University of Bristol

DEPARTMENT OF COMPUTER SCIENCE

How secure are elliptic curves over composite extension fields?

N. P. smart

November 2000 | CSTR-00-017

HOW SECURE ARE ELLIPTIC CURVES OVER COMPOSITE EXTENSION FIELDS?

N.P. SMART

ABSTRACT. We compare the method of Weil descent for solving the ECDLP against the standard method of parallelised Pollard rho. We give details of a theoretical and practical comparison and then use this to analyse the difficulty of actually solving the ECDLP for curves of the size needed in a practical cryptographic systems. In particular we examine the elliptic curves proposed in the Oakley key determination protocol.

1. INTRODUCTION

Ever since its invention, in 1986 by Koblitz [9] and Miller [12], elliptic curve cryptography (ECC) has attracted considerable interest since it enables improved security (compared to conventional systems such as RSA) with the added benefit of smaller key sizes, less bandwidth and less computing power, see [4] for a complete treatment of ECC. Various standards bodies, both government sponsored and industry led, for example NIST [2] and SECG [3], have standardised on elliptic curves defined over fields of the form $\mathbb{F}_{2^p}$ and $\mathbb{F}_p$, where p denotes a prime.

Despite this standardisation effort various people still propose using curves defined over so called composite extension fields, i.e. fields of the form $\mathbb{F}_{q^n}$ where q is some non-trivial power of the characteristic and $n > 1$. Composite extension fields are chosen because they provide greater computational efficiency for what at first glance appears to be the same security. The improved efficiency is particularly pronounced in characteristic two, where one chooses $q = 2^l$ and $n = 4$ or 5. In which case the use of look up tables to represent the subfield of degree 4 or 5 over $\mathbb{F}_2$ can significantly improve the efficiency of the resulting cryptographic scheme.

However, work of Frey, Galbraith, Gaudry, Hess and Smart, see [5], [6] and [8], has cast doubt on the claim that composite extension fields offer about the same security as those fields defined in the standards. This work is based on the technique of Weil descent. Even though the work on Weil descent is now well known in the community there still

appears to be a reluctance to drop composite extension fields in certain quarters.

In this paper we investigate in detail the security of such systems and try to quantify by how much the techniques based on Weil restriction weaken the cryptographic system. We shall concentrate solely on the case of characteristic two, which is important in applications. In section 2 we shall review the method of Weil restriction from [8]. In section 3 we examine in more detail Gaudry's method and explain a very efficient implementation of it. In section 4 we compare Gaudry's method, for the hyperelliptic curves arising from Weil restriction, to the method of Pollard rho on the original elliptic curve. In section 5 we discuss the curve over $\mathbb{F}_{2^{155}}$ proposed in the Oakley key determination protocol. Finally we give some conclusions.

2. The method of Weil restriction

Let $k = \mathbb{F}_q$ denote a finite field of characteristic two and let $K = \mathbb{F}_{q^n}$ denote an extension of degree $n \geq 4$. Suppose we are given an elliptic curve, over K,

$$E : Y^2 + XY = X^3 + \alpha X^2 + \beta,$$

which is suitable for use in cryptography, i.e. $E(K)$ contains a large cyclic subgroup of prime order $s \approx q^n/2$. In particular this means that E must be defined over K and not over some proper subfield since otherwise the order of $E(K)$ would not be almost prime, unless n were prime and $q = 2$. The elliptic curve discrete logarithm problem (ECDLP) for such curves is the following: Given $P, Q \in E(K)$ such that

$$[s]P = [s]Q = \mathcal{O}$$

find $\lambda \in (\mathbb{Z}/s\mathbb{Z})^*$ such that

$$Q = [\lambda]P.$$

Now let H denote a (imaginary quadratic) hyperelliptic curve of genus g, defined over $\mathbb{F}_q$,

$$H : Y^2 + h(X)Y = f(X)$$

where $\deg h(X) \leq g$ and $\deg f(X) = 2g + 1$. The Jacobian of H has about q^g elements and one can also consider a hyperelliptic curve discrete logarithm problem (HCDLP) for such curves. We let the degree-zero divisor D_1 generate some large cyclic subgroup of $\mathrm{Jac}_k(H)$ and let $D_2 \in \langle D_1 \rangle$. The HCDLP is to find the integer λ such that

$$D_2 = [\lambda]D_1.$$

Further details on the hyperelliptic group law and the HCDLP can be found in [10] and [4].

