Chapter 8
Elliptic and Hyperelliptic Curve Cryptography

Nigel Boston and Matthew Darnall

8.1 Introduction

Suppose two parties, Alice (A) and Bob (B), want to send messages between themselves without an eavesdropper Eve (E) reading the messages. Private-key (symmetric) cryptography relies on establishing a known secret between A and B before they can communicate. The term symmetric describes the fact that the information known to A and B is the same, namely the private key. We have seen an example of a private-key system, advanced eneryption standard (AES), in chapter 1. What if, as often happens in practice, it is infeasible for A and B to have a prearranged secret? In the development of cryptography it became apparent that a mechanism for A and B to agree upon a private key over an insecure channel would be important.

The area of cryptography devoted to the ways Alice and Bob can share information without a prearranged secret is called public-key (or asymmetric) cryptography. The term public key refers to the fact that in all current systems, some public piece of information is needed for the encryption to occur. Examples of this public information are the modulus in the famous RSA algorithm [46] or the group generator raised to a power for Diffie – Hellman, described later. The term asymmetric means that the private information known to A and B is different, i.e., A and B each start with information the other does not know. Public-key cryptography was introduced to the world at large in the seminal paper [10] of Whitfield Diffie and Martin Hellman in 1978, although shortly before then these ideas were known to the researchers at the British intelligence agency GCHQ*. Many other methods of public-key cryptography have since been introduced and current research is still searching for better protocols for the exchange of private keys.

University of Wisconsin,
e-mail: boston,darnall@math.wisc.edu

*(UK) Government Communications Headquarters

Ç.K. Koç (ed.) *Cryptographic Engineering*, DOI 10.1007/978-0-387-71817-0_8,

8.2 Diffie – Hellman Key Exchange

Though it was the first method of public-key cryptography known, the Diffie – Hellman key exchange protocol is still used widely and makes up the basis for both elliptic and hyperelliptic curve cryptography. Let G be a finite cyclic group of order n with generator g. The discrete logarithm problem (DLP) for the group G is to determine a given g^a, where a is a positive integer less than n. The simple idea observed by these researchers is that, if there is a group where performing the group operation is computationally easy, but solving the DLP is hard, then a secret between two parties can be shared. A and B come up with private keys k_A and k_B respectively – these are positive integers less than n. A publishes g^{k_A} and B publishes g^{k_B}. The shared secret between A and B is $g^{k_A k_B}$, which A computes as $(g^{k_B})^{k_A}$ and B computes as $(g^{k_A})^{k_B}$. Since Eve apparently cannot get k_A or k_B without solving a DLP, obtaining the shared secret is hard. It is widely believed that solving the DLP is equivalent to determining $g^{k_A k_B}$ given g, g^{k_A}, g^{k_B}, although this is not known.

The groups originally considered for Diffie–Hellman Key Exchange were large cyclic subgroups of multiplicative groups of finite fields. As seen in Chapter 7, the multiplication in finite fields can be efficiently computed. Unfortunately, the DLP in this case can be solved in time subexponential in the size of the group using an index calculus attack. A result of Victor Shoup [52] says that for an arbitrary group of order n, computing a discrete log will take $\sqrt{p}$ group operations, where p is the largest prime divisor of n. This result assumes that no structure of the group can be taken advantage of, so groups attackable only in exponential time should exist. Currently, the most suitable groups that provide quick encryption and for which only exponential time attacks are known, come from elliptic and hyperelliptic curves.

8.3 Introduction to Elliptic and Hyperelliptic Curves

Elliptic curves have been studied by mathematicians for centuries. Neal Koblitz and Victor Miller independently discovered that their rich structure makes them useful for a wide range of cryptographic applications [26], [38]. This structure can also lead to several attacks, so care must be taken by any would-be cryptographer.

Let $\mathbf{k}$ be a field. An *elliptic curve* E, over $\mathbf{k}$, is a non-singular projective curve of genus 1. For an arbitrary field $\mathbf{k}$, E can be thought of as the set of points $(X,Y) \in \mathbf{k}^2$ that satisfy an equation of the form:

$$Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6$$

where the coeffients a_i are in $\mathbf{k}$, together with a "point at infinity," P_∞. We must also assume that the curve defined by the equation is non-singular, which is equivalent to having the partial derivatives of the equation never vanish simultaneously. If the characteristic of $\mathbf{k}$ is not 2 or 3, we can perform a change of variables to get an equation to the form

$$Y^2 = X^3 + aX + b$$

Here the non-singular condition just says that the cubic $X^3 + aX + b$ must have distinct roots.

A hyperelliptic curve is a special type of non-singular, projective curve. For our purposes, a *hyperelliptic curve*, of genus $g \geq 1$ over $\mathbf{k}$ is the set of points $(X,Y) \in \mathbf{k}^2$ that satisfy

$$y^2 + h(X)Y = f(X)$$

where h and f are polynomials in $\mathbf{k}[X]$ with $deg(f) = 2g+1$, $deg(h) \leq g$, together with a point "at infinity", P_∞. An elliptic curve is just a hyperelliptic curve of genus 1.

To understand why there is a point at infinity, notice that the definition of elliptic or hyperelliptic curves includes the word 'projective'. We consider the curves as living in projective space, say with coordinates $(X : Y : Z)$. The point at infinity is the unique point where the projective curve defined by homogenizing our equation intersects the line $Z = 0$. If the reader has no background in projective geometry, simply think of the point P_∞ as a point infinitely far up the Y-axis that 'compactifies' the curve.

8.4 The Jacobian of a Curve

A priori, a hyperelliptic curve over a field $\mathbf{k}$, is a set of points in $\mathbf{k}^2$ with a special point at infinity. A beautiful fact noticed a long time ago by algebraic geometers is that a group can be attached to each curve. These groups are "made up" of collections of points on the curve and the group law can be performed using only operations in the field $\mathbf{k}$. For an elliptic curve, the group consists of the points on the elliptic curve, and the group law can be viewed geometrically. For hyperelliptic curves, the group consists of g-tuples of points on the curve. For both kinds of curves, the DLP is in general very hard to solve.

