Hyperelliptic Curve Cryptography

Paul Sheridan

April 29, 2003

Abstract

We Begin with a brief overview of El Gamal public key cryptography, and proceed to apply it to the jacobian of a hyperelliptic curve. The main goal is to present an implementation of El Gamal over the jacobian of a hyperelliptic curve in Maple 8. We conclude by giving a concise survey of current results on how to choose a curve suitable for cryptographic purposes.

1 Introduction

Curve based cryptosystems became popular with the advent of elliptic curve cryptography by Koblitz [8] in 1987. The security of the elliptic curve cryptosystem relies directly on the difficulty of calculating discrete logarithms in the groups of points on an elliptic curve. Its efficiency lies in the existence of simple formula for adding points on the curve, allowing for fast arithemetic for addition in the group. Further study in the field was directed toward finding other groups for which the discrete logarithm problem is computationally infeasible. Any such group would easily give rise to a cryptosystem via El Gamal. The group of points on an elliptic curve is precisely the jacobian of a hyperelliptic curve of genus 1. It is natural to consider hyperelliptic curves of higher genus as alternate avenues for the application of El Gamal. In general, there is no known subexponential algorithm for computing discrete logarithms in the jacobian of this class of curves. Hence, so long as we can preform calculations in the jacobian quickly, hyperelliptic curves are of interest in cryptography. In 1989 Cantor [1] unveiled a quadratic running time algorithm for adding points in the jacobian of a hyperelliptic curve. Hyperelliptic curve cryptography was introduced by Koblitz [9] in 1989.

In this paper, we begin Chapter 2 with a basic overview of public key cryptography, brought to life via the El Gamal cryptosystem. El

Gamal can be adapted for use in any finite cyclic group, however only those groups for which the discrete logarithm problem is hard will be of interest for cryptographic purposes. We first apply this cryptosystem to the group of units of a finite field, followed by the group of points on an elliptic curve. The goal being to ease, as naturally as possible, into El Gamal over the jacobian of a hyperelliptic curve. Also, we discuss some benefits of applying El Gamal in more and more general settings.

In Chapter 3 we cover some basic theory about hyperelliptic curves over finite fields. The aim is twofold: first to construct the jacobian of a hyperelliptic curve, and secondly to introduce Cantor's algorithms [1] for adding elements in the jacobian. Complete with a finite cyclic group and a feasible means to add the elements, El Gamal can be implemented with relative ease.

A complete description and implementation of the cryptosystem, in Maple 8, is presented in Chapter 4. We discuss the implementation and give examples. The program is of little pratical value. Calculations would be far too time consuming in a jacobian which is invulnerable to a brute force attack, however, it does illustrate how hyperelliptic curve cryptography works in practice. In addition, the reader may find some of the code to be of interest independently of any cryptographic applications. Embedded in the code is the basic arithemetic on polynomials over a finite field. The operations range from simple addition and multiplication of polynomials, to calculating the gcd and preforming the extended euclidean algorithm upon two polynomials.

Finally, we conclude in Chapter 5 with a merry discussion on means to ensure the security of a cryptosystem based on the discrete logarithm problem in a finite abeilan group. Several serious restrictions limit the class of hyperelliptic curves which are considered to give rise to a secure cryptosystem. The reader is encouraged to study [15] for a detailed means of constructing good curves.

2 Cryptography

Cryptography is the study of concealing information via ciphers which can later be revealed only by legitimate receivers employing a secret key.

The basic communication scenario is outlined in Figure 1. Alice wishes to send a message, called the **plaintext**, to Bob. Alice's **encryption key** gives her a rule by which she can encipher the plaintext so as it remains unintelligible to Eve. She securely transmits the resulting **ciphertext** to Bob, who may decipher the transmission with

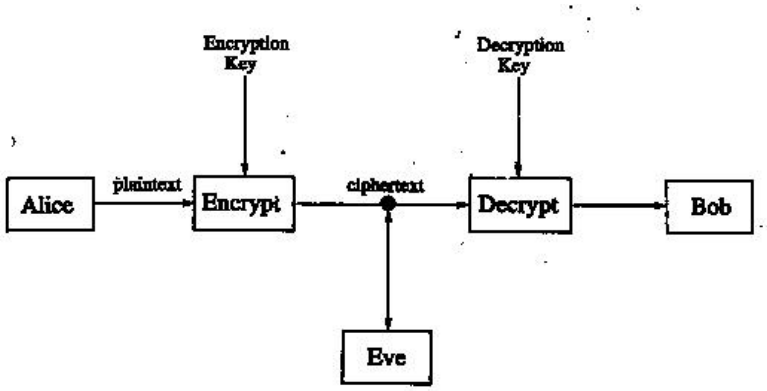

Figure 1:

a related **decryption key**.

Traditionally Alice and Bob agree upon a key and keep it secret at all cost. Alice enciphers the plaintext using the key. Bob simply applies the key in reverse to decrypt the message. Any such means of communication is called a **private key** cryptosystem because the security of the system depends upon the secrecy of the key between the two parties.

Efforts to circumvent such a precarious practice led to the development of what is called a **public key** cryptosystem. The idea is to find a mathematical process that is fast to preform in one direction, but extraordinarily slow to preform in the inverse direction. Such a process would yield a two-part key. The first part is used for encryption and is made public. The other is kept private and is used for decryption. Suppose Alice wishes to send a message to Bob. Bob makes the encryption key public to anyone including Eve. Alice then encrypts the plaintext using Bob's public key. Bob can then use his private key to decrypt the message. Unfortunately for Eve, she may not read the Alice's message to Bob. Theoretically, with knowledge of his public key alone she is unable to reverse the mathematical process on which the transition from Bob's private key to his public key is based.

The form of public key cryptography which we adopt throughout this paper is based on calculating discrete logarithms in finite cyclic groups. The cryptosystem was initially proposed in the literature by El Gamal in 1985 [2]. In the following sections we begin with an outline of the inner workings of this cryptosystem. Then we proceed to give examples of El Gamal in several different settings.

2.1 The El Gamal cryptosystem

El Gamal is based on the following one way process. Let $\mathbb{G}$ be a finite cyclic group and $g \in \mathbb{G}$. Given $y \in \mathbb{G}$, the **discrete logarithm problem to the base** g, is to find an integer a which satisfies $y = g^a$. a is called the **discrete logarithm of** y **to the base** g.

Calculating discrete logarithms is generally considered to be difficult. When the group $\mathbb{G}$ is chosen carefully, there is no known polynomial time algorithm for calculating discrete logarithms. However, the inverse problem of exponentiating can be easily computed using a basic square and multiply scheme.

