A reduction of the space for the parallelized Pollard lambda search on elliptic curves over prime finite fields and on anomalous binary elliptic curves

Igor Semaev
Department of Mathematics
University of Leuven,Celestijnenlaan 200B
3001 Heverlee,Belgium
Igor.Semaev@wis.kuleuven.ac.be

August 11, 2003

Abstract

Let E be an elliptic curve defined over a prime finite field F_p by a Weierstrass equation. In this paper we introduce a new partition of $E(F_p)$ into classes which are generally larger than $\{\pm R\}$. We give an effective procedure to compute representatives of such classes. So one can iterate the pseudorandom function, related to a discrete logarithm problem in $E(F_p)$, on the set of representatives of classes and get probably some speed up in computing discrete logarithms. The underlying idea how to enlarge known classes on anomalous binary elliptic curves is given.

1 Introduction

Let E be the elliptic curve defined over the prime field F_p of characteristic p by the equation

$$eY^2 = f(X), \text{ where } \quad f(X) = a_1X^3 + a_2X^2 + a_3X + a_4. \tag{1}$$

Let e, a_1, a_2, a_3, a_4 be integer numbers. We denote $|f| = \max_i |a_i|$. Let Q be a point from the group $E(F_p)$ that generates a subgroup of $E(F_p)$ of order m. The discrete logarithm problem is given a point $P \in < Q >$ find the residue $r(\bmod m)$ such that $P = rQ$ in $E(F_p)$. This problem is one of the hard problem in the public key cryptography, [1] and [2]. The best algorithm for solving the discrete logarithm problem on a general elliptic curve over a large prime finite field is the parallelized Pollard lambda-method, see [3] and [4]. In this algorithm each processor involved in the computation iterates a pseudorandom function on

$< Q >$. This function is chosen such that the logarithm of $\psi(R)$ is expressed by affine functions in the logarithm of R and the unknown logarithm of P, which we just want to compute. In [5] it was found that it is faster to iterate the function ψ on the classes $\{\pm R\}$ of points in $< Q >$. It resulted in the reduction of the space for the parallelized Pollard lambda search for two times. This reduction speeds up the discrete logarithm computation by a factor of $\sqrt{2}$.

In the present paper we introduce a new partition of $E(F_p)$ into classes which are generally larger than $\{\pm R\}$. We give an effective procedure to compute representatives of such classes. So one can iterate the pseudorandom function on the set of representatives of classes and get probably some speed up in computing discrete logarithms. The number of points in a class doesn't exceed $O(\sqrt{\log p})$. So the speed up doesn't exceed a factor bounded by $O(\sqrt[4]{\log p})$. In reality it is about a constant closed to 1 because most of classes should be like $\{\pm R\}$. We should also mention here that the number of new classes and the number of points in each class apparently depend on the size $|f|$ of the polynomial f. Namely, the less $|f|$, the bigger some classes and so the less the number of classes.

In the last Section of the paper the underlying idea how to enlarge known classes on anomalous binary elliptic curves is given.

2 Auxiliary algorithms

We consider the curve (1) as an elliptic curve over the field $\mathbb{Q}$ of rational numbers. Let $R = (x, y)$ be a point from $E(\mathbb{Q})$ and $x = x'/x''$ be the representation of the rational number x in its lowest terms. We put $|x| = \max\{|x'|, |x''|\}$ and $|R| = |x|$. We assume that the integer numbers e, a_1, a_2, a_3, a_4 take $O(\log |x|)$ bits for their representation.

First we consider if there exists a natural number $n > 1$ and a point $R_1 \in E(\mathbb{Q})$ such that $nR_1 = R$. Then we show how to find such a point R_1 in the case when it exists. It is obvious that we can restrict ourselves to working with x-coordinates of the points R, R_1. So to find the point R_1 is to find its x-coordinate x_1. For the sake of the application here we are searching R_1 such that $|R_1| < |R|/2$. Intuitively it is the most probable case.