The main result from [8] is the following: From an ECDLP in $E(K)$, i.e. $P_2 = [\lambda]P_1$ with $\lambda \in (\mathbb{Z}/s\mathbb{Z})^*$, one can construct a hyperelliptic curve H of genus g over k and two divisors, D_1 and D_2 of order s in $\mathrm{Jac}_k(H)$ such that

- $g = 2^{m-1}$ or $2^{m-1} - 1$ where $1 \leq m \leq n$.
- $D_2 = [\lambda]D_1$.
- s divides $\#\mathrm{Jac}_k(H)$.

We note that the construction of [8] is very fast and that the genus g is almost always equal to 2^{n-1} for curves of cryptographic interest. There is a small probability that the construction does not actually work in practice, but for real life examples this can usually be ignored.

Why this result is interesting is that it maps the discrete logarithm problem from a group, $E(K)$, where the only known solution has exponential complexity, in the size of q^n, to a group, $\mathrm{Jac}_k(H)$, where the best known solution has sub-exponential complexity, albeit in the size of

$$q^g = q^{2^{n-1}}.$$

However for fixed genus there is an algorithm due to Gaudry which solves the HCDLP in time $O(q^{2+\epsilon})$, which is much better than the algorithm for the equivalent ECDLP which takes time $O(q^{n/2}(\log q)^2)$. In [8] it is argued that for small fixed n, and hence essentially fixed g, this provides evidence for the weakness of the ECDLP on curves defined over composite extension fields, at least asymptotically. However the asymptotic complexity hides a very bad dependence on g, and hence such a conclusion may not be able to be substantiated on curves over field sizes of cryptographic interest. In [8] a single experiment was reported on which involved an elliptic curve over a field of the form $\mathbb{F}_{q^4}$ which gave rise to a hyperelliptic curve of genus four. This experiment was conducted for an elliptic curve which is not typical of elliptic curves over fields of the form $\mathbb{F}_{q^4}$, which would usually give rise to a hyperelliptic curve of genus eight. It is this latter problem that we aim to address here.

3. Analysing and implementing Gaudry's method

We refer to [7] for a detailed explanation of Gaudry's method for the HCDLP. Essentially one takes a factor base of all the degree one prime divisors on H up to the equivalence

$$D_1 \equiv D_2 \text{ if } D_1 = -D_2.$$

This gives approximately $q/2$ such divisors, but one selects by some appropriate means (see [8]) a proportion, say $1/l$, of them. Giving a total factorbase size of

$$F = q/(2l).$$

Then one collects relations amongst the factor base elements by performing a random walk. Once $F+1$ relations have been found one can solve the HCDLP by using a linear algebra technique for finding elements of the kernel of a large sparse matrix over $\mathbb{F}_s$, such as Lanczos [17].