Formally, the El Gamal cryptosystem works as follows. Suppose Alice wishes to send a message represented as an element m in $\mathbb{G}$ to Bob.

1. Alice and Bob agree on the cryptosystem parameters by choosing a group $\mathbb{G}$, and a base g.
2. Alice makes two keys:
   (a) She chooses an integer n_A as her private decryption key.
   (b) She computes g^{n_A} for her public encryption key.
3. Bob makes two keys:
   (a) He chooses an integer n_B as his private decryption key.
   (b) He computes g^{n_B} for his public encryption key.
4. Alice sends a message $m \in \mathbb{G}$ to Bob:
   $c = m\,(\text{ Bob's public key }) \overset{\text{Alice's private key}}{} = m\,(g^{n_B})^{n_A}$.
5. Bob reads the message:
   $m = c\left[(\text{ Alice's public key }) \overset{\text{Bob's private key}}{}\right]^{-1} = c\,[(g^{n_A})^{n_B}]^{-1}$.

In slightly less technical terms El Gamal works by disguising the plaintext m by multiplying it by $(g^{n_B})^{n_A}$, giving the ciphertext c. Bob can then easily uncloak the ciphertext by computing the inverse of $(g^{n_A})^{n_B}$, which is done by exponentiating Alice's public key g^{n_A} by his own private key n_A.

Clearly, if Eve can compute discrete logarithms in the finite cyclic group $\mathbb{G}$, then El Gamal becomes insecure. However, whether or not the security of the cryptosystem relies on Eve's ability to compute discrete logarithms alone remains an open question. We do not concern ourselves with the second possibility. Instead, we give examples of El Gamal in two different group settings where the discrete logarithm problem appears to be intractable.

2.2 The Galois fields

The purpose of the current section is to give a concise review of the theory behind Galois fields. Our point of departure will be the familiar finite fields $\mathbb{F}_p$, for p a prime. From there we will proceed to give a method of representing any finite field based on polynomials in the integral domain $\mathbb{F}_p[x]$. Our goal is to apply El Gamal to the group of multiplicative units of these fields.

We can easily construct finite fields with a prime number of elements by taking the quotient of the integral domain $\mathbb{Z}$ with the ideal $p\mathbb{Z}$, for $p \in \mathbb{Z}$ a prime. Since every prime ideal of $\mathbb{Z}$ is maximal, it follows that $\frac{\mathbb{Z}}{p\mathbb{Z}} = \mathbb{F}_p$ is a field. The elements of the finite field are precisely the equivalence classes $a+p\mathbb{Z}$ with $a \in \mathbb{Z}$ with the usual addition and multiplication. Finally, the field has a prime number of elements because the classes are determined uniquely for $a \in \{0, 1, \ldots, p-1\}$.

Any finite field is called a **Galois field**. There are Galois fields other than those with a prime number of elements, however, any finite field must have prime characteristic. The standard route for constructing these fields begins with the polynomial ring $\mathbb{F}_p[x]$. Proceeding by analogy to the above construction we take a polynomial $f \in \mathbb{F}_p[x]$, of degree n, which is irreducible over $\mathbb{F}_p$. Then the quotient of $\mathbb{F}_p[x]$ with the ideal generated by f is indeed a field. The notation is $\frac{\mathbb{F}_p[x]}{<f>} = \mathbb{F}_{p^n}$. Now, elements in the field are of the form $g+ < f >$ with $g \in \mathbb{F}_p[x]$, again with the standard addition and multiplication for a residue class ring. Furthermore, each equivalence class is guaranteed a unique representative g with degree less than that of f by the remainder theorem. Therefore, by a simple counting argument, we find the Galois field $\mathbb{F}_{p^n}$ has precisely p^n elements.

To complete our description of finite fields we give some elementary results without any indication of proof. First, for each $n \in \mathbb{N}$ there is at least one irreducible polynomial f of degree n over $\mathbb{F}_p$. Secondly, any two fields obtained from irreducible polynomials of the same degree are isomorphic. Finally, all finite field are of this form. Hence, we conclude that for any prime p and integer n, there is a unique Galois field $\mathbb{F}_{p^n}$. Furthermore, all finite fields are of this form.

2.3 El Gamal over the group of units of a Galois field

In this section we breathe life into El Gamal by applying it to the group of units of a Galois field denoted by $(\mathbb{F}_{p^n})^*$. Recall that El Gamal operates in a finite cyclic group. For the Galois fields this is indeed the case. The group of units is cyclic with $p^n - 1$ elements.

Suppose Alice and Bob have agreed on a way to represent a message as a polynomial $m(x) \in \mathbb{F}_{p^n}$. For Alice to send the plaintext to Bob the setup is identical.

1. Alice and Bob agree on the cryptosystem parameters by choosing a Galois field $\mathbb{F}_{p^n}$, and a base polynomial $f(x)$.
2. Alice makes two keys:
   (a) She chooses an integer n_A as her private decryption key.
   (b) She computes $f(x)^{n_A}$ for her public encryption key.
3. Bob makes two keys:
   (a) He chooses an integer n_B as his private decryption key.
   (b) He computes $f(x)^{n_B}$ for his public encryption key.
4. Alice sends a message $m(x) \in \mathbb{F}_{p^n}$ to Bob:
   $c(x) = m(x)\,(\text{ Bob's public key }) \text{ Alice's private key } = m(x)\,(f(x)^{n_B})^{n_A}$.
5. Bob reads the message:
   $m(x) = c(x)\left[(\text{ Alice's public key }) \text{ Bob's private key }\right]^{-1} = c(x)\,[(f(x)^{n_A})^{n_B}]^{-1}$.

The reader should not be overly concerned with the amount of computation involved to transmit a plaintext. In practice, the Galois fields $\mathbb{F}_{2^n}$ are used to speed up the calculations. Furthermore, the exponentiations are independent of the plaintext and can be preformed in advance. We should also make note that we are calculating with equivalence classes of polynomials. Thus following each exponentiation we may use the division algorithm to take the unique representative with degree less than n. With a typical square and multiply exponentiation algorithm the computations are manageable.

El Gamal in $\mathbb{F}_{2^n}$ is considered to be secure when $n > 800$ and the order of the group contains a large prime factor. The first condition is to ensure discrete logarithms cannot be calculated directly or using the index calculus algorithm [11]. The large prime factor is required for the cryptosystem to be invulnerable to the Pohlig-Hellman attack which is discussed in Section 5.1.

