Finding Secure Curves with the Satoh-FGH Algorithm and an Early-Abort Strategy

Mireille Fouquet[1], Pierrick Gaudry[1], and Robert Harley[2]

[1] LIX, École polytechnique, 91128 Palaiseau Cedex, France
[2] ArgoTech, 26 ter rue Nicolaï, 75012 Paris, France

Abstract. The use of elliptic curves in cryptography relies on the ability to count the number of points on a given curve. Before 1999, the SEA algorithm was the only efficient method known for random curves. Then Satoh proposed a new algorithm based on the canonical p-adic lift of the curve for $p \geq 5$. In an earlier paper, the authors extended Satoh's method to the case of characteristics two and three. This paper presents an implementation of the Satoh-FGH algorithm and its application to the problem of finding curves suitable for cryptography. By combining Satoh-FGH and an early-abort strategy based on SEA, we are able to find secure random curves in characteristic two in much less time than previously reported. In particular we can generate curves widely considered to be as secure as RSA-1024 in less than one minute each on a fast workstation.

1 Introduction

Since elliptic curve cryptosystems were first proposed in the mid-eighties by Koblitz [Kob87] and Miller [Mil87], their efficiency and security have been the focus of intense study. In recent years, they have become widely accepted as an alternative to cryptosystems based on factorisation or discrete logarithms in finite fields, especially for constrained environments.

One of the initial steps in protocols based on elliptic curve cryptography is to generate a suitable curve defined over a finite field. To ensure that the system is secure, the curve must be chosen to have a number of points which is divisible by a large prime so that computing discrete logarithms on the curve is intractable using known attacks. Hence it is necessary to know the cardinality of the curve.

Among the elliptic curves defined over a given finite field, there are some classes of curves with particular properties that are useful for counting points or for accelerating arithmetic operations occurring in the protocols. However choosing such curves can be dangerous.

Perhaps the most striking example is trace 1 curves. The number of points over $\mathbb{F}_q$ is simply q. However Smart [Sma99], Satoh-Araki [SA98] and Semaev [Sem98] independently discovered a polynomial-time attack.

Another attack due to Menezes-Okamoto-Vanstone [MOV91], and generalised by Frey-Rück [FR94], reduces discrete logs on supersingular and trace 2 curves to discrete logs in a small-degree extension of $\mathbb{F}_q$. This yields an algorithm that runs in sub-exponential time.

A minor weakness is known for curves with many automorphisms [vOW99], [GLV], [DGM99] including curves defined over a small subfield, proposed by Koblitz, and some complex-multiplication curves. Attacks on these curves take less time than for generic curves, but remain in exponential time.

It has recently been shown by Gaudry-Hess-Smart [GHS00] that curves defined over composite extension fields are also weak in certain cases, using a reduction via hyperelliptic curves.

These results suggest that for maximum security one should avoid curves with special properties and instead choose a random curve whose number of points is divisible by a large prime, over a prime field or an extension of prime degree. This ideal procedure was made possible in practice by the SEA algorithm due to Schoof [Sch85], [Sch95], Elkies [Elk98], Atkin [Atk92] and others [Cou94] [Cou96], [Mor95], [Ler97a], [Mül95], [Dew98], etc. With this method, counting points on one given curve is reasonably fast.

However finding a cryptographically suitable curve requires testing many curves and this takes much more time. For instance, Johnson and Menezes [JM99] recently described this process as a "complicated and cumbersome task" requiring "a few hours on a workstation" for 200 bits.

Recently, a new algorithm for counting points on curves in small characteristic $p \geq 5$ was designed by Satoh [Sat00] and we extended it to characteristics two and three in [FGH00]. An independent extension to characteristic two is described by Skjernaa [Skj].

Satoh's algorithm is asymptotically superior to SEA for fixed p, requiring $O(\log^{3+\varepsilon} q)$ deterministic time, instead of $O(\log^{4+\varepsilon} q)$ under reasonable hypotheses. As demonstrated in [FGH00], the Satoh-FGH algorithm is much faster in practice in characteristic two. Indeed we were able to count points over much larger fields (up to 8009 bits) than had previously been possible, and could match the largest size reached with SEA (i.e. 1999 bits) in just three hours.

In the following we will describe a method for generating cryptographically suitable curves, over fields of 113 to 571 bits, using an implementation of the Satoh-FGH algorithm combined with an efficient early-abort strategy based on ideas from SEA. In this manner we reduce substantially the time required for curve-generation, finding suitable 200-bit curves in minutes rather than hours on a workstation, for instance.

In section 2, we recall some basic facts about elliptic curves defined over finite fields of characteristic two. Next we review some algorithms that can be used to compute the cardinality of a curve, and in particular we give a description of the Satoh-FGH algorithm. Section 4 gives the conditions that a curve must satisfy in order to be suitable for cryptographic applications. It also describes the early-abort strategy first used by Lercier in [Ler97a] for selecting good curves. Last but not least we describe our implementation and the results we obtained by combining a more aggressive early-abort strategy and the Satoh-FGH algorithm.

2 Elliptic Curves over Finite Fields of Characteristic Two

In this section, we recall some basic facts about elliptic curves defined over $\mathbb{F}_q$ where $q = 2^d$. We will only be concerned with characteristic two. For more informations on elliptic curves, the reader can refer to [Men93], [Sil86], [BSS99].