The group associated to a hyperelliptic curve, C, is a quotient of the larger group called the *degree zero divisor group* of C, denoted $Div^0(C)$. This group is made up of elements, D, called divisors, of the form:

$$D = \sum_{P \in C} n_P P$$

where:

1. The formal sum is over points $P = (x_P, y_P)$ on C with coordinates x_P, y_P in $\bar{\mathbf{k}}$, an algebraic closure of $\mathbf{k}$. Here P_∞ is included as a point on C.
2. n_P is an integer for each P, with all but finitely many $n_P = 0$.
3. If there is an element σ of the Galois group of $\bar{\mathbf{k}}$ over $\mathbf{k}$ such that $\sigma P := (\sigma(x_P), \sigma(y_P)) = (x_Q, y_Q) = Q$, then $n_Q = n_P$.
4. $deg(D) = \sum_P n_P = 0$.

For readers unfamiliar with Galois groups, an element σ, of the Galois group of $\bar{\mathbf{k}}$ over $\mathbf{k}$ is an automorphism of $\bar{\mathbf{k}}$ that fixes $\mathbf{k}$. Thus, for any $a,b \in \bar{\mathbf{k}}$, we have $\sigma(a+b) = \sigma(a) + \sigma(b)$ and $\sigma(ab) = \sigma(a)\sigma(b)$. We also have that $\sigma k = k$ for any $k \in \mathbf{k}$.

We add two divisors by adding the coefficients corresponding to each point:

$$\sum_P m_P P + \sum_P n_P P = \sum_P (m_P + n_P) P$$

Notice that the new divisor still satisfies the conditions above.

8.4.1 The Principal Subgroup and $Jac(C)$

Let $F(X,Y) = Y^2 + h(X)Y - f(X)$ be the polynomial defining the curve C. Let $p(X,Y) \in \mathbf{k}[X,Y]$ be a polynomial in X and Y with coefficients in $\mathbf{k}$ that is not divisible by F. We shall get a divisor, $div(p) \in Div^0(C)$ from this polynomial. For every point $P = (x_P, y_P)$, we define $ord_p(P)$ to be the order at which p vanishes at P. This order has a rigorous definition that is beyond the scope of the book. Loosely speaking, at each point P, we can define something analogous to the Taylor expansion of the function p on C. The order, $ord_P(p)$, at which p vanishes at P is then the smallest exponent in the 'Taylor expansion' with a nonzero coefficient. Thus, $ord_P(p) = 0$ if and only if $p(x_P, y_P) \neq 0$. The order of a function can also be computed at P_∞, and in fact $ord_{P_\infty}(p)$ is the unique integer such that $\sum_{P \in C} ord_P(p) = 0$.

Definition 8.1. The divisor $div(p)$ associated to a polynomial $p(X,Y)$, p not divisible by F, is:

$$\sum_P ord_p(P) P$$

This divisor satisfies the conditions above to be an element of $Div^0(C)$.

We call a divisor principal if it can be written as $div(p) - div(q)$ for two polynomials p,q as above. The set of principal divisors forms a subgroup of $Div^0(C)$, denoted $Prin(C)$.

Definition 8.2. The quotient group $Div^0(C)/Prin(C)$, is called the *Jacobian* of C and is denoted $Jac(C)$.

If $\mathbf{k}$ is a finite field, as it will be for our purposes, then $Jac(C)$ is finite. It is this group that we use for our Diffie – Hellman key exchange.

8.5 Computing on $Jac(C)$

By the celebrated theorem of Riemann-Roch, which is beyond the scope of this book, we know a lot about the structure of $Jac(C)$. Namely, if C has genus g, every element of $Div^0(C)$ is equivalent in $Jac(C)$ to exaclty one divisor of the form:

$$\sum_{i=1}^{m} P_i - mP_\infty$$

where $P_i = (x_i, y_i)$, $m \leq g$, and $P_i \neq (x_j, -y_j - h(x_j)) = -P_j$ for $i \neq j$. When $m = 0$, we get the identity element of the group, the divisor with all coefficients equal to zero. Divisors of the above form are called *reduced* divisors. This gives us a method for representing points in $Jac(C)$ as unordered g-tuples of points on the curve over $\bar{\mathbf{k}}$. The condition that $n_P = n_Q$ for Galois conjugates P and Q ensures that every point occurring in a reduced divisor of $Jac(C)$ has coordinates in at most a degree g extension of $\mathbf{k}$. Thus, when $g = 1$ and we have an elliptic curve E, then the group $Jac(E)$ consists of the points on the curve E over $\mathbf{k}$.

It is easy to see that, if we add two reduced divisors D_1 and D_2 in the fashion described above, we are not guaranteed that the new divisor $D_1 + D_2$ will be reduced. We need a method for finding the unique reduced divisor corresponding to $D_1 + D_2$. To do this, we use a different representation for a divisor than the one given above. As before, let $Y^2 + h(X)Y = f(X)$ be the defining equation for C.

Definition 8.3. The Mumford representation of a reduced divisor $\sum_{i=1}^{m} P_i - mP_\infty$, $P_i = (x_i, y_i)$ is the unique pair of polynomials $[u(x), v(x)]$ in $\mathbf{k}[x]$ that satisfy the following:

1. $u(x) = \prod_{i=1}^{m}(x - x_i)$.
2. $deg(v) < deg(u) = m$.
3. $v(x_i) = y_i$.
4. $u(x)$ divides $v(x)^2 + h(x)v(x) - f(x)$.

The fact that the Mumford representation is unique follows from the fact that the m coefficients defining v can be determined uniquely by conditions 3 and 4. When $m = 0$, we have the identity element, and the Mumford representation is $[1, 0]$. Also every pair of polynomials $[u(x), v(x)]$ in $\mathbf{k}[x]$ satisfying

1. $deg(u) \leq g$ and $deg(v) < deg(u)$,
2. $u(x)$ divides $v(x)^2 + h(x)v(x) - f(x)$,
3. $u(x)$ is monic,

corresponds to a unique reduced divisor D. If $u(x) = \prod_{i=1}^{m}(x - x_i)$, then the points that make up D are $P_i = (x_i, v(x_i))$. Condition 2 will guarantee that that these points are on the curve C.

The benefit of using the Mumford represention for reduced divisors is the following algorithm for computing the sum of two reduced divisors. The algorithm is originally due to Cantor [5] for $h(X) = 0$, and in its most general form it is due to Koblitz [27]. The proof of the correctness of the algorithm, which we do not cover, can be found in [36].

Algorithm

Input: Two reduced divisors $D_1 = [u_1, v_1]$ and $D_2 = [u_2, v_2]$ given in Mumford representation.

Output: A reduced divisor $D = [u, v]$ in Mumford representation that satisfies $D = D_1 + D_2$ in $Jac(C)$.