2.4 Elliptic curves over Galois fields

The theory of elliptic curves is vast and dates back over 150 years. A more complete introduction to elliptic curves would begin with a geometric foundation. For brevity's sake we leap directly into material of cryptographic interest.

Definition 1 *An **elliptic curve** E over a Galois field $\mathbb{F}_q$ is a curve given by an equation of the form*

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6,\ a_i \in \mathbb{F}_q \tag{1}$$

with the added condition that any solution $(x, y) \in \overline{\mathbb{F}_q}^2$ to 1 cannot simultaneously satisfy the partial derivatives

$$a_1y = 3x^2 + 2a_2x + a_4, \tag{2}$$

$$2y + a_1x + a_3 = 0. \tag{3}$$

$\overline{\mathbb{F}_q}^2$ is the closure of $\mathbb{F}_q$ and q is shorthand for the previous notation p^n. The reason for the change in notation is because we may desire to look at solutions to 1 over field extensions $\mathbb{F}_{q^r}$. Formally, we let $E(\mathbb{Z}_q)$ denote the set of points $(x, y) \in \mathbb{F}_q^2$ which satisfy 1 along with $\mathbb{O}$ the point at infinity.

The object $E(\mathbb{F}_q)$ is in reality more than a mere set of points. Elliptic curves were first studied over less exotic fields such as the rationals and the real numbers. In that setting it becomes clear that we can geometrically impose an addition rule on the elements of $E(\mathbb{F}_q)$ which fashions it into an abelian group.

When the characteristic of the underlying Galois field is not 2 or 3 the elliptic curve equation reduces to

$$E :\ y^2 = x^3 + ax + b,\ a, b \in \mathbb{F}_q. \tag{4}$$

In this case the sum of two points $P+Q = (x_1, y_1)+(x_2, y_2) = (x_3, y_3)$ in $E(\mathbb{F}_q)$ is given by the formulas:

1. If $P \neq Q$, then
   (a) $x_3 = \left(\frac{y_2-y_1}{x_2-x_1}\right)^2 - x_1 - x_2$;
   (b) $y_3 = -y_1 + \left(\frac{y_2-y_1}{x_2-x_1}\right)(x_1 - x_3)$.
2. If $P = Q$, then
   (a) $x_3 = \left(\frac{3x_1^2}{2y_1}\right)^2 - 2x_1$;
   (b) $y_3 = -y_1 + \left(\frac{3x_1^2}{2y_1}\right)(x_1 - x_3)$.
3. If $x_1 = x_2$ and $y_1 = -y_2$, then we define $P + Q = \mathbb{O}$ the point at infinity.

Finally, by defining $P + \mathbb{O} = P$ for any point P it follows that as we would expect $\mathbb{O}$ acts as group identity. The corresponding formulas when E is defined over a field of characteristic 2 or 3 are more cumbersome.

2.5 El Gamal over the group of points on an elliptic curve

In this section we break with the theoretical motif to give a concrete example of El Gamal in our current setting. The elliptic curve cryptosystem has several advantages over El Gamal in the group of units of a Galois field. First, elliptic curves give rise to a wider selection of groups. That is for each finite field there is exactly one group of units. But we can find many elliptic curves over a given finite field. Secondly, additions in $E(\mathbb{F}_q)$ can be computed efficiently by using the formulas from section 2.4. Finally, there is no known subexponential algorithm for calculating discrete logarithms in $E(\mathbb{F}_q)$ when the curve is chosen properly.

For an elliptic curve E defined over $\mathbb{F}_q$ there is no guarantee the group of points $E(\mathbb{F}_q)$ is cyclic. To account for this fact, we make a slight alteration to the discrete logarithm problem.

Let $E(\mathbb{F}_q)$ be the group of points on an elliptic curve E with base $Q \in E(\mathbb{F}_p)$. Given $P \in E(\mathbb{F}_q)$, the **elliptic curve discrete logarithm problem to the base** Q is to find an integer n so that $nQ = \underbrace{Q + \cdots + Q}_{n \text{ times}} = P$. (if such an n exists)

Under these assumptions the discrete logarithm problem will have no solution exactly when the base point Q and P are in different cyclic subgroups. The consequences for El Gamal are disastrous. Bob can decrypt a message encoded as a point M from Alice only when it lies in $< Q >$. Basically, El Gamal is restricted to operate in the cyclic subgroup generated by the base point Q. Therefore, in the elliptic curve version of El Gamal the cyclic group used is precisely the subgroup of $E(\mathbb{F}_q)$ generated by the base point Q.

As promised we finish this section with an example. The group operations are written additively as is always the case with $E(\mathbb{F}_q)$.

Example 1 *1. Alice and Bob agree on the elliptic curve cryptosystem parameters by choosing a curve* $E : y^2 = x^3 + 3x + 45$ *over a finite field, say* $\mathbb{F}_{8831}$ *and a base point* $Q = (4, 11)$.

2. Alice creates keys:

(a) Private key: The integer $n_A = 8$.

(b) Public key: The point $n_A Q = 8(4, 11) = (5415, 6321)$.

3. Bob creates keys:

(a) Private key: The integer $n_B = 3$.

(b) Public key: The point $n_B Q = 3(4, 11) = (413, 1808)$.

4. *Alice sends a message* $P_m = (5, 1743)$ *in* $E(\mathbb{F}_{8831})$ *to Bob:* $C_m = P_m + (\text{ Bob's public key })(\text{ Alice's private key }) = P_m + (n_B Q)(n_A) = (6626, 3576)$.

5. *Bob reads the message:* $P_m = C_m - (\text{ Alice's private key })(\text{ Bob's public key }) = C_m - (n_A Q)(n_B) = (5, 1743)$.

2.6 Discussion

We have presented El Gamal in the group of units of a Galois field and over the group of points on an elliptic curve. Both have their respective advantages and disadvantages. However, they both satisfy the two main requirements for El Gamal to be useful. Namely, (1) the discrete logarithm problem is difficult, and (2) computations in the group are efficient. To date the elliptic curve cryptosystem is considered to be secure. Hyperelliptic curves are a generalization of elliptic curves which come from algebraic geometry. In the next chapter we present the theory required to apply El Gamal to this new class of curves.