We'll give an heuristic algorithm that solves this problem in $O(\log^2 |x|)$ binary operations. It is well-known that if $nR_1 = R$ then

$$n^2 \log |R_1| + O(1) = \log |R|, \tag{2}$$

where $O(1)$ depends on E and n, see [8]. Euristically it implies that $n = O(\sqrt{\log |x|})$. Let n be a natural number that satisfies the restriction above. First we test if a point R_1 exists for this n. Let q be a prime number of the size about the size of n. We consider the reduction of the rational elliptic curve (1) modulo q. That is a homomorphic map

$$E(\mathbb{Q}) \to E(F_q),$$

where $E(F_q)$ is the group of points on the elliptic curve (1) modulo q or a group isomorphic to the additive or the multiplicative group of the finite field F_q. Let n divides $N_q = |E(F_q)|$. It is obvious that if

$$(N_q/n)R \neq P_\infty$$

in $E(F_q)$ then R isn't nth power of any point in $E(\mathbb{Q})$. If the point R passes l such tests for different q one can say that R is a nth power with the probability of the mistake about $1/n^l$. To accomplish this test one doesn't need the y-coordinate of the point R. See the formulae in the end of this Section. So the complexity of this test is $O(\log^3 q) = o(\log |x|)$ binary operations. It takes some time to find such a q. One can show that it is no more than $O(\log^{1+\varepsilon} |x|)$ binary operations. But we can neglect this time in the forthcoming application, for q can be precomputed. We see now that the complexity of a few tests for all possible n is bounded by $O(\log^{1/2+\varepsilon} |x|)$ binary operations for small ε.

We take the maximal n that has passed a number of tests above. We'll explain now how to find a point $R_1 \in E(\mathbb{Q})$ such that $nR_1 = R$. We consider two possibilities.

Let $E(\mathbb{Q})$ not have any nontrivial n-torsion points. Then we fix a list of prime numbers q such that n is coprime to $N_q = |E(F_q)|$ and $\prod_q q > |x|^2$. For each q we solve the equation

$$R_q^n \equiv R \,(\operatorname{mod} q)$$

in $R_q \in E(F_q)$ by formulae

$$R_q \equiv R^{m_q} \,(\operatorname{mod} q),$$

where m_q is a natural number such that $nm_q \equiv 1 (\operatorname{mod} N_q)$. Let x_q be the x-coordinate of R_q. Then using the Chinese Remainder Theorem and the extended Euclidean Algorithm we find a rational number x_1 congruent to x_q modulo q for all q from the list such that $|x_1| < \sqrt{\prod_q q}$. If $|x_1| \geq |x|/2$ we try smaller n or say that the problem hasn't solution. If $|x_1| < |x|/2$, which is more probable, we terminate. The complexity of this step is $O(\log^2 |x|)$ binary operations.

Let $E(\mathbb{Q})$ have a nontrivial n-torsion point now. Then the number $N_q = |E(F_q)|$ isn't coprime to n for any prime number q. So if we use the same step as above we have too many solutions as the result of the Chinese Remainder Theorem. So we apply Henzel Lemma. It is obvious that we can restrict ourselves to the case when n is a prime number. According to Mazur's Theorem, see [8], $n = 2, 3$ or 5. We'll give an example of the application of Henzel Lemma when $n = 2$. When $n = 3$ or 5 the method is similar.

Using the duplication formula on E we get

$$x = \frac{f'(x_1)^2 - 4a_2 f(x_1) - 8a_1 x_1 f(x_1)}{4a_1 f(x_1)}.$$

That is x_1 is a root of the polynomial

$$a_1^2 X^4 - 4x a_1^2 X^3 - (2a_1 a_3 + 4a_1 a_2 x)X^2 -$$
$$(4a_1 a_3 x + 8a_1 a_4)X + a_3^2 - 4a_2 a_4 - 4a_1 a_4 x.$$

in X. We fix a prime number q and a natural number t such that $q^t > |x|^2$. For each root modulo q of the polynomial above we apply Henzel Lemma to get a root x_1' of this polynomial modulo q^t. With some small probability the polynomial may have a lot of solutions modulo some q^s, where $1 < s \leq t$. In this case we take another prime number q and repeat the computations. Then with the extended Euclidean Algorithm we find a rational $x_1 \equiv x_1' (\operatorname{mod} q^t)$, where $|x_1| \leq \sqrt{q^t}$. If $|x_1| < |x|/2$, which is more probable, we terminate. The complexity of this step is $O(\log^2 |x|)$ binary operations as above.