We define the following estimates of the bit-complexity of certain algorithms:

- c_q = Cost of an arithmetic operation in $\mathbb{F}_q$. For fields of cryptographic interest this is given by
$$c_q = (\log q)^2$$
- $c_{q,g}$ = Cost of an operation on a polynomial of degree g over $\mathbb{F}_q$. For fields of cryptographic interest and polynomials of degree $g \leq 32$ the actual methods used have cost, using a Karatsuba style multiplication,
$$c_{q,g} = g^{1.59} c_q.$$
- c_J = Cost of a doubling/addition in the Jacobian of H. By work of [15] this is given by
$$c_J = 22 c_{q,g}.$$
- c_s = Cost of operation in $\mathbb{Z}/s\mathbb{Z}$, for values of s of cryptographic interest namely $s \approx q^n$ we have
$$c_s = (n \log q)^2.$$

Arguing as in [7] one can see that Gaudry's algorithm then takes around

$$F l^g g! c_J$$

bit operations to compute the matrix and then

$$F^2 c_s g$$

bit operations to actually compute an element in the kernel. Here we have assumed, as is born out by experiment, that the operations in the Jacobian dominate the time needed to compute the matrix.

The idea of the parameter l is to balance the time for finding the matrix with the time for solving the matrix. Assuming we have X times more computing power available to perform the relation finding, this gives the equation

$$2 l^{g+1} g! c_J = c_s g q / X.$$

In theory one should choose $X = 1$ but in practice a given organisation probably has more spare idle time available on desk top computers than on a single big server like that needed to run the matrix step. When $X = 1$ this means we should choose our proportion of good divisors as

$$l \approx \left(\frac{n^2 q}{44 g! g^{0.59}} \right)^{1/(g+1)} = \ell.$$

But since we must have $l \geq 1$, we shall choose $l \approx \min(1, \ell)$. In particular this means that the overall complexity of the attack on the ECDLP based on Weil descent, is given by

$$\begin{aligned} C &= \frac{(qn\ln(q))^2 g}{4} \left(\frac{n^2 q g^{-0.59}}{44 g!} \right)^{(-2/(g+1))} \\ &= (qn\ln(q))^2 2^{n-3} \left(\frac{n^2 q \left(2^{n-1}\right)^{-0.59}}{(44 \cdot 2^{n-1})!} \right)^{\left(-2/(2^{n-1}+1)\right)}, \end{aligned}$$

since $g \leq 2^{n-1}$. Therefore, for fixed n we obtain a complexity of

$$O(q^{2+\epsilon}(\log q)^2).$$

We implemented Gaudry's algorithm with the following optimisations

- The field arithmetic in $\mathbb{F}_q$ was implemented using very fast hand coded loops for the particular finite fields we where interested in, namely $q = 2^i$ with $i \leq 31$. This on its own provided nearly a 200% improvement in performance.
- The polynomial code was also optimised heavily for the case where the polynomials have degree less than twenty, using Karatsuba type techniques.
- The linear algebra step was run using the code used in the McCurley challenge [18]. We thank T. Denny and D. Weber for allowing us to use this code. This was run on a machine with 6 processors and 8GB of RAM running HP-UX.

4. Comparison with Pollard rho

To have something concrete to compare the method of Weil descent to, we implemented the parallel version of Pollard's rho method [16] for the ECDLP. We used the method of distinguished points due to Wiener and van Oorschot [13] which has been used in recent years to solve various challenge ECDLP examples set by Certicom.

Since we are using elliptic curves defined over fields of the form $\mathbb{F}_{q^n}$ where $n = 4$ or 5 we implemented very efficient techniques for these fields, using lookup tables for the subfields of degree 4 or 5 over $\mathbb{F}_2$. In table 1 we give the time needed to solve an elliptic curve discrete

logarithm problem on various elliptic curves over $\mathbb{F}_{q^4}$. This was for an implementation on a network of 80 Sun Sparc-5 and Sparc-10s, for comparison we also give the time to run the program on a single Sparc-10.

TABLE 1. Pollard rho for $E(\mathbb{F}_{q^4})$

q	2^7	2^{11}	2^{13}	2^{17}	2^{19}	2^{21}
80 Sparcs	00:00	00:00	00:06	38:32	≈11d	≈621d
Single Sparc	00:00	00:11	04:50	≈38d	≈3y	≈71y

Times are given either in the format *hrs:mins* rounded to the nearest minute, or in the format xd or xy to denote a certain number of days or years. A $\approx$ in the table denotes an approximate run time deduced from running the program for a reasonable length of time and then calculating the expected run time from this empirical data. One should note that since the rho method is heuristic in nature the running times represent an average for the small values of q.