1. $d_1 \leftarrow gcd(u_1, u_2) = e_1u_1 + e_2u_2$
2. $d \leftarrow gcd(d_1, v_1 + v_2 + h) = c_1d_1 + c_2(v_1 + v_2 + h)$
3. $s_1 \leftarrow c_1e_1$, $s_2 \leftarrow c_1e_2$, $s_3 \leftarrow c_2$
4. $u \leftarrow (u_1u_2)/d^2$
5. $v \leftarrow (s_1u_1v_2 + s_2u_2v_1 + s_3(v_1v_2 + f))/d \bmod u$
6. **DO**
7. $u' \leftarrow (f - vf - v^2)/u$
8. $v' \leftarrow (-h - v) \bmod u'$
9. $u \leftarrow u'$ and $v \leftarrow v'$
10. **WHILE** $deg(u) \leq g$
11. make u monic by dividing by the leading coefficient
12. return $[u, v]$

A key fact to notice is that all the operations involved in adding the two reduced divisors can be reduced to multiplication, division and addition of polynomials in $\mathbf{k}[x]$. Thus, using the techniques for finite field arithmetic given in the previous chapters, we can perform the group operations on $Jac(C)$ quickly. Cantor's algorithm given above is completely general; it works for any hyperelliptic curve over any field. In a practical implementation, properties of the curve and field are used to speed up the algorithm.

8.6 Group Law for Elliptic Curves

Computations in the Jacobian of an arbitrary hyperelliptic curve can be complicated. In this section, we give a simple geometric interpretation of the group law for elliptic curves. We give explicit algorithms for adding two points on an elliptic curve. The following chapter covers the various speedups and optimizations in more detail. The reader interested in implementation specifics should consult that chapter.

Let E be an elliptic curve over a field $\mathbf{k}$. We assume that the field $\mathbf{k}$ has characteristic not equal to 2 or 3, so that we can make a change of coordinates to make the defining equation for E of the form

$$Y^2 = X^3 + aX + b$$

with $a, b \in \mathbf{k}$. Elliptic curves over characteristic two fields are important for cryptographic uses, but for the geometric description of the group law it is easier to assume $char(\mathbf{k}) > 3$.

The map that sends a point P to the reduced divisor $P - P_\infty$ is a bijection between the points on E and $Jac(E)$, the Jacobian of E. We use this bijection and the definition of $Jac(E)$ to give a geometric meaning to the sum of two points. Recall that two divisors D_1 and D_2 are equivalent if $D_1 = D_2 + div(f) - div(g)$, where f and g are polynomials in $\mathbf{k}[X, Y]$ not divisible by $Y^2 - X^3 - aX - b$.

For two constants m, $c \in \mathbf{k}$, consider the line $Y = mX + c$ in the same plane as the elliptic curve E. By Bezout's theorem, we know that the line intersects E in

exactly three points, if we count the points with appropriate multiplicity. So the function $f(X,Y) = Y - mX - c$ has three (not necessarily unique) points on E where it vanishes. We can then write the divisor of f as

$$div(f) = P_1 + P_2 + P_3 - 3P_\infty$$

where P_1, P_2 and P_3 are the three (not necessarily unique) points where the line intersects E. The three points can be non-unique in only certain examples, such as when the line lies tangential to the curve E. In this case, the line intersects the curve E in only two actual points, though the tangential point has multiplicity 2. In analogy with the Taylor series, this is because at the tangential point, f and $Y^2 - X^3 - aX - b$ not only have the same value, they also have the same first derivative.

Since $P_1 + P_2 + P_3 - 3P_\infty = (P_1 - P_\infty) + (P_2 - P_\infty) + (P_3 - P_\infty)$ is the divisor of a function, it represents the identity in the group $Jac(E)$. Using the given bijection with E, we see that

Lemma 8.1. *The sum of two points P_1 and P_2 on E is equal to $-P_3$, where P_3 is the unique other point on E that intersects the line through P_1 and P_2.*

The question remains of what $-P_3$ means, i.e., what is the inverse of a point on E? We first notice that our bijection with $Jac(E)$ sends P_∞ to the divisor $P_\infty - P_\infty$, which is the identity of $Jac(E)$. Thus, P_∞ is the identity of E. Now, let $P = (x_P, y_P)$ be a point on E. Consider the function $X - x_P$. This function intersects the curve E at the points $P = (x_P, y_P)$, $Q = (x_P, -y_P)$, and P_∞. The first two interesect points are easy to see, the last one follows from looking at the projectivization of the curve and the line. Thus, in $Jac(E)$, the divisor $P + Q - 2P_\infty$ equals the identity, so $Q = -P$ on E.

The work above gives us the following algorithm for adding two points, P_1 and P_2. We simply take the unique line through P_1 and P_2, find the unique other point, (x,y), that is on the elliptic curve and the line, and return $P = (x, -y)$. Remember that if $P_1 = P_2$, the line through the point should have the same slope as the defining equation for E.

Algorithm

Input: Two points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ on the curve E defined by $Y^2 = X^3 + aX + b$.

Output: The point $P = (x,y) = P_1 + P_2$.

1. IF P_1 or $P_2 = P_\infty$ THEN $P \leftarrow P_2$ or P_1.
2. IF P_1 = P_2
3. IF $y_1 = 0$ THEN $P \leftarrow P_\infty$
4. $\lambda \leftarrow (3x_1^2 + a)/2y_1$
5. $x \leftarrow \lambda^2 - 2x_1$
6. $y \leftarrow \lambda(x_1 - x_3) - y_1$
7. IF $P_1 \neq P_2$
8. $\lambda \leftarrow (y_1 - y_2)/(x_2 - x_1)$
9. $x \leftarrow \lambda^2 - x_1 - x_2$
10. $y \leftarrow \lambda(x_1 - x_3) - y_1$

8.7 Techniques for Computations in Hyperelliptic Curves

Optimization of hyperelliptic curve arithmetic is a current area of research with several papers appearing each year in the top cryptography conferences. This section gives a brief survey of the techniques and ideas behind them. The reader interested in implementing one of the methods should consult the references given. As the genus gets larger, the computational costs rise significantly. This computational cost, as well as the existence of subexponential time index calculus attacks on high genus hyperelliptic curves, makes low genus curves the most practical for cryptography.