Let $P = (x, y)$ be any affine point on E defined by (1). We denote $nP = (x_n, y_n)$, so $x_1 = x$ and $y_1 = y$. If $nP = P_\infty$ we put $x_n = \infty$. The x-coordinate of nP doesn't depend on y. We are going to give an algorithm for computing x_n without using y-coordinates. This algorithm is a slight generalization of that due to Montgomery [7] designed for a particular case when $f(0) = 0$. Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be two affine points on E such that $x_1 \neq x_2$. In fact we need to consider only this case. We denote

$$P_3 = (x_3, y_3) = P_1 + P_2,$$
$$P_4 = (x_4, y_4) = P_1 - P_2.$$

One can see that x_3, x_4 are roots of a quadratic polynomial, whose coefficients are symmetric functions in x_1 and x_2. Really, we derive

$$x_3 = \frac{e\lambda_3^2 - a_2}{a_1} - x_1 - x_2,$$
$$x_4 = \frac{e\lambda_4^2 - a_2}{a_1} - x_1 - x_2,$$

where $\lambda_3 = (y_1 - y_2)/(x_1 - x_2)$ and $\lambda_4 = (y_1 + y_2)/(x_1 - x_2)$. Then

$$\begin{gathered} x_3 + x_4 = \\ \tfrac{e(\lambda_3^2+\lambda_4^2)-2(a_2+a_1(x_1+x_2))}{a_1} = \\ A(x_1, x_2), \end{gathered}$$

and

$$\begin{gathered} x_3 x_4 = \\ \tfrac{e^2\lambda_3^2\lambda_4^2 - e(a_2+a_1(x_1+x_2))(\lambda_3^2+\lambda_4^2)+(a_2+a_1(x_1+x_2))^2}{a_1^2} = \\ B(x_1, x_2), \end{gathered}$$

Since $\lambda_3^2\lambda_4^2$ and $\lambda_3^2 + \lambda_4^2$ are symmetric functions in x_1, x_2, the x-coordinates x_3 and x_4 are roots of the polynomial

$$X^2 - A(x_1, x_2)X + B(x_1, x_2),$$

where $A(x_1, x_2)$ and $B(x_1, x_2)$ are symmetric functions in x_1 and x_2.

For the duplication on E we use the usual formulae. That is

$$x_{2m} = \frac{e\lambda^2 - a_2}{a_1} - 2x_m,$$

where $\lambda^2 = (f'(x_m))^2/(4ef(x_m))$. Therefore

$$x_{2m} = \frac{(f'(x_m))^2 - 4a_2 f(x_m)}{4a_1 f(x_m)} - 2x_m \tag{3}$$

At each step of the algorithm that computes x_n we have a pair of x-coordinates (x_m, x_{m+1}). The initial pair is (x_1, x_2). If $x_m \neq x_{m+1}$, then we compute (x_{2m}, x_{2m+1}) or (x_{2m+1}, x_{2m+2}) by formulae (3) and

$$x_{2m+1} = A(x_m, x_{m+1}) - x_1 \quad \text{or} \quad x_{2m+1} = B(x_m, x_{m+1})/x_1.$$

