Solving Elliptic Curve Discrete Logarithm Problems Using Weil Descent

Michael Jacobson
University of Manitoba
jacobs@cs.umanitoba.ca

Alfred Menezes
Certicom Research & University of Waterloo
ajmeneze@uwaterloo.ca

Andreas Stein
University of Illinois
andreas@math.uiuc.edu

May 16, 2001

Abstract

We provide a concrete instance of the discrete logarithm problem on an elliptic curve over $\mathbb{F}_{2^{155}}$ which resists all previously known attacks, but which can be solved with modest computer resources using the Weil descent attack methodology of Frey. We report on our implementation of index-calculus methods for hyperelliptic curves over characteristic two finite fields, and discuss the cryptographic implications of our results.

1 Introduction

Let E be an elliptic curve defined over a finite field $K = \mathbb{F}_{q^n}$. The elliptic curve discrete logarithm problem (ECDLP) in $E(K)$ is the following: given E, $P \in E(K)$, $r = \mathrm{ord}(P)$ and $Q \in \langle P \rangle$, find the integer $s \in [0, r-1]$ such that $Q = sP$. The ECDLP is of interest because its apparent intractability forms the basis for the security of elliptic curve cryptographic schemes.

The elliptic curve parameters have to be carefully chosen in order to circumvent some known attacks on the ECDLP. In order to avoid the Pohlig-Hellman [34] and Pollard's rho [35, 32] attacks, r should be a large prime number. To avoid the Weil pairing [27] and Tate pairing [13] attacks, r should not divide $q^{ni} - 1$ for each $1 \le i \le C$, where C is large enough so that it is computationally infeasible to find discrete logarithms in $\mathbb{F}_{q^{nC}}$. Finally, the curve should not be $\mathbb{F}_{q^n}$-anomalous (i.e., $\#E(\mathbb{F}_{q^n}) \neq q^n$) in order to avoid the attack of [36, 37, 38]. For the remainder of this paper, we assume that the elliptic curve parameters satisfy these conditions. In particular, we assume that $r \approx q^n$.

Frey [11, 12] first proposed using Weil descent as a means to reduce the ECDLP in elliptic curves over characteristic two finite fields $\mathbb{F}_{q^n}$ to the discrete logarithm problem in an abelian variety over the smaller field $\mathbb{F}_q$. Frey's method, which we refer to as the *Weil descent attack* *methodology*, was further elaborated by Galbraith and Smart [14]. In 2000, Gaudry, Hess and Smart (GHS) [17] showed how Frey's methodology could be used to reduce any instance of the ECDLP to an instance of the discrete logarithm problem in the Jacobian of a hyperelliptic curve over $\mathbb{F}_q$. Since subexponential-time algorithms for the latter problem are known, this could have important implications to the security of elliptic curve cryptographic schemes.

In this paper, we focus our attention on determining the practicality of the GHS method for solving the ECDLP in elliptic curves over $\mathbb{F}_{2^{155}}$. We offer two justifications for this restriction. First, as proven in [29], the GHS attack is certain to fail for *all* elliptic curves defined over $\mathbb{F}_{2^n}$ where n is a prime in the interval $[160, 600]$. Second, a specific elliptic curve over $\mathbb{F}_{2^{155}}$ is one of the two elliptic curves allowed in an IETF standard [21] for key establishment (the other elliptic curve is defined over $\mathbb{F}_{2^{185}}$).

The remainder of the paper is organized as follows. §2 provides a brief introduction to the relevant theory of hyperelliptic curves. The Weil descent attack methodology of Frey and the GHS attack are described in §3. An overview of index-calculus algorithms for solving the hyperelliptic curve discrete logarithm problem is presented in §4, and a report of our implementation for hyperelliptic curves over characteristic two finite fields is given in §5. The cryptographic implications of our results are discussed in §6. Our conclusions are stated in §7.

2 Hyperelliptic Curves

We provide a brief overview of the theory of hyperelliptic curves that is relevant to this paper. For a more detailed (but elementary) exposition, see [30].

HYPERELLIPTIC CURVES. Let $k = \mathbb{F}_q$ denote the finite field of order q. The *algebraic closure* of $\mathbb{F}_q$ is $\overline{k} = \bigcup_{n \ge 1} \mathbb{F}_{q^n}$. A *hyperelliptic curve C of genus g over k* is defined by a non-singular equation

$$v^2 + h(u)v = f(u),$$

where $h, f \in k[u]$, $\deg f = 2g+1$, and $\deg h \le g$. Let L be an extension field of k. The set of *L-rational points* on C is $C(L) = \{(x,y) : x, y \in L,\ y^2 + h(x)y = f(x)\} \cup \{\infty\}$. The *opposite* of $P = (x,y) \in C(L)$ is $\widetilde{P} = (x, -y - h(x))$; we also define $\widetilde{\infty} = \infty$. Note that $\widetilde{P} \in C(L)$. There is no natural group law on the set of points $C(L)$[1]. Instead, one considers the Jacobian of C over k which is a finite group.