8.7.1 Explicit Formulae

As we showed with elliptic curves, the group law can be implemented using only additions, multiplications and inversions in the base field, **k**. To improve the runtime of Cantor's algorithm, it helps one to have exact formulae for the computations in Cantor's algorithm, i.e., a description of the algorithm in terms of only additions, multiplications and inversions in **k**. Explicit formulae have been completed for genus 2, 3 and 4 hyperelliptic curves. In genus 2 , the first work was done by [57], though improved methods have been found by Harley [22], Lange [29], Matsuo, Chao and Tsujii [34], Takahashi [58], and many others. In genus 3, Pelzl et al. [41] generalized work by Kuroki et al. [28] to give the first explicit formulae that work in all positive characteristic. The paper [41] also gives improvements on implementations of genus 2 hyperelliptic curves. A full description of genus 3 formulae can be found in Wollinger's PhD thesis [62]. For genus 4, Pelzl, Wollinger, and Paar gave the first explicit formulae in [42]. Their computations show that genus 4 arithmetic can compete with lower genus curves as far as computation costs are concerned.

8.7.2 Projective Coordinates

The operation of inverting elements in a finite field is much more costly than addition or multiplication. This has initiated research into finding ways to trade inversions for extra multiplications, additions, and storage in elliptic and hyperelliptic curve cryptography. If we wanted to compute nP for some element $P \in Jac(C)$ and $n \in \mathbf{Z}$ using the standard double-and-add method, we would be forced to use $O(\log n)$ inversions. By introducing another variable, Z, it is possible to delay performing inversions until the last step of the algorithm. For elliptic curves, this extra coordinate Z is equivalent to storing the point in projective coordinates. For higher genus this is no longer the case, but we still call these coordinates projective, due to the similarity with elliptic curves. Projective coordinates for elliptic curves will be covered in the following chapter of this book. Miyamoto et al. [39] first described

an algorithm for projective coordinates on genus 2 curves. This work has been improved by Lange [31] and others. Projective coordinates have also been described by Fan, Wollinger, and Wang [13] for genus 3 curves.

8.7.3 Other Optimization Techniques

Due to the rich structure of hyperelliptic curves, many other techniques exist for performing the group operation. Lange [30], has expanded upon projective coordinates for genus 2 curves in characteristic 2 to give better results. This work introduces several new variables to save on inversions, combining them to also save on other costs. Using special curves can also give remarkable speedups. Certain elliptic curves can be transferred to the Montgomery form, which aids considerably in computations. Gaudry has given a similar form for certain hyperelliptic curves in [19]. In Ref. [2] Bernstein and Lange showed that genus 2 hyperelliptic curves in Gaudry form can even outperform elliptic curves. The fact that hyperelliptic curves need smaller fields to obtain cryptographically secure group sizes is what potentially gives hyperelliptic curves the edge over elliptic curves. Koblitz curves provide another useful way of saving on implementation costs. A nice description of this theory for hyperelliptic curves can be found in [32], while the original idea can be found in Koblitz's work [27]. Additional methods for implementing the hyperelliptic curves will continue to be developed as cryptography in smaller, resource-constrained environments becomes necessary.

8.8 Counting Points on $Jac(C)$

In this section, we introduce methods for counting how many points there are on a given (hyper)elliptic curve over $\mathbf{k} = \mathbf{F}_q$. We begin with Schoof's method [50] for elliptic curves, which reduces the time taken from $O(q^{1/4+\varepsilon})$ to $O(\log^8(q))$. Refinements of this due to Elkies and Atkin [11], [1] reduce this further to $O(\log^6(q))$.

Let E be an elliptic curve defined over $\mathbf{F}_q$. By Hasse's theorem [53], the number of points n on it is $q+1-t$, where $|t| \leq 2\sqrt{q}$. Define ℓ_0 to be the smallest prime such that

$$\prod \ell > 4\sqrt{q}$$

where the product is over primes $\leq \ell_0$. Schoof's idea is to find $t \pmod{\ell}$ for all these primes, in which case the Chinese remainder theorem determines $t \pmod{\prod \ell}$ and so t (and hence n) exactly, since $\prod \ell$ is larger than the range t is confined to. By the Prime Number theorem, $\ell_0 = O(\log(q))$.

If $\ell = 2$, $n \equiv 1 \pmod{2}$ if and only if E has no point of order 2, which is easy to check. For example, if the curve has equation $Y^2 = f(X)$, then the points of order 2 correspond to roots of f, so $n \equiv 1 \pmod{2}$ if and only if f is irreducible over $\mathbf{F}_q$,

which happens if and only if $\gcd(f(X), X^q - X) = 1$. Likewise, for each ℓ, there is a polynomial $f_\ell(X)$ whose roots are the X-coordinates of points of order ℓ.

Suppose $\ell > 2$. We consider the Frobenius mapping ϕ on the points of E over $\bar{\mathbf{k}}$ defined by $\phi(x,y) = (x^q, y^q)$ and sending P_∞ to itself. The proof of Hasse's theorem establishes that

$$\phi^2(P) - t\phi(P) + qP = P_\infty$$

Let $q_\ell = q \pmod{\ell}$ and $t_\ell = t \pmod{\ell}$. For each $\tau \in \{0, 1, ..., \ell - 1\}$, we compute the x-coordinates of both $(x^{q^2}, y^{q^2}) + q_\ell$ and $\tau(x^q, y^q)$. Thanks to f_ℓ, these are both rational functions of x and y. Clearing denominators and using the equation of the curve to eliminate any nonlinear powers of y yields an equation of the form $a(x) - yb(x) = 0$. Substituting this into the curve equation produces a polynomial equation h just in x. Since we are seeking a point of order ℓ, all these calculations, so in particular h, can be taken $\pmod{f}_\ell$, which has degree $O(\ell^2)$.

To check if h has a solution that is a point of order ℓ, $\gcd(h, f_\ell)$ is computed. Only if it is nontrivial do we get a viable value of τ, i.e., $\tau = \pm t_\ell$. Either sign is possible since the x-coordinates are the same. A similar analysis of the y-coordinates determines which. This also means that τ only need run as far as $(\ell - 1)/2$. Most of the work is in computing $x^q, y^q, x^{q^2}, y^{q^2} \pmod{f_\ell}$ and, since f_ℓ is $O(\ell^2)$ and ℓ is $O(\log(q))$, the complexity is polynomial in $\log(q)$, namely $O(\log^8(q))$.