Really, x_{2m+1}, x_1 are roots of the polynomial $X^2 - A(x_m, x_{m+1})X + B(x_m, x_{m+1})$. If $x_m = x_{m+1}$, then $(2m+1)P = P_\infty$ and we put $x_{2m+1} = \infty$. Let $x_{m+1} = \infty$ now. It is easy to see that $x_{2m+1} = x_m$. Similarly when $x_m = \infty$ then $x_{2m+1} = x_{m+1}$. This gives a way to compute x_n. Let $n_0, \ldots, n_{t-1}, 1$ be the 2-adic expansion of the natural number n, that is

$$n = n_0 + \ldots + n_{t-1}2^{t-1} + 2^t,$$

where $n_i = 0$ or 1. Then we put $m_0 = 1$ and compute

$$(x_{m_i}, x_{m_i+1}) = \begin{cases} (x_{2m_i}, x_{2m_i+1}), & \text{if } n_{t-i} = 0, \\ (x_{2m_i+1}, x_{2m_i+2}), & \text{if } n_{t-i} = 1, \end{cases}$$

for $i \geq 1$. Finally we have $x_n = x_{m_t}$.

3 The algorithm

We define a pseudorandom function ψ on the set of x-coordinates of points. Let's denote by x_P the x-coordinate of the point $P \in E(F_p)$. We partition F_p into $t \geq 3$ roughly equal size sets $S_1, S_2, \ldots, S_t$ and fix t small different natural numbers $r_1, r_2, \ldots, r_t$ bigger than 1. To define ψ we put

$$\psi(x_P) = x_{r_k P},$$

when $x_P \in S_k$. According to the terminology introduced in [6] this is a multiplicative method of getting a pseudorandom function on $E(F_p)$. To compute $x_{r_k P}$ one needs nothing but x_P and formulae given in the previous Section. As we'll see there is a step in the algorithm, when we should use usual formulae involving y-coordinates for the arithmetic operation on E. It is only to set up initial points for processors iterating the pseudorandom function ψ.

Before presenting the algorithm we discuss the representation of residues modulo p by rational numbers. Let x be a nonzero residue modulo p. By means of the extended Euclidean Algorithm applied to x and p one finds a pair of integer numbers x' and x'' such that

$$x \equiv \frac{x'}{x''} (\operatorname{mod} p) \quad \mathit{and} \quad 1 \leq |x'|, x'' < \sqrt{p}. \tag{4}$$

It is easy to understand that there exists one or two such pairs. Let they be two, namely x', x'' and x'_1, x''_1. Then the numbers x' and x'_1 have different signs and

$$x'x''_1 - x'_1x'' = \pm p.$$

Therefore if

$$x \equiv \frac{x'}{x''} (\operatorname{mod} p) \quad \mathit{and} \quad 1 \leq |x'|, x'' < \sqrt{p}/2$$

then there exists only one pair satisfying (4). We are able to define the representation of a point $P \in E(F_p)$ by a rational number. It is $\frac{x'}{x''}$ in its lowest terms satisfying (4), where $x = x_P$. The integer numbers x' and x'' are the result of a fixed variant of the extended Euclidean Algorithm applied to x_P and p. If $x_P \equiv 0(\operatorname{mod} p)$ we represent x_p by 0. There won't be any confusion even if there are two pairs of integer numbers satisfying (4). Therefore we put $|x_P| = \max\{|x'|, |x''|\}$.

We'll give an informal description of the algorithm. For each natural number s, where $1 \leq s \leq O(\sqrt{\log p})$, one should have precomputed a few small prime numbers q such that s divides $N_q(d)$. Here $N_q(d)$ is the order of the group of points defined over F_q on the rational elliptic curve

$$dY^2 = f(X) \tag{5}$$

taken modulo q. Only two variants for d should be taken into account. Namely, when d is a square modulo q and a nonsquare. Obviously one can restrict himself to the case when s is a prime number. It reduces the number of tests described in Section 2.

Then for each s, coprime to the order of the torsion group on the rational elliptic curve (5), which only depends on the polynomial $f(X)$, one should have precomputed a list of small prime numbers q, together with s_q, such that $\gcd(s, N_q(d)) = 1$, and

$$ss_q \equiv 1(\operatorname{mod} N_q(d)),$$

and $\prod_q q > p$. We conjecture that such a list of small prime numbers may be found. Therefore one solves the equation $sR_1 = R$ on rational curve (5) in one step by using the Chinese Remainder Theorem.

