

# Aspects of Elliptic Curve Cryptography and Schoof's Algorithm

Submitted to the Mathematics Department of  
the United States Naval Academy in Partial  
Fulfillment of the Requirements for a Degree  
in Mathematics with Honors

by Blake Wanier  
23 April 2007

---

Accepted by the Honors Committee

---

Advisor

---

Chairman

---

First Reader

---

Chairman of Mathematics Department

---

Second Reader

---

Date

## Introduction

Since the beginning of society, there has been a need for information to be passed swiftly and securely. To solve this problem people searched for ways to mask what they were communicating. Cryptology is the composition of cryptography and cryptanalysis, or the making and breaking of codes. One of the earliest uses of cryptography was in the Roman Empire where they simply shifted the alphabet, so that each letter stood for another letter a certain number of positions down the alphabet. While it would not be a secure cipher today, it was adequate for the day. This project outlines the basic theory of a modern cipher by analyzing elliptic curve cryptography. It also discusses Rene Schoof's algorithm [SE] which counts the number of points of an elliptic curve over a finite field.

A serious problem that arises from creating secure cryptosystems is the ability to communicate and manage the encryption and decryption keys. When dealing with symmetric ciphers, the decryption key can be easily derived when the encryption key is known. The problem arises in trying to establish a key where there is no secure communication already established between the entities that want to interact. Up until recently, there were no asymmetric cryptosystems, in other words, there were no systems in which knowing the encryption key did not allow easy access to the decryption key. The ability to publish a key for which others could use to encrypt without giving away the decryption key solved the problem of communicating keys that are used for symmetric ciphers. Now it is possible to exchange keys for secure symmetric ciphers by using an asymmetric, or public key, cipher. Currently, methods in creating public encryption keys that are resistant to attack are based on mathematical problems that are believed to be computationally hard. Two such mathematical problems are the factorization of integers and the discrete logarithm problem. We analyze the discrete logarithm problem before analyzing the El Gamal public key cryptosystems.

## Goals

In this project we analyze of the discrete logarithm problem and factoring, and how it influences public key cryptography. Then we discuss elliptic curve cryptography, analyzing how the discrete logarithm problem changes, and the different problems that arise. Finally we analyze Schoof's Algorithm using both his original paper [SC] and Washington's book on elliptic curves and cryptography [WA], and discuss how it impacts elliptic curve cryptography.

## Discrete Logarithm Problem

The discrete logarithm problem comes from the difficulty in finding the power to which to raise a generator of a cyclic group in order to find a specific element of that group. We begin by defining the group and a generator of that group:

Let  $n$  be a positive integer for which the group of invertible elements modulo  $n$ ,  $\mathbf{Z}_n^*$ , is cyclic. The following theorem of Gauss determines when  $\mathbf{Z}_n^*$  is cyclic.

### Theorem 1

$\mathbf{Z}_n^*$  is cyclic with respect to multiplication modulo  $n$  if and only if:

$n = 2, 4, p^l, 2p^l$ , where  $l=1, 2, \dots$ , and  $p$  is an odd prime

Let  $\alpha$  be a generator of the cyclic group  $\mathbf{Z}_n^*$ :

$$\langle \alpha \rangle = \mathbf{Z}_n^*.$$

Then we choose which element we wish to find the discrete logarithm of:

Let  $\beta$  be an element of  $\mathbf{Z}_n^*$ .

Then there is a unique  $a$  such that:

$$\alpha^a = \beta \text{ in } \mathbf{Z}_n^* \text{ where } 1 \leq a \leq n-1.$$

This  $a$  is the discrete logarithm of  $\beta$  to the base  $\alpha$  or:

$$a = \log_{\alpha} \beta.$$

#### Example 1:

$\mathbf{Z}_{97}^*$  is a cyclic group of order 96. A generator is  $\alpha=5$ .

$$5^{32} = 35 \bmod 97,$$

$$\therefore \log_5 35 = 32 \text{ in } \mathbf{Z}_{97}^*.$$

The discrete logarithm problem (DLP) is written as such:

Given a prime  $p$ , a generator  $\alpha$  of  $\mathbf{Z}_p^*$  and an element  $\beta \in \mathbf{Z}_p^*$ , find  $a$  such that  
 $\alpha^a \equiv \beta \pmod{p}$  where  $0 \leq a \leq p-1$ .

For small  $p$  it is possible to do an exhaustive search in a short time span, but as  $p$  grows large, then the difficulty of finding  $a$  is exponentially harder. It is the difficulty of finding  $a$  which provides the security for several modern asymmetric cryptosystems including the El Gamal cryptosystem.

## The El Gamal Cryptosystem

Let  $p$  be a prime such that the DLP for  $\mathbf{Z}_p^*$  is infeasible. Let  $\alpha$  be a generator of  $\mathbf{Z}_p^*$ . Let  $a$  be an integer:

$$K = \{(p, a, \alpha, \beta) : \alpha^a = \beta \pmod{p}\}.$$

In the cryptosystem  $(p, \alpha, \beta)$  is the public key and  $a$  is the private key. The encrypter chooses a random number  $k \in \mathbf{Z}_{p-1}$ .

Encryption consists of :

$$\varepsilon_k(x, k) = (y_1, y_2),$$

$$y_1 = \alpha^k \pmod{p},$$

$$y_2 = x\beta^k \pmod{p},$$

where  $x$  is the message,  $y_1$  is the header and  $y_2$ .

Decryption consists of:

$$d_k(y_1, y_2) = y_2(y_1^a)^{-1} \pmod{p}.$$

This works since:

$$y_1 = \alpha^k \pmod{p},$$

$$y_2 = x\beta^k \pmod{p},$$

$$d_k(y_1, y_2) = y_2(y_1^a)^{-1} \pmod{p} = x\beta^k(\alpha^{ka})^{-1} \pmod{p} = x\alpha^{ak}(\alpha^{ka})^{-1} \pmod{p} = x \pmod{p}.$$

### Example 2

Let  $p=3571$ ,  $\alpha=5$ ,  $a=601$ , our random  $k$  will be 233, and our message will be:  $x=1221$ .

$$\beta = 3^{601} \pmod{3571} = 825 \pmod{3571}.$$

$$\varepsilon_k(1221, 233) = (y_1, y_2),$$

$$y_1 = \alpha^k \pmod{p} = 3^{233} \pmod{3571} = 2546 \pmod{3571},$$

$$y_2 = x\beta^k \pmod{p} = 1221 * 825^{233} \pmod{3571} = 3542 \pmod{3571},$$

$$\varepsilon_k(1221, 233) = (2546, 3542).$$

Decryption is a single simple operation:

$$d_k(2546, 3542) = y_2(y_1^a)^{-1} \pmod{p} = 3542(2546^{601})^{-1} \pmod{3571} = 1221 \pmod{3571}.$$

# Elliptic Curves

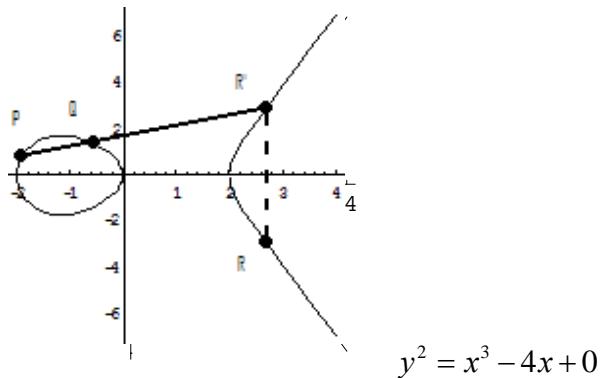
One of the difficulties of previous public key cryptosystems is the ability to find unique groups which can be used for encryption. While elliptic curves are a well studied area of mathematics, it was not until the mid 1980s when V. Miller [Mi] and N. Koblitz [Ko], working independently found a solution for this problem. Elliptic curves are solution sets for certain polynomials, and can be defined over  $\mathbf{Z}_p$ . While not a recent discovery, their use in cryptology has proved to be quite useful because current effective solutions for the discrete logarithm problem over the field  $\mathbf{Z}_q$  does not work over elliptic curves. First, we analyze elliptic curves over the  $\mathbf{R}$ , the set of real numbers.

A non singular elliptic curve  $E$  is the set of solutions  $(x, y) \in \mathbf{R} \times \mathbf{R}$  to the following:

Let  $a, b \in \mathbf{R}$  such that  $4a^3 + 27b^2 \neq 0$ ,

$$y^2 = x^3 + ax + b \quad ,$$

and the point O, also called the point at infinity.



**Figure 1**

We now define an operation to make  $E$  an abelian group. The identity element of  $E$  is defined as  $O$ .

Let  $P, Q \in E$ , where  $P = (x_1, y_1)$ ,  $Q = (x_2, y_2)$ .

We then have three cases which must be considered:

- 1)  $x_1 \neq x_2$
  - 2)  $x_1 = x_2 \quad y_1 = -y_2$
  - 3)  $x_1 = x_2 \quad y_1 = y_2 \quad \dots$

In the first case we look at the line  $L$  which intersects  $E$  at the two points  $P$  and  $Q$ . It is clear that  $L$  also intersects  $E$  at a third point  $R'$ . We define  $P+Q=R$ , where  $R$  is the reflection of  $R'$  about the  $x$ -axis (See Figure 1).

The elliptic curve would not be very useful if we could not define  $R$  algebraically, in order to do so we begin by analyzing the equation for  $L$ :

$$y = \lambda x + v,$$

where:

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1},$$

is the slope and:

$$v = y_1 - \lambda x_1 = y_2 - \lambda x_2.$$

We then solve for the intersections of  $L$  with  $E$  by substituting  $y = \lambda x + v$  into the equation for  $E$ :

$$\begin{aligned} y^2 &= x^3 + ax + b, \\ x^3 - \lambda^2 x^2 + (a - 2\lambda v)x + b - v^2 &= 0. \end{aligned}$$

When we solve for  $x$  in this equation we will get the three  $x$ -coordinates of  $E \cap L$ , and since we already know two of them are real from  $P$  and  $Q$ , we know the third root is real as well. Furthermore we know that the sum of the three roots must be equal to:

$$\begin{aligned} -(-\lambda^2) &= \lambda^2, \\ x_1 + x_2 + x_3 &= \lambda^2, \\ x_3 &= \lambda^2 - x_1 - x_2. \end{aligned}$$

Now that we have solved for the  $x$ -coordinate of  $R'$ , we let  $y_3$  be the  $y$ -coordinate of  $R'$ , and compute it by using  $\lambda$ :

$$\begin{aligned} \lambda &= \frac{y_2 - y_1}{x_2 - x_1} = \frac{-y_3 - y_1}{x_3 - x_1}, \\ -y_3 &= \lambda(x_3 - x_1) + y_1 \\ y_3 &= \lambda(x_1 - x_3) - y_1. \end{aligned}$$

We now have a simple formula to solve for  $R = (x_3, y_3)$ , where:

$$x_3 = \lambda^2 - x_1 - x_2,$$

$$y_3 = \lambda(x_1 - x_3) - y_1,$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}.$$

For the second case, we simply define:

$$(x, y) + (x, -y) = O, \text{ the point at infinity.}$$

In the third case we are adding  $P$  to itself and assume that  $y_1 \neq 0$ . In this case we need to define  $L$  to be the line tangent to  $E$  at  $P$  (See Figure 2). While most of the analysis is identical to case 1, the slope needs to be calculated through implicit differentiation:

$$y^2 = x^3 + ax + b,$$

$$2y \frac{dy}{dx} = 3x^2 + a,$$

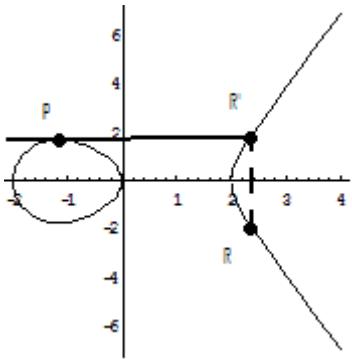
$$\lambda = \frac{dy}{dx} = \frac{3x^2 + a}{2y}.$$

therefore in the case of  $P = (x_1, y_1)$ :

$$x_3 = \lambda^2 - x_1 - x_2,$$

$$y_3 = \lambda(x_1 - x_3) - y_1,$$

$$\lambda = \frac{3x_1^2 + a}{2y_1}.$$



**Figure 2**

By combining the three cases we can summarize addition to:

Let  $P = (x_1, y_1)$  then  $-P = (x_1, -y_1)$ . If  $Q = (x_2, y_2)$  and  $Q \neq -P$ , then  $P + Q = (x_3, y_3)$ , where:

$$x_3 = \lambda^2 - x_1 - x_2,$$

$$y_3 = \lambda(x_1 - x_3) - y_1,$$

$$\lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1}, & \text{if } P \neq Q \\ \frac{3x_1^2 + a}{2y_1}, & \text{if } P = Q \end{cases}.$$

Example 1: we compute  $P(-1.93, -.0762) + Q(-0.308, 1.1)$  for the elliptic curve  
 $y^2 = x^3 - 4x$ .