For our purposes, we can choose the equation of an elliptic curve E (with non-zero j-invariant) to be:

$$E: \ y^2 + xy = x^3 + a_6 \qquad \text{where } a_6 \in \mathbb{F}_q^*.$$

Its *twist* curve is:

$$E^*: \ y^2 + xy = x^3 + a_2x^2 + a_6$$

where a_2 is some fixed element of trace 1.

An important invariant of the curve is its j-invariant $j(E) = 1/a_6$. In the following we assume $j(E) \notin \mathbb{F}_4$ and in particular that curves are ordinary i.e., not supersingular.

The set of points $E(\mathbb{F}_q)$ of the curve is:

$$E(\mathbb{F}_q) = \{(x,y) \in \mathbb{F}_q^2 \,|\, (x,y) \text{ satisfies the equation of } E\} \cup \{\mathcal{O}_E\},$$

where $\mathcal{O}_E$ is the *point at infinity.*

The Frobenius automorphism F is the map $x \mapsto x^q$ on $\mathbb{F}_q$. It can be extended to an endomorphism of E:

$$\begin{aligned} F: \quad & E \to E \\ & (x,y) \mapsto (x^q, y^q) \end{aligned}$$

Its characteristic equation is of the form:

$$F^2 - cF + q = 0.$$

One can show that the number of points on E is

$$N = q + 1 - c, \quad \text{with} \quad |c| \le 2\sqrt{q}$$

where c is the trace of Frobenius on E. The bound on c is due to Hasse [Has33]. Note that $4 \,|\, N$ since the point $(\sqrt[4]{a_6}, \sqrt{a_6})$ on E has order four. The number of points on E^* is $N^* = q + 1 + c$ and one has $2 \parallel N^*$.

The *little* Frobenius automorphism σ is the map $x \mapsto x^2$. It can be extended to an isogeny from E to the conjugate curve $E^\sigma : \ y^2 + xy = x^3 + a_6^2$ as follows:

$$\begin{aligned} \sigma: \quad & E \to E^\sigma \\ & (x,y) \mapsto (x^2, y^2). \end{aligned}$$

3 Counting the Number of Points

3.1 The Schoof-Elkies-Atkin Algorithm

The first polynomial-time algorithm for counting points on elliptic curves over finite fields was described by Schoof in [Sch85]. The basic idea is to find the trace of the curve modulo small primes ℓ by studying the action of F on the ℓ–torsion part of E. Restricting the characteristic equation of F to the ℓ–torsion results in

$$(X^{q^2}, Y^{q^2}) - [q](X, Y) = [c_\ell](X^q, Y^q)$$

for each point (X, Y), where $c_\ell \equiv c \mod \ell$. This equality can be tested, for each candidate $c_\ell \in [0 \ldots \ell - 1]$, by doing polynomial arithmetic modulo the ℓ–division polynomial. Now, it suffices to compute c_ℓ for many small primes ℓ and then to recover the exact result using the Chinese Remainder Theorem. The time required for point-counting over $\mathbb{F}_q$ with this algorithm is $O(\log^{5+\varepsilon} q)$ using asymptotically fast methods for arithmetic (or $O(\log^8 q)$ using naïve arithmetic). The degree of the ℓ–division polynomial is $O(\ell^2)$, which grows quickly and causes this algorithm to be slow in practice.

In large characteristic, Elkies [Elk98] and Atkin [Atk92] improved Schoof's method yielding the so-called SEA algorithm (see [Sch95]) with run-time reduced to $O(\log^{4+\varepsilon} q)$ (or $O(\log^6 q)$) under reasonable hypotheses. Their idea is to construct a factor of degree $O(\ell)$ of the division polynomial and work with it instead. Such a factor can be found by factoring the modular polynomial to find eigenspaces of the Frobenius endomorphism F restricted to $E[\ell]$.

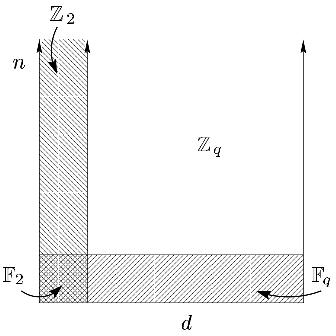
Further work by Morain [Mor95] and others led to practical implementations of SEA for prime fields. Couveignes extended SEA to work in small characteristic using the formal group [Cou94] or the p-torsion [Cou96] and Lercier found an efficient method for characteristic two [Ler97a].

3.2 The Satoh-FGH Algorithm

Here we present our adaptation of Satoh's algorithm to the case of characteristic two. The reader can find more details, including for odd characteristic, in [Sat00] and [FGH00].

The principal idea of this new algorithm is to lift E to a curve $\mathcal{E}$ over a 2–adic ring $\mathbb{Z}_q$ and to compute the trace of the Frobenius on $\mathcal{E}$.

Canonical Lift of the Curve Just as $\mathbb{F}_q$ is obtained from $\mathbb{F}_2$ by taking an algebraic extension modulo an irreducible polynomial $f(x)$, one can obtain $\mathbb{Z}_q$ from the 2–adic integers $\mathbb{Z}_2$ by taking an extension modulo a polynomial $g(x)$ which reduces modulo 2 to $f(x)$. Thus we have $\mathbb{Z}_q = \mathbb{Z}_2[x]/(g(x))$. We represent this situation with the following figure.



A Frobenius morphism $\mathcal{F}$ can also be defined on $\mathbb{Z}_q$. In this case it is not a simple q-th powering operation but something much more complicated. We do not define it explicitly since we will never have to compute it. Similarly, there exists a little Frobenius morphism Σ. For further details on $\mathbb{Z}_q$ and its Frobenius maps, see [Ser68].