When $t^2 - 4q$ is a square $\pmod{\ell}$, the Frobenius map has an eigenvalue in $\mathbf{F}_\ell$, in which case a factor of degree $(\ell - 1)/2$ of f_ℓ can be used, as noted by Elkies. Atkin found a similar method in the case that $t^2 - 4q$ is not a square $\pmod{\ell}$, and together these yield the SEA (Schoof–Elkies–Atkin) algorithm with complexity $O(\log^6(q))$.

Now, let C be a hyperelliptic curve of genus g defined over $\mathbf{F}_q$. The theory of zeta functions tells us that there exist g complex numbers $\alpha_1, ..., \alpha_g$ with absolute value $\sqrt{q}$ such that, if N_r is the number of points on $C(\mathbf{F}_{q^r}$, then

$$N_r = q^r + 1 - \alpha_1^r - \overline{\alpha_1}^r - ... - \alpha_g^r - \overline{\alpha_g}^r$$

Note that when $g = 1$ and $t = \alpha_1 + \overline{\alpha_1}$, we get Hasse's inequality.

In general, $N_1, ..., N_g$ determine $\alpha_1, ..., \alpha_g$ (and so N_r for all r). They also determine the order of the group $Jac(C)$, which turns out to be $\prod_1^g (1 - \alpha_i)(1 - \overline{\alpha_i}) = \prod_1^g (1 + q - \alpha_i - \overline{\alpha_i})$.

Over finite fields with small characteristic, Satoh's p-adic approach [48] is asymptotically faster than the SEA algorithm. It was extended to Characteristic 2 by Skjernaa [55] and Fouquet, Gaudry, and Harley [14] with a memory-efficient version introduced by Vercauteren [61]. Mestre's AGM (arithmetic-geometric mean) method [37] gave the same asymptotic behavior but with a better constant, while Satoh, Skjernaa, and Taguchi [49] gave a quicker method if one allows precomputations.

As regards higher genus curves, Pila [43] gave an impractical generalization of Schoof's algorithm. Satoh's approach does not work well since the Serre–Tate canonical lift of the Jacobian need not be a Jacobian. Mestre's AGM method is only practical in genus ≤ 2.

This led to the introduction of new techniques. Kedlaya [25] used Washnitzer–Monsky cohomology to count points in small, odd characteristic in time

$$O(g^{4+\varepsilon}\log^{3+\varepsilon}(q)) .$$

This was extended to characteristic 2 by Denef and Vercauteren [7] and Vercauteren [60]. Using Dwork cohomology, Lauder and Wan [33] produced a practical method to count points on Artin–Schreier curves. Both these approaches take $(g^{5+\varepsilon}\log^{3+\varepsilon}(q))$ time.

8.9 Attacks

Having introduced elliptic and hyperelliptic curve cryptography, we now consider potential vulnerabilities of these systems. Over time, researchers have discovered several possible attacks on ECC and HCC that someone looking to implement these systems should be aware of. Avoiding them informs our choice of suitable (hyper)elliptic curve. In general, Shanks' baby-step giant-step method and Pollard's methods (see Sections 9.1 and 9.2) improve on sheer brute force attack by exploiting an idea called the birthday attack to solve discrete logarithm problems in any abelian group. These take on the order of $\sqrt{n}$ operations, where n is the size of the group (so about $q^{g/2}$ in the case of a curve of genus g over $\mathbf{F}_q$). Certain (hyper)elliptic curves are vulnerable to other methods, described later in this section, and any user of ECC or HCC should avoid this choice of curve.

8.9.1 Baby-Step Giant-Step Attack

Shanks' baby-step giant-step algorithm [3] works for any abelian group G. Let us assume G has prime order n (as is recommended by the results in the next subsection) and that we wish to solve the discrete logarithm problem $Q = mP$ for m. Write $m = a\lceil\sqrt{n}\rceil + b$ with $0 \le a,b < \lceil\sqrt{n}\rceil$. Then $Q - bP = a\lceil\sqrt{n}\rceil P$. We make a table of baby steps $Q - bP, b = 0,1,...,\lceil\sqrt{n}\rceil - 1$, and we then start computing giant steps $a\lceil\sqrt{n}\rceil P, a = 0,1,...,\lceil\sqrt{n}\rceil - 1$. Once we find a point that also occurred in one of our baby-step tables, we have found a,b and so have solved the DLP.

It should be noted that an exact value for n is not needed, just an upper bound – in fact the method can be adapted to yield n assuming G is cyclic [12]. A drawback of the baby-step giant-step method is the large amount of memory required if n is large (about $\sqrt{n}$ entries of length $\log n$).

8.9.2 Pollard Rho and Lambda Attacks

Pollard's rho method [45] avoids this by employing a single random walk that eventually self-intersects solving the problem (and that looks like the Greek letter

rho). Pollard's lambda method (or tame kangaroo/wild kangaroo method) uses two random walks, the goal again being to find a collision, but this time with the tame kangaroo laying traps for the wild kangaroo (so that they produce the Greek letter lambda). Various authors [4], [59] have experimented with the parameters of this method. In particular, van Oorschot and Wiener [40] provide a significant speedup by using several kangaroos in parallel.

The main idea is as follows. Let P, Q be points in G, an abelian group of order n, such that $Q = mP$. Let $f : G \to \{1, ..., s\}$ be a function equidistributed in the sense that

$$\sum_{i=1}^{s} || \{g \in G : f(g) = i\} \mid -n/s \mid = O(\sqrt{n})$$

Given a starting point $g_0 \in G$, we define a random walk $g_k = F(g_{k-1})$, where $F(g) = g + M_{f(g)}$. Here $M_i = a_i P + b_i Q$ for $i = 1, ..., s$ is a set of multipliers. Teske [59] found that s approximately 20 worked best. You set off two kangaroos performing these jumps.

Applying this to two kangaroos, you reach a collision so that $x_{i1}P + x_{i2}Q = y_{j1}P + y_{j2}Q$, so that $(x_{i1} - y_{j1})P = (y_{j2} - x_{i2})Q = (y_{j2} - xi2)mP$. Since $\gcd(x_{i1} - y_{j1}, n) = 1$, this can then be solved for m.

8.9.3 Pohlig–Hellman Attack

The Pohlig–Hellman algorithm [44] reduces the discrete logarithm problem in any abelian group to a subgroup of prime order. Thus, the order of the group we choose (the number of points on the elliptic curve or Jacobian of the hyperelliptic curve) should have a very large prime divisor and we will take our generating point P to be of that order.