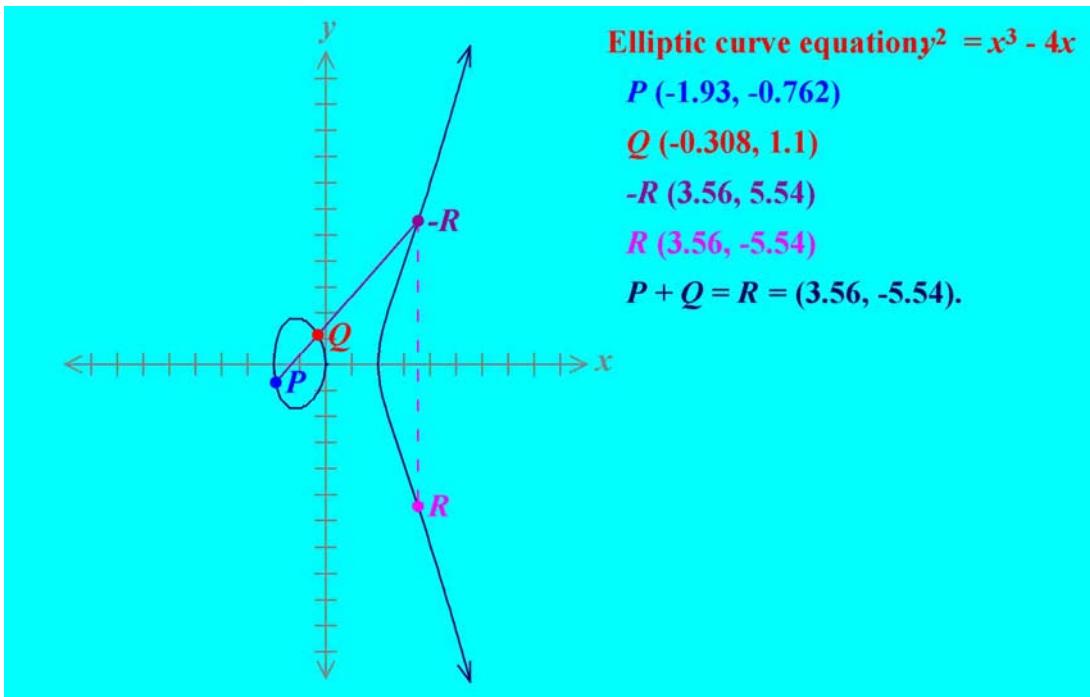


Figure 3 [CE]

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} = \frac{1.1 + .762}{-.308 + 1.93} = \frac{1.862}{1.622} = 1.148.$$

$$x_3 = \lambda^2 - x_1 - x_2 = 1.148^2 + .308 + 1.93 = 3.56,$$

$$y_3 = \lambda(x_1 - x_3) - y_1 = 1.148(-1.93 - 3.56) + .762 = -5.54.$$

## Elliptic Curve Cryptography

Elliptic curves can be defined over any field by the following adaptation of the previous definition.

Let  $K$  be any field, and let  $a, b \in K$ . By  $E(K)$  we denote the pairs  $(x, y) \in K \times K$  that satisfy the cubic equation:

$$y^2 = x^3 + ax + b ,$$

where  $4a^3 + 27b^2 \neq 0$  in  $K$ .

The addition law is defined as before, with the obvious adaptation in the notation for inverses for fields.

Let  $P = (x_1, y_1)$  and  $Q = (x_2, y_2)$ . Recall from the previous section that if  $x_1 = x_2$  and  $y_1 = -y_2$  then  $P + Q = O$  otherwise  $P + Q = (x_3, y_3)$ , where:

$$\begin{aligned} x_3 &= \lambda^2 - x_1 - x_2 , \\ y_3 &= \lambda(x_1 - x_3) - y_1 , \\ \lambda &= \begin{cases} (y_2 - y_1)(x_2 - x_1)^{-1} , & \text{if } P \neq Q \\ (3x_1^2 + a)(2y_1)^{-1} , & \text{if } P = Q \end{cases} , \end{aligned}$$

and  $P + O = O + P = P$  for all  $P \in E$ .

Elliptic curves over finite fields are used for cryptography because of the difficulty of trying to solve the discrete logarithm problem over an elliptic curve. While generic algorithms apply, the index calculus algorithm has no adaptation, effectively eliminating one of the most powerful tools used in solving DLP.

From now on, we only consider elliptic curves over a finite field  $\mathbf{F}_q$ , where  $q = p^n$ , and  $p$  is an odd prime. If  $q$  is small enough, it is possible to calculate all of the elements of  $E$  using a brute force method. After we determine the elliptic curve, we need to calculate its order. The brute force method is extremely slow when  $p$  is a large prime, and there are several theorems which allow us to approximate the order, but our most powerful tool for doing this is Schoof's algorithm. Schoof's algorithm computes  $|E|$  with a running time of  $O(\log p)^8$ , and is efficient for primes  $p$  up to several hundred digits. After we have determined  $|E|$  we can easily see if the elliptic curve is cyclic, because if the order of a group is prime, then the group is cyclic.

Next we describe an El-Gamal cryptosystem for the special case  $\mathbf{F}_q = \mathbf{Z}_p$ ,  $p$  being prime. Now that we have defined addition of  $E$  over  $\mathbf{Z}_p$ , establishing it as a group, and we can determine relatively easily if it is a cyclic group, we can look at the El Gamal cryptosystem, which translates nicely to the elliptic curve.

The first thing we need to do is to change the encryption and decryption from multiplicative to additive notation.

Let  $\alpha$  be a generator of the cyclic elliptic curve  $E$ , and let  $a$  be the private key:

$$\beta = a\alpha .$$

Given a plain text to be encrypted  $x$ , and a secret number  $k$   $0 < k \leq |E| - 1$ , encryption is as follows:

$$\varepsilon_k(x, k) = (k\alpha, x + k\beta),$$

and decryption is:

$$d_k(y_1, y_2) = y_2 - ay_1.$$

It is important to note that  $x$  must be an element of  $E$ , and the encoding of  $x$  onto  $E$  is not trivial when  $E$  is defined on  $\mathbf{Z}_p$ , rather than  $\mathbf{R}$ .

### Schoof's Algorithm

When analyzing an elliptic curve, one of the first steps is to determine the number of points on the elliptic curve. This is very important because it gives us the order of the group we are working with. In 1985 Schoof published an algorithm that greatly sped up the process of determining the number of points on elliptic curves for very large  $q$ . Schoof's algorithm requires at the most  $\log^8 q$  bit operations compared to the  $q^{1/4}$  that the Baby Step, Giant Step algorithm uses. We will analyze Schoof's method in determining the number of points on an elliptic curve.

### Background

Before we are able to do an in depth analysis of Schoof's Algorithm, we need to build a few tools. The first of these tools is Hasse's Theorem, which is the starting point for our analysis.

Theorem 2 (Hasse 1934)

*Let  $E$  be an elliptic curve over the finite field  $\mathbf{F}_q$ . Then the order of  $E(\mathbf{F}_q)$  satisfies*

$$|q + 1 - \#E(\mathbf{F}_q)| \leq 2\sqrt{q} . \quad [\text{WA p. 91}]$$

Another important tool is torsion points, or those points whose order is finite.

Let  $E$  be an elliptic curve defined over a field  $K$ . Let  $n$  be a positive integer. The  $n$ -torsion points are all  $P$  that satisfy:

$$E[n] = \{P \in E(\overline{K}) \mid nP = \infty\},$$

Where  $\overline{K}$  is the algebraic closure of  $K$ . It is important to note that  $E[n]$  contains points in the closure, not just in  $K$ . An important theorem comes from the concept of torsion.

**Theorem 3** [WA p. 74]

*Let  $E$  be an elliptic curve over a field  $K$  and let  $n$  be a positive integer. If the characteristic of  $K$  does not divide  $n$ , or is 0, then:*

$$E[n] \cong \mathbf{Z}_n \oplus \mathbf{Z}_n.$$

*If the characteristic of  $K$  is  $p > 0$  and  $p \mid n$ , write  $n = p^r n'$  with  $\gcd(p, n') = 1$ . Then:*

$$E[n] \cong \mathbf{Z}_{n'} \oplus \mathbf{Z}_{n'} \quad \text{or} \quad \mathbf{Z}_n \oplus \mathbf{Z}_{n'}.$$

We must also introduce some important polynomials that are used in computing

$$nP = \underbrace{P + P + \cdots + P}_n.$$

The division polynomials,  $\psi_n$ , are important for calculating  $nP$ .

We begin with variables  $A, B$ , and define the so called division polynomials (see reference [WA p. 77])  $\psi_m \in \mathbf{Z}[x, y, A, B]$  by:

$$\psi_0 = 0,$$

$$\psi_1 = 1,$$

$$\psi_2 = 2y,$$

$$\psi_3 = 3x^4 + 6Ax^2 + 12Bx - A^2,$$

$$\psi_4 = 4y(x^6 + 5Ax^4 + 20Bx^3 - 5A^2x^2 - 4ABx - 8B^2 - A^3),$$

$$\psi_{2m+1} = \psi_{m+2}\psi_m^3 - \psi_{m-1}\psi_{m+1}^3 \quad \text{for } m \geq 2,$$

$$\psi_{2m} = (2y)^{-1} (\psi_m) (\psi_{m+2}\psi_{m-1}^2 - \psi_{m-2}\psi_{m+1}^2) \quad \text{for } m \geq 2.$$

Theorem 4 [WA p. 79]

Let  $P = (x, y)$  be a point on the elliptic curve  $y^2 = x^3 + Ax + B$  (over some characteristic not 2), and let  $n$  be a positive integer. Then:

$$nP = \left( \frac{\phi_n(x)}{\psi_n^2(x)}, \frac{\omega_n(x, y)}{\psi_n^3(x, y)} \right),$$

Where:

$$\begin{aligned} \phi_n &= x\psi_n^2 - \psi_{n+1}\psi_{n-1} \\ \omega_n &= (4y)^{-1} (\psi_{n+2}\psi_{n-1}^2 - \psi_{n-2}\psi_{n+1}^2). \end{aligned}$$

Theorem 5

Let  $n$  be odd. Then  $\psi_n$  is a polynomial in  $x$  only and for  $(x, y) \in E(\overline{\mathbf{F}_q})$  we have:

$$(x, y) \in E[n] \Leftrightarrow \psi_n(x) = 0.$$

In other words, an element of  $E(\overline{\mathbf{F}_q})$  is a torsion point if and only if it equals 0 for one of the division polynomials.

Endomorphisms are another important tool that we need to develop. An endomorphism of  $E$  is a homomorphism from the closure of  $E$  to the closure of  $E$ . For an endomorphism  $\alpha$  there are rational functions (quotients of polynomials)  $R_1(x, y), R_2(x, y)$ , with coefficients in  $\overline{K}$  such that:

$$\alpha(x, y) = (R_1(x, y), R_2(x, y)),$$

for all  $(x, y) \in E(\overline{\mathbf{F}_q})$ . The Frobenius homomorphism,  $\phi_q(x) = x^q$ ,  $x \in \overline{\mathbf{F}_q}$ , is used in our analysis, and more details on it can be found in Appendix A.

The Frobenius homomorphism can be defined for  $E(\overline{\mathbf{F}_q})$  as follows,

$$\phi_q(x, y) = (x^q, y^q), x \in \overline{\mathbf{F}_q}.$$

The last tool that we must present before we can begin our analysis explains an important relationship about the Frobenius endomorphism on an elliptic curve over a finite field.

Theorem 6

Let  $E$  be an elliptic curve defined over  $\mathbf{F}_q$ . Let  $a = q + 1 - \#E(\mathbf{F}_q) = q + 1 - \deg(\phi - 1)$ . Then:

$$\phi_q^2 - a\phi_q + q = 0,$$

as endomorphisms of  $E$ , and  $a$  is the unique integer  $k$  such that:

$$\phi_q^2 - k\phi_q + q = 0.$$

In other words, if  $(x, y) \in E(\overline{\mathbf{F}_q})$ , then:

$$(x^{q^2}, y^{q^2}) - a(x^q, y^q) + q(x, y) = \infty,$$

and  $a$  is the unique integer such that this relation holds for all  $(x, y) \in E(\overline{\mathbf{F}_q})$ . [WA p. 95]

Now that we have the necessary tools, we can begin the analysis of Schoof's Algorithm

### Analysis

We begin by taking the basic form of an elliptic curve  $E$  over a finite field  $\mathbf{F}_q$ :

$$y^2 = x^3 + Ax + B.$$

By Hasse's Theorem we know that the number of points in an elliptic curve,  $\#E(\mathbf{F}_q)$ , is:

$$q + 1 - a, \text{ where } |a| \leq 2\sqrt{q}.$$

Let  $S = \{2, 3, 5, 7, \dots, L\}$  be a set of primes such that:

$$\prod_{\ell \in S} \ell > 4\sqrt{q}.$$

If we are able to determine  $a \pmod{\ell}$  for all  $\ell \in S$  then we know  $a \pmod{\prod_{\ell \in S} \ell}$  by the Chinese Remainder theorem, and therefore we know  $a$  because  $|a| \leq 2\sqrt{q}$ .

Example 3:

Why are we using  $\prod_{\ell \in S} \ell > 4\sqrt{q}$  rather than  $\prod_{\ell \in S} \ell > 2\sqrt{q}$ ?

Suppose  $a=-3$  and  $4\sqrt{q}=8$ .

If we are able to determine that  $a \equiv -3 \pmod{8}$ , then we know that either:

$$a \equiv -3 \pmod{8} \quad \text{or} \quad a \equiv 5 \pmod{8}.$$

Since  $|a| \leq 2\sqrt{q}$  then  $|a| \leq 4$ ,

therefore, it must be that  $a = -3$ , and not  $a = 5$ .

If we had let  $\prod_{\ell \in S} \ell > 2\sqrt{q}$ , then when we had determined  $a \equiv -3 \pmod{4}$ , we would have had two choices:

$$a \equiv -3 \quad \text{and} \quad a \equiv 1,$$

both of which satisfy the requirement  $|a| \leq 2\sqrt{q}$ .

Let us assume that  $\ell \neq p$  so that we do not get exceptional torsion cases. We begin by assuming  $\ell = 2$  which gives us two cases.

In the first case:

$$x^3 + Ax + B \text{ has a root } e \in \mathbf{F}_q \Rightarrow (e, 0) \in E[2]$$

Since  $e$  is a root then

$$y^2 = x^3 + Ax + B = 0,$$

therefore  $(e, 0)$  is a point in  $E(\mathbf{F}_q)$  of order 2. Thus 2 divides  $\#E(\mathbf{F}_q)$ , therefore  $\#E(\mathbf{F}_q)$  is even. Furthermore we know that  $a \equiv 0 \pmod{2}$ .