A theorem of Lubin, Serre and Tate [LST64] guarantees the existence and uniqueness of a canonical lifted curve $\mathcal{E}$ over $\mathbb{Z}_q$ such that $\mathrm{End}(\mathcal{E}) = \mathrm{End}(E)$, via a canonical lift of the j-invariant. Indeed $J = j(\mathcal{E})$ is characterised by $J \equiv j(E)$ modulo 2 and $\Phi_2(J, \Sigma(J)) = 0$, where Φ_2 is the 2–modular polynomial.

A crucial part of Satoh's contribution is an efficient algorithm for lifting j-invariants. Instead of lifting $j(E)$ in isolation, he suggests lifting the whole cycle of conjugate j's simultaneously. He also proposes considering the duals $\hat{\Sigma}_i$ of the little Frobenius isogenies instead of Σ_i themselves. Indeed the duals are separable and hence are determined by their kernel. After having lifted the j-invariants using Satoh's method, we lift the coefficients of the curves and then compute the kernels by lifting a 2–torsion point on each conjugate curve, using the methods from [FGH00]. As a result, we compute the following diagram:

$$\begin{array}{ccccccccc}
\mathcal{E}_0 & \xrightarrow{\hat{\Sigma}_0} & \mathcal{E}_1 & \xrightarrow{\hat{\Sigma}_1} & \cdots & \xrightarrow{\hat{\Sigma}_{d-2}} & \mathcal{E}_{d-1} & \xrightarrow{\hat{\Sigma}_{d-1}} & \mathcal{E}_0 \\
\downarrow{\scriptstyle \pi} & & \downarrow{\scriptstyle \pi} & & & & \downarrow{\scriptstyle \pi} & & \\
E_0 & \xrightarrow{\hat{\sigma}_0} & E_1 & \xrightarrow{\hat{\sigma}_1} & \cdots & \xrightarrow{\hat{\sigma}_{d-2}} & E_{d-1} & \xrightarrow{\hat{\sigma}_{d-1}} & E_0
\end{array}$$

Here the top row is over $\mathbb{Z}_q$ to precision $O(2^{d/2+o(d)})$ and π is reduction modulo 2 down to $\mathbb{F}_q$.

Computing the Trace in $\mathbb{Z}_q$ Since traces are preserved by taking the dual and by canonical lifting, we have the equation:

$$\mathrm{Tr}(F) = \mathrm{Tr}(\hat{F}) = \mathrm{Tr}(\hat{\mathcal{F}}).$$

Moreover $\hat{\mathcal{F}}$ can be written as the composition

$$\hat{\mathcal{F}} = \hat{\Sigma}_{d-1} \circ \ldots \circ \hat{\Sigma}_1 \circ \hat{\Sigma}_0.$$

To find its trace we go to the formal groups of the curves. In formal groups, isogenies are represented by power series and composing isogenies is done by composing the power series. The first coefficient c_1 of the power series of $\hat{\mathcal{F}}$ is related to its trace as follows:

$$\operatorname{Tr}\hat{\mathcal{F}} = c_1 + \frac{q}{c_1}.$$

Therefore, computing the trace can be done by computing c_1, and the latter can be computed by composing all the power series of the $\hat{\Sigma}_i$. Only the first coefficients g_i of the $\hat{\Sigma}_i$ have to be determined, and this can be done with Vélu's formulae [Vél71]. More precisely, g_i^2 is given by an explicit formula involving the lifted curves and 2–torsion. Taking one of the square roots of $\prod g_i^2$ produces the trace to sufficient precision for it to be recovered exactly using Hasse's bound.

3.3 Description of the Algorithm

In this section, we give a synthetic description of the algorithm. For a more detailed one, we refer the reader to [FGH00]. The general procedure is:

Procedure MainAlgorithm
Input: An elliptic curve E defined over $\mathbb{F}_q$, with $j(E) \notin \mathbb{F}_4$.
Output: The trace of the curve.

1. Compute the cycle of d curves E_i and their j-invariants j_i.
2. Lift all the j_i's simultaneously, yielding J_i.
3. Lift each curve by lifting its a_6 coefficient.
4. Lift the kernel of each $\hat{\Sigma}_i$.
5. Compute the trace from the lifted data.

In this procedure, points 2, 3 and 4 concern the lifting of the cycle of curves and of the kernels. We will detail these first. An essential ingredient is *Newton's iteration* for improving the (2–adic) precision of a root of a function.

Procedure LiftCurvesAnd2Torsion
Input: A cycle of d conjugate curves, and their j-invariants.
Output: The canonical lift of this cycle over $\mathbb{Z}_q$.

1. Lift the j-invariants simultaneously using an adaptation of the Newton iteration to the multivariate case. The function to be considered acts on a $1 \times d$ vector: $\Theta(x_0, \ldots, x_{d-1}) = (\Phi_2(x_0, x_1), \Phi_2(x_1, x_2), \cdots, \Phi_2(x_{d-1}, x_0))$ and the initial approximation of the root is the vector $(j_0, j_1, \ldots, j_{d-1})$ modulo 2.
2. Lift each curve E_i by lifting its a_6 coefficient, yielding A_i, using a Newton iteration with the function $f(x) = 1 + J(x + 432x^2)$ and the initial approximation $-1/J_i$ modulo 16.
3. Lift the 2–torsion point in the kernel of each $\hat{\Sigma}_i$ yielding (X_i, Y_i) on $\mathcal{E}_i$, using a Newton iteration based on the function $f(x) = 8x^3 + x^2 + A_i$ with initial approximation $1/J_{i+1}$ modulo 4.