3 Hyperelliptic curves

In this chapter we establish notation and give some basic definitions and results about hyperelliptic curves over finite fields. Unlike elliptic curves, the set of points on a hyperelliptic curve does not form a group. The goal of the present chapter is to associate a group structure, called the **jacobian**, to each hyperelliptic curve. The utility of El Gamal hinges upon our ability to compute efficiently in the jacobian. We address the issue directly by manufacturing a representation of elements in the jacobian via reduced divisors. Finally, we present Cantor's algorithms [1] for adding elements in the jacobian of a hyperelliptic curve.

3.1 Definitions and examples

Definition 2 *A* ***hyperelliptic curve*** C ***of genus g*** *over a Galois field* $\mathbb{F}_{p^n} = \mathbb{F}_q$ *is a curve in* $\mathbb{F}_q[x, y]$ *given by an equation of the form*

$$C : y^2 + h(x)y = f(x), \tag{5}$$

where

1. $h \in \mathbb{F}_q[x]$ *has degree at most* g,

2. $f \in \mathbb{F}_q[x]$ *is monic with degree* $2g+1$.

and where any solution $(x,y) \in \overline{\mathbb{F}_q}^2$ *to 2 cannot simultaneously satisfy the partial derivatives*

$$2y + h(x) = 0 \tag{6}$$

$$h'(x)y - f'(x) = 0. \tag{7}$$

The genus of C has geometric significance when we extend the definition of a hyperelliptic curve to fields such as the real and complex numbers. For our objectives it suffices to consider it as a strictly algebraic quantity. It should also be noted that we obtain an elliptic curve precisely when the genus is 1.

The definition reduces to a more manageable form when the characteristic of the underlying finite field is not 2, and $h(x) = 0$.

Definition 3 *A curve*

$$C: \ y^2 = f(x) \tag{8}$$

defined over $\mathbb{F}_q$, $char(\mathbb{F}_q) \neq 2$, *is* ***hyperelliptic of genus*** g *when* f *has degree* $2g+1$, *and* f *has no repeated roots in* $\overline{\mathbb{F}_q}$.

We use the next example as an excuse to introduce a little terminology.

Example 2 *Consider the hypereliptic curve* $C: \ y^2 = f(x) = x^5 + 1$ *defined over* $\mathbb{F}_3$. C *is well defined because* f *has no repeated roots. The* $\mathbb{F}_{3^2}$-***points*** *of* C *are the points on the curve which lie in* $\mathbb{F}_{3^2}$, *a field extension of* $\mathbb{F}_3$. *Using the irreducible polynomial* $\alpha^2 - \alpha - 1$ *to construct the extension field* $\mathbb{F}_{3^2}$, *we find 10* $\mathbb{F}_{3^2}$*-points on the curve:* $C(\mathbb{Z}_{3^2}) = \{(0,1),(0,2),(1,1+\alpha),(2,0),(2+\alpha,2+\alpha),(2+\alpha,1+2\alpha),(2\alpha,\alpha),(2\alpha,2\alpha),\mathbb{O}\}$.

Definition 4 *A point* $P = (x,y)$ *on* $\mathbb{C}$ *has an* ***opposite*** *point given by* $\widetilde{P} = (x, -y - h(x))$. P *is called a* ***special*** *point when* $P = \widetilde{P}$. *By convention we say* $\infty = \widetilde{\infty}$.

In example 2 $(0,1)$ is opposite to $(0,2)$, and $(2\alpha,\alpha)$ is opposite to $(2\alpha,2\alpha)$. The only special point is (2,0). All other points on C are said to be **ordinary**.

3.2 The field of rational functions over a hyperelliptic curve

In algebraic geometry, a common means of studying a curve is to look at the rational functions defined on it. We choose this approach in building the jacobian of a hyperelliptic curve. In what follows C is a hyperelliptic curve over $\mathbb{F}_q$ as stated in definition 3.1.

Definition 5 *A **rational function defined over** C is defined by*

$$R = F/G : \ \mathbb{C}(\overline{\mathbb{F}_q}) \rightarrow \mathbb{C}(\overline{\mathbb{F}_q}) \tag{9}$$

where F and G are polynomials in $\overline{\mathbb{F}_q[x, y]}$.

Example 3 *Continuing with example 3.1 we define the rational function $R : \ C(\overline{\mathbb{F}_3}) \rightarrow \mathbb{C}(\overline{\mathbb{F}_3})$ by*

$$R(x, y) = \frac{F(x, y)}{G(x, y)} = \frac{x - \alpha - 2}{y - 2}. \tag{10}$$

R can easily be evaluated at any finite point in the domain. For instance, $R(2\alpha, 2\alpha) = (2\alpha - \alpha - 2)/(\alpha - 2) = 1$. The only issue is what happens at the point at infinity $\mathbb{O}$. We define $R(\mathbb{O})$ in analogy with how it is defined in projective geometry. Basically, if the order of infinity in the denominator is greater than that in the numerator, then $R(\mathbb{O}) = 0$. If the opposite holds, then $R(\mathbb{O})$ is undefined. In the final case when the orders of infinity are equal we define $R(\mathbb{O})$ to be the ratio of leading coefficients. In our case $R(\mathbb{O}) = 1$.

The reader may observe that many of the rational functions we are at liberty to define over C will harbor the factor $y^2 + h(x)y - f(x)$ in either the numerator or denominator. Since it will evaluate to zero or be undefined at any point in the domain there little sense of having it there in the first place. Hence, it will suffice to study quotients of remainders of polynomials in $\mathbb{F}_q[x, y]$ upon division by $y^2 + h(x)y - f(x)$. We accomplish this by defining a quotient ring of equivalence classes. The representatives will be precisely the remainders.

Definition 6 *The **coordinate ring** of C over $\overline{\mathbb{F}_q}$ is the quotient ring*

$$\overline{\mathbb{F}_q}[C] = \frac{\overline{\mathbb{F}_q}[x, y]}{< y^2 + h(x)y - f(x) >}. \tag{11}$$

The coordinate ring is an integral domain because $y^2 + h(x)y - f(x)$ is irreducible over $\overline{\mathbb{F}_q}$. If it were not, then the polynomial would

factor as $(y - a(x))(y - b(x))$. Upon expansion we find that $-h(x) = a(x) + b(x)$ and $-f(x) = a(x)b(x)$. Hence $deg(ab) = 2g + 1$ and $deg(a+b) \leq g$, which cannot possibly hold. Therefore we are justified to deem the next structure a field.