JACOBIAN OF A HYPERELLIPTIC CURVE. The set D^0 of *zero divisors* of C is the set of formal sums $\sum_{P \in C(\overline{k})} m_P P$, where $m_P \in \mathbb{Z}$ and only a finite number of the m_P's are non-zero. D^0 is a group under the addition rule $\sum m_P P + \sum n_P P = \sum (m_P + n_P) P$. Let $\sigma : \overline{k} \to \overline{k}$ be the *Frobenius map* defined by $x \mapsto x^q$. The map σ extends to $C(\overline{k})$ by $(x,y) \mapsto (x^\sigma, y^\sigma)$ and $\infty^\sigma \mapsto \infty$, and to D^0 by $\sum m_P P \mapsto \sum m_P P^\sigma$. The set of zero divisors defined over k is $D^0_k = \{D \in D^0 : D^\sigma = D\}$. The *function field* of C over k, denoted $k(C)$, is the field of fractions of the integral domain of polynomial functions $k[u,v]/(v^2 + h(u)v - f(u))$. For $f \in k(C)$, the *divisor of f* is $\mathrm{div}(f) = \sum_{P \in C(\overline{k})} v_P(f) P$,

[1]Except for the case $g = 1$, since a genus 1 hyperelliptic curve is precisely an elliptic curve.

where $v_P(f)$ denotes the multiplicity of P as a root of f. Now the set $\mathrm{Prin}_k = \{\mathrm{div}(f) : f \in k(C)\}$ is a subgroup of D_k^0. The *Jacobian* of C (over k) is the quotient group $J_C(k) = D_k^0/\mathrm{Prin}_k$.

PROPERTIES OF THE JACOBIAN. $J_C(k)$ is a finite group. A theorem of Weil's implies that $(\sqrt{q}-1)^{2g} \le \#J_C(k) \le (\sqrt{q}+1)^{2g}$ so $\#J_C(k) \approx q^g$. If D_1 and D_2 are in the same equivalence class of divisors in $J_C(k)$ we write $D_1 \sim D_2$. Each equivalence class has a unique divisor in *reduced form*, i.e., a divisor $\sum_{P\neq\infty} m_P P - (\sum_{P\neq\infty} m_P)\infty$ satisfying (i) $m_P \ge 0$ for all P; (ii) if $m_P \ge 1$ and $P \neq \widetilde{P}$, then $m_{\widetilde{P}} = 0$; (iii) $m_P = 0$ or 1 if $P = \widetilde{P}$; and (iv) $\sum m_P \le g$. Such a *reduced divisor* D can be uniquely represented by a pair of polynomials $a, b \in k[u]$ where (i) $\deg b < \deg a \le g$; (ii) a is monic; and (iii) $a|(b^2 + bh - f)$. We write $D = \mathrm{div}(a,b)$ to mean $D = \gcd(\mathrm{div}(a), \mathrm{div}(b - v))$ where the gcd of two divisors $\sum m_P P$ and $\sum n_P P$ is defined to be $\sum \min(m_P, n_P) P$. The *degree* of D is $\deg a$. Cantor's algorithm [5] can be used to efficiently compute the sum of two reduced divisors, and express the sum in reduced form.

3 Weil Descent Attack

Let l and n be positive integers. Let $q = 2^l$, and let $k = \mathbb{F}_q$ and $K = \mathbb{F}_{q^n}$. Consider the (non-supersingular) elliptic curve E defined over K by the equation

$$E \;:\; y^2 + xy = x^3 + ax^2 + b, \qquad a \in K,\, b \in K^*.$$

We assume that $\#E(K) = dr$ where d is small (e.g., $d = 2$ or $d = 4$) and r is prime. Hence $r \approx q^n$. Let $b_i = \sigma^i(b)$, where $\sigma : K \to K$ is the Frobenius automorphism defined by $\alpha \mapsto \alpha^q$, and define

$$m(b) = \dim_{\mathbb{F}_2}(\mathrm{Span}_{\mathbb{F}_2}\{(1, b_0^{1/2}), \ldots, (1, b_{n-1}^{1/2})\}). \tag{1}$$

Assume now that either n is odd, or $m(b) = n$, or $\mathrm{Tr}_{K/\mathbb{F}_2}(a) = 0$. Gaudry, Hess and Smart [17] showed how Weil descent can be used to reduce the ECDLP problem in the subgroup of order r of $E(K)$ to the discrete logarithm problem in a subgroup of order r of the Jacobian $J_C(k)$ of a hyperelliptic curve C of genus g defined over k. One first constructs the Weil restriction $W_{E/k}$ of scalars of E, which is an n-dimensional abelian variety over k. Then, $W_{E/k}$ is intersected with $n-1$ hyperplanes to obtain the hyperelliptic curve C. We call their reduction algorithm the *GHS attack* on the ECDLP. The genus g of C is either 2^{m-1} or $2^{m-1}-1$, where $m = m(b)$.

The discrete logarithm problem in the subgroup of order r in $J_C(k)$ can be solved using Pollard's rho algorithm [35, 32] which has an expected running time of $O(g^2 q^{n/2} \log^2 q/M)$ bit operations where M is the number of processors available for a parallel attack. However, since the group operation in $E(K)$ can be performed faster than the group operation in $J_C(k)$, it is more efficient to apply Pollard's rho algorithm directly in $E(K)$. The other alternative is to use index-calculus algorithms (see §4). These algorithms have subexponential running time for large genus curves, and therefore may be more efficient than Pollard's rho algorithm for some parameters of practical interest.