To see this, suppose that the order of the group $n = \prod_{i=1}^{r} p_i^{k_i}$ and that we wish to solve the problem $Q = mP$ for m. Letting $n' = n/p_1^{k_1}$ and $m_1 = m \pmod{p_1}$, we can solve $Q' = n'Q = m_1 P'$ where $P' = n'P$ is a point of prime order p_1 to get m_1. Then $m_i = \pmod{p_1^i}, i = 2, 3, ...$ are successively computed as follows. Say m_i is known and $m = m_i + cp_1^i$. Then we know $Q - m_i P = cp_1^i P = cR$ and R, which has order $n_i = n/p_1^i$. So $c \pmod{p_1}$ is found and we have m_{i+1}. Once $m \pmod{p_i^{k_i}}$ is known for all i, the Chinese remainder theorem determines m.

8.9.4 Menezes–Okamoto–Vanstone Attack

The Menezes–Okamato–Vanstone (MOV) attack [35] uses the Weil pairing to embed the group of points on an elliptic curve over $\mathbf{F}_q$ in the multiplicative group of a larger finite field. If this finite field is only a small degree extension of $\mathbf{F}_q$, then we will be vulnerable to index calculus or number field sieve attacks on the discrete

logarithm problem in the multiplicative group of the extension field. This happens in particular with supersingular elliptic curves, where the degree is at the most 6. These curves were favored because addition in them involves fewer operations, but now should be avoided. For supersingular hyperelliptic curves of genus 2, the extension degree is at the most 30 [18].

The way the MOV attack works is as follows. Suppose we wish to solve $Q = mP$ for m, where because of the Pohlig–Hellman attack we are assuming P has prime order n. Let e be the smallest positive integer such that $q^e \equiv 1 \pmod{n}$. This ensures that $\mathbf{F}_{q^e}$ contains primitive nth roots of 1. For supersingular curves, Menezes [35] showed with case-by-case consideration that $e \leq 6$. Curves of trace 2 are also bad since $n = q + 1 - 2 = q - 1$ and so $e = 1$.

Say $e > 1$. The Weil pairing is a pairing

$$E(\mathbf{F}_{q^e})/nE(\mathbf{F}_{q^e}) \times E[n] \to \mathbf{F}_{q^e}^*/(\mathbf{F}_{q^e}^*)^n$$

This injects $< P >$ into a subgroup of $\mathbf{F}_{q^e}^*/(\mathbf{F}_{q^e}^*)^n$ and so maps the discrete logarithm problem over to the multiplicative group of a finite field, where subexponential methods can be used if e is reasonably small. This includes index calculus methods where a set of elements is chosen to act as a factor base. Enge [12] has a form of index calculus for attacking elliptic curve cryptosystems directly, but these are ineffective for large field sizes. The case $e = 1$ is similar but runs into a small technicality involving non-degeneracy of the Weil pairing [3]. Frey and Rück [16] introduced a similar method that uses the Tate pairing.

8.9.5 Semaev, Satoh-Araki, Smart Attack

An anomalous elliptic curve is one with exactly q points. The Semaev, Satoh-Araki, and Smart attack [51], [47], [56] maps the group to the additive group of $\mathbf{F}_q$, yielding a polynomial-time attack (the other attacks listed here are at best subexponential). The main idea is as follows. Suppose we wish to solve $Q = mP$ in the elliptic curve E over $\mathbf{F}_q$. There is a unique smallest complete local ring of characteristic zero, $\mathbf{Z}_q$, the q-adic integers, and by Hensel's lemma we can lift P and Q to points defined over $\mathbf{Z}_q$, say $\tilde{P}$ and $\tilde{Q}$. Then $qE(\mathbf{Z}_q)/q^2E(\mathbf{Z}_q) \cong \mathbf{F}_q$ and denoting this map by log we have that $\log(q\tilde{P}) = m\log(q\tilde{Q})$. Then solving the discrete logarithm problem in the additive group of a field is trivial, using Euclid to invert $\log(q\tilde{Q})$.

8.9.6 Attacks employing Weil descent

More recently, Weil descent has been used for some special finite fields. In 2000, Gaudry, Hess, and Smart [20], extending the work of Frey [15] and Galbraith [17], showed how to reduce a discrete logarithm problem in $E(\mathbf{F}_{q^e})$ to a discrete logarithm

problem in $\mathrm{Jac}(C)(\mathbf{F_q})$ where C is a hyperelliptic curve. The idea is to use the Weil restriction of E, which is an e-dimensional abelian variety over $\mathbf{F}_q$, and intersect it with $e-1$ hyperplanes to obtain C. For example, if $q = 2^{31}$ and $e = 5$, we obtain a curve C of genus at most 16, so it is possible to attack many elliptic curves over 2^{155} this way. The GHS attack has yet to be shown to be effective in practice – in particular none of the ten elliptic curves in the standards is vulnerable to it. Thanks to the work of Gaudry, and later Diem [8], elliptic curves over fields $\mathbf{F}_{p^n}$, where both p and n are large, are vulnerable to this method. Diem and Thomé [9] have also introduced an index calculus method for non-hyperelliptic curves of genus 3.

8.10 Good Curves

Putting together the attacks from the previous section leads to design criteria for the underlying elliptic or hyperelliptic curve. To set up an ECC, we should use an elliptic curve over $\mathbf{F}_q$ and a subgroup of order n where:

(i) n should be prime (Pohlig-Hellman);
(ii) q should be of the order of 1000 bits to be truly considered secure, but in practice 160 bits is considered equivalent to about 1024-bit RSA and 190 bits to 2048-bit RSA. Thanks to baby-step giant-step and Pollard's methods, these are considered equivalent to 80-bit and 95-bit symmetric cryptosystems respectively;
(iii) the curve should not be anomalous, so n should not equal q (Semaev, Satoh-Araki, Smart);
(iv) the smallest positive integer e such that $q^e \equiv 1 \pmod{n}$ should be large so the curve should not be of trace zero or two nor supersingular (Menezes–Okamoto–Vanstone);
(v) certain ground fields, e.g., $\mathbf{F}_{2^{155}}$, should be avoided (Weil descent).

The standards provide suitable curves. For example, FIPS 186-2 lists 10 fields and methods to choose elliptic curves that will produce secure cryptosystems.