With these algorithms, one can perform the lifting efficiently. Once this is done, it remains to compute the trace of $\hat{\mathcal{F}}$. The equations in the following algorithm are derived from Vélu's formulae.

Procedure COMPUTETRACE
Input: A cycle of d curves, given by A_i, and 2–torsion abscissae X_i.
Output: The trace of $\hat{\mathcal{F}}$.

1. Compute the square of the first coefficient of the expansion of each $\hat{\Sigma}_i$ in the formal group of $\mathcal{E}_i$ using Vélu's formulae. The result is:
$$g_i^2 = \frac{1 - 252X_i + 19008A_i}{(1 + 120(X_i + 6X_i^2))\,(1 + 864A_{i+1})}.$$
2. Compute $c^2 = \prod g_i^2$.
3. Compute c by computing a square root of c^2 and by determining the sign using $c \equiv 1 \mod 4$.

4 Good Elliptic Curves in Cryptography

The security of elliptic curve cryptosystems depends on the difficulty of solving the elliptic curve discrete logarithm (ECDL) problem. As mentioned in the introduction, there are several attacks against curves with special properties such as the one against trace 1 curves, or the MOV reduction for supersingular curves, etc.

For random curves, the chance that one of these methods can apply is vanishingly small. However there are other attacks that work for generic abelian finite groups.

The first is Pohlig-Hellman reduction [PH78]. When the group order N has all its prime factors small, discrete logs can be computed quickly by working in small subgroups. Thus for good security it is essential to pick a group whose order is divisible by a large prime.

The other attacks are algorithms that run in time $O(\sqrt{N})$. They include Shanks' baby-step giant-step algorithm (see [Coh96]) and Pollard's ρ method [Pol78]. In practice, the most difficult ECDL that has been computed is on a Koblitz curve over $\mathbb{F}_{2^{109}}$ using a distributed version of Pollard–ρ [Har00].

By extrapolating the work required to larger sizes and allowing safety margins for future increases in computing power, it is generally believed (see [FIPS186], [LV00], [P1363], [Sil00]) that a random curve whose order is divisible by a prime of at least 160 bits will offer reasonable security, comparable to 80-bit symmetric systems or 1024-bit RSA. For applications with the highest security requirements, one may take larger safety margins.

To find a secure curve, Lercier [Ler97a] proposed an early-abort strategy to use when computing the cardinality of the curve using SEA. The idea is to test on the fly if $q + 1 - c \equiv 0 \bmod \ell$. If the test is true, then we throw away the curve and try again with another one. Since SEA computes $c \bmod \ell$, this test is easy to implement and costs no extra run-time. In large characteristic where

Satoh-FGH does not apply this is still the best known method and we refer to the existing literature on the subject [LM95], [IKNY98], [MP98].

A difficulty that arises when designing an early-abort strategy to use with the Satoh-FGH algorithm is that $c \bmod \ell$ is not available (except for ℓ a power of p). Our solution is to implement a simplified version of SEA to determine whether the curve has a rational point of ℓ-torsion or not for the first few primes ℓ, as a preliminary step before launching Satoh-FGH. There is a trade-off to be made between the extra cost of these calculations and the benefit to be gained by avoiding an entire cardinality computation. In practice we found this strategy to be very worthwhile and obtained run-times lower than those previously reported in the literature.

5 Implementation and Results

5.1 Implementation Details

We wrote optimised implementations of the early-abort strategy and the Satoh-FGH algorithm for characteristic two, in the C programming language. This implementation of the early-abort strategy is independant of Lercier's one. For multiplication in $\mathbb{F}_q$ we used Karatsuba's algorithm; in $\mathbb{Z}_q$ we used Toom's algorithm. To ensure that modular reduction took very little time, we chose the irreducible polynomial to be a trinomial or pentanomial. For division we used the binary Euclidean algorithm in $\mathbb{F}_q$, and inversion by Newton iterations in $\mathbb{Z}_q$.

Most of our timing tests were run on a 750 MHz EV6 Alpha. In order to compare results with [Ler97a], we also ran some tests on a 266 MHz EV4 Alpha identical to the one Lercier used. Note that the difference between these processors is more than what we could think by just comparing the clock speeds: for usual applications, the gain is by a factor of about 15. Finally we timed curve generation for one small field on a 275 MHz StrongARM chip.

In the early-abort part, as explained below, the most time consuming parts are lazy factorizations of small-degree polynomials over $\mathbb{F}_q$. The most frequent operation is multiplication in $\mathbb{F}_q$. We give relevant timings obtained on the 750 MHz Alpha in Table 1.

Field size	163 bits	193 bits	239 bits	409 bits	571 bits
Cost of a multiplication in $\mathbb{F}_q$	0.488 μs	0.639 μs	0.917 μs	2.632 μs	4.685 μs

Table 1. Cost of a multiplication in $\mathbb{F}_q$ on a 750 MHz EV6 Alpha.

The most frequent operation in the point-counting part is multiplication in $\mathbb{Z}_q$. In Table 2, we give the time for one such operation at the highest 2–adic precision required i.e., $\lceil d/2 \rceil + 3$ bits, for various field sizes d. These measurements were also done on the 750 MHz Alpha.