In order for the GHS attack to be successful in solving the ECDLP in $E(K)$, the discrete logarithm problem in $J_C(k)$ should be tractable using the known index-calculus algorithms. Note that $1 \leq m \leq n$. In general, $m \approx n$ whence $g \approx 2^{n-1}$ and $\#J_C(k) \approx q^{2^{n-1}}$ and the GHS attack fails. The GHS attack will only succeed if m is small, say $m \approx \log_2 n$, because then $g \approx n$ and $\#J_C(k) \approx q^n$. The formula (1) was analyzed in [29], and the following result was obtained for the case n prime.

Theorem 1 ([29]) Let n be an odd prime, let t be the multiplicative order of 2 modulo n, and let $s = (n-1)/t$. Then

(i) $x^n + 1$ factors over $\mathbb{F}_2$ as $(x+1)f_1 f_2 \cdots f_s$, where the f_i's are distinct irreducible polynomials of degree t.

(ii) Let $\sigma : \mathbb{F}_{q^n} \to \mathbb{F}_{q^n}$ be the Frobenius map defined by $x \mapsto x^q$. Define $B = \{b \in \mathbb{F}_{q^n} : (\sigma+1)f_i(\sigma)(b) = 0 \text{ for some } 1 \leq i \leq s\}$, and let $a \in \mathbb{F}_{q^n}$ be an element of trace 1. Then for all $b \in B$, the elliptic curves $y^2 + xy = x^3 + b$ and $y^2 + xy = x^3 + ax^2 + b$ have $m(b) = t+1$.

(iii) The cardinality of the set B is $qs(q^t - 1)$.

Consider now the case $q = 2^5$ and $n = 31$ (so $q^n = 2^{155}$). We have $t = 5$ and $s = 6$. It follows from Theorem 1 that there are approximately 2^{32} elliptic curves over $\mathbb{F}_{2^{155}}$ for which the GHS attack efficiently reduces the ECDLP to the DLP in the Jacobian of a genus 31 or 32 hyperelliptic curve defined over $\mathbb{F}_{2^5}$. In §5 we provide convincing evidence that the latter problem is quite tractable, which means that the original ECDLP is also tractable. The next section provides an overview of index-calculus methods for the hyperelliptic curve discrete logarithm problem.

4 Index-Calculus Methods

Problem Definition. Let C be a genus g hyperelliptic curve over $k = \mathbb{F}_q$. The hyperelliptic curve discrete logarithm problem (HCDLP) is the following: given C, $D_1 \in J_C(k)$, $r = \text{ord}(D_1)$, and $D_2 \in \langle D_1 \rangle$, find the integer $s \in [0, r-1]$ such that $D_2 = sD_1$. We shall assume that r is prime, and $\#J_C(k) \approx r$.

Index-Calculus Methods for HCDLP. Adleman, DeMarrais and Huang (ADH) [1] presented the first index-calculus algorithm for solving the HCDLP. Their algorithm was described for the case q an odd prime, and was later extended by Bauer [3] to arbitrary q. The (heuristic) expected running time of the ADH algorithm is $L_{q^{2g+1}}[c]$ for $g \to \infty$ and $\log q \leq (2g+1)^{0.98}$, where $c < 2.313$ and $L_n[c] = O(\exp((c+o(1))\sqrt{\log n \log\log n}))$. The algorithm does not assume that the group order $\#J_C(k)$ is known, necessitating an expensive Smith Normal Form computation on a sparse integer matrix. Index-calculus algorithms with rigorously proved running times were presented by Müller, Stein and Thiel [31] and Enge [7]. Their algorithms have an expected running time of $L_{q^{2g+1}}[1.44]$ and are superior, both in theory and in practice, to the ADH algorithm.

Gaudry [16], building on earlier work of Adleman, DeMarrais and Huang [1] and Hafner and McCurley [18], presented an algorithm specifically suited for very small genus curves. Gaudry's algorithm has an expected running time of $O(g^3 q^2 \log^2 q + g^2 g! q \log^2 q)$ bit operations. It becomes impractical for large genera, e.g., $g \geq 10$, because of the large multiplicative factor $g!$. Gaudry's algorithm was extended and analyzed by Enge and Gaudry [8]. The extended algorithm has an expected running time of $L_{q^g}[\sqrt{2}] = L_{q^{2g+1}}[1]$ bit operations for $g/\log q \to \infty$. The primary reason for the improved running time over the ADH algorithm is that the order and structure of $J_C(k)$ is assumed to be known, whereby one only needs to solve a sparse system of equations modulo r instead of an expensive Smith Normal Form computation.

It is the Enge-Gaudry index-calculus algorithm that we describe and have implemented. We first need to introduce the notions of a prime divisor and a smooth divisor.

PRIME DIVISORS. A reduced divisor $D = \text{div}(a, b) \in J_C(k)$ is called a *prime divisor* if a is irreducible over k. The set of all prime divisors of degree $\leq t$ can be found as follows. For each monic irreducible polynomial $a \in k[u]$ of degree $\leq t$, find the roots of $v^2 + h(u)v - f(u)$ modulo $a(u)$. For each root $b(u)$ (there are either 0, 1 or 2 such roots), $\text{div}(a, b)$ is a prime divisor.