Definition 7 *The **rational function field** $\overline{\mathbb{F}_q}(C)$ **of** C **over** $\overline{\mathbb{F}_q}$ is the field of fractions of the coordinate ring $\overline{\mathbb{F}_q}[C]$.*

Example 4 *Here we ground our theory with an example using the familiar curve $C: \ y^2 = f(x) = x^5 + 1$ defined over $\mathbb{F}_3$.*

The coordinate ring of C is $\overline{\mathbb{F}_3}[C] = \frac{\overline{\mathbb{F}_3}[x,y]}{<y^2 - x^5 - 1>}$. The remainders of polynomials in $\overline{\mathbb{F}_q}[x, y]$ upon division by $y^2 - x^5 - 1$ are polynomials of the form $G(x, y) = a(x) - b(x)y$ where $deg(a), deg(b) \leq 4$. Therefore, we may use the unique representation $[G(x, y)] = [a(x) - b(x)y]$ for elements of the ring.

Similarly, equivalence classes in $\overline{\mathbb{F}_3}(C)$, the rational function field of C over $\overline{\mathbb{F}_3}$, can be uniquely represented by quotients of representatives from the coordinate ring.

3.3 Divisors

Definition 8 *A **divisor** D is a finite formal sum of points on $\mathbb{C}$,*

$$D = \sum_{P \in C} m_P P \tag{12}$$

with $m_P \in \mathbb{Z}$.

In this section we show how a rational function $R = F/G$ in $\overline{\mathbb{F}_q}(C)$ gives rise to a divisor by looking at the zeros and poles of F and G.

Definition 9 *A polynomial F in $\overline{\mathbb{F}_q}[C]$ has a **zero** at $P \in C(\overline{\mathbb{F}_q})$ when $F(P) = 0$ and a **pole** at P when $F(P) = \mathbb{O}$, the point at infinity on C.*

According to the rules stated for evaluating polynomials in example 3 the only pole of F is $\mathbb{O}$. For polynomials over $\mathbb{R}$ the number of zeros and poles match when we take multiplicities into account. For instance, $f: \ \mathbb{R} \to \mathbb{R}$ defined by $x \mapsto (x - 1)^2$ has a double zero at 1 and a double pole at ∞. The same holds for polynomials defined over a curve. We use this fact in associating a divisor to F.

Definition 10 *The **divisor** of F in the coordinate ring $C[\overline{\mathbb{F}_q}]$ is defined by $div(F) = \sum m_i P_i - (\sum m_i)\,\mathbb{O}$ where each P_i is a zero of F with multiplicity m_i.*

To associate a divisor to each element $R = F/G$ in the rational function field $C(\overline{\mathbb{F}_q})$ we simply take the difference of the of the divisors of the polynomials F and G. A divisor which has an associated rational function in $C(\overline{\mathbb{F}_q})$ is called a **principal divisor**.

Example 5 *Let C be as in example 2.*
Recall we defined $R(x,y) = \frac{F(x,y)}{G(x,y)} = \frac{x-\alpha-2}{y-2}$.
Here are some divisors:

1. $div(F) = (2+\alpha, 2+\alpha) + (2+\alpha, 1+2\alpha) - 2\mathbb{O}$;
2. $div(G) = 2(2,0) - 2\mathbb{O}$;
3. $div(R) = div(F) - div(G) = (2+\alpha, 2+\alpha) + (2+\alpha, 1+2\alpha) - 2(2,0)$.

Note that in $div(G)$, (2,0) has coefficient 2 because it is a special point.

3.4 The jacobian of a hyperelliptic curve

So far we have restricted our discussion to individual divisors of a curve C. In this section we focus on structural properties of divisors. The set of all divisors of C, denoted $\mathbb{D}$, forms an abelian group under the addition rule:

$$\sum_{P \in C} m_P P + \sum_{P \in C} n_P P = \sum_{P \in C} (m_P + n_P) P.$$

The **degree** of a divisor is the sum of its coefficients. The set of all divisors of degree 0, denoted $\mathbb{D}^0$, obviously forms a subgroup of $\mathbb{D}$. Furthermore, addition of principal divisors corresponds to multiplication of their corresponding rational functions in $C(\overline{\mathbb{F}_q})$. This provides us with the closure property we need to conclude the set of principal divisors, denoted $\mathbb{P}$, is indeed a subgroup of $\mathbb{D}^0$.

The reader may anticipate what is to follow. Whenever we have a subgroup of objects like principal divisors embedded inside a more robust group it is an open invitation to take the quotient. Here is no exception. Define the **jacobian** of a hyperelliptic curve C as the quotient group $\mathbb{J} = \mathbb{D}^0/\mathbb{P}$. The jacobian is a group of equivalence classes of degree 0 divisors with $\mathbb{P}$ as identity.

We close this section by introducing a representation for elements in $\mathbb{J}$. We simply stream through the basic argument behind the representation. The curious reader is directed to Koblitz [7] for a rigorous demonstration.

Definition 11 *A divisor $D = \sum m_i P_i - (\sum m_i)\,\mathbb{O}$ in $\mathbb{D}^0$ is called a **semi-reduced divisor** when each $m_i > 0$ and if P is an ordinary point in the formal sum, then $\widetilde{P}$ is not in the sum.*

Definition 12 *A semi-reduced divisor* $D = \sum m_i P_i - (\sum m_i)\,\mathbb{O}$ *is called* ***reduced*** *when* $\sum_{P \in \mathbb{C}} m_P \leq g$ *(the genus of* C*).*

Recall that two elements D_1, D_2 of $\mathbb{D}^0$ are equivalent, written $D_1 \sim D_2$, when they lie in the same equivalence class. It takes little effort to show that every $D \in \mathbb{D}^0$ has an equivalent semi-reduced divisor. Then with significantly more work it can be shown, with the aid of the Riemann-Roch Theorem, that of all semi-reduced divisors in an equivalence class exactly one is reduced. We take the reduced divisors as our representatives in $\mathbb{J}$.

Two observations can be made. First, the quotient of an abelian group with a subgroup is itself abelian. Secondly, the finiteness of reduced divisors implies the jacobian is finite. Hence, $\mathbb{J}$ is a finite abelian group.

3.5 Computation in the jacobian

Here we describe a polynomial representation from semi-reduced divisors. It will prove its usefulness in the next section when we give algorithms for adding elements in the jacobian.

A semi-reduced divisor can be represented as a pair of polynomials a and b in $\mathbb{F}_q[x]$.