8.11 Exercises

1. Prove that for an elliptic curve, the "chord and tangent" addition rule described in Section 8.6 is the same as the one given in Cantor's algorithm.
2. If the characteristic of $\mathbf{k}$ is not 2, show that any hyperelliptic curve C_1 with equation:
$$Y^2 + h_1(X)Y = f_1(X)$$
is isomorphic to a hyperelliptic curve C_2 with equation:
$$Y^2 = f_2(X)$$
An isomorphism between hyperelliptc curves is a linear map $(X,Y) \to (aX + bY, cX + dY)$ such that the points on C_1 are mapped to the points on C_2.

3. Let E be the elliptic curve $y^2 = x^3 + 81x + 103$ defined over $\mathbf{F}_{1013}$. Show that it has 962 points. Since 962 factors as $2 \times 13 \times 37$, you can use Pohlig–Hellman to solve the following discrete logarithm problem. Let $P = (1,728)$ and $Q = (769,175)$. Find an integer m such that $Q = mP$.
4. Let E be the same elliptic curve as in Question 3. Find the smallest positive integer e (embedding degree) such that with $q = 1013$ and $n = 962$, $q^e \equiv 1 \pmod{n}$. Convert the discrete logarithm problem of question 1 into one in $\mathbf{F}_{q^e}$ and estimate how long it will take to solve using index calculus methods.
5. Let E be the elliptic curve $y^2 = x^3 + 141x + 30$ defined over $\mathbf{F}_{1013}$. Show that this curve is anomalous. Construct an explicit isomorphism between this and the additive group of $\mathbf{F}_{1013}$. Using this, if $P = (1,292)$ and $Q = (316,412)$, find an integer m such that $Q = mP$.
6. Suppose E is an elliptic curve with equation $y^2 = x^3 + ax$ over prime field $\mathbf{F}_p$ with $p \equiv 3 \pmod{4}$. Show that it has exactly $p + 1$ points. [Hint: how many points do x and $-x$ together contribute?]. What is the embedding degree? Show how one can map arbitrary ID-strings to points on $E(\mathbf{F}_p)$ (so that it can be used in identity-based cryptography).

8.12 Projects

1. To show why Elliptic Curve Cryptography has had such an impact, implement both ECC and RSA and compare their timings. For details on RSA, consult [46]. Remember that for similar security, a key size of 160 bits in ECC is equivalent to 1024-bit RSA. Thus, even if the speed of RSA is better, transmission and storage costs for ECC are lower. For previous results on RSA vs ECC for 8-bit processors, see [21].
2. Consider the curve $x^2 + y^2 = a^2(1 + x^2y^2)$, where $a^5 \neq a$. Show that this defines an elliptic curve and that every elliptic curve is, possibly over an extension field, isomorphic to such a curve. Figure out explicit rules for addition and point-doubling. A very recent article by Harold Edwards in the Bulletin of the AMS (July 2007) carries out these calculations and more. Bernstein and Lange have suggested that this form could be very good for ECC, ensuring it is better than genus 2 HCC. Test out this claim in practical implementations, comparing this ECC with state-of-the-art HCC as found in papers on Gaudry's and Lange's websites.

References

1. A. O. L. Atkin. *The number of points on an elliptic curve modulo a prime*, Series of emails to the NMBRTHRY mailing list, 1992
2. D. J. Bernstein and T. Lange. *Elliptic vs. hyperelliptic*, (parts 1 and 2), talks at ECC-06

3. I. Blake, G. Seroussi, and N. Smart. *Elliptic Curves in Cryptography*, London Mathematical Society Lecture Note Series, Cambridge University Press, 1999
4. I. Blake, G. Seroussi, and N. Smart. *Advances in Elliptic Curve Cryptography*, London Mathematical Society Lecture Note Series, Cambridge University Press, 2004
5. D. G. Cantor. Computing in the Jacobian of a hyperelliptic curve. In: *Mathematics of Computation*, 48(177): 95–101, 1987
6. H. Cohen. A Course in Computational Algebraic Number Theory, *Graduate Texts in Mathematics* 138, 1993
7. J. Denef and F. Vercauteren. An extension of Kedlaya's algorithm to Artin-Schreier curves in characteristic 2, in ANTS-V, 2002
8. C. Diem. The GHS attack in odd characteristic, *Journal of Ramanujan Mathematical Society* 18(1): 1–32, 2003
9. C. Diem and E. Thomé. "Index calculus attacks in class groups of non-hyperelliptic curves of genus three", *Journal of Mathematical Cryptology* 2, to appear, 2008
10. W. Diffie and M. E. Hellman. New directions in cryptography, IEEE Transaction Information Theory, IT-22, 6: 644–654, 1976
11. N. Elkies. Elliptic and modular curves over finite fields and related computational issues In: *Computational Perspectives on Number Theory*, 21–76, 1998
12. A. Enge. *Elliptic Curves and Their Applications to Cryptography, An Introduction*, Kluwer Academic Publishers 1999
13. X. Fan, T. Wollinger, and Y. Wang. Inversion-Free Arithmetic on Genus 3 Hyperelliptic Curves and Its Implementations, *International Conference on Information Technology: Coding and Computing - ITCC*, April 11–13, 2005
14. M. Fouquet, P. Gaudry, and R. Harley. On Satoh's algorithm and its implementation, *Journal of Ramanujan Mathematical Society* 15: 281–318, 2000
15. G. Frey. *How to disguise an elliptic curve*, talk at ECC '98, 1998
16. G. Frey and H. Rück. A remark concerning *m*-divisibility and the discrete logarithm in the divisor class group of curves. *Mathematics of Computation*, 62(206): 865–874 (1994)
17. S. Galbraith. Limitations of constructive Weil descent. In: *Public-Key Cryptography and Computational Number Theory*, 59–70, de Gruyter, 2000
18. S. Galbraith. "Supersingular curves in cryptography", *LNCS* 2248: 200-217, 2002
19. P. Gaudry. Fast genus 2 arithmetic based on theta functions, *Journal of Mathematical Cryptology*, 1: 243–266, 2007
20. P. Gaudry. F. Hess, and N. Smart, Constructive and destructive facets of Weil descent on elliptic curves. *Journal of Mathematical Cryptology*, 2000
21. N. Gura. A. Patel, A. Wander, H. Eberle, and S. Shantz, *Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs*, CHES2004, Cambridge (Boston), 2004
22. R. Harley. *Fast Arithmetic on Genus Two Curves*, http://cristal.inria.fr/∼harley/hyper/, (2000)