In tables 2 and 3 we give the run times for Gaudry's algorithm using the same set of 80 Sun Sparc-5 and Sparc-10s to compute the matrix, we also give the estimate of the time needed for a single Sparc-10 to compute the matrix. We also give the time needed for the matrix step using a HP-UX machine which had 8 GBytes of RAM. These times should be compared to the time needed to solve the equivalent problem on the elliptic curve using Pollard rho.

TABLE 2. Numerical data for $n = 4$ and $g = 4$

q	2^7	2^{11}	2^{13}	2^{17}	2^{19}	2^{21}
$\min(1, \ell)$	1	1.6	2.2	3.8	5.1	6.78
l used	1	2	2	4	4	8
$F = \#FB$	65	513	2049	16428	65537	131283
Time for relation step						
80 Sparcs	00:00	00:00	00:01	00:55	05:15	68:00
Single Sparc	00:00	00:02	00:10	16:50	70:00	≈115d
Time for matrix step	00:00	00:00	00:01	00:06	02:10	13:00

We first examine the case of $n = 4$ and $g = 4$, this case occurs for around $1/q$ of all elliptic curves defined over the field $\mathbb{F}_{q^4}$. As can be seen from the table the method of Weil descent provides a far more efficient way of attacking such elliptic curves than the standard method of Pollard rho for all values of q.

TABLE 3. Numerical data for $n = 4$ and $g = 8$

q	2^7	2^{11}	2^{13}	2^{17}	2^{19}
$\min(1, \ell)$	1	1	1	1	1.03
l used	1	1	1	1	1
$\#F$	64	1024	4096	65536	262144
Time for relation step					
80 Sparcs	00:05	01:20	05:45	43:45	≈8d
Single Sparc	01:30	19:20	95:10	≈62d	≈250d
Time for matrix step	00:00	00:00	00:02	31:00	≈20d

For the case $n = 4$ and $g = 8$, which is the most common case for elliptic curve systems over fields of the form $\mathbb{F}_{q^4}$, we see that the cross over point between Pollard rho and the method of Weil descent occurs at a value of q just over 2^{17}. This, therefore, provides the missing evidence from [8] that all curves over fields of composite extension degree divisible by four should be avoided in cryptographic applications.

Hence we now have a complete experimental treatment of the case $n = 4$ in the method of Weil descent. The next case to consider is $n = 5$, which in fact turns out to be the most interesting in practical applications. In the next section we turn to this case.

5. The Oakley 'Well-Known Groups' 3 and 4

In [1] two elliptic curve groups are proposed for use in a key agreement protocol used as part of the IPSEC set of protocols. These groups, denoted 'Well-Known Group' 3 and 'Well-Known Group' 4, are defined as elliptic curves over fields of composite degree over $\mathbb{F}_2$. The first group is defined over the field $\mathbb{F}_{2^{155}}$, whilst the second is defined over the field $\mathbb{F}_{2^{185}}$. Since the extension degree of these fields over $\mathbb{F}_2$ are composite it is an open question as to whether these curves should still be used within the IPSEC family of protocols. In this section we shall concentrate solely on group 3.