Proposition 1 *Let* $D = \sum m_i P_i - (\sum m_i)\,\mathbb{O}$ *be a semi-reduced divisor with* $P_i = (x_i, y_i)$*. Let* $a(x) = \prod (x - x_i)^{m_i}$*. There is a unique polynomial* $b(x)$ *such that:*

1. $deg(b) \leq deg(a)$*;*
2. $b(x_i) = y_i$ *for all* i *for which* $m_i \neq 0$*;*
3. $a(x)$ *divides* $b(x)^2 + b(x)h(x) - f(x)$*.*

With

- $div(a(x)) = \sum_{P \in C} a_i P_i - (\sum a_i)\,\mathbb{O}$ *and*
- $div(b(x) - y) = \sum_{P \in C} b_i P_i - (\sum b_i)\,\mathbb{O}$

it follows that $D = div(a, b) = \sum_{P \in C} min(a_i, b_i) P_i - (\sum min(a_i, b_i))\,\mathbb{O}$.

$D = div(a, b)$ is merely notation used to express that we are representing D by the polynomials a and b.

Example 6 *Continuing with our example the divisor* $D = 2(0, 1) + (2, 0) - 3\mathbb{O}$ *has the polynomial representation*

- $a(x) = x^3 - 2x^2$ *and*
- $b(x) = -2x^2 + 1$*.*

3.6 Cantor's addition algorithms

In the final section of this chapter we present Cantor's algorithms for adding reduced divisors in polynomial form. First there is a small point which must be addressed. The algorithms only work in a subgroup of $\mathbb{J}$. We outline this subgroup and then show how to add its elements in quadratic running time.

As usual we take a curve C of genus g defined over $\mathbb{F}_q$ with jacobian $\mathbb{J}$. For a point on the curve $P = (x, y)$ and an automorphism $\sigma : \mathbb{F}_q \to \overline{\mathbb{F}_q}$ we define $\sigma(P) = (\sigma(x), \sigma(y))$. Moreover, we define the automorphism σ for a divisor by $\sigma(D) = \sum m_P \sigma(P)$. A divisor D is **defined** over $\mathbb{F}_q$ when $D = \sigma(D)$ for all automorphisms from $\mathbb{F}_q$ to $\overline{\mathbb{F}_q}$. The set $\mathbb{J}(\mathbb{F}_q)$ of equivalence classes in $\mathbb{J}$ which have representative defined over $\mathbb{F}_q$ forms a subgroup of $\mathbb{J}$.

Adding elements in $\mathbb{J}(\mathbb{F}_q)$ is divided into two parts. The first algorithm takes two semi-reduced divisors D_1, D_2 and returns their sum as a semi-reduced divisor $D \sim D_1 + D_2$. The second algorithm returns a reduced divisor D' which is equivalent to D.

Algorithm 1 ***Addition algorithm***
Input: *$D_1 = div(a_1, b_1)$ and $D_2 = div(a_2, b_2)$ semi-reduced divisors defined over $\mathbb{F}_q$.*
Output: *$D = div(a, b)$ a semi-reduced divisor defined over $\mathbb{Z}_q$ equivalent to $D_1 + D_2$.*

1. *Using the Euclidean algorithm compute polynomials $d_1(x)$, $e_1(x)$, $e_2(x)$ such that $d_1 = gcd(e_1, e_2)$ and $d_1 = e_1 a_1 + e_2 a_2$.*
2. *Using the Euclidean algorithm compute polynomials $d(x)$, $c_1(x)$, $c_2(x)$ such that $d = gcd(d_1, b_1 + b_2 + h)$ and $d = c_1 d_1 + c_2(b_1 + b_2 + h)$.*
3. *Let $s_1 = c_1 e_1$, $s_2 = c_2 e2$, and $s_3 = c_2$, so that*
$$d = s_1 a_1 + s_1 a_2 + s_3(b_1 + b_2 + h).$$
4. *Finally set*
$$a = \frac{a_1 a_2}{d^2}$$
and
$$b = \frac{s_1 a_1 b_2 + s_2 a_2 b_1 + s_3(b_1 b_2 + f)}{d} \ mod\ a.$$

Algorithm 2 ***Reduction algorithm***
Input: *$D = div(a, b)$ a semi-reduced divisor defined over $\mathbb{F}_q$.*
Output: *$D' = div(a', b')$ the reduced divisor equivalent to D.*

1. *Set*

$$a' = \frac{f - bh - b^2)}{a}$$

and

$$b' = -h - b \ mod \ a'.$$

2. *If* $deg(a') > g$, *then assign* $a \leftarrow a'$, $b \leftarrow b'$ *and repeat step 1.*
3. *Let* c *be the leading coefficient of* a', *and assign* $a' \leftarrow c^{-1}a'$.
4. *Return* a', b'.

For a detailed exposition of the correctness of the algorithms with running time analysis the reader is encouraged to read both [7] and [9]. These algorithms are in effect a generalization of Cantor's algorithms due to Koblitz. A simplified version where $h(x) = 0$ and the characteristic of the underlying field is not 2 is given by Cantor in [1].

4 Implementation in Maple

In this chapter we describe an implementation of El Gamal in the jacobian of a hyperelliptic curve. Following an outline of the cryptosystem we describe the implementation in some detail.

4.1 El Gamal over the jacobian of a hyperelliptic curve

The basic requirements necessary to apply El Gamal to the jacobian have been satisfied. Using the polynomial representation discussed in the previous section we have an efficient means of adding elements in $\mathbb{J}(\mathbb{F}_q)$. Furthermore, there is no subexponential algorithm known for calculating discrete logarithms in the jacobian. The discrete logarithm problem and El Gamal carry over almost word for word from $E(\mathbb{F}_q)$ to the jacobian of a hyperelliptic curve.

Let $\mathbb{J}$ be the jacobian of the curve C over $\mathbb{F}_q$ with base $Q = [div(a, b)] \in \mathbb{J}$. Given $P = [div(a', b')] \in \mathbb{J}$, the discrete logarithm problem to the base Q is to find an integer n so that $nQ = P$. (if such an n exists)

El Gamal message transmission works as follows:

1. Alice and Bob agree on the hyperelliptic curve cryptosystem parameters by choosing a curve $C : y^2 + h(x)y = f(x)$ over a finite field $\mathbb{F}_q$ and a base element Q in $\mathbb{J}(\mathbb{F}_q)$.
2. Alice creates keys
   (a) Private key: An integer n_A.
   (b) Public key: The point $n_A Q$.
3. Bob creates keys:
   (a) Private key: An integer n_B.
   (b) Public key: The point $n_B Q$.
4. Alice sends a message P_m in $\mathbb{J}(\mathbb{F}_q)$ to Bob:
   $C_m = P_m +$ (Bob's public key) (Alice's private key) $= P_m + (n_B Q)\,(n_A)$.
5. Bob reads the message:
   $P_m = C_m -$ (Alice's private key) (Bob's public key) $= C_m - (n_A Q)\,(n_B)$.