Base field size	163 bits	193 bits	239 bits	409 bits	571 bits
Maximal precision	85	100	123	208	289
Cost of a multiplication in $\mathbb{Z}_q$	0.19 ms	0.24 ms	0.36 ms	4.6 ms	8.0 ms

Table 2. Cost of a multiplication in $\mathbb{Z}_q$ on a 750 MHz EV6 Alpha.

Field size	SEA (timings from [Ler97b])			Satoh-FGH	Ratio
	Min	Max	Avg		
155 bits	58.8 s	132 s	86.5 s	36.3 s	2.4
196 bits	212 s	1029 s	308 s	68.8 s	4.5
300 bits	1519 s	3686 s	2434 s	408.4 s	6

Table 3. Times for point-counting on a 266 MHz EV4 Alpha

5.2 Counting the Number of Points on One Curve

When computing the cardinality of a curve, one has to decide whether to use SEA or Satoh. Two cases have to be dealt with differently: the case of large characteristic and the case of small characteristic.

The complexity of Satoh's algorithm has a bad dependency in the characteristic p of the base field and when p is large, it is not efficient at all. This is due to the use of the modular equation Φ_p for the lifting of the curves. This equation has $O(p^2)$ coefficients that have to be known at least modulo $p^{(d/2)+O(1)}$. Hence a complexity which is exponential in p appears to be unavoidable. On the other hand, the SEA algorithm is polynomial-time independently of p. For instance, Morain succeeded in counting the number of points of a curve over a field of size $10^{499} + 153$ [Mor95].

However in small characteristic Satoh's algorithm is efficient. In particular in characteristic two, Satoh-FGH is clearly faster than SEA in practice. To illustrate the difference in speed between the two algorithms, we compare Lercier's results [Ler97b] with the timings we get over the same fields, using an identical 266 MHz Alpha. The results are given in Table 3. We do not give minimal or maximal times for Satoh-FGH since the runtime of this algorithm is essentially constant when treating different curves over the same field. These results show that the bigger the field the greater the advantage for Satoh-FGH, as expected from the asymptotics.

We give timings for point-counting on the 750 MHz Alpha in Table 4. Most of the field sizes that we chose are recommended in cryptographic standards (ANSI X9.63, IEEE P1363, IPSec, NIST, WAP).

Remark: In some cases, the SEA and Satoh-FGH algorithms can be combined to speed-up point-counting. This works particularly well when the field size is such that the maximum precision required in Satoh-FGH is a little more than a multiple of the machine word-size. A good example is $q = 2^{251}$: the maximum precision in the lifting calculations is $\lceil \frac{251}{2} \rceil + 3 = 129$ bits. In this case, computing

Field size	Satoh-FGH	Field size	Satoh-FGH	Field size	Satoh-FGH
157 bits	2.39 s	197 bits	4.45 s	283 bits	26.5 s
163 bits	2.76 s	233 bits	6.57 s	409 bits	76.3 s
193 bits	4.10 s	239 bits	6.94 s	571 bits	257 s

Table 4. Times for point-counting on a 750 MHz EV6 Alpha

the trace modulo 3 with the SEA algorithm allows the precision to be reduced to 128 bits which fits perfectly in a whole number of words. This approach could certainly be pushed further, although implementation complexity would appear to outweigh the moderate gain in speed.

5.3 Finding a Good Curve

The naïve strategy to find a curve suitable for cryptographic use is to count the number of points for many curves, until one with almost prime order is found. As mentioned before, if the SEA algorithm is used then many bad curves can be detected early; this nice property does not hold for the Satoh-FGH algorithm.

Hence, for small to medium sizes, the naïve strategy using Satoh-FGH is not better than the early-abort strategy with SEA. For instance over $\mathbb{F}_{2^{155}}$, Lercier [Ler97b] was able to select the good curves among a set of 1000 random ones in 14112 seconds. On the same computer, the Satoh-FGH method takes 36.5 seconds per curve, so that selecting the good ones would take 36500 seconds with the naive strategy, and would be worse by a factor 2.5. (For larger sizes, this phenomenon vanishes and Satoh-FGH is always better.)

To counter this, we take advantage of both methods: we first eliminate many candidate curves by an early-abort strategy based on SEA's techniques, and then run Satoh-FGH on the remaining ones.

Let E be a curve over $\mathbb{F}_q$. For a small prime ℓ, E is called ℓ-good if its order is coprime to ℓ, and ℓ-bad otherwise. Early-abort works as follows for each ℓ:

1. Compute the number of roots of $\Phi_\ell(X, j(E))$. It can be 0, 1, 2 or $\ell+1$. (The cases 1 or $\ell+1$ cannot occur unless q is a square modulo ℓ.)
2. If there are no roots, E is ℓ-good.
3. Otherwise, for each root of Φ_ℓ, build the corresponding factor of the ℓ-division polynomial and search for a root x of the factor. If there is such an x in $\mathbb{F}_q$ and a corresponding y too, then (x, y) is an ℓ-torsion point over $\mathbb{F}_q$ and E is ℓ-bad.
4. Otherwise E is ℓ-good.

The major cost in step 1 is that of computing X^q modulo $\Phi_\ell(X, j(E))$, which has degree $\ell+1$. To accelerate the calculation, we replace Φ_ℓ by the *canonical* modular polynomial Φ_ℓ^c, which has the same degree but is sparser and involves lower powers of j. We refer to [Mor95] for the construction and the properties of these equations.