Group 3 is defined by the equation

$$Y^2 + XY = X^3 + \beta$$

where

$$\beta = \omega^{18} + \omega^{17} + \omega^{16} + \omega^{13} + \omega^{12} + \omega^9 + \omega^8 + \omega^7 + \omega^3 + \omega^2 + \omega + 1,$$

where $\omega^{155} + \omega^{62} + 1 = 0$. This has group order

$$E(\mathbb{F}_{2^{155}}) = 12 \cdot 3805993847215893016155463826195386266397436443.$$

We carried out a number of experiments on elliptic curves over fields of the form $\mathbb{F}_{q^5}$. For the Pollard rho method, using the various optimisations available in such fields, we obtained the times in Table 4. Extrapolating our experimental results on the Pollard rho algorithm to

TABLE 4. Pollard rho for $E(\mathbb{F}_{q^5})$

q	2^7	2^{11}	2^{13}	2^{17}	2^{19}
80 Sparcs	00:00	00:06	06:30	≈376d	≈41y
Single Sparc	00:00	02:05	≈20d	≈58y	≈4000y

'Well Known Group' 3, it would appear that we would require

$$10^{11} \text{ years}$$

to solve the discrete logarithm problem using our network of 80 Sparc 5 and Sparc 10 computers, or

$$10^{13} \text{ years}$$

using a single Sparc-10. Hence it is clearly currently infeasible to attack this curve using the Pollard rho algorithm.

We now turn out attention to whether it is feasible to attack 'Well Known Group' 3 using techniques based on Weil descent. Applying the method of [8] to this curve we obtain the hyperelliptic curve

$$\begin{aligned} H: \quad y^2 \quad & + y\left(\begin{array}{l} 1258097243x^{16} + 1177011841x^8 + 540379308x^4 \\ +1555798523x^2 + 613019365x \end{array}\right) \\ & + 558654746x^{33} + 1390366357x^{32} + 577010024x^{28} \\ & + 1211700991x^{26} + 2017104043x^{25} + 1674361774x^{24} \\ & + 993950732x^{22} + 1777282797x^{21} + 1982857394x^{20} \\ & + 144558341x^{19} + 693983331x^{18} + 1937134056x^{16} \\ & + 1947274294x^8 + 31687647x^4 + 1217310851x^2 + 493932675x \end{aligned}$$

defined over the field $\mathbb{F}_{2^{31}}$, where $w^{31} + w^3 + 1 = 0$ and the curve H has genus 16. In the above equation to convert the decimal coefficients to field elements one should first convert the decimal to binary, and then use the binary representation to define the polynomial in w which gives the corresponding field element. For example

$$\begin{aligned} 1258097243 \quad \equiv \quad & w^{30} + w^{27} + w^{25} + w^{23} + w^{22} + w^{21} + w^{20} + w^{19} \\ & + w^{18} + w^{16} + w^{11} + w^9 + w^6 + w^4 + w^3 + w + 1 \end{aligned}$$

In our experiments using curves of genus 16 we found that it would take over three years for the network of 80 workstations to compute a single relation for a curve over a field of size 2^7. Hence, it makes very little sense to extrapolate from actual run times for Gaudry's algorithm.

However, we can give a rough estimate as to how long it would take to perform the two steps for the curve over $\mathbb{F}_{2^{155}}$ considered above.

Firstly we note that for such a curve we would take $l = 1$ and hence the factor base would have size,

$$F \approx 2^{30}.$$

This on its own would imply that the matrix step would require around

$$10^7 \text{ years}$$

to process using the code used to produce the examples in the last section. To produce the matrix we estimate would take the network of 80 Sparcs over

$$10^{10} \text{ years.}$$

Hence, although the method of Weil descent would appear to produce a more efficient way to attack systems based on 'Well Known Group 3', it would appear that such curves are secure. However, this assumes there is no further algorithmic improvements in either the method of Weil descent or the method of Gaudry for solving HCDLP.

6. Conclusion

The 'Well Known Groups' 3 and 4 in IPSEC may still be considered secure, however they are made less secure by the method of Weil descent. This does not pose an immediate threat, but future algorithmic improvements could render them insecure. It should be noted that since both Weil descent and Gaudry's algorithm are comparatively recent advances one cannot rule out further algorithmic improvements in the coming years.