23. M. Jacobson, N. Koblitz, J. Silverman, A. Stein, and E. Teske. Analysis of the xedni calculus attack. *Designs, Codes, and Cryptography*, 20(1): 41–64, 2000
24. M. Jacobson, A. Menezes, and A. Stein. "Solving elliptic curve discrete logarithm problems using Weil descent", *Journal of Ramanujan Mathematical Society* 16(3): 231–260, 2001
25. K. Kedlaya. "Counting points on hyperelliptic curves using Monsky-Washnitzer cohomology", *Journal of Ramanujan Mathematical Society* 16: 323–338, 2001
26. N. Koblitz. Elliptic curve cryptosystems. In: *Mathematics of Computation* 48: 203–209, 1987
27. N. Koblitz. Hyperelliptic cryptosystems. *Journal of Mathematical Cryptology* 1: 139–150, 1989
28. J. Kuroki, M. Gonda, K. Matsuo, J. Chao, and S. Tsujii. Fast Genus Three Hyperelliptic Curve Cryptosystems. In *Proceedings of SCIS*, 2002
29. T. Lange. *Efficient Arithmetic on Hyperelliptic Curves*, PhD Thesis. Universitat-Gesamthochschule Essen, 2001
30. T. Lange. Weighted Coordinates on Genus 2 Hyperelliptic Curves. *Cryptology ePrint Archive*, Report 2002/153, 2002
31. T. Lange. Inversion-Free Arithmetic on Genus 2 Hyperelliptic Curves. Preprint, 2002
32. T. Lange, C. Günther, and A. Stein. Speeding up the arithmetic on hyperelliptic Koblitz curves of genus 2, SAC 2001, LNCS 2012, Springer 106–117, 2001
33. A. Lauder and D. Wan. Computing zeta functions of Artin-Schreier curves over finite fields, London Math Soc. JCM 5: 34–55, 2002
34. K. Matsuo J. Chao, and S. Tsujii. Fast Genus Two Hyperelliptic Curve Cryptosystems, *Proc. Second Int'l Symp. Electronic Commerce* (ISEC 2001), 2001
35. A. Menezes, T. Okamoto, and S. Vanstone. Reducing elliptic curve logarithms to a finite field. *IEEE Transaction on Information Theory*, 39: 1639-1646, 1993
36. A. Menezes, Y-H. Wu, and R. Zuccherato. *An Elementary Introduction to Hyperelliptic Curves*. Technical Report CORR 96-19, Department of Combinatorics and Optimization, University of Waterloo, Ontario, Canada, (1996)
37. J. F. Mestre. *AGM pour le genre 1 et 2*, lettre à Gaudry et Harley, Dec 2000
38. V. Miller. Use of elliptic curves in cryptography, *CRYPTO* 85, 1985
39. Y. Miyamoto, H. Doi, K. Matsuo, J. Chao, and S. Tsuji. A Fast Addition Algorithm of Genus Two Hyperelliptic Curve, *Proceedings of SCIS 2002*, 497-502, in Japanese, 2002
40. P.van Oorschot and M. Wiener. Parallel collision search with cryptanalytic applications. *Journal of Mathematical Cryptology*, 12, no. 1, 1-28 1999
41. J. Pelzl, T. Wollinger, J. Guajardo, and C. Paar. Hyperelliptic curves cryptosystems: closing the performance gap to elliptic curves. *Cryptology ePrint Archive*, 2003, http://eprint.iacr.org/

42. J. Pelzl, T. Wollinger, and C. Paar. Low Cost Security: Explicit Formulae for Genus-4 Hyperelliptic Curves, In Tenth Annual Workshop on Selected Areas in Cryptography, 2003
43. J. Pila. Frobenius maps of abelian varieties and finding roots of unity in finite fields. *Mathematics of Computation* 55: 745-763, 1990
44. G. Pohlig and M. Hellman. An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. *IEEE Transaction on Information Theory*, 24: 106-110, 1978
45. J. Pollard. Monte Carlo methods for index computation mod p. *Mathematics of Computation*: 918-924 (1978)
46. R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21 (2): 120-126, 1978
47. T. Satoh and K. Araki. Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves, Comm. Math. Univ. Sancti Pauli, 47(1): 81-92, 1998
48. T. Satoh. The canonical lift of an ordinary elliptic curve over a finite field and its point counting, *Journal of Ramanujan Mathematical Society*. 15: 247-270, 2000
49. T. Satoh, B. Skjernaa, and Y. Taguchi. Fast computation of canonical lifts of elliptic curves and its application to point counting, *Finite Fields and Their Applications* 9: 89-101, 2003
50. R. Schoof. Elliptic curves over finite fields and the computation of square roots mod p, *Mathematics of Computation* 44: 483-494, 1985
51. I. A. Semaev. Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p, 67(221): 353-356, 1998
52. V. Shoup. Lower bounds for discrete logarithms and related problems. In *Proc. Eurocrypt* '97, pp. 256-266, 1997
53. J. H. Silverman. The arithmetic of elliptic curves. *Graduate Texts in Mathematics*, vol 106, Springer-Verlag, 1986
54. J. H. Silverman. The xedni calculus and the elliptic curve discrete logarithm problem. *Designs, Codes, and Cryptography*, 20: 5–40, 2000
55. B. Skjernaa. Satoh's algorithm in characteristic 2. *Mathematics of Computation* 72: 477–488, 2003
56. N. Smart. The discrete logarithm on elliptic curves of trace one. *Journal of Mathematical Cryptology*, 12: 193–206, 1999
57. A. M. Spallek. *Kurven vom Geschlecht 2 und ihre Anwendung in Public-Key-Kryptosystemen*, PhD Thesis. Universitat Gesamthochschule Essen, 1994
58. M. Takahashi. *Improving Harley Algorithms for Jacobians of Genus 2 Hyperelliptic Curves*, In SCIS, IEICE Japan, 2002. in Japanese.
59. E. Teske. Speeding up Pollard's rho method for computing discrete logarithms. *LNCS*, 1423: 541–554, 1998
60. F. Vercauteren. Computing zeta functions of hyperelliptic curves over finite fields of characteristic 2. In "Advances in cryptology - CRYPTO 2002", *LNCS* 2442: 369–384, 2002

61. F. Vercauteren, B. Preneel, and J. Vandewalle, A memory efficient version of Satoh's algorithm. In "Advances in Cryptology - EUROCRYPT 2001", *LNCS* 2045, 1–13 (2001)
62. T. Wollinger. *Software and Hardware Implementation of Hyperelliptic Curve Cryptosystems*, Ph.D. Thesis, Ruhr-Universitt Bochum, Germany, July 2004