	$q = 2^{163}$		$q = 2^{239}$	
ℓ	Root finding of $\Phi_\ell^c(X, j)$	Average total time	Root finding of $\Phi_\ell^c(X, j)$	Average total time
3	0.17 ms	0.17 ms	0.28 ms	0.28 ms
5	0.34 ms	0.38 ms	0.61 ms	0.68 ms
7	0.34 ms	1.18 ms	0.56 ms	2.18 ms
11	4.42 ms	6.93 ms	9.14 ms	14.1 ms
13	1.07 ms	4.19 ms	1.94 ms	8.36 ms
17	3.71 ms	8.63 ms	7.34 ms	17.9 ms
19	4.97 ms	11.6 ms	10.1 ms	23.7 ms

Table 5. Average runtime for checking if E is ℓ-good (EV6 – 750 MHz)

Heuristically, in half of the cases there will be no root (in such a case ℓ is called an Atkin prime) and we are done. Otherwise, we have to continue to step 3. The factor of the division polynomial corresponding to a root of the modular polynomial is calculated using a system of formulae due to Lercier [Ler97a]. For small ℓ the solution to this system can be written explicitly, and the factor is obtained at almost no cost. (For larger ℓ the system could be solved efficiently by an algorithm also due to Lercier.) The cost of searching for a root is dominated by the computation of X^q modulo the factor, which has degree $(\ell - 1)/2$.

In Table 5 we give the run-time for this procedure, measured on the 750 MHz Alpha.

It is necessary to bound the maximum size of ℓ in order to balance the cost of early-abort against the gain obtained by avoiding point-counting. In theory, it would be beneficial to increase ℓ until the above early-abort procedure took approximately one ℓ-th of the time required for point-counting. Hence the maximum size of ℓ would grow with the field size.

However almost all of the advantage to be gained comes from using the first few primes and in practice we found $\ell \leq 19$ to be a good trade-off. For these primes it is not difficult to determine if curves are ℓ-good: Lercier's construction of isogenies is relatively easy, as in the search for ℓ-torsion points. Thus we were able to keep our code simple and reliable.

For comparison with Lercier's results reported in [Ler97b], we ran some further tests on the 266 MHz Alpha. We chose a similar early-abort strategy, searching for good curves with order $4p$ without considering the twist curves at all (but see below). The results can be found in Table 6. As a first step in the early-abort, we determine whether the order is divisible by 8. This can be decided very quickly by computing $\operatorname{Tr} a_6$. Note that we measured our timings for 157 and 197 bits instead of 155 and 196 because composite extension fields may be weak in certain cases, as mentioned in the introduction.

Next, in order to maximise the performance of curve generation we decided to search simultaneously for twist curves with order $2p$ and this allowed us roughly to double the speed. As is clear from section 2, the cardinality of the twist can be found immediately from that of the curve itself. Furthermore, the early-abort

Field size	SEA (from [Ler97b])	Satoh-FGH + early-abort
155 bits	14112 s	4490 s
196 bits	30254 s	7850 s

Table 6. Time to select good curves among 1000 (EV4 – 266 MHz)

Field size (in bits)	Time for e.-a. on 10000 curves	Remaining curves	Time to count remaining curves	Good curves	Average time to find a good curve
157	21.1 s	435	17.3 min	45	23.6 s
163	23.1 s	473	21.7 min	55	24.1 s
193	25.1 s	402	27.5 min	33	50.7 s
197	30.8 s	415	30.8 min	43	43.6 s
233	40.3 s	402	44 min	29	92,4 s
239	43.5 s	435	50.3 min	29	105.6 s
283	122 s	418	3h 4 min	20	9.2 min
409	245 s	467	9h 54 min	22	27 min
571	524 s	375	26h 40 min	11	146 min

Table 7. Average time to find a good curve (EV6 – 750 MHz)

strategy can easily be adapted to take the twist into account since it has the same j-invariant and the same division polynomials. (This is because the curve and its twist are isomorphic over an algebraic closure and the isomorphism preserves the abscissae.)

One possibility would be to reject a pair consisting of a curve and its twist only when the early-abort strategy determines that both curves are cryptographically unsuitable. Alternatively one may pursue a more aggressive strategy by rejecting them both as soon as either one is found to be unsuitable, and immediately moving on to a new pair. Using the latter method for 10000 random curve pairs on the 750 MHz Alpha, we measured the timing results shown in Table 7.

Although the $O(d^3)$ space complexity of Satoh's algorithm grows quickly, the tricks described in [FGH00] keep the constant factor small. With these tricks, the largest key size we dealt with (571 bits) requires under 10 megabytes and for moderate key sizes the memory usage was only a few hundred kilobytes. We chose a different trade-off, using more memory in exchange for slightly higher speed.

To investigate the possibility of generating curves in constrained environments, we ran some tests at 113 bits on an ARM chip. This small key size is recommended for key-exchange in the Wireless Application Forum's WTLS standard (WAP) and can be used for short-term security at a level comparable to DES. The results can be seen in Table 8.