SMOOTH DIVISORS. A reduced divisor $D = \text{div}(a, b) \in J_C(k)$ can be efficiently expressed as a sum of prime divisors as follows. First factor a into monic irreducibles over k: $a = a_1^{e_1} a_2^{e_2} \cdots a_L^{e_L}$. Let $b_i = b \bmod a_i$ for $1 \leq i \leq L$. Then $D = \sum_{i=1}^{L} e_i \text{div}(a_i, b_i)$. D is said to be *t-smooth* if $\max\{\deg a_i\} \leq t$.

ENGE-GAUDRY INDEX-CALCULUS ALGORITHM. The main ideas of the Enge-Gaudry index-calculus algorithm are the following. First build a factor base $S = \{P_1, P_2, \ldots, P_w\}$ consisting of all prime divisors of degree $\leq t$ for some bound t. One then performs a random walk (á la Teske [41]) in the set of reduced divisors equivalent to divisors of the form $\alpha D_1 + \beta D_2$ and stores the t-smooth divisors encountered in this walk—each t-smooth divisor yields a relation $\alpha_i D_1 + \beta_i D_2 \sim R_i = \sum_j e_{ij} P_j$. When $w + 1$ different relations have been found, one can find by linear algebra modulo r a non-trivial linear combination $\sum_{i=1}^{w+1} \gamma_i (e_{i1}, e_{i2}, \ldots, e_{iw}) = (0, 0, \ldots, 0)$. Thus $\sum_{i=1}^{w+1} \gamma_i R_i = 0$, whence $\sum \gamma_i (\alpha_i D_1 + \beta_i D_2) = 0$ and $\log_{D_1} D_2 = -(\sum \gamma_i \alpha_i)/(\sum \gamma_i \beta_i) \bmod r$.

5 Implementation Results

Our implementation was done in C++ using Victor Shoup's NTL library.

5.1 Implementation Details

We provide some details of our implementation of the Enge-Gaudry index-calculus method for solving the HCDLP in the Jacobian of genus 31 hyperelliptic curves over $k = \mathbb{F}_q$ for $q = 4$, 8, 16 and 32. The hyperelliptic curves over these fields are denoted C62, C93, C124 and C155. They all have $\#J_C(k) = 2r$ where r is prime. The hyperelliptic curves were obtained by applying the GHS attack to an instance of the ECDLP on elliptic curves E62, E93, E124 and E155 over $\mathbb{F}_{2^{62}}$, $\mathbb{F}_{2^{93}}$, $\mathbb{F}_{2^{124}}$ and $\mathbb{F}_{2^{155}}$, respectively. The elliptic curve and hyperelliptic curve parameters are presented in Table 1. See Appendix A for an example of how the elliptic curves were selected, and how the GHS attack was used to reduce an instance of the ECDLP to an instance of the HCDLP.

E62, $N=62$, $\mathbb{F}_{2^{62}}=\mathbb{F}_2[z]/(z^{62}+z^{29}+1)$, $a=z^{33}$ $b=z^{59}+z^{55}+z^{48}+z^{47}+z^{45}+z^{43}+z^{42}+z^{40}+z^{39}+z^{38}+z^{37}+z^{36}+z^{34}+z^{30}+z^{29}+z^{27}+$ $z^{25}+z^{24}+z^{22}+z^{21}+z^{20}+z^{19}+z^{18}+z^{16}+z^{13}+z^{12}+z^{11}+z^{10}+z^{8}+z^{6}+z^{5}+z+1$
C62, $q=4$, $\mathbb{F}_{2^2}=\mathbb{F}_2[w]/(w^2+w+1)$ $f(u)=u^{63}+w^2u^{62}+u^{48}+w^2$ $h(u)=u^{31}+u^{30}+wu^{28}+u^{24}+w^2u^{16}+w^2$ $\#\mathrm{E62}(\mathbb{F}_{2^{62}})=\#J_{\mathrm{C62}}(\mathbb{F}_{2^2})=2\cdot 2305843007560748609$
E93, $N=93$, $\mathbb{F}_{2^{93}}=\mathbb{F}_2[z]/(z^{93}+z^{2}+1)$, $a=1$ $b=z^{79}+z^{78}+z^{73}+z^{65}+z^{64}+z^{62}+z^{61}+z^{60}+z^{55}+z^{53}+z^{51}+z^{50}+z^{49}+z^{48}+z^{41}+z^{40}+$ $z^{38}+z^{37}+z^{36}+z^{34}+z^{33}+z^{29}+z^{26}+z^{24}+z^{22}+z^{21}+z^{16}+z^{14}+z^{12}+z^{11}+z^{10}+z^{9}+$ $z^{8}+z^{7}+z^{5}+z^{3}+z$
C93, $q=8$, $\mathbb{F}_{2^3}=\mathbb{F}_2[w]/(w^3+w+1)$ $f(u)=w^4u^{63}+w^5u^{62}+w^5u^{60}+w^3u^{56}+w^5u^{48}+wu^{32}+w^5$ $h(u)=w^2u^{31}+w^5u^{30}+u^{28}+w^6u^{24}+w^6$ $\#\mathrm{E93}(\mathbb{F}_{2^{93}})=\#J_{\mathrm{C93}}(\mathbb{F}_{2^3})=2\cdot 4951760157141611728579495009$
E124, $N=124$, $\mathbb{F}_{2^{124}}=\mathbb{F}_2[z]/(z^{124}+z^{19}+1)$, $a=z^{105}$ $b=z^{108}+z^{106}+z^{102}+z^{101}+z^{99}+z^{93}+z^{87}+z^{85}+z^{75}+z^{70}+z^{68}+z^{67}+z^{66}+z^{64}+z^{62}+$ $z^{59}+z^{58}+z^{56}+z^{55}+z^{54}+z^{53}+z^{51}+z^{50}+z^{49}+z^{48}+z^{46}+z^{45}+z^{44}+z^{42}+z^{41}+$ $z^{40}+z^{33}+z^{32}+z^{29}+z^{27}+z^{24}+z^{23}+z^{22}+z^{20}+z^{18}+z^{16}+z^{15}+z^{14}+z^{9}+z^{8}+z^{7}+$ $z^{6}+z^{3}+z^{2}+z$
C124, $q=16$, $\mathbb{F}_{2^4}=\mathbb{F}_2[w]/(w^4+w+1)$ $f(u)=w^3u^{63}+w^7u^{60}+w^3u^{56}+w^3u^{48}+1$ $h(u)=w^9u^{31}+w^{12}u^{30}+w^8u^{28}+w^{13}u^{24}+w^6u^{16}+w^6$ $\#\mathrm{E124}(\mathbb{F}_{2^{124}})=\#J_{\mathrm{C124}}(\mathbb{F}_{2^4})=2\cdot 10633823966279326985483775888689817121$
E155, $N=155$, $\mathbb{F}_{2^{155}}=\mathbb{F}_2[z]/(z^{155}+z^{62}+1)$, $a=1$ $b=z^{16}+z^2+z$
C155, $q=32$, $\mathbb{F}_{2^5}=\mathbb{F}_2[w]/(w^5+w^2+1)$ $f(u)=w^4u^{63}+w^6u^{62}+w^{15}u^{60}+w^{26}u^{56}+w^{25}u^{48}+w^7u^{32}+w^{13}$ $h(u)=w^2u^{31}+w^7u^{30}+w^{30}u^{28}+w^{22}u^{24}+w^3u^{16}+w^{22}$ $\#\mathrm{E155}(\mathbb{F}_{2^{155}})=\#J_{\mathrm{C155}}(\mathbb{F}_{2^5})=$ $2\cdot 22835963083295358096932727763065266972881541089$

