peter Shor's algorithm for solving discrete logarithm problem in the multiplicative group (mod p). However, there are a few concerns that we need to point out. First, the number N (the number of points on the elliptic curve) used in the elliptic curve analog of Shor's algorithm must be a prime in order for the algorithm to work exactly as proposed by Shor. But since we can choose from an infinite number of elliptic curves, maybe we can choose the curve in such a way that, the number of points on the curve is prime; we leave this open to investigation.

Another thing that came to our notice is that, in the original version of Shor's algorithm for solving discrete logarithm problem (in the multiplicative group mod p), in recovering r from the pair (c, d), Shor uses $\text{mod}(p-1)$ where p is a prime. However, if p is prime, then $p-1$ is even for $p > 2$ and so at least two numbers between 1 and $p-1$ divide $p-1$ and therefore do not have multiplicative inverses. This will cause a problem when the quantum computer attempts to deduce the value of r. This calls for an adjustment of the algorithm.

The third and final comment is about the problem that arose in the given example where Charlie, the evesdropper, tried to use the quantum computer to solve the discrete logarithm problem. Instead of obtaining $r = 5(\text{mod}33)$, he consistently obtained -5. Maybe something is not working quite right in the algorithm.

References

[1] Certicom home page, www.certicom.com.

[2] Koblitz, Neal. *A Course in Number Theory and Cryptography.* New York: Springer-Verlag, 1987.

[3] Koblitz, Neal "Elliptic Curve Cryptosystems." *Mathematics of Computation.* 1987. vol. 48, no. 177, 203-209.

[4] Richard L. Liboff. *Introductory Quantum Mechanics.* Addison-Wesley Publishing Company, 2nd edition.

[5] RSA home page, www.rsa.com, quote.

[6] Shor, Peter W. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer." *Proceedings of the 35th Annual Symposium on Foundations of Computer Science.* 1994. IEEE Computer Society Press, 124-134.

[7] Silverman, Joseph H., and John Tate. *Rational Points on Elliptic Curves.* New York: Springer-Verlag, 1992.

[8] Spiller, Timothy P. *Quantum Information Processing: Cryptography Computation and Teleportation.* Bristol, UK: Hewlett-Packard Labratories, 1996.