Field size	Frequency	Time to count one curve	Average time to find a good curve	RAM + ROM used
113 bits	275 MHz	5.9 s	38 s	240 KB + 136 KB

Table 8. Time to find a good WAP curve on an ARM chip

Field size	Time	Field size	Time	Field size	Time
157 bits	0.50 s	197 bits	0.91 s	283 bits	6.32 s
163 bits	0.56 s	233 bits	1.39 s	409 bits	19.4 s
193 bits	0.84 s	239 bits	1.47 s	571 bits	58.2 s

Table 9. New times for point-counting on a 750 MHz EV6 Alpha

6 Conclusion

The Satoh-FGH algorithm has proven to be the method of choice whenever one wants to compute the cardinality of a random elliptic curve defined over a finite field of characteristic two. But in spite of Satoh-FGH's excellent performance (see Table 4), the SEA algorithm should not be abandoned too quickly. In the case of large characteristic it is the only practical method available. Moreover the early-abort strategy, which is closely related to it, is valuable when looking for a curve for cryptographic use, even in small characteristic. By combining this technique and the Satoh-FGH algorithm, we obtain an efficient way of computing secure curves (see Table 7). We conclude that it is no longer necessary to use precomputed curves in cryptography since one can easily compute new curves as desired. Finding a curve with a security level comparable with RSA-1024 takes minutes or less. Curve generation for short-term security, with a level equivalent to DES, is feasible on a low-power chip. Finally, very high security levels similar to the highest AES level are now possible albeit in several hours.

Remark

We have recently implemented a new and quite different point-counting algorithm with lower memory requirements and a gain in speed by a factor ranging from 4 to 5 depending on key-size. For instance a secure 113-bit curve can be found in 8 seconds using 36 KB of RAM on the 275 MHz StrongARM. Repeating the calculations from Tables 4 and 7 gave the times in Table 9 and Table 10.

Acknowledgements

We would like to thank François Morain for his continuous support and many invaluable suggestions during this work.

We are also grateful to Rajit Manohar from Cornell Computer Systems Laboratory. He provided the computer resources needed for many of our calculations.

Field size (in bits)	Average time to find a good curve
157	5 s
163	5 s
193	10 s
197	10 s
233	21 s
239	22 s
283	138 s
409	7 min
571	34 min

Table 10. New times to find good curves (EV6 – 750 MHz)

References

[FIPS186] FIPS 186-2. Digital Signature Standard. Federal Information Processing Standards publication, january 2000. U.S. Departement of Commerce/National Institute of Standards and Technology. Available at `http://csrc.nist.gov/cryptval/dss.htm`.

[P1363] IEEE P1363. Standard specifications for public key cryptography. Available at `http://www.manta.ieee.org/groups/1363/`.

[Atk92] A. O. L. Atkin. The number of points on an elliptic curve modulo a prime. Series of e-mails to the `NMBRTHRY` mailing list, 1992.

[BSS99] I. Blake, G. Seroussi, and N. Smart. *Elliptic curves in cryptography*, volume 265 of *London Math. Soc. Lecture Note Ser.* Cambridge University Press, 1999.

[Coh96] H. Cohen. *A course in algorithmic algebraic number theory*, volume 138 of *Graduate Texts in Mathematics*. Springer–Verlag, 1996. Third printing.

[Cou94] J.-M. Couveignes. *Quelques calculs en théorie des nombres*. Thèse, Université de Bordeaux I, July 1994.

[Cou96] J.-M. Couveignes. Computing ℓ-isogenies using the p-torsion. In H. Cohen, editor, *Algorithmic Number Theory*, volume 1122 of *Lecture Notes in Comput. Sci.*, pages 59–65. Springer Verlag, 1996. Second International Symposium, ANTS-II, Talence, France, May 1996, Proceedings.

[Dew98] L. Dewaghe. Remarks on the Schoof-Elkies-Atkin algorithm. *Math. Comp.*, 67(223):1247–1252, July 1998.

[DGM99] I. Duursma, P. Gaudry, and F. Morain. Speeding up the discrete log computation on curves with automorphisms. In Kwok Yan Lam, Eiji Okamoto, and Chaoping Xing, editors, *Advances in Cryptology – ASIACRYPT '99*, volume 1716 of *Lecture Notes in Comput. Sci.*, pages 103–121. Springer-Verlag, 1999. International Conference on the Theory and Applications of Cryptology and Information Security, Singapore, November 1999, Proceedings.

[Elk98] N. Elkies. Elliptic and modular curves over finite fields and related computational issues. In D.A. Buell and eds. J.T. Teitelbaum, editors, *Computational Perspectives on Number Theory*, pages 21–76. AMS/International Press, 1998. Proceedings of a Conference in Honor of A.O.L. Atkin.

[FGH00] M. Fouquet, P. Gaudry, and R. Harley. An extension of Satoh's algorithm and its implementation. *J. Ramanujan Math. Soc.*, 15:281–318, 2000.

[FR94] G. Frey and H.-G. Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Math. Comp.*, 62(206):865–874, April 1994.

[GHS00] P. Gaudry, F. Hess, and N. Smart. Constructive and destructive facets of Weil descent on elliptic curves. Submitted to J. Crypt. and available at `http://www.cs.bris.ac.uk/~nigel/weil_descent.html`, 2000.

[GLV] R. Gallant, R. Lambert, and S. Vanstone. Improving the parallelized Pollard lambda search on binary anomalous curves. To appear in *Math. Comp.*

[Har00] R. Harley. `http://cristal.inria.fr/~harley/ecdl7/q`, 2000.

[Has33] H. Hasse. Beweis des Analogons der Riemannschen Vermutung für die Artinschen und F. K. Smidtschen Kongruenzzetafunktionen in gewissen elliptischen Fällen. *Ges. d. Wiss. Narichten. Math.-Phys. Klasse*, pages 253–262, 1933.