Table 1: Hyperelliptic curves C62, C93, C124 and C155 of genus $g=31$ over $\mathbb{F}_q$ for $q=4,8,16$ and 32. These curves were obtained by applying the GHS attack to an instance of the ECDLP on elliptic curves E62, E93, E124 and E155 over $\mathbb{F}_{2^{62}}$, $\mathbb{F}_{2^{93}}$, $\mathbb{F}_{2^{124}}$ and $\mathbb{F}_{2^{155}}$, respectively (cf. Appendix A). "EN" denotes an elliptic curve over $\mathbb{F}_{2^N}$. The equation of EN is $y^2+xy=x^3+ax^2+b$ where $a,b\in\mathbb{F}_{2^N}$, The equation of CN is $v^2+h(u)v=f(u)$, where $h,f\in\mathbb{F}_q[u]$. The prime factorizations of $\#\mathrm{E}N(\mathbb{F}_{2^N})$ and $\#J_{\mathrm{C}N}(\mathbb{F}_q)$ are also listed.

GROUP LAW. We implemented Cantor's algorithm [5] with Tenner's reduction algorithm [33] for adding reduced divisors.

RANDOM WALK. 40 integers $a_0, a_1, \ldots, a_{19}$, $b_0, b_1, \ldots, b_{19}$ are randomly selected from $[0, r-1]$, and the divisors $T_i = a_i D_1 + b_i D_2$, $0 \le i \le 19$, are computed. The walk commences at a divisor $R_0 = \alpha_0 D_1 + \beta_0 D_2$ where α_0 and β_0 are randomly selected from $[0, r-1]$. A divisor R_i on the walk is computed from the previous divisor R_{i-1} as $R_i = R_{i-1} + T_j$, where j is obtained by taking the integer formed from the 5 least significant bits of the binary representation of a, where $R_{i-1} = \mathrm{div}(a, b)$, and reducing it modulo 20. Note that $R_i = \alpha_i D_1 + \beta_i D_2$ where $\alpha_i = (\alpha_{i-1} + a_j) \bmod r$ and $\beta_i = (\beta_{i-1} + b_j) \bmod r$. Thus the pair (α_i, β_i) can be efficiently computed from the pair $(\alpha_{i-1}, \beta_{i-1})$.

FACTOR BASE. Let $a \in k[u]$ be a monic irreducible polynomial for which

$$v^2 + h(u)v - f(u) \equiv 0 \pmod{a(u)} \tag{2}$$

has a solution $v = b(u) \in k[u]$. Then $D = \mathrm{div}(a, b)$ and $-D = \mathrm{div}(a, b + h)$ are the only prime divisors with first component a.[2] We store exactly one of D and $-D$ in the factor base. Let A_l denote the number of prime divisors of degree l in the factor base for $1 \le l \le t$. Heuristically, one would expect that half of all equations (2) have solutions, and hence one expects A_l to be equal to half the number $I_q(l)$ of monic irreducible polynomials of degree l in $k[u]$. That is,

$$A_l \approx \frac{1}{2}\left(\frac{1}{l}\sum_{d|l}\mu(l/d)q^d\right), \tag{3}$$

where μ is the Möbius function. In fact, this estimate is a good one for the following reasons. Theorem 2 of [9] states that if

$$0 \le \epsilon \le \frac{1}{4} \quad \text{and} \quad l \ge \frac{1}{\epsilon}\log_q(2g + 6 + \sqrt{2}), \tag{4}$$

then $A_l \in [F_1, G_1]$ where

$$F_1 = \frac{q^l}{2l}\left(1 - q^{l(\epsilon - \frac{1}{2})}\right) \quad \text{and} \quad G_1 = \frac{q^l}{2l}\left(1 + q^{l(\epsilon - \frac{1}{2})}\right).$$