In the second case:

$$x^3 + Ax + B \text{ has no roots in } \mathbf{F}_q.$$

Therefore there is no point of order 2 in  $E(\mathbf{F}_q)$ . Thus 2 does not divide  $\#E(\mathbf{F}_q)$ , therefore  $\#E(\mathbf{F}_q)$  is odd, and  $a \equiv 1 \pmod{2}$ .

The question now becomes, How do we find the roots of  $x^3 + Ax + B$ ? We could try an exhaustive search, but there is a more efficient way. Since  $a \in \mathbf{F}_q$ ,  $a$  is a root of  $x^q - x$ .

Therefore, if  $x^3 + Ax + B$  has a root in  $\mathbf{F}_q$ , then it has a root in common with  $x^q - x$ , and we can use the Euclidean Algorithm to determine the greatest common denominator. If the GCD has degree greater than 1 then there is a common root. So we compute:

$$\gcd(x^q - x, x^3 + Ax + B).$$

Here we run into similar problems as we did before, as computing  $x^q - x$  is tedious for large  $q$ . Let us first compute:

$$x_q = x^q \bmod(x^3 + Ax + B).$$

This greatly simplifies the problem of taking the GCD as degree of  $x_q$  is less than or equal to 2. This works because :

$$\gcd(p_1, p_2) = \gcd(p_1 \bmod p_2, p_2).$$

So if the GCD is equal to 1, then there are no common roots, and  $a$  is odd, otherwise there is a root, and  $a$  is even, so we have shown how to determine,  $a \bmod 2$ .

Note: In the following, when  $x^q, x^{q^2}$ , etc. are used, they will be computed modulo a polynomial, just as we did with  $\ell = 2$ .

From the division polynomials  $\psi_n$ , we know that when  $n$  is odd,  $\psi_n$  is a polynomial in  $x$ , and for  $(x, y) \in E[\overline{\mathbf{F}_q}]$ :

$$(x, y) \in E[n] \Leftrightarrow \psi_n(x) = 0.$$

Let the Frobenius endomorphism be represented by:

$$\phi_q(x, y) = (x^q, y^q).$$

By Theorem 6,

$$\phi_q^2 - a\phi_q + q = 0,$$

holds for the Frobenius endomorphism. This implies

$$\begin{aligned} \phi_q^2(x, y) - a\phi_q(x, y) + q(x, y) &= \infty, \\ (x^{q^2}, y^{q^2}) - a(x^q, y^q) + q(x^q, y^q) &= \infty, \\ (x^{q^2}, y^{q^2}) + q(x^q, y^q) &= a(x^q, y^q). \end{aligned}$$

Since  $(x, y)$  has order  $\ell$ , then:

$$q(x, y) = (q \bmod \ell)(x, y).$$

In order to simplify notation, let:

$$q_\ell = q \bmod \ell.$$

Further, let us center  $q_\ell$  around 0, therefore:

$$|q_\ell| < \ell/2.$$

So from above we now have:

$$(x^{q^2}, y^{q^2}) + q_\ell(x, y) = a(x^q, y^q).$$

Since  $\phi_q$  is an endomorphism:

$$\begin{aligned} \ell(x^q, y^q) &= \ell\phi_q(x, y) = \underbrace{\phi_q(x, y) + \cdots + \phi_q(x, y)}_{\ell} = \\ &\phi_q\left(\underbrace{(x, y) + \cdots + (x, y)}_{\ell}\right) = \phi_q(\ell(x, y)) = \phi_q(\infty) = \infty. \end{aligned}$$

So we can determine  $a$  modulo  $\ell$  from the relation:

$$(x^{q^2}, y^{q^2}) + q_\ell(x, y) = a(x^q, y^q).$$

This idea allows us to compute all of the terms except  $a$  in this relationship, and then solve for the unique value of  $a$  that makes the relationship hold. If we are able to find a point  $(x, y) \in E[\ell]$  that works, then this would determine  $a \bmod \ell$  so it would hold for all  $(x, y) \in E[\ell]$ .

Assume that:

$$(x^{q^2}, y^{q^2}) \neq \pm q_\ell(x, y), \text{ for some } (x, y) \in E[\ell].$$

Then:

$$(x^{q^2}, y^{q^2}) + q_\ell(x, y) \neq \infty,$$

and we can define:

$$(x', y') \equiv (x^{q^2}, y^{q^2}) + q_\ell(x, y) \neq \infty.$$

Therefore  $a \neq 0 \pmod{\ell}$ . We can then find the sum of the two points using the line through them, rather than a tangent line because the  $x$ -coordinates are distinct. For integers  $j$ , we write the  $j$ th addition as:

$$j(x, y) = (x_j, y_j).$$

We are able to compute  $x_j$  and  $y_j$  using the division polynomials. Moreover,  $x$  and  $y$  are rational functions of  $x$ , namely:

$$x_j = r_{1,j}(x),$$

and

$$y_j = r_{2,j}(x)y.$$

Therefore we have:

$$\begin{aligned} x' &= \left( \frac{y^{q^2} - y_{q\ell}}{x^{q^2} - x_{q\ell}} \right)^2 - x^{q^2} - x_{q\ell}, \\ (y^{q^2} - y_{q\ell})^2 &= y^2 (y^{q^2-1} - r_{2q\ell}(x))^2 \\ &= (x^3 + Ax + B) \left( (x^3 + Ax + B)^{(q^2-1)/2} - r_{2q\ell}(x) \right)^2. \end{aligned}$$

Now we need to determine  $j$  for which

$$(x', y') = (x_j^q, y_j^q)$$

We begin by looking at the  $x$ -coordinates. We know that  $(x', y') = \pm(x_j^q, y_j^q)$  if and only if  $x' = x_j^q$ . As stated before if this is true for one point in  $E[\ell]$  then it will hold for all of the points. We know that the roots of the division polynomials are the  $x$ -coordinates, therefore:

$$x' - x_j^q \equiv 0 \pmod{\psi_\ell}.$$

Note: The roots of  $\psi_\ell$  must be simple. We know this because there are  $\ell^2 - 1$  distinct points of order  $\ell$ , as we assumed that  $\ell \neq q$ . There are  $(\ell^2 - 1)/2$  distinct  $x$ -coordinates, as we are dealing with  $(x', y') = \pm(x_j^q, y_j^q) = (x_j^q, \pm y_j^q)$ , and since every point is a torsion point, they are all roots of  $\psi_\ell$  which has degree  $(\ell^2 - 1)/2$ , therefore the roots are simple.

Once we have determined  $j$  such that the above holds true, then:

$$(x', y') = \pm(x_j^q, y_j^q) = (x_j^q, \pm y_j^q)$$

We note that both  $y'/y$  and  $y_j^q/y$  can be written as functions of  $x$ . So if:

$$\frac{(y' - y_j^q)}{y} \equiv 0 \pmod{\psi_\ell},$$

then:

$$a \equiv j \pmod{\ell}, \text{ otherwise, } a \equiv -j \pmod{\ell},$$

and therefore we know  $a(\bmod \ell)$ .

We now consider the case where

$$(x^{q^2}, y^{q^2}) = \pm q(x, y) \text{ for all } (x, y) \in E[\ell].$$

If:

$$\phi_q^2(x, y) = (x^{q^2}, y^{q^2}) = q(x, y),$$

Then:

$$\begin{aligned} a\phi_q(x, y) &= \phi_q^2(x, y) + q(x, y) = 2q(x, y), \\ a^2q(x, y) &= a^2\phi_q^2(x, y) = (2q)^2(x, y). \end{aligned}$$

Therefore  $a^2q \equiv 4q^2 \equiv (2q)^2 \pmod{\ell}$ , thus  $q$  must be a square mod  $\ell$ , so let  $w^2 \equiv q \pmod{\ell}$ . We can then expand  $\phi_q^2(x, y) - q(x, y)$ :

$$(\phi_q + w)(\phi_q - w)(x, y) = (\phi_q^2 - q)(x, y) = \infty \text{ for all } (x, y) \in E[\ell].$$

For a point  $P$  in  $E[\ell]$ , we now have one of two cases, either  $(\phi_q - w)P = \infty$ , and therefore  $\phi_q P = wP$ , or  $(\phi_q + w)P = P'$  is a finite point with  $(\phi_q + w)P' = \infty$ . Either way, there exists  $P \in E[\ell]$ ,  $\phi_p P = \pm wP$ .

In the first instance,  $\phi_q P = wP$ , and from above we know for all elements of  $E[\ell]$ ,

$$\begin{aligned} & (\phi_q^2 - a\phi_q + q)(x, y) = 0, \\ & \infty = (\phi_q^2 - a\phi_q + q)P = (q - aw + q)P, \end{aligned}$$

So  $aw \equiv 2q \equiv 2w^2 \pmod{\ell}$ , and thus  $a \equiv 2w \pmod{\ell}$ . Now let  $\phi_q P = -wP$ , then it is clear that by the same process,  $a \equiv -2w \pmod{\ell}$ . In order to check to see if we are in the case, we only need to check:

$$(x^q, y^q) = \pm w(x, y) = \pm (x_w, y_w) = (x_w, \pm y_w),$$

for some  $(x, y) \in E[\ell]$ . It is important to note that unlike the previous relationship, this relationship does not need to hold for every  $(x, y) \in E[\ell]$ , and the fact that it holds for one point, does not mean that it holds for all  $(x, y) \in E[\ell]$ . Therefore we compute  $x^q - x_w$ . Since this is a rational function of  $x$ , we can use similar methods as before. If:

$$\gcd(\text{numerator}(x^q - x_w), \psi_\ell) \neq 1,$$

then there is a  $(x, y) \in E[\ell]$  such that  $\phi_q(x, y) = \pm w(x, y)$ , then all we need to do is use the  $y$ -coordinates to determine the sign. In this case we only need to find if the GCD exists and is greater than 1, because if it does exist then the relation holds for at least one point, whereas if we were looking for  $0 \pmod{\psi_\ell}$ , checks for the relation to hold for all points simultaneously.

If the GCD=1, then we cannot be in the above case, and therefore we must be in the last case,  $(x^{q^2}, y^{q^2}) = -q_\ell(x, y)$ . In this case,  $aP = (\phi_q^2 + q)P = \infty$  for all  $P \in E[\ell]$ . Thus,  $a \equiv 0 \pmod{\ell}$ .

This ends the detailed description of Schoof's Algorithm.

### Schoof's Algorithm

Summarizing Schoof's algorithm, we wish to compute  $\#E(\mathbf{F}_q) = q+1-a$ . We begin with an elliptic curve  $E$  over  $\mathbf{F}_q$ :

$$y^2 = x^3 + Ax + B.$$

First we choose a set of primes  $S = \{2, 3, 5, \dots, L\}$ , with  $p \notin S$ , such that  $\prod_{\ell \in S} \ell > 4\sqrt{q}$ .

Next we test if for  $\ell = 2$ ,  $\gcd(x^q - x, x^3 + Ax + B) \neq 1$ , because  $a \equiv 0 \pmod{2}$  if and only if that is true.

Next we analyze the odd primes of  $\ell \in S$  by,

1) Let  $q_\ell \equiv q \pmod{\ell}$ , with  $|q_\ell| < \ell/2$ .

2) Compute the  $x$  coordinate of  $x'$  of

$$(x', y') = (x^{q^2}, y^{q^2}) + q_\ell(x, y) \pmod{\psi_\ell}.$$

3) For  $j = 1, 2, \dots, (\ell-1)/2$ , do the following:

a) Compute the  $x$  coordinate of  $x_j$  of  $(x_j, y_j) = j(x, y)$ .

b) If  $x' - x_j^q \equiv 0 \pmod{\psi_\ell}$ , continue to the next step, otherwise try the next value of  $j$ . If all values of  $j$  have been tried, and none work, then move on to step 4.

c) Compute  $y'$  and  $y_j$ . If  $(y' - y_j)/y \equiv 0 \pmod{\psi_\ell}$ , then  $a \equiv j \pmod{\psi_\ell}$ , otherwise  $a \equiv -j \pmod{\psi_\ell}$ .

4) Let  $w^2 \equiv q \pmod{\ell}$ . If such a  $w$  does not exist, then  $a \equiv 0 \pmod{\ell}$ .

5) If  $\gcd(\text{numerator}(x^q - x_w), \psi_\ell) = 1$ , then  $a \equiv 0 \pmod{\ell}$ . If it does not, then compute  $\gcd(\text{numerator}(y^q - y_w/y), \psi_\ell)$ . If this gcd is not 1 then  $a \equiv 2w \pmod{\ell}$ . Otherwise,  $a \equiv -2w \pmod{\ell}$ .

Now that we know  $a \pmod{\ell}$  for every  $\ell \in S$ , we are able to use the Chinese Remainder Theorem to calculate  $a \pmod{\prod \ell}$ . We can now choose the  $a$  that satisfies  $|a| < 2\sqrt{q}$ . Then the number of points in  $E(\mathbf{F}_q) = q+1-a$ .

### Example of Schoof's Algorithm

Let  $E$  be the elliptic curve  $y^2 = x^3 + 2x + 1 \pmod{19}$ . Then:

$$\#E(\mathbf{F}_{19}) = 19 + 1 - a$$

We begin with  $\ell = 2$ , and we compute:

$$x^{19} = x^2 + 13x + 14 \pmod{x^3 + 2x + 1},$$

using successive squaring, using that to compute:

$$\gcd(x^{19} - x, x^3 + 2x + 1) = \gcd(x^2 + 12x + 14, x^3 + 2x + 1) = 1.$$

Since there are no common denominators greater than 1, then  $x^3 + 2x + 1$  has no roots in  $\mathbf{F}_{19}$ , so there are no 2-torsion in  $E(\mathbf{F}_{19})$ , and therefore  $a \equiv 1 \pmod{2}$ .