Let s not be coprime to the order of the torsion group on the rational elliptic curve (5) and l be their gcd. Then one should have precomputed a list of small prime numbers q, together with s_q, such that $\gcd(s, N_q(d) = l$, and

$$ss_q \equiv l(\operatorname{mod} N_q(d)),$$

and $\prod_q q > p$. We conjecture that such a list of small prime numbers may be found. Therefore one solves the equation $sR_1 = R$ on the rational curve (5) in two steps. At the first one finds lR_1 by using the Chinese Remainder Theorem. At the second step computes R_1 by using Henzel Lemma.

The input of the algorithm is the points $Q, P \in E(F_p)$ such that Q generates a group of order m and $P \in < Q >$. The output is the residue r modulo m such that $P = rQ$. We suppose using $T \geq 2$ processors and one central processor.

Step 1 On processor i generate random numbers u_i and v_i, compute the point $P_{i0} = u_i P + v_i Q$ in $E(F_p)$ and put x_{i0} to be the x-coordinate of P_{i0}. Set up the initial values for other parameters $r_{i0} = 1$ and $s_{i0} = 1$.

Iterate ψ on the processor i with the starting point x_{i0}. That is given $x_{ij} \in S_k$, where $j \geq 0$, compute

$$x = \psi(x_{ij}) \qquad \text{and} \qquad r_{ij+1} \equiv r_{ij} r_k (\operatorname{mod} m).$$

Compute the representation (4) of the residue x modulo p by using the extended Euclidean Algorithm. Then the rational point

$$R = (\frac{x'}{x''}, \frac{1}{(x'')^2})$$

is on the rational elliptic curve E_d defined by the equation (5), where $d = (x'')^4 f(\frac{x'}{x''})$. We remark that one needs to compute neither d nor $\frac{1}{(x'')^2}$ as integer numbers. One should only decide weather d is a square modulo small prime numbers q from the list or not.

Using the precomputed list of small q find, by tests described in Section 2, the maximal s such that $sR_1 = R$ in $E_d(\mathbb{Q})$. When $s > 1$ with another precomputed list of small q find $R_1 = (x_1, y_1)$ for $|x_1| < |x|/2$, if such a point exists. In the case when R_1 doesn't exist, which is very unlikely, take smaller $s > 1$ and repeat the computation. We remark that all computations are being done on E_d modulo some small prime numbers q and they don't make any use of the y-coordinates of points involved. In particular one only needs to compute x_1 at this step. Put

$$x_{ij+1} \equiv \begin{cases} x_1, & \text{if } s > 1 \text{ and } R_1 \text{ exists}, \\ x, & \text{if } s = 1 \text{ or } R_1 \text{ doesn't exist}, \end{cases}$$

and $s_{ij+1} \equiv s_{ij} s (\operatorname{mod} m)$, where $s = 1$ if R_1 doesn't exist.

At each iteration on processor i compute the string of five numbers - $(x_{ij}, r_{ij}, s_{ij}, u_i, v_i)$. Send this string to the central processor, when x_{ij} belongs to a set of distinguished numbers. We consider x_{ij} to be distinguished if $|x_{ij}| < 2^l$, where l depends on the memory space available on the central processor and the number T of processors working in parallel.

Step 2 A useful collision occurs when there are two strings

$$(x_{ij}, r_{ij}, s_{ij}, u_i, v_i) \quad \text{and} \quad (x_{i_1 j_1}, r_{i_1 j_1}, s_{i_1 j_1}, u_{i_1}, v_{i_1})$$

in the central processor's memory such that $x_{ij} = x_{i_1j_1}$ and $i \neq i_1$. It implies that

$$r_{i_1j_1}s_{ij}P_{i_10} = \pm r_{ij}s_{i_1j_1}P_{i0}$$

and therefore

$$r_{i_1j_1}s_{ij}(ru_{i_1} + v_{i_1}) \equiv \pm r_{ij}s_{i_1j_1}(ru_i + v_i)$$

modulo m. Solving this congruence get(with a good probability) two variants for r. Then test which of them is true by comparing the x-coordinates of the points rP and Q.