[IKNY98] T. Izu, J. Kogure, M. Noro, and K. Yokoyama. Efficient implementation of Schoof's algorithm. In K. Ohta and D. Pei, editors, *Advances in Cryptology – ASIACRYPT '98*, volume 1514 of *Lecture Notes in Comput. Sci.*, pages 66–79. Springer-Verlag, 1998. International Conference on the theory and application of cryptology and information security, Beijing, China, October 1998.

[JM99] D. Johnson and A. Menezes. The elliptic curve digital signature algorithm (ECDSA). Technical Report CORR 99-34, U. Waterloo, 1999. Available at `http://www.cacr.math.uwaterloo.ca/`.

[Kob87] N. Koblitz. Elliptic curve cryptosystems. *Math. Comp.*, 48(177):203–209, January 1987.

[Ler97a] R. Lercier. *Algorithmique des courbes elliptiques dans les corps finis.* Thèse, École polytechnique, June 1997.

[Ler97b] R. Lercier. Finding good random elliptic curves for cryptosystems defined over $\mathbb{F}_{2^n}$. In W. Fumy, editor, *Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *Lecture Notes in Comput. Sci.*, pages 379–392. Springer-Verlag, 1997. International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 1997, Proceedings.

[LM95] R. Lercier and F. Morain. Counting the number of points on elliptic curves over finite fields: strategies and performances. In L. C. Guillou and J.-J. Quisquater, editors, *Advances in Cryptology – EUROCRYPT '95*, volume 921 of *Lecture Notes in Comput. Sci.*, pages 79–94, 1995. International Conference on the Theory and Application of Cryptographic Techniques, Saint-Malo, France, May 1995, Proceedings.

[LST64] J. Lubin, J. P. Serre, and J. Tate. Elliptic curves and formal groups. In *Lecture notes prepared in connection with the seminars held at the Summer Institute on Algebraic Geometry, Whitney Estate, Woods Hole, Massachusetts, July 6-July 31, 1964*, 1964. Scanned copies available at `http://www.ma.utexas.edu/users/voloch/lst.html`.

[LV00] A. Lenstra and E. Verheul. Selecting cryptographic key sizes, January 2000. Presented at PKC2000.

[Men93] A. J. Menezes. *Elliptic curve public key cryptosystems.* Kluwer Academic Publishers, 1993.

[Mil87] V. Miller. Use of elliptic curves in cryptography. In A. M. Odlyzko, editor, *Advances in Cryptology – CRYPTO '86*, volume 263 of *Lecture Notes in Comput. Sci.*, pages 417–426. Springer-Verlag, 1987. Proceedings, Santa Barbara (USA), August 11–15, 1986.

[Mor95] F. Morain. Calcul du nombre de points sur une courbe elliptique dans un corps fini : aspects algorithmiques. *J. Théor. Nombres Bordeaux*, 7:255–282, 1995.

[MOV91] A. Menezes, T. Okamoto, and S. A. Vanstone. Reducing elliptic curves logarithms to logarithms in a finite field. In *Proceedings 23rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 80–89. ACM Press, 1991. May 6–8, New Orleans, Louisiana.

[MP98] V. Müller and S. Paulus. On the generation of cryptographically strong elliptic curves. Preprint, 1998.

[Mül95] V. Müller. *Ein Algorithmus zur Bestimmung der Punktanzahl elliptischer Kurven ber endlichen Körpern der Charakteristik gröer drei.* PhD thesis, University of Saarland, 1995.

[PH78] S. Pohlig and M. Hellman. An improved algorithm for computing logarithms over $\mathrm{GF}(p)$ and its cryptographic significance. *IEEE Trans. Inform. Theory*, IT–24:106–110, 1978.

[Pol78] J. M. Pollard. Monte Carlo methods for index computation mod p. *Math. Comp.*, 32(143):918–924, July 1978.

[SA98] T. Satoh and K. Araki. Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. *Comment. Math. Univ. St. Paul.*, 47:81–92, 1998.

[Sat00] T. Satoh. The canonical lift of an ordinary elliptic curve over a finite field and its point counting. *J. Ramanujan Math. Soc.*, 15:247–270, 2000.

[Sch85] R. Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Math. Comp.*, 44:483–494, 1985.

[Sch95] R. Schoof. Counting points on elliptic curves over finite fields. *J. Théor. Nombres Bordeaux*, 7:219–254, 1995.

[Sem98] I. A. Semaev. Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curves in characteristic p. *Math. Comp.*, 67(221):353–356, January 1998.

[Ser68] J. P. Serre. *Corps locaux.* Hermann, 1968.

[Sil86] J. H. Silverman. *The arithmetic of elliptic curves*, volume 106 of *Graduate Texts in Mathematics.* Springer–Verlag, 1986.

[Sil00] R. Silverman. A cost-based security analysis of symmetric and assymetric key lengths. Bulletin Number 13 of RSA Security, April 2000.

[Skj] B. Skjernaa. Satoh's algorithm in characteristic 2. Copies available at `http://www.imf.au.dk/~skjernaa/`.

[Sma99] N. Smart. The discrete logarithm problem on elliptic curves of trace one. *J. Cryptology*, 12:193–196, 1999.

[Vél71] J. Vélu. Isogénies entre courbes elliptiques. *C. R. Acad. Sci. Paris Sér. I Math.*, 273:238–241, July 1971. Série A.

[vOW99] P. C. van Oorschot and M. J. Wiener. Parallel collision search with cryptanalytic applications. *J. of Cryptology*, 12:1–28, 1999.