For  $\ell = 3$ , we continue through Schoof's Algorithm, until we get to  $j=1$ . We calculate  $q^2 = 19^2 = 361$ , and  $q_\ell \equiv 19 \pmod{3} = 1 \pmod{3}$ . Now we need to check if:

$$(x^{361}, y^{361}) + (x, y) = \pm(x^{19}, y^{19}),$$

For  $(x, y) \in E[3]$ . The third division polynomial is:

$$\psi_3 = 3x^4 + 12x^2 + 12x - 4.$$

We compute the  $x$ -coordinate of  $(x^{361}, y^{361}) + (x, y)$ :

$$\left( \frac{y^{361} - y}{x^{361} - x} \right)^2 - x^{361} - x = (x^3 + 2x + 1) \left( \frac{(x^3 + 2x + 1)^{180} - 1}{x^{361} - x} \right)^2 - x^{361} - x.$$

In order to reduce this  $\pmod{\psi_3}$  we use the Euclidean algorithm to find the inverse of  $x^{361} - x \pmod{\psi_3}$ . But,

$$\gcd(x^{361} - x, \psi_3) = x - 8 \neq 1,$$

so a multiplicative inverse does not exist. We could remove the common factor from the numerator and denominator, but in this case there is a shortcut. We notice that  $x=8$  is a root of  $\psi_3$ , therefore  $(8, 4) \in E(\mathbf{F}_{19})$  has order 3, and thus:

$$\#E(\mathbf{F}_{19}) = 19 + 1 - a \equiv 0 \pmod{3},$$

so  $a \equiv 2 \pmod{3}$ .

Finally we look at  $\ell = 5$ , and following the first three steps of Schoof's Algorithm, we get:

$$q_\ell \equiv 19 \equiv 4 \equiv -1 \pmod{5}.$$

When  $j=1$ , then  $x' - x_j^q \neq 0 \pmod{\psi_\ell}$ , so we continued on to  $j=2$ . In this case we need to check if:

$$x' - x_j^q \equiv 0 \pmod{\psi_\ell},$$

or in other words if:

$$(x', y') = (x^{361}, y^{361}) + (x, -y) = \pm 2(x^{19}, y^{19}) = (x'', y'') \pmod{\psi_\ell},$$

For all  $(x, y) \in E(5)$ . Using the fifth division polynomial, we can determine the  $x$  coordinate:

$$x' = \left( \frac{y^{361} + y}{x^{361} - x} \right)^2 - x^{361} - x = \left( \frac{3x^{38} + 2}{2y^{19}} \right)^2 - 2x^{19} = x'' \pmod{\psi_\ell}.$$

We are able to reduce this to a polynomial relationship in  $x$  by substituting  $x^3 + 2x + 1$  in for  $y^2$ , and I can be verified that this relationship holds, therefore:

$$a \equiv \pm 2 \pmod{5}.$$

In order to determine the sign, we check the  $y$ -coordinates. For  $(x', y') = (x^{361}, y^{361}) + (x, -y)$ , the  $y$ -coordinate is:

$$y(9x^{11} + 13x^{10} + 15x^9 + 15x^7 + 18x^6 + 17x^5 + 8x^4 + 12x^3 + 8x + 6) \pmod{\psi_5}.$$

For  $(x'', y'') = 2(x, y)$  the  $y$ -coordinate is:

$$y(13x^{10} + 15x^9 + 16x^8 + 13x^7 + 8x^6 + 6x^5 + 17x^4 + 18x^3 + 8x + 18) \pmod{\psi_5}.$$

Then we compute:

$$(y' + y''^{19})/y \equiv 0 \pmod{\psi_5}.$$

Therefore

$$(x', y') \equiv (x_2^{19}, -y_2^{19}) = -2(x^q, y^q) \pmod{\psi_5},$$

and thus  $a \equiv -2 \pmod{5}$ .

Now using the Chinese remainder theorem, we are able to show that  $a = -7$ . Therefore,  $\#E(\mathbf{F}_{19}) = 19 + 1 - a = 27$ .

## Appendix A

### Algebraic Extensions

Before we are able to explain an algebraic extension, we must first explain the concept of an element being algebraic.

Let K and L be fields, where  $K \subseteq L$  and let  $\alpha \in L$ . Then  $\alpha$  is algebraic over K if there exists a non constant polynomial:

$$f(x) = X^n + a_{n-1}X^{n-1} + \cdots + a_0,$$

with  $a_0, \dots, a_{n-1} \in K$  such that  $f(\alpha) = 0$ . We can then expand this concept to include algebraic extensions. We say that L is an algebraic extension of K if every element of L is algebraic over K. We can further say that the field  $\overline{K}$  is a an algebraic closure of K, if  $\overline{K}$  is a field containing K such that:

$\overline{K}$  is an algebraic extension of K and every non-constant polynomial  $g(X)$  with coefficients in  $\overline{K}$  has a root in  $\overline{K}$ .

### Frobenious Homomorphism

The Frobenius homomorphism,  $\phi_q$  of  $\overline{F_q}$  is a homomorphism from  $\overline{F_q}$  to  $\overline{F_q}$  denoted by the formula  $\phi_q(x) = x^q$ ,  $x \in \overline{F_q}$ . It is clear enough that  $\phi_q$  is from  $\overline{F_q}$  to  $\overline{F_q}$ , so we need only prove that  $\phi_q$  is a homomorphism. In order to prove that  $\phi_q$  is a field homomorphism, we need to prove that  $\phi_q(x+y) = \phi_q(x) + \phi_q(y)$  and  $\phi_q(xy) = \phi_q(x)*\phi_q(y)$ . We will begin by showing that  $\phi_q(x*y) = \phi_q(x)*\phi_q(y)$ :

$$\phi_q(x*y) = (xy)^q = x^q y^q = \phi_q(x)*\phi_q(y).$$

To show that  $\phi_q(x+y) = \phi_q(x) + \phi_q(y)$  is a little bit trickier, but still not that difficult:

$$\phi_q(x+y) = (x+y)^q = x^q + \binom{q}{1}x^{q-1}y + \binom{q}{2}x^{q-2}y^2 + \cdots + \binom{q}{q-1}xy^{q-1} + y^q.$$

Since we are working in  $\overline{F_q}$ , all of the inside terms reduce to 0 leaving us with:

$$x^q + y^q = \phi_q(x+y).$$

### Chinese Remainder Theorem

Let  $m_1, m_2, \dots, m_k$  be pairwise relatively prime positive integers, and let  $a_1, a_2, \dots, a_k$  be integers. Consider the system of congruences

$$\begin{aligned}x &= a_1 \pmod{m_1} \\x &= a_2 \pmod{m_2} \\&\vdots \\x &= a_k \pmod{m_k}.\end{aligned}$$

The Chinese Remainder theorem stats that the systems has a unique solution modulo  $M = m_1m_2\dots m_k$  given by

$$x = \sum_{i=1}^k a_i M_i y_i \pmod{M},$$

where

$$M_i = \frac{M}{m_i} \quad \& \quad y_i = M_i^{-1} \pmod{m_i} \quad \text{for } 1 \leq i \leq k.$$

### Example

Consider the following system of congruencies:

$$\begin{aligned}x &\equiv a_1 \pmod{m_1} = 1 \pmod{2} \\x &\equiv a_2 \pmod{m_2} = 2 \pmod{3} \\x &\equiv a_3 \pmod{m_3} = 3 \pmod{5},\end{aligned}$$

we will solve for  $x$  modulo 30.

$$\begin{aligned}x &= \sum_{i=1}^k a_i M_i y_i \pmod{M} = 1 * 15 * 1 + 2 * 10 * 1 + 3 * 6 * 1 = 15 + 20 + 18 = 53 = 23 = -7 \pmod{30}, \\x &= 23 = -7 \pmod{30}.\end{aligned}$$

## Appendix B: Schoof's Algorithm in Practice

```

/* MAGMA code for the example. */

/* Let E be the elliptic curve  $y^2=x^3+2x+1 \pmod{19}$ . Then: #E(F_19)=19+1-a.
We begin with l=2 and we compute: */

q:=19;

R<x> := PolynomialRing(ResidueClassRing(q));

/* The equation for the elliptic curve. */

ell := x^3+2*x+1;

/* l=2 */

Gcd(x^q-x,ell);

1

/* The gcd is 1, so a=1 mod 2.*/

/* Now let l=3. */

/* Here is psi3 */

psi := 3*x^4+12*x^2+12*x-4;

/* Starting with l=3. q=1 mod l. So, ql=1. Let j =1. */

/* We need to check the relation  $(x^{q^2}, y^{q^2}) + 1(x, y) = 1(x^q, y^q)$  modulo
psi. The left hand side is computed by  $x_3 = m^2 - x_1 - x_2$ ,  $y_3 = m(x_1 - x_3)$ 
-  $y_1$ , where  $m = (y_2 - y_1)/(x_2 - x_1)$ ; or easier to keep track:  $m = (y_1 -
y_2)/(x_1 - x_2)$ . In this case  $m = (y^{(q^2)} - y)/(x^{(q^2)} - x)$ . */

/* We simplify the fraction in m by wiping any from both the numerator
and denominator any common factors between the denominator and psi. So
we divide out by the gcd of  $x^{(q^2)} - x$  and psi. This way the denominator
becomes always invertible modulo psi. */

xr1 := ((ell^((q^2)-1) div 2)-1) div Gcd(x^(q^2)-x,psi));

xr2 := ((x^(q^2)-x) div Gcd(x^(q^2)-x,psi));

/* Inversion of xr2 modulo psi. The answer is in xq. */

xp,xq,xr := Xgcd(xr2,psi);

/* The left hand side of the x components in the addition of
 $(x^{q^2}, y^{q^2}) + (x, y)$ . */

xlhs := ell*xq^2*xr1^2-x^(q^2)-x;

/* Checking that the x on the left minus  $x^q$  on the right is zero
mod psi. */

```

```

xdiv := (xlhs-x^q) mod (psi);

xdiv;

0

/* It is, so we proceed to compute the y's */

x1 := x^(q^2);

x3 := ell*xq^2*xr1^2-x^(q^2)-x;

/* The left hand side of the y components in the addition of (x^(q^2),y^(q^2))
+ (x,y).
Recall that y3 = m(x1-x3) - y1, where m = (y2-y1)/(x2-x1). */

ylhs := xq*xr1*(x1-x3) - ell^((q^2-1) div 2);

/* Checking whether the y component on the left minus y^q on the right
is zero mod psi. */

ydiv:= (ylhs-ell^((q-1) div 2)) mod (psi);

ydiv;

16*x^3 + 2*x^2 + 12*x + 18

/* It is not, so out of the two options a +1, or -1 we pick -1 mod l=3.
So a = 2 mod 3. */

/* Let l=5. Then q=4=-1 mod l. We pick ql=-1 since |ql|<=l/2|. */

/* Next we need psi5. Starting with j=1. */

/* Here is psi5 */

psi := 5*x^12+10*x^10+17*x^8+5*x^7+x^6+9*x^5+12*x^4+2*x^3+5*x^2+8*x+8;

/* We repeat to get the x's */

xr1 := ((ell^((q^2-1) div 2)-1) div Gcd(x^(q^2)-x,psi));

xr2 := ((x^(q^2)-x) div Gcd(x^(q^2)-x,psi));

xp,xq,xr := Xgcd(xr2,psi);

xlhs := ell*xq^2*xr1^2-x^(q^2)-x;

xdiv:= (xlhs-x^q) mod (psi);

xdiv;

7*x^11 + 9*x^10 + 17*x^9 + 7*x^8 + 18*x^7 + 13*x^6 + x^4 + 6*x^3 + 5*x^2 + 12*x
+ 2

/* This time we get nonzero. So, we increment j. Let j=2. */