This finishes the description of the algorithm. We are going to explain why this algorithm may be better than the previous variants of the Pollard lambda search. Let's consider the particular case when

1. coefficients of the polynomial $f(X)$ are small, for example, they are bounded by a constant when p is large,

2. the point Q generates the whole group $E(F_p)$, which is a cyclic group of prime order.

Let's take a small rational number $x = \frac{x'}{x''}$ in its lowest terms and consider the rational point

$$R_1 = (\frac{x'}{x''}, \frac{1}{(x'')^2})$$

on the rational elliptic curve E_d defined by the equation (5), where $d = (x'')^4 f(\frac{x'}{x''})$. Then with the probability about 1/2 the integer number ed is a square modulo p. If it is the case there exists a point $\bar{R}_1 \in E(F_p) =< Q >$, whose x-coordinate is just $\frac{x'}{x''}$ taken modulo p. This rational number is the representation of $x_{\bar{R}_1}$, introduced above. Because of the coefficients of the polynomial $f(X)$ are small, some small powers sR_1 in $E_d(\mathbb{Q})$ have x-coordinates bounded by $\sqrt{p}/2$. From (2) it follows that $|s|$ should be bounded by $O(\sqrt{\log p/ \log |x|})$. Let R, the point from the first step of the algorithm taken modulo p, be in the set

$$\{s\bar{R}_1/ \, |s| = O(\sqrt{\log p/ \log |x|})\}. \tag{6}$$

Then one finds the point R_1 and the number s such that $sR_1 = R$ in an effective way described in Section 2. One can say that sets like (6) are classes in a partition of $E(F_p)$ and $\bar{R}_1$ or its x-coordinate is a representative of the class. It really looks like a partition. We'll give an euristic argument for this. The condition 2) implies that curves (5) have not any nontrivial torsion. So it follows from Mordell's Theorem, see [8], that if R_1 and R_2 are two rational points on the same rational elliptic curve (5) and $l_1R_1 = l_2R_2$ for some integer numbers l_1, l_2, then there exists a rational point R_3 on the same curve such that $R_1, R_2 \in < R_3 >$. It means that, when classes like (6) intersect, they are subclasses of a bigger class like (6).

The previously known classes, introduced in [5], look like

$$\{\pm \bar{R}_1\}. \tag{7}$$

We see that classes (6) are generally larger than classes (7). Therefore the number of classes in the partition of $< Q >$ is generally less, at least for the case defined by conditions 1) and 2). Using this partition of $E(F_p)$ reduces the space for the Pollard lambda search and may slightly affect its effectiveness for this reason.

4 A remark about implementation

When the point R taken modulo p is in the trivial class like (7), one only needs few tests to recognize this event. It takes a negligible time to do. When R taken modulo p is in a nontrivial class like (6) it takes more time to find a representative of this class. One should use the Chinese Remainder Theorem with the modulus about p. So it takes time comparable with few multiplications modulo p to do.

There are two points why the application of the Euclidean Algorithm doesn't increase much the cost of iterating the function ψ or probably even reduces it.

1 Let x_{ij} be given by the representation (4). Then the computation of $x = \psi(x_{ij})$ is going faster because the algorithm described in Section 2 deals with smaller numbers.

2 The Euclidean Algorithm can be incorporated into the algorithm implementing the arithmetic operation on E, see formulas in Section 2. Let this algorithm compute a division modulo p at its last step. Then one changes this division by the reduction of a 2-dimensional lattice. Really, let the algorithm should compute $x \equiv a/b$ modulo p, where a, b are integer numbers. In order to do this computation one defines the lattice in $\mathbb{Z}^2$ generated by vectors $(a, b), (0, p), (p, 0)$. Then one applies the 3-dimensional lattice reduction algorithm, see [9], which manages such lattices as well, or the pairwise Gaussian Reduction, described in the same paper. The reduction algorithm computes the Minkowski reduced basis of the lattice above. The shortest nonzero vector of the basis gives a representation (4) for x in most cases. If it is not the case then sum or difference of basis' vectors will work. If there are two such representations for x one takes the vector with the positive coordinates.