Now, by Theorem 6.5.1 of [2], we have $\frac{1}{2}I_q(l) \in [F_2, G_2]$ where

$$F_2 = \frac{q^l}{2l}\left(1 - \frac{2}{q^{l/2}}\right) \quad \text{and} \quad G_2 = \frac{q^l}{2l}.$$

Clearly, $G_2 \le G_1$. And, it is easy to see that $F_1 \le F_2$ when (4) holds. Thus, when (4) holds, the estimate $\frac{1}{2}I_q(l)$ lies in the interval $[F_1, G_1]$ which is known to contain A_l.

The following lemma gives an efficiently computable expression for the number of t-smooth reduced divisors in $J_C(k)$ where $C \in \{$C62,C93,C124,C155$\}$.

[2] For the curves C62, C93, C124 and C155, $h(u)$ is irreducible over k. Thus $h \not\equiv 0 \pmod{a}$ when $1 \le \deg a < \deg h$, and so $D \ne -D$.

Lemma 2 Let $C \in \{C62, C93, C124, C155\}$. Let A_l, $1 \le l \le t$, denote the number of prime divisors of degree l in the factor base. Then the number of t-smooth reduced divisors in $J_C(k)$ is

$$M(t) = \sum_{i=1}^{31} \left([x^i] \prod_{l=1}^{t} \left(\frac{1+x^l}{1-x^l} \right)^{A_l} \right),$$

where $[\,]$ denotes the coefficient operator.

Proof: Suppose that $a \in k[u]$ is a t-smooth monic polynomial of degree ≤ 31 for which (2) has a solution. Let $a = a_1^{e_1} a_2^{e_2} \cdots a_L^{e_L}$ be the factorization of a into monic irreducibles over k. Then the number of t-smooth reduced divisors in $J_C(k)$ having first component a is exactly 2^L; these divisors are $D = \sum_{i=1}^{L} e_i \mathrm{div}(a_i, b_i)$ where each b_i is one of the two solutions to $v^2 + h(u)v - f(u) \equiv 0 \pmod{a_i}$.

For each l, $1 \le l \le t$, let $P_l = \{a(u) : \mathrm{div}(a,b) \text{ is a prime divisor of degree } l\}$. Note that $\#P_l = A_l$. Let $c_{i,j}$ be the number of monic polynomials of degree i in $k[u]$ having exactly j distinct monic irreducible factors all of which are in $\bigcup_{l=1}^{t} P_l$. Then

$$\sum_{i,j \ge 0} c_{i,j} x^i y^j = \prod_{l=1}^{t} \left(1 + x^l y + x^{2l} y + x^{3l} y + \cdots \right)^{A_l} = \prod_{l=1}^{t} \left(1 + \frac{x^l y}{1-x^l} \right)^{A_l}.$$

Since there are exactly two prime divisors $\mathrm{div}(a,b)$ for each monic irreducible polynomial a in $\bigcup_{l=1}^{t} P_l$, it follows that

$$M(t) = \sum_{i=1}^{31} \sum_{j \ge 0} c_{i,j} 2^j = \sum_{i=1}^{31} \left([x^i] \prod_{l=1}^{t} \left(1 + \frac{2x^l}{1-x^l} \right)^{A_l} \right),$$

as required. □

For known values of A_l, $1 \le l \le t$, $M(t)$ can be efficiently obtained by computing the first 32 terms of the Taylor series expansion about $x = 0$ of

$$\prod_{l=1}^{t} \left(\frac{1+x^l}{1-x^l} \right)^{A_l},$$

and then summing the coefficients of $x, x^2, \ldots, x^{31}$.

Smoothness Bound Selection. The divisors encountered in the random walk all lie in the prime order subgroup $\langle D_1 \rangle$ of order r. We make the heuristic (and reasonable) assumption that the proportion of t-smooth divisors in $\langle D_1 \rangle$ is the same as the proportion of t-smooth divisors in the full group $J_C(k)$. Then, the expected number of random walk iterations before a t-smooth divisor is encountered is $E(t) = \#J_C(k)/M(t)$. Table 2 presents, for various choices of the smoothness bound t, the factor base size $F(t)$, $E(t)$, and the expected number $T(t) = (F(t)+5)E(t)$ of random walk iterations to generate $F(t)+5$ relations[3]. In the table, an asterisk signifies that the factor base

[3]Some of the relations generated may be linearly dependent on previous relations. Heuristically, we expect that if $F(t)+5$ relations are generated, then the resulting system of linear equations will have a unique solution.

size $F(t)$ was estimated using (3). Taking into account both the expected running time and the storage requirements for the factor base, it appears that the optimal choices of smoothness bounds are $t = 7, 5, 5$ and 4 for C62, C93, C124 and C155, respectively.