```

```
/* Now the relation is (x^q^2,y^q^2) - 1(x,y) = plus minus 2 * (x^q,y^q)
because ql=-1 and j=2.
```

```
This is the same as (x^q^2,y^q^2) + (x,-y) = plus minus 2 * (x^q,y^q)
So we need to compute: (x^q^2,y^q^2) + (x,-y) and 2 * (x^q,y^q). */
```

```
/* First we need the x component of (x^q^2,y^q^2) + (x,-y). This is done
just as before. Watch out: this is not the same as xlhs before because
we have -y, instead of y. The only difference in the code is that now
we have plus one instead of minus 1 in xr1.*/
```

```
xr1 := ((ell^(q^2-1) div 2) + 1) div Gcd(x^(q^2)-x,psi));
```

```
xr2 := ((x^(q^2)-x) div Gcd(x^(q^2)-x,psi));
```

```
/* xq below is the denominator inverted.*/
```

```
xp,xq,xr := Xgcd(xr2,psi);
```

```
/* The left hand side. */
```

```
xlhs := ell*xq^2*xr1^2-x^(q^2)-x;
```

```
xlhs;
```

```
11*x^1105 + x^1104 + 8*x^1103 + 4*x^1102 + 8*x^1101 + 4*x^1100 + 17*x^1098 +
4*x^1097 + 5*x^1096 + 16*x^1095 + 15*x^1094 + 15*x^1093 + 3*x^1092 +
13*x^1091 + 3*x^1090 + 14*x^1089 + x^1088 + 5*x^1087 + x^1086 + 2*x^1085 +
13*x^1084 + 9*x^1083 + 3*x^565 + 2*x^564 + 2*x^562 + 3*x^561 + 12*x^560 +
5*x^559 + 3*x^558 + 2*x^557 + 18*x^556 + 5*x^555 + 17*x^554 + 16*x^553 +
9*x^552 + 5*x^551 + 17*x^550 + 11*x^549 + 7*x^548 + 3*x^547 + 13*x^546 +
17*x^545 + 15*x^544 + 6*x^543 + 18*x^542 + 4*x^541 + 6*x^540 + 6*x^539 +
16*x^538 + 17*x^537 + 3*x^536 + 18*x^535 + 12*x^534 + 11*x^532 + x^531 +
5*x^530 + 17*x^529 + 9*x^528 + 3*x^527 + 17*x^526 + 10*x^525 + 6*x^524 +
6*x^523 + 17*x^522 + 18*x^521 + 3*x^520 + 12*x^519 + 16*x^518 + 6*x^517 +
13*x^516 + 5*x^515 + 16*x^514 + 13*x^513 + 2*x^512 + 8*x^511 + 12*x^510 +
16*x^509 + 14*x^508 + x^507 + 4*x^506 + 12*x^505 + 9*x^504 + 9*x^503 + x^502 +
2*x^501 + 2*x^500 + 12*x^499 + 4*x^498 + 2*x^497 + 18*x^496 + 5*x^495 +
15*x^494 + 18*x^492 + 8*x^491 + 18*x^490 + 11*x^489 + 4*x^488 + 11*x^487 +
11*x^486 + 15*x^485 + 6*x^484 + 10*x^483 + 5*x^482 + 13*x^480 + 14*x^479 +
15*x^478 + 16*x^477 + 17*x^476 + 3*x^475 + 6*x^474 + 14*x^473 + 2*x^472 +
2*x^470 + 8*x^468 + 8*x^467 + 8*x^466 + 5*x^464 + x^463 + x^462 + 7*x^461 +
16*x^460 + 9*x^459 + 3*x^458 + x^457 + 15*x^456 + 8*x^455 + 5*x^454 +
17*x^453 + 18*x^452 + 15*x^451 + 7*x^450 + 9*x^449 + 12*x^448 + 8*x^447 +
5*x^446 + 2*x^445 + 14*x^444 + 17*x^443 + 5*x^442 + 8*x^441 + 5*x^440 +
6*x^439 + 16*x^438 + 2*x^437 + x^436 + 15*x^435 + 13*x^434 + x^432 +
11*x^431 + 14*x^430 + 6*x^429 + 2*x^428 + 9*x^427 + 3*x^426 + 4*x^425 +
8*x^424 + 2*x^423 + 8*x^422 + 3*x^420 + 18*x^419 + 11*x^418 + 3*x^417 +
17*x^416 + 16*x^415 + 10*x^414 + 3*x^413 + 3*x^412 + 8*x^411 + 13*x^410 +
4*x^409 + 8*x^408 + 10*x^407 + 15*x^406 + 15*x^405 + 4*x^404 + 4*x^403 +
11*x^402 + 5*x^401 + 5*x^400 + 9*x^399 + 15*x^398 + 4*x^397 + 6*x^396 +
7*x^395 + 8*x^394 + 9*x^391 + 14*x^390 + 12*x^389 + 3*x^388 + 2*x^386 +
17*x^385 + 12*x^384 + 16*x^383 + 17*x^382 + 16*x^381 + 9*x^380 + 4*x^379 +
10*x^378 + 3*x^377 + 7*x^376 + 10*x^375 + 9*x^373 + 3*x^372 + 14*x^371 +
11*x^370 + 10*x^369 + 3*x^367 + 18*x^366 + 4*x^365 + 13*x^364 + 18*x^363 +
```

$$\begin{aligned}
& 9*x^{362} + x^{361} + 17*x^{360} + 4*x^{359} + 6*x^{358} + 15*x^{357} + 6*x^{356} + \\
& 8*x^{355} + 16*x^{354} + 8*x^{353} + x^{352} + 14*x^{351} + 12*x^{350} + 14*x^{349} + \\
& 17*x^{348} + 3*x^{347} + 18*x^{346} + 8*x^{345} + 18*x^{344} + 6*x^{342} + 4*x^{341} + \\
& 4*x^{340} + 6*x^{339} + x^{338} + x^{337} + x^{336} + 7*x^{335} + 12*x^{334} + x^{333} + \\
& 5*x^{332} + 9*x^{331} + 12*x^{329} + 5*x^{328} + 6*x^{327} + 5*x^{326} + 7*x^{325} + \\
& 3*x^{324} + 10*x^{323} + 7*x^{322} + x^{321} + 11*x^{320} + 10*x^{319} + 3*x^{318} + \\
& 7*x^{316} + 9*x^{315} + 8*x^{314} + 9*x^{313} + 9*x^{312} + 8*x^{311} + 7*x^{309} + \\
& 4*x^{308} + x^{307} + 2*x^{306} + 8*x^{305} + 2*x^{304} + 17*x^{303} + 13*x^{302} + \\
& 10*x^{301} + 5*x^{300} + 13*x^{299} + 11*x^{298} + 12*x^{297} + 5*x^{296} + 10*x^{295} + \\
& 4*x^{294} + 10*x^{293} + 18*x^{292} + 18*x^{291} + 9*x^{290} + 18*x^{289} + 5*x^{288} + \\
& 15*x^{287} + 13*x^{286} + 10*x^{285} + 6*x^{284} + 7*x^{283} + x^{282} + 12*x^{281} + \\
& 15*x^{280} + 17*x^{279} + 9*x^{278} + 17*x^{277} + 11*x^{276} + 14*x^{275} + 10*x^{273} + \\
& 9*x^{272} + 18*x^{271} + x^{270} + 11*x^{269} + 8*x^{268} + 4*x^{267} + 18*x^{266} + \\
& 4*x^{265} + 14*x^{264} + 2*x^{263} + 11*x^{262} + 13*x^{261} + 17*x^{260} + 11*x^{259} + \\
& 15*x^{258} + 11*x^{257} + 15*x^{256} + 11*x^{255} + 15*x^{254} + 11*x^{253} + 18*x^{252} + \\
& 6*x^{251} + 8*x^{250} + 10*x^{249} + 9*x^{248} + 6*x^{247} + 5*x^{246} + 17*x^{245} + \\
& 16*x^{244} + 18*x^{243} + 15*x^{242} + 14*x^{241} + 5*x^{240} + 15*x^{239} + 5*x^{238} + \\
& 9*x^{237} + 17*x^{236} + 5*x^{235} + 14*x^{234} + 17*x^{233} + 8*x^{232} + 3*x^{231} + \\
& 13*x^{230} + 9*x^{229} + 12*x^{228} + 9*x^{227} + 13*x^{226} + 10*x^{225} + 11*x^{224} + \\
& 3*x^{222} + 5*x^{221} + 14*x^{220} + 3*x^{219} + 7*x^{218} + 15*x^{217} + 17*x^{216} + \\
& x^{215} + 15*x^{214} + 16*x^{213} + x^{212} + 5*x^{211} + 7*x^{210} + 12*x^{209} + \\
& 13*x^{208} + 17*x^{207} + 16*x^{206} + 12*x^{205} + 6*x^{204} + x^{203} + 12*x^{202} + \\
& 4*x^{201} + 9*x^{200} + 13*x^{199} + 7*x^{198} + 3*x^{197} + 10*x^{196} + 12*x^{195} + \\
& 5*x^{194} + 9*x^{193} + 7*x^{192} + 6*x^{191} + 5*x^{190} + 4*x^{189} + 11*x^{188} + \\
& 7*x^{187} + 8*x^{186} + 11*x^{185} + 12*x^{184} + 6*x^{183} + 12*x^{182} + 2*x^{181} + \\
& 7*x^{180} + 8*x^{179} + 14*x^{178} + 13*x^{177} + 18*x^{176} + 15*x^{175} + 13*x^{174} + \\
& 17*x^{173} + 12*x^{172} + x^{171} + 2*x^{170} + 13*x^{169} + 10*x^{168} + 2*x^{167} + \\
& 5*x^{166} + 18*x^{165} + 6*x^{164} + 13*x^{163} + 18*x^{162} + 11*x^{161} + 15*x^{160} + \\
& 4*x^{159} + x^{158} + 14*x^{157} + 17*x^{156} + 4*x^{155} + 6*x^{154} + x^{153} + 7*x^{152} \\
& + 9*x^{151} + 9*x^{150} + 4*x^{149} + 7*x^{148} + 18*x^{147} + 8*x^{146} + 11*x^{145} + \\
& 9*x^{144} + 18*x^{143} + 3*x^{142} + 18*x^{141} + 11*x^{140} + 9*x^{139} + 6*x^{138} + \\
& 16*x^{137} + 15*x^{136} + 16*x^{134} + 6*x^{133} + 10*x^{132} + 3*x^{131} + 14*x^{130} + \\
& 5*x^{129} + 15*x^{128} + 16*x^{127} + 9*x^{126} + 7*x^{125} + 5*x^{124} + 15*x^{123} + \\
& 4*x^{122} + 18*x^{121} + 6*x^{120} + 11*x^{119} + 15*x^{118} + 18*x^{117} + 4*x^{116} + \\
& 9*x^{115} + 5*x^{114} + 17*x^{113} + 16*x^{112} + 5*x^{111} + 8*x^{110} + 6*x^{108} + \\
& 17*x^{107} + 17*x^{106} + 4*x^{105} + 9*x^{104} + 7*x^{103} + 4*x^{102} + 6*x^{101} + \\
& 14*x^{100} + 3*x^{99} + 10*x^{98} + 4*x^{97} + 3*x^{96} + 9*x^{95} + 13*x^{93} + 10*x^{92} + \\
& 13*x^{91} + 15*x^{90} + 9*x^{89} + 9*x^{88} + 13*x^{87} + x^{86} + 3*x^{85} + 13*x^{84} + \\
& 17*x^{83} + 4*x^{82} + 18*x^{80} + 10*x^{79} + 14*x^{78} + 6*x^{77} + 9*x^{76} + 13*x^{75} + \\
& 11*x^{74} + 7*x^{73} + 6*x^{72} + 8*x^{71} + 14*x^{70} + 2*x^{69} + 2*x^{68} + 16*x^{67} + \\
& 2*x^{66} + 11*x^{65} + 7*x^{64} + 18*x^{63} + 10*x^{62} + 14*x^{61} + 15*x^{60} + 15*x^{59} \\
& + 11*x^{58} + 5*x^{57} + 13*x^{56} + 6*x^{55} + 9*x^{54} + x^{53} + 18*x^{52} + 4*x^{51} + \\
& 8*x^{49} + 8*x^{48} + 4*x^{47} + 5*x^{46} + 6*x^{45} + 7*x^{44} + 14*x^{43} + 9*x^{42} + \\
& 6*x^{41} + 7*x^{40} + 18*x^{39} + 2*x^{38} + 16*x^{37} + 6*x^{36} + 9*x^{35} + 13*x^{34} + \\
& 7*x^{33} + 17*x^{32} + 5*x^{31} + 17*x^{30} + 9*x^{29} + 13*x^{28} + 2*x^{27} + 13*x^{25} + \\
& 3*x^{24} + 3*x^{23} + 16*x^{22} + 17*x^{21} + 3*x^{20} + 17*x^{19} + 4*x^{18} + 17*x^{17} + \\
& 18*x^{16} + 10*x^{15} + x^{14} + 18*x^{12} + 5*x^{11} + 10*x^{10} + 18*x^{9} + 18*x^{8} + \\
& 7*x^7 + 7*x^6 + 10*x^5 + 17*x^4 + 7*x^3 + 13*x^2 + 11*x + 17
\end{aligned}$$

```

/* Next we need the x component of 2*(x^q,y^q). */
/* The formula is x3=m^2-2x1, y3=m(x1-x3)-y1, m = (3x1^2+A)/(2y1). A=2
here.*/

```

A := 2;

```

/* So x3 = (3*(x^q)^2+A)^2/(2*y^q)^2 - 2*x^q; */

```

```

/* We need to replace  $y^2$  and write it in terms of  $x$  and find inverse of
the denominator modulo psi. After all that we get the right hand side
of the  $x$  components. */

/* Denominator:  $(2*y^q)^2 = 4*(y^2)^q = 4*ell^q */$ 
```

$$dd := 4*ell^q;$$

```

/* Numerator:  $(3*(x^q)^2+A)^2 */$ 
```

$$nn := (3*(x^q)^2+A)^2;$$

$$xr1 := (nn \text{ div } Gcd(dd,psi));$$

$$xr2 := (dd \text{ div } Gcd(dd,psi));$$

```

/*  $xq$  below is the denominator inverted.*/
```

$$xp,xq,xr := Xgcd(xr2,psi);$$

```

/* This is the right hand side. */
xrhs := xq*xr1-2*x^q;
```

$$xrhs;$$

$$13*x^{87} + x^{85} + 18*x^{84} + 9*x^{83} + 4*x^{82} + 12*x^{81} + 11*x^{80} + 5*x^{79} + 2*x^{78} + 3*x^{77} + 12*x^{76} + 11*x^{49} + 14*x^{47} + 5*x^{46} + 12*x^{45} + 18*x^{44} + 16*x^{43} + 2*x^{42} + 13*x^{41} + 9*x^{40} + 4*x^{39} + 16*x^{38} + 17*x^{19} + 10*x^{11} + 11*x^9 + 8*x^8 + 4*x^7 + 6*x^6 + 18*x^5 + 7*x^4 + 17*x^3 + 3*x^2 + 14*x + 18$$

```

/* Now check the difference if it is zero mod psi */

xdiv:= (xlhs-xrhs) mod (psi);

xdiv;
0

/* OK! We got zero, as it was claimed.
 ****
/* Now we compute the  $y$ 's to determine whether  $a$  is  $+2$  or  $-2$  mod  $5$  */

/* First we need the  $y$  component of  $(x^{(q^2)}, y^{(q^2)}) + (x, -y)$ . Recall that
 $x_3 = m^2 - x_1 - x_2$ ,  $y_3 = m(x_1 - x_3) - y_1$ , where  $m = (y_1 - y_2)/(x_1 - x_2)$ .
```

Here,  $x_1 = x^{(q^2)}$ ,  $x_2 = x$ ,  $y_1 = y^{(q^2)}$  and  $y_2 = -y$

So  $m = (y^{(q^2)} + y)/(x^{(q^2)} - x) = y * (y^{(q^2-1)} + 1) / (x^{(q^2)} - x)$ ;  
 $m = y * (ell^{((q^2-1)/2)} + 1) / (x^{(q^2)} - x)$ ;  
In magma:  $ell^{((q^2-1)/2)}$  is  $ell^{((q^2-1) \text{ div } 2)}$ .