5 Curves recommended by FIPS

FIPS 186-2 recommends two sorts of elliptic curves for using in the public key cryptography. They are elliptic curves defined over a large prime finite field F_p and anomalous elliptic curves over the field F_2 of two elements. The elliptic curve E over F_p is defined by the equation

$$Y^2 = X^3 - 3X + b, \tag{8}$$

where $b^2c \equiv -27(\operatorname{mod} p)$. The integer number c is being produced with the SHA-1 algorithm and looks like a random residue modulo p. The equation has an equivalent form

$$3bY^2 = X^3 + cX + c. \tag{9}$$

We see that the polynomials on the right-hand sides of the equations (8) and (9) depend on large numbers of the size about the size of p. For such curves new classes found in the present paper coincide with the classes $\{\pm R\}$ introduced in [5].

The picture is quite different for an anomalous curve E over F_2. Such a curve is defined by the equation

$$Y^2 + XY = X^3 + \alpha X + 1,$$

where $\alpha = 0$ or 1. Let F_{2^l} be the finite field of 2^l elements. We fix some irreducible polynomial $h(X)$ of degree l. So we fix a representation of the field's elements by polynomials because $F_{2^l} \cong F_2[X]/h(X)$. Let

$$a = a(X) = a'(X)/a''(X)$$

be a rational function in $F_2(X)$ in its lowest terms. Let a be a small function, that is its size $|a| = \max\{\deg a'(X), \deg a''(X)\}$ is small. With the probability about 1/2 the equation in Y

$$Y^2 + aY = a^3 + \alpha a + 1,$$

considered modulo $h(X)$, has a solution in F_{2^l}. In this case there is a point $R \in E(F_{2^l})$, whose x-coordinate is just $a(X)$ taken modulo $h(X)$. Because a is small, some small powers sR have x-coordinates represented by rational functions, the size of which is bounded by $l/2$. One can see that $|s|$ should be bounded by $\sqrt{l/|a|}$. So one defines the class

$$\{sR/|s| = O(\sqrt{l/|a|}\}.$$

Such classes are similar to classes (6) and can be used in the same way. In particular the representative of the class can be computed in an effective way by means introduced in Section 2. One only need to change small prime numbers by irreducible polynomials over F_2 of small degrees. Naturally one can use this idea to enlarge classes introduced in [5], [6]. Those classes include points derived from a fixed one by applying automorphisms from the Galua group of the field F_{2^l} over F_2. We are going to give a detailed presentation of this method in our next paper.

References

[1] V.Miller, *Use of elliptic curves in cryptography. Advances in cryptology—CRYPTO '85 (Santa Barbara, Calif., 1985)*, 417–426, Lecture Notes in Comput. Sci., **218**(1986), Springer, Berlin, 417–426.

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

[3] J.Pollard *Monte-Carlo methods for index computation* mod *p,* Math.Comp. **32** (1978), 918–924.

[4] P.van Oorschot and M.Wiener, *Parallel collision search with cryptanalytic applications,* J. Cryptology **12** (1999), no. 1, 1–28.

[5] M.Wiener and R.Zuccherato, *Faster attacks on elliptic curve cryptosystems. Selected areas in cryptography (Kingston, ON, 1998)*, Lecture Notes in Comput. Sci., **1556**(1999), Springer, Berlin, 190–200.

[6] R.Gallant, R.Lambert,and S.Vanstone, *Improving the parallelized Pollard lambda search on anomalous binary curves.* Math. Comp. **69** (2000), no. 232, 1699–1705.

[7] P.Montgomery, *Speeding the Pollard and elliptic curve methods of factorization.* Math. Comp. **48** (1987), no. 177, 243–264.

[8] J.Silverman, *The arithmetic of elliptic curves.* Graduate Texts in Mathematics, **106**. Springer-Verlag, New York, 1986.

[9] I.Semaev, *A 3-dimensional lattice reduction algorithm. Cryptography and lattices (Providence, RI, 2001),* Lecture Notes in Comput. Sci., **2146**(2001), Springer, Berlin, 181–193.