4.2 Documentation

Here we give an overview of a working model for El Gamal in the jacobian of a hyperelliptic curve in Maple 8. This model is by no means

practical, however, it does provide us with a way of working with small examples. Furthermore, some of the code is of practical use independently of the cryptosystem. The GF package in Maple allows the user to create a table of functions and constants for doing arithmetic in a Galois field. It is very often the case that we must perform calculations in polynomial ring of a finite field. Unfortunately, the GF package has no such capability. To remedy the situation, we extended GF to work many of the basic operations on polynomials with coefficients in a finite field. The methods range from simple spadework such as addition and subtraction in the ring to more complicated operations like the extended Euclidean algorithm.

To outline the project code we begin with a description of some common data types followed by the method hierarchy.

- [*polynom*]: A polynomial in Maple.
- [*zppoly*]: Element of a finite field $\mathbb{Z}_q$.
- [$gfpoly$]: Element of the polynomial ring $\mathbb{Z}_q[x]$. The polynomial is stored as a list of [*zppoly*]'s beginning with the constant term.
- [*div*]: Element of the jacobian of a curve represented by a list of two [$gfpoly$]'s.
- [GF]: A Galois field represented by a table of [*zppoly*]'s.

The method calling sequence for encryption and decryption is for demonstration purposes only. It begins with the *cryptosystem* procedure which globally defines the cryptosystem parameters such as the curve, finite field, and base point. *make-keys* is subsequently called to create public and private keys for both Alice and Bob. Generation of the public keys requires the *exponentiate* procedure to add the base point Q to itself n_A and n_B times respectively. *semi-reduced* and *reduced* are Cantor's algorithms for addition in the jacobian.

Figure 2 shows the method hierarchy of Cantor's algorithms along with the encryption process. Many of the procedure names are self explanatory. For instance, *add-gfpoly* add two polynomials in $\mathbb{Z}_q[x]$. *div-gfpoly* returns the quotient and remained of two polynomials. For a more detailed description of each procedure see the attached documentation.

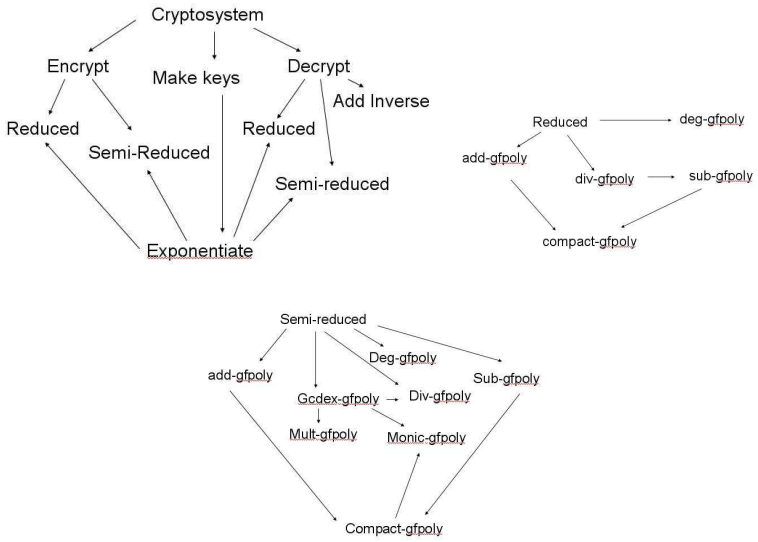

Figure 2:

4.3 Example in Maple

In this section we give an example which illustrates how to encrypt and decrypt a message using the program. Our methods are based on the GF package. Reading through an example will help to familiarize ourselves with the package and also the data types used in the program. The input and output is displayed in Maple format.

We begin by defining the finite field $\mathbb{F}_7$. Using the GF package this is accomplished as follows:

```
> gal:=GF(7,1):
> T:=gal[variable]:
```

Here we could define the necessary parameters via *cryptosystem*. However, it is just as easy to define them directly in this case. We take the curve defined by $f(x) = x^5 + 5x^4 + 6x^2 + x + 3$ and $h(x) = x$.

```
> F:=[gal[input](3),gal[input](1),gal[input](6),gal[input](0),gal[input
> ](5),gal[input](1)]; H:=[gal[input](0),gal[input](1)];
> C:=[F,H]:
```

$$F := [3,\ 1,\ 6,\ 0,\ 5,\ 1]$$

$$H := [0,\ 1]$$

To encrypt a plaintext first define the cryptosystem. This amounts to giving a base element in the jacobian of the curve to work with.

We take the reduced divisor represented by the polynomials
$a(x) = x^2 + x + 5$ and $b(x) = 4x + 4$.

```
> B:=[[gal[input](5),gal[input](1),gal[input](1)],[gal[input](4),gal[in
> put](4)]];
```

$$B := [[5, 1, 1], [4, 4]]$$

To begin the encryption and decryption process make public and private keys for the sender and receiver.

Public and private keys for sender. "make_keys" returns a public and private key.

```
> alice_keys:=make_keys(32):
> alice_public:=alice_keys[1];
> alice_private:=alice_keys[2];
```

$$alice\_public := [[6, 6, 1], [1, 6]]$$

$$alice\_private := [32, [[5, 1, 1], [4, 4]]]$$

Public and private keys for receiver.

```
> bob_keys:=make_keys(5):
> bob_public:=bob_keys[1];
> bob_private:=bob_keys[2];
```

$$bob\_public := [[1, 5, 1], [2, 3]]$$

$$bob\_private := [5, [[5, 1, 1], [4, 4]]]$$

Encryption of a plaintext P works as follows.

```
> P:=[[gal[input](3),gal[input](0),gal[input](1)],[gal[input](4),gal[in
> put](6)]];
```

$$P := [[3, 0, 1], [4, 6]]$$

```
> C:=encrypt(P,alice_private,bob_public);
```

$$C := [[1, 5, 1], [2, 3]]$$

Decryption is similar.

```
> plaintext:=decrypt(C,bob_private,alice_public);
```

$$plaintext := [[3, 0, 1], [4, 6]]$$

5 Pratical considerations

In order to make El Gamal over hyperelliptic curves practical there are two major considerations. The underlying curve selected must not compromise the security of the cryptosystem, and the computations in the underlying finite field should be efficient.