SMOOTHNESS TESTING. Given a reduced divisor $D = \text{div}(a, b)$, $a(u)$ is first subjected to a square-free factorization algorithm (e.g., see [2]). The square-free portion $\overline{a}(u)$ is then tested for t-smoothness using the fact that $x^{q^l} - x$ is the product of all monic irreducible polynomials in $\mathbb{F}_q[x]$ of degree dividing l. If $\overline{a}(u)$ is indeed t-smooth, then the factorization is obtained using the Cantor-Zassenhaus factoring algorithm [6]. Table 3 presents the time to generate and test 10,000 candidate reduced divisors for C62, C93, C124 and C155. Generating a candidate essentially involves one application of the Jacobian group law, while testing a candidate involves a square-free factorization and a distinct degree factorization. Also listed in Table 3 is the proportion of time spent on the Jacobian group law and on the smoothness testing.

PARALLELIZATION. The relation gathering portion of the algorithm can be effectively parallelized, i.e., yielding a factor-m speedup when m processors are used. A different random walk is performed on each machine (i.e., with different divisors $T_0, T_1, \ldots, T_{19}$ and different initial divisors R_0). Any relations are reported to a central processor which also discards duplicates.

LINEAR ALGEBRA. For C62, C93, and C124, we used our unoptimized implementation of Wiedemann's algorithm [42] as described in [23] to compute a vector in the kernel of the matrix modulo the large prime divisor r of the group order. For C155, it will be necessary to optimize our implementation and most likely add structured Gaussian elimination [25] to reduce the size of the matrix before applying Wiedemann. Nevertheless, we do not anticipate major difficulties with this stage of the algorithm. Joux and Lercier [22] report on performing structured Gaussian elimination on a sparse matrix with 2,900,000 rows, followed by Lanczos on a $172,049 \times 171,061$ matrix, all modulo a 100-decimal digit prime. This was a parallel computation (four 500 Mhz Dec Alpha processors), and took 20 days. By comparison, the sparse matrix for the C155 discrete logarithm computation has only 136,528 rows, and the linear algebra is performed modulo a 155-bit prime. Thus, the linear algebra stage of the discrete logarithm computations for C155 is well within the realm of feasibility.

5.2 Numerical Experiments

Table 4 presents timings from our experiments with solving instances of the HCDLP in the genus 31 curves C62, C93 and C124. Note that the average number of random walk iterations before a smooth divisor is encountered is very close to the predicted numbers in Table 2.

From Table 4, we conclude that the HCDLP for each of the three curves C62, C93 and C124 is quite tractable. In fact, the HCDLP in C124 (and hence also the ECDLP in E124; cf. Appendix A) was solved in far less CPU time that the estimated 200,000 days on a single 450 MHz Pentium PC expended on solving the significantly easier Certicom ECC2-108K ECDLP challenge[4] [19].

[4]Koblitz curves [24, 40] are elliptic curves defined over $\mathbb{F}_2$. ECC2-108K is an instance of the ECDLP in a Koblitz curve of order twice a prime over $\mathbb{F}_{2^{109}}$. By exploiting properties of the Frobenius endomorphism, Pollard's rho

Curve	t	$F(t)$	$E(t)$	$T(t)$
C62	1	2	2324438515686238	16271069609803669
	2	4	27837587014206	250538283127858
	3	14	1794233002	34090427031
	4	42	2889490	135806029
	5	144	36296	5408075
	6	474	2614	1251872
	7	1644	421	694997
	8	*5724	117	672969
	9	*20284	46	932866
	10	*72661	23	1647615
C93	1	4	$1.15035222 \times 10^{22}$	1.3531699×10^{23}
	2	16	5594986379814614	117494713976106894
	3	100	2237298251	234916316328
	4	596	1830509	1100135670
	5	3872	28668	111146195
	6	*25670	2139	54917739
	7	*175466	370	64977373
	8	*1223786	107	130753664
C124	1	8	$3.33693830 \times 10^{28}$	$4.33801984 \times 10^{28}$
	2	64	$6.48579145 \times 10^{15}$	$4.44751961 \times 10^{18}$
	3	744	1781948118	1334679140141
	4	8872	1498799	13304838571
	5	113728	25876	2942900859
	6	*1511468	2001	3024499495
	7	*20685428	354	7320993345
	8	*289116788	103	29880384177
C155	1	16	$1.15149568 \times 10^{32}$	$2.24181409 \times 10^{32}$
	2	256	4105255075208737	1071471574629480605
	3	5712	1549820999	8860326649526
	4	136528	1378374	188193560220
	5	*3491968	24746	86410841791
	6	*92967640	1945	180781004858
	7	*2547234664	347	883799233900
	8	*71266645874	102	7257807696673

Table 2: For each of the curves C62, C93, C124, C155, this table lists the factor base size $F(t)$, the expected number $E(t)$ of random walk iterations before a t-smooth divisor is encountered, and the expected number $T(t) = (F(t) + 5)E(t)$ of random walk iterations to generate $F(t) + 5$ relations for various choices of the smoothness bound t. An asterisk signifies that $F(t)$ is an estimate of the factor base size.