First we want to make sure that all common factors between  $x^{(q^2)} - x$  and  $\psi$ . \*/

$$dd := x^{(q^2)} - x;$$

```

/* Numerator: ell^((q^2-1) div 2) + 1 */

nn := ell^((q^2-1) div 2) + 1;

yrl := (nn div Gcd(dd,psi));

yr2 := (dd div Gcd(dd,psi));

/* yq is the denominator inverted. */

yp,yq,yr := Xgcd(yr2,psi);

/* Now the left hand side: Recall that y3 = m(x1-x3) - y1, m= (y1-y2)/(x1-x2). */
/* So m = (y^(q^2)+y)/(x^(q^2)-x) */
/* Recall that x3=xlhs. So the lhs of y is up to a factor of y: */

ylhs := yq*yrl*(x^(q^2) - xlhs) - ell^((q^2 - 1) div 2);

ylhs;

18*x^1656 + 18*x^1655 + 15*x^1654 + 4*x^1653 + 3*x^1652 + 13*x^1651 + x^1650 +
11*x^1649 + 7*x^1648 + 8*x^1647 + 14*x^1646 + 10*x^1645 + x^1644 + 4*x^1643 +
+ 17*x^1642 + 5*x^1641 + 10*x^1640 + 12*x^1639 + 9*x^1638 + 13*x^1637 +
8*x^1636 + 7*x^1635 + 3*x^1634 + 16*x^1633 + 12*x^1632 + 14*x^1631 +
14*x^1630 + 15*x^1629 + 17*x^1628 + 10*x^1627 + 6*x^1626 + 14*x^1625 +
14*x^1624 + 4*x^1623 + 9*x^1622 + 2*x^1621 + 4*x^1620 + 10*x^1619 +
13*x^1618 + 5*x^1617 + 5*x^1616 + 6*x^1615 + 3*x^1614 + x^1613 + 11*x^1612 +
18*x^1611 + 6*x^1610 + 3*x^1609 + 6*x^1608 + 6*x^1607 + x^1606 + 4*x^1605 +
11*x^1604 + 9*x^1603 + x^1602 + 2*x^1601 + 4*x^1600 + 16*x^1599 + 7*x^1598 +
17*x^1597 + 3*x^1596 + 11*x^1595 + 10*x^1594 + 14*x^1593 + 9*x^1592 +
10*x^1591 + 17*x^1590 + 2*x^1589 + 8*x^1588 + 14*x^1587 + 3*x^1586 +
16*x^1585 + 17*x^1584 + 12*x^1583 + 9*x^1582 + 14*x^1581 + 18*x^1580 +
18*x^1578 + 4*x^1577 + 9*x^1575 + 11*x^1574 + 9*x^1573 + 9*x^1572 + 2*x^1571 +
+ 15*x^1570 + 15*x^1569 + 2*x^1568 + 10*x^1567 + 5*x^1566 + 8*x^1565 +
3*x^1564 + 2*x^1563 + 5*x^1562 + 12*x^1561 + 13*x^1560 + x^1559 + 7*x^1558 +
16*x^1557 + 2*x^1556 + 7*x^1555 + 4*x^1554 + 17*x^1553 + 13*x^1552 +
7*x^1551 + 16*x^1550 + 12*x^1549 + 4*x^1548 + 8*x^1547 + 14*x^1546 +
3*x^1545 + 17*x^1544 + 3*x^1543 + 4*x^1542 + x^1541 + 7*x^1539 + 10*x^1538 +
9*x^1537 + 6*x^1536 + 15*x^1535 + x^1534 + 6*x^1533 + x^1532 + 7*x^1531 +
10*x^1530 + 15*x^1529 + 12*x^1527 + 3*x^1526 + 11*x^1525 + 13*x^1524 +
6*x^1523 + 5*x^1522 + 18*x^1521 + 8*x^1520 + 16*x^1519 + 18*x^1518 +
13*x^1517 + x^1516 + 16*x^1515 + 5*x^1514 + 18*x^1513 + 7*x^1512 + 17*x^1511 +
+ 10*x^1510 + 3*x^1509 + 14*x^1508 + 8*x^1507 + 15*x^1506 + 12*x^1505 +
4*x^1504 + 14*x^1503 + 16*x^1502 + 18*x^1501 + 6*x^1500 + 10*x^1499 +
10*x^1498 + 6*x^1497 + 12*x^1496 + 16*x^1495 + 12*x^1494 + 16*x^1493 +
17*x^1492 + 7*x^1491 + 6*x^1490 + 6*x^1489 + 6*x^1488 + 9*x^1487 + 12*x^1486 +
+ 4*x^1485 + 18*x^1484 + 14*x^1483 + x^1482 + 6*x^1481 + 10*x^1480 +
2*x^1479 + 6*x^1478 + 11*x^1477 + 7*x^1476 + 4*x^1475 + 18*x^1474 +
11*x^1473 + 15*x^1472 + 14*x^1471 + 11*x^1470 + 3*x^1469 + 16*x^1468 +
13*x^1467 + 8*x^1466 + 9*x^1465 + 14*x^1464 + 2*x^1463 + 4*x^1461 +
15*x^1460 + 4*x^1459 + 5*x^1458 + 12*x^1457 + 2*x^1456 + 11*x^1455 +
9*x^1454 + 7*x^1453 + 9*x^1452 + 7*x^1451 + 17*x^1450 + 15*x^1449 +
16*x^1448 + 15*x^1447 + x^1446 + 15*x^1445 + 15*x^1444 + 8*x^1443 +
15*x^1442 + 6*x^1441 + 16*x^1440 + 15*x^1439 + 16*x^1438 + 8*x^1437 +
2*x^1436 + 10*x^1434 + x^1432 + 9*x^1431 + 5*x^1430 + 2*x^1429 + 16*x^1428 +
3*x^1427 + 17*x^1426 + 17*x^1425 + 15*x^1424 + 9*x^1423 + 12*x^1422 +

```

$$\begin{aligned}
& 18*x^{1421} + 4*x^{1420} + 14*x^{1419} + 12*x^{1418} + 8*x^{1417} + 18*x^{1416} + \\
& 9*x^{1415} + 8*x^{1414} + 17*x^{1413} + 5*x^{1412} + 14*x^{1411} + 10*x^{1410} + \\
& 4*x^{1409} + 17*x^{1408} + 9*x^{1406} + 9*x^{1405} + 14*x^{1404} + 10*x^{1403} + \\
& 8*x^{1402} + 16*x^{1401} + 2*x^{1399} + 18*x^{1398} + 14*x^{1397} + 10*x^{1396} + \\
& 8*x^{1395} + 11*x^{1394} + 12*x^{1393} + 15*x^{1392} + 10*x^{1391} + 14*x^{1390} + \\
& 5*x^{1389} + 11*x^{1387} + 7*x^{1385} + 12*x^{1384} + 7*x^{1383} + 4*x^{1382} + 2*x^{1381} \\
& + 13*x^{1380} + 5*x^{1379} + 11*x^{1378} + 17*x^{1377} + 11*x^{1376} + 17*x^{1375} + \\
& 6*x^{1374} + 12*x^{1373} + 9*x^{1372} + x^{1371} + 5*x^{1370} + 6*x^{1369} + 18*x^{1368} + \\
& 9*x^{1367} + 3*x^{1366} + 12*x^{1365} + 16*x^{1364} + 13*x^{1363} + 2*x^{1362} + \\
& 16*x^{1361} + 9*x^{1360} + 11*x^{1359} + 14*x^{1358} + 16*x^{1357} + 14*x^{1356} + \\
& 7*x^{1355} + 3*x^{1354} + 17*x^{1353} + 17*x^{1352} + x^{1351} + 18*x^{1350} + 9*x^{1349} \\
& + 4*x^{1348} + 17*x^{1347} + 6*x^{1346} + 8*x^{1345} + 15*x^{1344} + 9*x^{1343} + \\
& 7*x^{1342} + 5*x^{1341} + 7*x^{1340} + 2*x^{1339} + 16*x^{1338} + x^{1337} + 8*x^{1336} + \\
& 7*x^{1335} + 3*x^{1334} + 4*x^{1333} + 8*x^{1331} + 11*x^{1330} + 3*x^{1329} + 6*x^{1328} \\
& + 6*x^{1327} + 4*x^{1326} + 17*x^{1325} + 18*x^{1324} + 18*x^{1323} + 15*x^{1322} + \\
& 17*x^{1321} + 8*x^{1320} + 8*x^{1319} + 13*x^{1318} + 8*x^{1317} + 16*x^{1316} + \\
& 5*x^{1315} + 15*x^{1314} + 17*x^{1313} + 9*x^{1312} + x^{1311} + 9*x^{1310} + 10*x^{1309} \\
& + 8*x^{1308} + 4*x^{1307} + 15*x^{1306} + 5*x^{1305} + 13*x^{1304} + 18*x^{1303} + \\
& 10*x^{1302} + 9*x^{1301} + 5*x^{1300} + 3*x^{1299} + 7*x^{1298} + 10*x^{1297} + 9*x^{1296} \\
& + 4*x^{1295} + 4*x^{1294} + 12*x^{1293} + 3*x^{1292} + 10*x^{1291} + 9*x^{1290} + \\
& 9*x^{1289} + 15*x^{1288} + 18*x^{1287} + 7*x^{1286} + 4*x^{1285} + 11*x^{1284} + \\
& 17*x^{1283} + 12*x^{1282} + 16*x^{1281} + 3*x^{1280} + 16*x^{1279} + 6*x^{1278} + \\
& 15*x^{1277} + 13*x^{1276} + 5*x^{1275} + 14*x^{1274} + 14*x^{1273} + 7*x^{1272} + \\
& 2*x^{1270} + 8*x^{1269} + 11*x^{1268} + 4*x^{1267} + 9*x^{1266} + 3*x^{1265} + 14*x^{1264} \\
& + 17*x^{1263} + 10*x^{1262} + 11*x^{1261} + 8*x^{1260} + 6*x^{1259} + 8*x^{1258} + \\
& 12*x^{1257} + 6*x^{1256} + 6*x^{1255} + 9*x^{1254} + 6*x^{1252} + 6*x^{1251} + 6*x^{1250} \\
& + 9*x^{1249} + 3*x^{1248} + 15*x^{1247} + 4*x^{1246} + 13*x^{1245} + 8*x^{1244} + \\
& 11*x^{1243} + 3*x^{1242} + 11*x^{1241} + 12*x^{1240} + 13*x^{1239} + 7*x^{1238} + \\
& 15*x^{1237} + 13*x^{1236} + 2*x^{1235} + 2*x^{1234} + 3*x^{1233} + 18*x^{1232} + \\
& 8*x^{1231} + 3*x^{1230} + 7*x^{1229} + 15*x^{1228} + 4*x^{1227} + 3*x^{1226} + 8*x^{1225} \\
& + 17*x^{1224} + 17*x^{1223} + 2*x^{1222} + 9*x^{1221} + 8*x^{1220} + 7*x^{1219} + \\
& 10*x^{1218} + 17*x^{1217} + 13*x^{1216} + 11*x^{1215} + 15*x^{1214} + 14*x^{1212} + \\
& 8*x^{1211} + 10*x^{1210} + 5*x^{1209} + 5*x^{1208} + 9*x^{1207} + 16*x^{1206} + 3*x^{1205} \\
& + 10*x^{1204} + 18*x^{1203} + 9*x^{1201} + 4*x^{1200} + 18*x^{1199} + 16*x^{1198} + \\
& 9*x^{1197} + x^{1196} + 16*x^{1195} + 11*x^{1194} + 9*x^{1193} + 11*x^{1192} + 5*x^{1191} \\
& + 17*x^{1190} + 14*x^{1189} + 18*x^{1188} + x^{1187} + 8*x^{1186} + x^{1185} + 13*x^{1184} \\
& + 10*x^{1183} + 16*x^{1182} + 11*x^{1181} + 4*x^{1180} + x^{1179} + 15*x^{1178} + \\
& 6*x^{1177} + 4*x^{1176} + 11*x^{1175} + 11*x^{1173} + 9*x^{1172} + 4*x^{1171} + \\
& 15*x^{1170} + 14*x^{1169} + 11*x^{1168} + 7*x^{1167} + x^{1166} + 17*x^{1164} + \\
& 10*x^{1163} + 13*x^{1162} + 9*x^{1161} + 3*x^{1160} + 8*x^{1159} + 3*x^{1158} + x^{1157} + \\
& 3*x^{1156} + 18*x^{1155} + 5*x^{1154} + 13*x^{1153} + 17*x^{1152} + 8*x^{1151} + \\
& 14*x^{1150} + 12*x^{1149} + 14*x^{1147} + 17*x^{1146} + 9*x^{1145} + 5*x^{1144} + \\
& 4*x^{1143} + x^{1142} + 10*x^{1141} + 11*x^{1140} + 15*x^{1139} + 3*x^{1138} + 5*x^{1137} \\
& + 12*x^{1136} + 2*x^{1135} + 17*x^{1134} + 15*x^{1133} + 9*x^{1132} + 9*x^{1131} + \\
& 2*x^{1130} + 17*x^{1129} + 17*x^{1128} + 15*x^{1127} + 10*x^{1126} + 9*x^{1125} + x^{1124} \\
& + 2*x^{1123} + x^{1122} + 5*x^{1121} + 10*x^{1120} + 17*x^{1119} + 2*x^{1118} + 4*x^{1117} \\
& + 18*x^{1116} + 3*x^{1115} + 13*x^{1114} + 7*x^{1113} + 9*x^{1112} + 16*x^{1111} + \\
& 17*x^{1110} + 4*x^{1109} + 18*x^{1108} + 16*x^{1107} + x^{1106} + 18*x^{1105} + \\
& 11*x^{1104} + 7*x^{1103} + 5*x^{1102} + 6*x^{1101} + 11*x^{1100} + 3*x^{1099} + \\
& 15*x^{1098} + 13*x^{1097} + 6*x^{1096} + 10*x^{1095} + 4*x^{1094} + 10*x^{1093} + \\
& 9*x^{1092} + 6*x^{1091} + 8*x^{1090} + 14*x^{1089} + 13*x^{1088} + 17*x^{1087} + \\
& 6*x^{1086} + 17*x^{1085} + 17*x^{1084} + 13*x^{1083} + 17*x^{934} + 17*x^{933} + \\
& 11*x^{932} + 8*x^{931} + 6*x^{930} + 7*x^{929} + 2*x^{928} + 3*x^{927} + 14*x^{926} + \\
& 16*x^{925} + 9*x^{924} + x^{923} + 2*x^{922} + 8*x^{921} + 15*x^{920} + 10*x^{919} + x^{918} \\
& + 5*x^{917} + 18*x^{916} + 7*x^{915} + 16*x^{914} + 14*x^{913} + x^{912} + 5*x^{911} + \\
& 17*x^{910} + x^{909} + 7*x^{908} + 11*x^{907} + 3*x^{906} + 10*x^{905} + 13*x^{904} + \\
& 3*x^{903} + 3*x^{902} + 12*x^{901} + 16*x^{900} + 13*x^{899} + 17*x^{898} + 18*x^{897} +
\end{aligned}$$