For large genus the method of Gaudry will only be asymptotically better than Pollard rho, as $q \to \infty$. This is due to the bad dependence on g. For values of g where $g >> n$ the current techniques of Weil descent produce a major problem, namely the ECDLP is in a group of order q^n, whilst using Weil descent we have mapped it into a subgroup (of order q^n) of a group of order

$$q^g = q^{2^{n-1}}.$$

Hence we seem to have made our problem more difficult. It may be that the best algorithm for the HCDLP in this setting may be the ones which have asymptotic complexity

$$O\left(L_{q^g}(1/2, c)\right) = O\left(\exp((c + o(1))\sqrt{(\log q^g)(\log\log q^g)})\right)$$

as q is fixed and $g \to \infty$. However, there has been little work on practical implementations of these methods, the only one in the literature

being described in [14]. The algorithm in [14] does not appear practical for the curve which arose above when we considered the Oakley group.

We end by stating that for curves over characteristic two fields of size 2^p, where p is prime, the method of Weil descent does not apply. This was already pointed out in the paper [8] but further explanation is given in a paper of Menezes and Qu, [11].

References

[1] IETF. The Oakley Key Determination Protocol. *IETF RFC 2412*, Nov 1998.

[2] NIST. FIPS PUB 186-2 : DIGITAL SIGNATURE STANDARD (DSS). *National Institute for Standards and Technology*, 2000.

[3] SECG. SEC 2: Recommended Elliptic Curve Domain Parameters. *Standards for Efficient Cryptography Group*, 1999.

[4] I.F. Blake, G. Seroussi and N.P. Smart. *Elliptic curves in cryptography.* Cambridge University Press, 1999.

[5] G. Frey. How to disguise an elliptic curve. Talk at Waterloo workshop on the ECDLP, 1998. **http://cacr.math.uwaterloo.ca/conferences/1998/ecc98/slides.html**

[6] S.D. Galbraith and N.P. Smart. A cryptographic application of Weil descent. *Cryptography and Coding, 7th IMA Conference*, Springer-Verlag, LNCS 1746, 191–200, 1999. The full version of the paper is *HP Labs Technical Report, HPL-1999-70.*

[7] P. Gaudry. An algorithm for solving the discrete logarithm problem on hyperelliptic curves. In *Advanced in cryptology - EUROCRYPT 2000*, Springer-Verlag LNCS 1807, 19–34, 2000.

[8] P. Gaudry, F. Hess and N.P. Smart. Constructive and destructive facets of Weil descent on elliptic curves. *Preprint*, 2000.

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

[10] N. Koblitz. Hyperelliptic cryptosystems. *J. Crypto.*, **1**, 139–150, 1989.

[11] A. Menezes and M. Qu. Analysis of the Weil Descent Attack of Gaudry, Hess and Smart. *Preprint*, 2000.

[12] V. Miller. Use of elliptic curves in cryptography. In *Advances in Cryptology, CRYPTO - '85*, Springer LNCS 218, 47–426, 1986.

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

[14] S. Paulus. An algorithm of sub-exponential type computing the class group of quadratic orders over principal ideal domains. In *ANTS-2: Algorithmic Number Theory*, Springer-Verlag, LNCS 1122, 243–257, 1996.

[15] S. Paulus and A. Stein. Comparing real and imaginary arithmetics for divisor class groups of hyperelliptic curves. In *ANTS-3: Algorithmic Number Theory*, Springer-Verlag, LNCS 1423, 576–591, 1998.

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

[17] J. Teitelbaum. Euclid's algorithm and the Lanczos method over finite fields. *Math. Comp.*, **67**, 1665-1678, 1998.

[18] D. Weber and T. Denny. The solution of McCurley's discrete log challenge. In *Advanced in cryptology - CRYPTO '98*, Springer-Verlag LNCS 1462, 458–471, 1998.

Dept. Computer Science, University of Bristol, Merchant Venturers Building, Woodland Road, Bristol, BS8 1UB

E-mail address: nigel@cs.bris.ac.uk