5.1 Known attacks

At an elementary level, determining which curves are safe for cryptography relies on our knowledge of known attacks on the cryptosystem. Most of the attacks discussed below are not particular to hyperelliptic curve cryptography. Indeed, many apply to El Gamal over any group. However, the attack due to Huang and Adleman is especially designed for calculating discrete logarithms in the jacobian.

The **Pohlig-Hellman algorithm** [11] for calculating discrete logarithms, in conjuncton with **Shank's algorithm** [11], is considered to be effective when the order of the jacobian contains only small prime factors, say < 40 digits [7].

Frey-Ruck is an alternate attack which reduces the problem of calculating discrete logarithms in $\mathbb{J}(\mathbb{F}_q)$ to calculating them in some extension field of the group of units of $\mathbb{F}_q$. However, this approach relies on the added structural property that any prime factor l of the group order does not divide $q^k - 1$ for sufficiently small values of k, say $1 \leq k \leq 2000/log_2 q$.

Another algorithm reduces computing discrete logarithms in the jacobian using Weil descent. To be secure from such an attack the underlying field, $\mathbb{F}_q$, should have $q = 2^r$ elements for r a prime. It turns out that this is not a terribly serious restriction. Arithmetic in Galois fields of characteristic 2 appears to be more efficient than in fields of higher characteristic.

Finally, **Huang-Adleman** is a subexponential algorithm which can in theory be used to calculate discrete logarithms in the jacobian of a hyperelliptic curve of genus 4 or higher.

The attacks outlined above place some serious restrictions on the curve and underlying finite field we may select. Basically we are looking for curves C of genus 2 or 3 over finite fields of the form $\mathbb{F}_{2^r}$ for a prime r. Also the order of $\mathbb{J}(\mathbb{F}_{2^r})$ must contain at least one prime factor with over 40 digits. But to avoid Frey-Ruck any prime factor p of $\mid \mathbb{J}(\mathbb{F}_{2^r}) \mid$ cannot divide $p^{2^k} - 1$ for small k.

5.2 Examples

hyperelliptic curve cryptography has been primarily restricted to academia because it is difficult to implement. Curve selection alone poses a major hurdle. In this section we conclude with examples of both a successful and failed attempt at choosing a curve and finite field.

Example 7 *Take* $C: \ y^2 + y = x^5 + x^3 + x$ *over* $\mathbb{F}_{2^{101}}$.
Without going into any details it is sometimes possible to calculate the order of the jacobian of C *using its zeta function. It this case the*

computation yields that the jacobian has order

642775217703596110216784836936718571128926843393416474761625 7

with a 58 digit prime factor

$p = 151276822241373525586440300526410583932437477852063185399 3.$

The genus 2 curve seems promising because the underlying field has characteristic 2 and 101 is prime. Furthermore, the order of the jacobian has a large prime factor. However, we find that p divides $(2^{101})^3 - 1$ *which makes the system vulnerable to Frey-Ruck.*

Example 8 *The family of hyperelliptic curves of the form* $y^2 + y = x^{2g+1}$ *over* $\mathbb{F}_q$ *were first introduced in cryptography by Koblitz [10]. They are useful because many contain large prime factors in the orders of their jacobians. For instance, consider the curve* $C: \ y^2 + y = x^7$ *over* $\mathbb{F}_{2^{47}}$*. The order of* $\mathbb{J}(\mathbb{F}_{2^{47}})$ *contains the 42 digit prime factor*

$$p = 398227592830903984669824190479460780961207.$$

p is large enough so that Pohlig-Hellman algorithm becomes impractical. Also, it can be checked that Frey-Ruck does not apply for this curve. Therefore, this choice for C satisfies the criteria for security.

6 Concluding remarks

Implementation of hyperelliptic curve cryptography remains in the research and development phase. There are no standard protocols as with elliptic curve cryptography. However, if implemented it does promise some advantages over other public key cryptosystems.

Hyperelliptic curves provide us a with richer source of groups for El Gamal than elliptic curves. Remember, finding more groups for El Gamal was a motivating factor for studying elliptic curve cryptography to begin with. Another benefit over elliptic curves lies in the size of the underlying field. Computation should become easier because as we take higher genus curves the underlying field can be smaller with equal security.

Before hyperelliptic curve cryptography is used in practice there are issues which should be addressed in addition to curve selection. Currently, arithmetic in the jacobian is slow in comparison to arithmetic in the groups of points on an elliptic curve. This leads to the open problem of how to speed up Cantor's algorithms. Also, we saw that to defend from various attacks it is critical to know the order of

the jacobian. Investigating point counting algorithms on the jacobian is another active area of research.

Obviously, this paper has nothing new to say about the subject. However, it is hoped this project can be of some value as a resource for those interested in studying hyperelliptic curve cryptography at the undergraduate level.

References

[1] D. Cantor, *Computing in the jacobian of a hyperelliptic curve*, Math. Comp. 48 (1987).

[2] T. El Gamal, *A public key cryptosystem and a signature scheme based on discrete logarithms*, IEEE Transactions on Information Theory, 31 (1985).

[3] A. Enge, *Hyperelliptic cryptosystems efficiency and subexponenial attacks*, Herstellung: Books on Demand, 2000.

[4] K. Dilcher, *Cryptography an introductory course*, unpublished lecture notes, 2001.

[5] D. Dummit, R. Foote, *Abstract algebra*, Prentice-Hall, 1991.

[6] M.D. Huang, *CS559 Curve based cryptography*, lecture notes, 2002.

[7] N. Koblitz, *Algebraic aspects of cryptography*, Springer, 1999.

[8] N. Koblitz, *Elliptic curve cryptosystems*, Math. Comp. 48 (1987), 203-209.

[9] N. Koblitz, *Hyperelliptic cryptosystems*, Journal of Cryptology. 1:139-150, 1989.

[10] N.Koblitz, *Jacobi sums, irreducible zeta-polynomials, and cryptography*, Canadian Math. Bull. 34 (1991), 229-235.

[11] D. Stinson, *Cryptography theory and practice*, CRC Press, 1995.

[12] W. Trappe, L. Washington. *Introduction to cryptography with codeing theory*, Prentice-Hall, 2002.

[13] L. Wagner, *Algebro-geometric attack methods in elliptic curve cryptography*, unpublished thesis, 2002.

[14] A. Weng, *Arithemetic on hyperelliptic curves*, unpublished thesis, 2001.

[15] A. Weng, *Constructing hyperelliptic curves of genus 2 suitable for cryptography*, Math. Comp. 72 (2003).