$$\begin{aligned}
& 5*x^{896} + 13*x^{895} + x^{894} + 15*x^{893} + 15*x^{892} + 17*x^{891} + 11*x^{890} + \\
& 16*x^{889} + 4*x^{888} + 15*x^{887} + 2*x^{886} + 3*x^{885} + 3*x^{884} + 13*x^{883} + \\
& 3*x^{882} + 8*x^{881} + 14*x^{880} + 18*x^{879} + 5*x^{878} + 17*x^{877} + 9*x^{876} + \\
& 4*x^{875} + 5*x^{874} + 11*x^{873} + 8*x^{872} + 17*x^{871} + x^{870} + x^{869} + 8*x^{868} \\
& + 14*x^{867} + 15*x^{866} + 15*x^{865} + 12*x^{864} + 9*x^{863} + 17*x^{862} + 15*x^{861} \\
& + 9*x^{860} + 11*x^{859} + 16*x^{857} + 7*x^{856} + 17*x^{855} + 14*x^{854} + 16*x^{853} + \\
& 18*x^{852} + x^{851} + 7*x^{850} + x^{849} + 7*x^{848} + 10*x^{847} + 6*x^{846} + 18*x^{845} \\
& + 8*x^{844} + 8*x^{843} + 18*x^{842} + 14*x^{841} + 14*x^{840} + 2*x^{839} + x^{838} + \\
& 8*x^{837} + 11*x^{836} + 6*x^{835} + 5*x^{834} + 17*x^{833} + 15*x^{832} + 13*x^{830} + \\
& 4*x^{829} + 5*x^{827} + 6*x^{826} + 3*x^{825} + 11*x^{824} + 4*x^{823} + 12*x^{822} + \\
& 14*x^{821} + 3*x^{820} + 9*x^{819} + 11*x^{818} + 14*x^{817} + 12*x^{816} + 11*x^{815} + \\
& 12*x^{814} + 16*x^{813} + 9*x^{812} + 17*x^{811} + 10*x^{810} + 2*x^{809} + 3*x^{808} + \\
& 6*x^{806} + 6*x^{805} + 9*x^{804} + 9*x^{803} + 9*x^{802} + 15*x^{801} + 7*x^{800} + \\
& 6*x^{799} + 15*x^{798} + 15*x^{797} + 14*x^{796} + 17*x^{795} + x^{793} + 11*x^{792} + \\
& 9*x^{791} + 15*x^{790} + 15*x^{789} + 7*x^{788} + 7*x^{787} + 3*x^{786} + 3*x^{785} + \\
& x^{784} + 4*x^{782} + 7*x^{781} + 18*x^{780} + 18*x^{779} + 11*x^{778} + 12*x^{777} + \\
& 14*x^{776} + 5*x^{775} + 3*x^{774} + 8*x^{773} + 2*x^{772} + 7*x^{771} + 14*x^{770} + \\
& 14*x^{769} + 16*x^{768} + 17*x^{767} + 5*x^{766} + 2*x^{765} + 7*x^{764} + 7*x^{763} + \\
& 14*x^{762} + 2*x^{761} + 10*x^{760} + x^{759} + 8*x^{758} + 11*x^{757} + 6*x^{756} + \\
& 14*x^{755} + 12*x^{754} + 11*x^{753} + x^{752} + 8*x^{751} + x^{750} + 7*x^{748} + \\
& 12*x^{747} + 2*x^{746} + 18*x^{745} + 15*x^{744} + 11*x^{743} + 4*x^{742} + 12*x^{741} + \\
& 12*x^{740} + 9*x^{739} + 11*x^{738} + 10*x^{737} + 9*x^{736} + 7*x^{735} + 11*x^{734} + \\
& 2*x^{733} + 15*x^{732} + 2*x^{731} + 9*x^{730} + 3*x^{729} + x^{728} + 2*x^{727} + \\
& 10*x^{726} + 16*x^{725} + 16*x^{724} + 18*x^{723} + 16*x^{722} + 11*x^{721} + 9*x^{720} + \\
& 14*x^{719} + 6*x^{718} + 10*x^{717} + 18*x^{716} + 16*x^{715} + 17*x^{714} + 8*x^{713} + \\
& 9*x^{711} + x^{710} + 16*x^{709} + 13*x^{708} + 11*x^{707} + 9*x^{706} + 5*x^{705} + \\
& 10*x^{704} + 4*x^{703} + 5*x^{702} + 8*x^{701} + 11*x^{700} + 7*x^{699} + 9*x^{698} + \\
& 16*x^{697} + 15*x^{696} + 2*x^{695} + 15*x^{694} + 6*x^{693} + 3*x^{692} + 18*x^{691} + \\
& 3*x^{690} + 16*x^{689} + 15*x^{688} + 14*x^{687} + 13*x^{686} + 14*x^{685} + 18*x^{684} + \\
& 6*x^{683} + 8*x^{682} + 16*x^{681} + x^{680} + x^{679} + 18*x^{678} + 7*x^{677} + 15*x^{676} \\
& + 8*x^{675} + 6*x^{674} + 5*x^{673} + 3*x^{672} + 3*x^{671} + 12*x^{670} + 2*x^{669} + \\
& 11*x^{668} + 18*x^{667} + 13*x^{666} + 9*x^{665} + 2*x^{664} + 11*x^{663} + 5*x^{662} + \\
& 8*x^{661} + 11*x^{660} + 17*x^{659} + 5*x^{658} + 13*x^{657} + 12*x^{656} + 13*x^{655} + \\
& 11*x^{654} + 10*x^{653} + 16*x^{652} + 13*x^{651} + 8*x^{650} + 6*x^{649} + 6*x^{648} + \\
& 10*x^{647} + 2*x^{646} + 14*x^{645} + 12*x^{644} + 18*x^{643} + 15*x^{642} + 10*x^{641} + \\
& 8*x^{640} + 13*x^{639} + 17*x^{638} + 17*x^{637} + 12*x^{636} + 5*x^{635} + 12*x^{634} + \\
& x^{633} + 16*x^{632} + 13*x^{631} + 8*x^{630} + 5*x^{629} + 13*x^{628} + 15*x^{627} + \\
& 14*x^{626} + 6*x^{625} + x^{624} + 5*x^{623} + 8*x^{622} + 17*x^{621} + 7*x^{620} + \\
& 6*x^{619} + 11*x^{618} + 12*x^{617} + x^{616} + 9*x^{615} + 3*x^{614} + 16*x^{613} + \\
& 18*x^{612} + 6*x^{611} + x^{610} + 8*x^{609} + 7*x^{608} + 11*x^{607} + 14*x^{606} + x^{605} \\
& + x^{603} + 11*x^{601} + 15*x^{599} + 17*x^{599} + 6*x^{598} + 7*x^{597} + x^{595} + \\
& 11*x^{594} + 8*x^{593} + 7*x^{592} + 18*x^{591} + 11*x^{590} + 12*x^{589} + 15*x^{588} + \\
& 15*x^{587} + 17*x^{586} + 6*x^{585} + 5*x^{584} + 14*x^{583} + 17*x^{582} + 18*x^{581} + \\
& 17*x^{580} + 18*x^{579} + 3*x^{578} + 2*x^{577} + 9*x^{576} + 7*x^{575} + 6*x^{574} + \\
& 7*x^{573} + 18*x^{572} + 12*x^{571} + 6*x^{570} + x^{569} + 6*x^{568} + 5*x^{567} + \\
& 13*x^{566} + 4*x^{565} + 11*x^{564} + 4*x^{563} + 11*x^{562} + 15*x^{561} + 8*x^{560} + \\
& 13*x^{559} + 17*x^{558} + 3*x^{557} + 2*x^{556} + 14*x^{555} + 5*x^{554} + 7*x^{553} + \\
& 15*x^{552} + 2*x^{551} + 18*x^{550} + 6*x^{549} + 16*x^{547} + 8*x^{546} + 11*x^{545} + \\
& 13*x^{544} + 2*x^{543} + 5*x^{542} + 10*x^{541} + 6*x^{540} + 2*x^{539} + 11*x^{538} + \\
& 11*x^{537} + 9*x^{536} + 18*x^{535} + 16*x^{534} + 15*x^{533} + 9*x^{532} + 15*x^{531} + \\
& 9*x^{529} + 10*x^{528} + 12*x^{527} + 5*x^{526} + 7*x^{525} + 13*x^{524} + 10*x^{523} + \\
& 8*x^{522} + 8*x^{521} + 2*x^{520} + 4*x^{519} + 2*x^{518} + 8*x^{517} + 4*x^{516} + \\
& 4*x^{515} + 17*x^{514} + 11*x^{513} + 11*x^{512} + 18*x^{511} + 12*x^{510} + 10*x^{509} + \\
& 14*x^{508} + 3*x^{506} + x^{505} + 6*x^{504} + 17*x^{503} + 5*x^{502} + 14*x^{501} + x^{500} \\
& + 7*x^{499} + 2*x^{498} + 7*x^{497} + 4*x^{495} + 6*x^{494} + 7*x^{493} + 11*x^{492} + \\
& 8*x^{491} + 6*x^{490} + 16*x^{489} + 12*x^{488} + 3*x^{487} + 14*x^{486} + 7*x^{485} + \\
& 4*x^{484} + 16*x^{483} + 6*x^{482} + 16*x^{481} + 5*x^{480} + 3*x^{479} + 6*x^{478} +
\end{aligned}$$

$$\begin{aligned}
& 7*x^{477} + 7*x^{476} + 6*x^{475} + 9*x^{474} + 8*x^{473} + x^{472} + 15*x^{471} + 9*x^{470} \\
& + 4*x^{469} + x^{468} + 2*x^{467} + 11*x^{466} + 13*x^{465} + 9*x^{464} + 4*x^{463} + \\
& 16*x^{462} + 4*x^{461} + 5*x^{460} + 16*x^{459} + 12*x^{458} + 11*x^{457} + 15*x^{456} + \\
& 14*x^{455} + 13*x^{454} + 2*x^{453} + 15*x^{452} + 7*x^{451} + 10*x^{450} + 7*x^{449} + \\
& 5*x^{448} + 6*x^{447} + 4*x^{446} + 13*x^{445} + 6*x^{444} + 2*x^{442} + 9*x^{441} + \\
& 6*x^{440} + 5*x^{439} + 16*x^{438} + 13*x^{437} + 18*x^{436} + 6*x^{435} + 16*x^{434} + \\
& 10*x^{433} + 4*x^{432} + 7*x^{431} + 8*x^{430} + 9*x^{429} + 4*x^{428} + 14*x^{427} + \\
& 8*x^{426} + x^{425} + 11*x^{424} + 4*x^{423} + 12*x^{422} + 10*x^{421} + 10*x^{420} + \\
& x^{419} + 2*x^{418} + 4*x^{417} + 10*x^{416} + 10*x^{415} + 3*x^{414} + 10*x^{413} + \\
& 8*x^{412} + 12*x^{411} + 17*x^{410} + 15*x^{409} + 3*x^{408} + 8*x^{407} + 11*x^{406} + \\
& 3*x^{405} + 9*x^{404} + 15*x^{403} + 16*x^{402} + 9*x^{401} + 9*x^{400} + 7*x^{399} + \\
& 11*x^{398} + 8*x^{397} + x^{396} + 8*x^{395} + 10*x^{394} + 7*x^{393} + 11*x^{391} + \\
& 18*x^{390} + 13*x^{389} + 3*x^{388} + 14*x^{387} + 9*x^{386} + 3*x^{385} + 9*x^{384} + \\
& 18*x^{383} + 10*x^{382} + 13*x^{381} + 9*x^{380} + 18*x^{379} + 17*x^{378} + 3*x^{377} + \\
& 9*x^{376} + 6*x^{375} + 12*x^{374} + 17*x^{373} + 4*x^{372} + 13*x^{371} + x^{369} + \\
& 6*x^{368} + 15*x^{367} + 10*x^{366} + 15*x^{365} + 8*x^{364} + 12*x^{363} + 4*x^{362} + \\
& 10*x^{361} + 6*x^{360} + x^{359} + 4*x^{358} + 11*x^{357} + 17*x^{356} + 3*x^{355} + \\
& 12*x^{354} + 3*x^{353} + 17*x^{352} + 8*x^{351} + x^{350} + 7*x^{349} + 6*x^{348} + \\
& 7*x^{347} + 15*x^{346} + 6*x^{345} + 18*x^{344} + 16*x^{343} + 3*x^{341} + 17*x^{340} + \\
& 2*x^{339} + 14*x^{338} + 9*x^{337} + 11*x^{336} + 4*x^{335} + x^{334} + 17*x^{333} + \\
& 4*x^{332} + 8*x^{331} + 2*x^{328} + 7*x^{327} + 6*x^{326} + 8*x^{325} + 16*x^{324} + \\
& 13*x^{323} + 2*x^{322} + 9*x^{321} + 7*x^{320} + 11*x^{319} + 10*x^{318} + 11*x^{317} + \\
& 7*x^{316} + 5*x^{315} + 14*x^{314} + 12*x^{313} + 10*x^{312} + 5*x^{311} + 10*x^{310} + \\
& 15*x^{309} + 17*x^{308} + x^{307} + 4*x^{305} + 8*x^{304} + 9*x^{303} + 3*x^{302} + \\
& 14*x^{301} + 3*x^{300} + 17*x^{299} + 8*x^{298} + 17*x^{297} + 8*x^{296} + 14*x^{294} + \\
& 6*x^{293} + 16*x^{292} + 8*x^{291} + 8*x^{290} + 14*x^{289} + 13*x^{288} + 12*x^{287} + \\
& 10*x^{286} + x^{285} + 5*x^{284} + 11*x^{283} + 15*x^{282} + 7*x^{281} + 13*x^{280} + \\
& 8*x^{279} + 4*x^{278} + 13*x^{277} + 4*x^{276} + 8*x^{275} + 8*x^{274} + 16*x^{273} + \\
& 3*x^{272} + 10*x^{271} + 17*x^{270} + 7*x^{269} + 15*x^{268} + 18*x^{267} + 8*x^{266} + \\
& 15*x^{264} + 11*x^{263} + 3*x^{262} + 9*x^{261} + 6*x^{260} + 18*x^{259} + 15*x^{258} + \\
& 15*x^{257} + 17*x^{256} + 3*x^{255} + 10*x^{254} + 10*x^{253} + 13*x^{252} + 4*x^{251} + \\
& 10*x^{250} + 14*x^{249} + 5*x^{248} + 16*x^{247} + 2*x^{246} + 3*x^{245} + 17*x^{244} + \\
& 13*x^{241} + 8*x^{240} + 4*x^{239} + 7*x^{238} + 5*x^{237} + 6*x^{236} + 17*x^{235} + \\
& 5*x^{233} + 9*x^{232} + 2*x^{231} + 18*x^{230} + 2*x^{229} + 8*x^{228} + 11*x^{227} + \\
& 17*x^{224} + 2*x^{223} + 11*x^{222} + 14*x^{221} + 6*x^{220} + x^{219} + 16*x^{218} + \\
& 7*x^{217} + 10*x^{216} + 7*x^{215} + 7*x^{214} + 9*x^{213} + 9*x^{212} + 13*x^{211} + \\
& x^{210} + 15*x^{209} + 10*x^{208} + 3*x^{207} + 13*x^{206} + 10*x^{205} + 4*x^{204} + \\
& 5*x^{203} + 16*x^{202} + 4*x^{201} + x^{200} + 7*x^{199} + 4*x^{198} + 11*x^{197} + \\
& 6*x^{196} + 17*x^{195} + 17*x^{194} + 2*x^{193} + 9*x^{192} + 5*x^{191} + 14*x^{190} + \\
& 2*x^{188} + 16*x^{187} + 3*x^{186} + 6*x^{185} + 7*x^{184} + x^{183} + 3*x^{182} + \\
& 17*x^{181} + 4*x^{180} + 12*x^{179} + 2*x^{178} + 16*x^{177} + 6*x^{176} + 3*x^{175} + \\
& 4*x^{174} + 5*x^{172} + 6*x^{171} + 12*x^{170} + 9*x^{169} + 10*x^{168} + 17*x^{167} + \\
& 3*x^{166} + 14*x^{165} + 11*x^{164} + 12*x^{163} + 7*x^{162} + 7*x^{161} + 7*x^{160} + \\
& 18*x^{159} + 9*x^{158} + 3*x^{156} + 3*x^{155} + 14*x^{154} + 9*x^{153} + 11*x^{152} + \\
& 16*x^{151} + 16*x^{150} + 4*x^{149} + 10*x^{148} + 2*x^{147} + 13*x^{146} + 16*x^{145} + \\
& 8*x^{144} + 17*x^{143} + 7*x^{142} + 9*x^{141} + 6*x^{140} + 12*x^{139} + 4*x^{138} + \\
& 11*x^{137} + 9*x^{136} + 3*x^{135} + 5*x^{134} + 13*x^{133} + 2*x^{132} + 4*x^{131} + \\
& 12*x^{130} + 18*x^{129} + 13*x^{128} + 12*x^{127} + 4*x^{126} + 6*x^{125} + 4*x^{124} + \\
& 18*x^{123} + 14*x^{122} + 12*x^{121} + 13*x^{120} + 4*x^{119} + 18*x^{118} + 9*x^{117} + \\
& 12*x^{116} + x^{115} + 13*x^{114} + 2*x^{113} + x^{112} + 14*x^{111} + 9*x^{110} + 3*x^{109} \\
& + x^{108} + 2*x^{107} + 10*x^{106} + 11*x^{105} + 9*x^{104} + 11*x^{103} + 2*x^{102} + \\
& 9*x^{101} + 14*x^{100} + 17*x^{99} + 11*x^{98} + 5*x^{97} + 15*x^{96} + 10*x^{95} + \\
& 12*x^{93} + 10*x^{92} + 13*x^{90} + 2*x^{89} + 4*x^{88} + 14*x^{87} + 4*x^{86} + 12*x^{85} + \\
& 18*x^{84} + 3*x^{83} + 3*x^{82} + 15*x^{81} + x^{80} + 13*x^{79} + 10*x^{78} + 8*x^{77} + \\
& 15*x^{76} + 9*x^{75} + 2*x^{74} + 2*x^{73} + x^{71} + 17*x^{70} + 10*x^{69} + 3*x^{68} + \\
& 15*x^{67} + 11*x^{66} + 5*x^{65} + 15*x^{63} + 7*x^{62} + 13*x^{61} + 6*x^{60} + 16*x^{59} + \\
& 15*x^{58} + 7*x^{57} + 11*x^{56} + 8*x^{54} + x^{53} + 9*x^{52} + 5*x^{51} + 8*x^{50} +
\end{aligned}$$

```

13*x^49 + 11*x^48 + 18*x^47 + 16*x^46 + 2*x^45 + 3*x^44 + 10*x^43 + 12*x^42
+ 8*x^41 + 5*x^40 + 3*x^39 + 12*x^38 + 3*x^37 + 8*x^36 + 12*x^35 + 16*x^34 +
2*x^33 + 2*x^32 + 16*x^31 + 9*x^30 + 12*x^29 + x^28 + 10*x^27 + 18*x^26 +
17*x^25 + 18*x^24 + 2*x^23 + 17*x^22 + 12*x^21 + 14*x^20 + 8*x^19 + 16*x^18
+ 2*x^17 + 2*x^16 + 17*x^15 + 6*x^14 + 17*x^13 + 4*x^12 + 9*x^10 + 9*x^9 +
7*x^8 + 14*x^6 + x^5 + 11*x^4 + 15*x^3 + 13*x^2 + x + 6

ylhs mod psi;

9*x^11 + 13*x^10 + 15*x^9 + 15*x^7 + 18*x^6 + 17*x^5 + 8*x^4 + 12*x^3 + 8*x + 6

/* The above is completely correct. */
/*********************************************
/* Then we need the y component of 2*(x^q,y^q) . */

/* Recall that y3=m(x1-x3)-y1, m = (3x1^2+A)/(2y1).
Recall that y3=m(x1-x3)-y1, m = (3(x^q)^2+A)/(2y^q).

We rewrite: m = y* (3*(x^q)^2+A)/(2*y^(q+1)) = y*
(3*(x^q)^2+A)/(2*ell^((q+1)/2)).
So the right hand side of y, is y3=m(x1-x3)-y1 */

/* The slope is
mm := y* (3*(x^q)^2+A) / (2*ell^((q+1) div 2)) */

/* The formula is x3=m^2-2x1, y3=m(x1-x3)-y1, m = (3x1^2+A)/(2y1). A=2
here.*/
/* y3= y * (3*(x^q)^2+A) / (2*ell^((q+1) div 2)) *(x^q-xrhs)-y^q*/
/* Divide out by y
y3= (3*(x^q)^2+A) / (2*ell^((q+1) div 2)) *(x^q-xrhs)-y^(q-1)
Then convert into x and divide. */

/* yrhs := (3*(x^q)^2+A) / (2*ell^((q+1) div 2)) *(x^q-xrhs)-ell^((q-1)
div 2); */

/* Denominator: */

dd := 2*ell^((q+1) div 2);

/* Numerator: */

nn := 3*(x^q)^2+A;

yr1 := (nn div Gcd(dd,psi));

yr2 := (dd div Gcd(dd,psi));

/* yq is the denominator inverted. */

yp,yq,yr := Xgcd(yr2,psi);

yrhs := yq*yr1*(x^q-xrhs)-ell^((q-1) div 2);

yrhs;

```

$$\begin{aligned}
& 10*x^{136} + 8*x^{135} + 6*x^{134} + 10*x^{133} + 5*x^{132} + 10*x^{131} + 7*x^{130} + \\
& 10*x^{129} + 6*x^{128} + 16*x^{127} + 4*x^{126} + 12*x^{125} + 4*x^{124} + 5*x^{123} + \\
& 9*x^{122} + 16*x^{121} + 17*x^{120} + 10*x^{119} + 2*x^{118} + 7*x^{117} + 12*x^{116} + \\
& 17*x^{115} + 13*x^{114} + x^{98} + 16*x^{97} + 12*x^{96} + x^{95} + 10*x^{94} + x^{93} + \\
& 14*x^{92} + x^{91} + 12*x^{90} + 13*x^{89} + 8*x^{88} + 5*x^{87} + 8*x^{86} + 10*x^{85} + \\
& 18*x^{84} + 13*x^{83} + 15*x^{82} + x^{81} + 4*x^{80} + 14*x^{79} + 5*x^{78} + 15*x^{77} + \\
& 7*x^{76} + 5*x^{68} + 4*x^{67} + 7*x^{66} + 8*x^{65} + x^{64} + 5*x^{63} + 16*x^{62} + \\
& 11*x^{61} + 3*x^{60} + 6*x^{59} + x^{58} + 18*x^{57} + 13*x^{56} + 7*x^{55} + 3*x^{54} + \\
& 7*x^{53} + 8*x^{52} + 15*x^{51} + 18*x^{50} + 16*x^{49} + 18*x^{48} + 13*x^{47} + 12*x^{46} \\
& + 15*x^{45} + 10*x^{44} + 7*x^{43} + 9*x^{42} + 3*x^{41} + 16*x^{40} + 10*x^{39} + 11*x^{38} \\
& + 16*x^{30} + 9*x^{29} + 11*x^{28} + 17*x^{27} + 7*x^{26} + 17*x^{25} + 8*x^{24} + 9*x^{23} \\
& + 9*x^{22} + 2*x^{21} + 13*x^{20} + 18*x^{19} + 12*x^{18} + 3*x^{17} + 15*x^{16} + 8*x^{15} \\
& + 4*x^{14} + 18*x^{13} + 7*x^{12} + 16*x^{11} + x^{10} + 10*x^9 + 4*x^8 + 5*x^7 + \\
& 16*x^6 + 5*x^5 + 8*x^4 + 10*x^3 + x^2 + 18*x + 12
\end{aligned}$$

```
ydiv:=(ylhs-yrhs) mod (psi);
```

```
ydiv;
```

```
18*x^11 + 7*x^10 + 11*x^9 + 11*x^7 + 17*x^6 + 15*x^5 + 16*x^4 + 5*x^3 + 16*x + 12
```

```
/* The answer is nonzero so we get a=-2 mod 5. Thus, a=3. */
```

## References

- [CE] Certicom, Online Elliptic Curve Cryptography Tutorial,  
[http://www.certicom.com/index.php?action=ecc\\_tutorial,home](http://www.certicom.com/index.php?action=ecc_tutorial,home). 10MAR2007.
- [Ga] Garret, Paul “Making, Breaking Codes: Introduction to Cryptology” Prentice Hall, 2000.
- [Ko] N. Koblitz, *Elliptic curve cryptosystems*, in *Mathematics of Computation* 48, 1987, pp. 203–209.
- [Mi] V. Miller, *Use of elliptic curves in cryptography*, CRYPTO 85, 1985.
- [Sc] Schoof, René(I-ROME2) Counting points on elliptic curves over finite fields. (English summary) Les Dix-huitièmes Journées Arithmétiques (Bordeaux, 1993). J. Théor. Nombres Bordeaux 7 (1995), no. 1, 219–254.
- [St] Stinson, Douglas “Cryptography: Theory and Practice” 1956 Chapman & Hall/CRC, 2002.
- [WA] Washington, Lawrence “Elliptic Curves: Number Theory and Cryptography” 2003 Chapman & Hall/CRC, 2003.