Curve	Smoothness bound t	Time to generate and test 10,000 candidate divisors	Proportion of time spent on Jacobian arithmetic	Proportion of time spent on smoothness testing
C62	7	54.0	40%	60%
C93	5	67.4	38%	62%
C124	5	89.0	32%	68%
C155	4	120.7	25%	75%

Table 3: Time (in sec) to generate and test 10,000 candidate reduced divisors for t-smoothness on a single 1 GHz Pentium III workstation having 512 MBytes of RAM.

Curve	C62	C93	C124
Smoothness bound t	7	5	5
Factor base size	1,644	3,872	113,728
Time to generate factor base	20s	34s	12m 3s
Number of relations generated	1,649	3,877	113,733
Avg. no. of iterations per relation	400	28,050	25,576
Total CPU time to generate all relations	1h 49m 29s	15d 20h 6m	379d 2h 1m
Time to solve linear system	46s	6m 23s	3d 17h 55m

Table 4: Timings from our experiments with implementing the Enge-Gaudry index-calculus algorithm for solving instances of the HCDLP in the genus 31 curves C62, C93 and C124 (see Table 1). The timings for factor base generation and for solving the sparse linear system were obtained using a single 800 MHz Pentium III workstation with 512 MBytes of RAM. The timings for relation generation for C62 and C93 were obtained using a cluster of 12 550 MHz Pentium III workstations each having 256 MBytes of RAM. The timing for relation generation for C124 was obtained using a cluster of 16 400 MHz Pentium II processors, 26 450 MHz Pentium II processors, 66 550 MHz Pentium III processors, and 100 1 GHz Pentium III processors. Seconds, minutes, hours, and days are denoted by “s”, “m”, “h”, and “d”, respectively.

Curve	C62	C93	C124	C155
Smoothness bound t	7	5	5	4
Factor base size	1,644	3,872	113,728	136,528
Time to generate factor base	15s	26s	9m 17s	8m 58s
Number of relations generated	1,649	3,877	113,733	136,533
Total CPU time to generate all relations	(1h 3m)	(8d 16h)	(303d)	(26,290d)
Time to solve linear system	(1m)	(6m)	(3d 12m)	(5d)

Table 5: Time to solve instances of the HCDLP on C62, C93, C124 and C155. The times for factor base generation are actual times obtained on a *single* 1 GHz Pentium III workstation with 512 MBytes of RAM. The times for generating relations are estimates on a *single* 1 GHz Pentium III workstation with 512 MBytes of RAM. These estimates were derived from our estimates for the number of random walk iterations required (see Table 2), and the actual time to generate and test a candidate divisor for smoothness (see Table 3). The times for solving the sparse linear system are estimates for a 1 GHz Pentium III workstation.

We did not solve an instance of the HCDLP in C155. However, we argue that this problem is quite feasible. For a smoothness bound of $t = 4$, the factor base size is $F(4) = 136,528$. From Table 2, the expected number of random walk iterations before a smooth divisor is encountered is $E(4) = 1,378,374$. Thus the expected number of random walk iterations before $F(4) + 5$ relations are obtained is $E(4)(F(4)+5) \approx 1.88\times10^{11}$. Since the average time to generate and test a candidate divisor is 1.207×10^{-2} sec on a 1 GHz Pentium III workstation (see Table 3), the expected time to generate the relations on a single such machine is approximately 26,290 days. The time to solve the resulting sparse linear system can be ignored since, as argued in §5.1, it is at most a couple of days. The estimated time for the C155 HCDLP computation is compared to the estimated time for the C62, C93 and C124 computations on the same workstation in Table 5.

We can conclude that instances of the HCDLP in C155 can be solved in about one month using a network of 1,000 1 GHz Pentium III workstations. This is the same order of magnitude as the work required to perform exhaustive search on the DES key space (estimated time is 110,000 days on a single 450 MHz Pentium PC [19]), and less that the estimated time of 200,000 days on a single 450 MHz Pentium PC spent on the Certicom ECC2-108K ECDLP challenge [19].

5.3 Further Optimizations

We did not make significant efforts to optimize our implementation. The following are some ways in which our implementation could be improved.

1. Experiment with different methods for selecting prime divisors for the factor base. For example, we might start with an empty factor base and add prime divisors as they are encountered as factors of smooth divisors.

algorithm for the ECDLP in Koblitz curves over $\mathbb{F}_{2^m}$ can be sped up by a factor of $\sqrt{m}$ [15, 43]. The expected number of elliptic curve operations to solve the ECC2-108K challenge using Pollard's rho algorithm is 1.5×10^{16}.

2. Experiment with the large prime variant for generating relations. In addition to storing the factorizations of the t-smooth divisors, we also store "partial relations" which arise from random divisors which are t-smooth except for one irreducible factor of high degree. Any two partial relations containing the same large irreducible factor can be combined to yield a relation. This method has been successfully employed in other index-calculus algorithms (e.g., see [26]), and initial experiments indicate that it may be useful in our setting as well.

3. Experiment with Bernstein's methods [4] for fast smoothness testing.

4. Experiment with sieving methods (see [10]) to determine if they can be used to generate relations faster than the random walk method.

6 Cryptographic Implications

Our experiments with our non-optimized implementation of index-calculus methods for the HCDLP in C155 indicate that the HCDLP for genus 31 hyperelliptic curves over $\mathbb{F}_{2^5}$ is quite tractable. Now, the ECDLP in the particular elliptic curve E155 over $\mathbb{F}_{2^{155}}$ (see Table 1) is intractable using Pollard's rho algorithm since the expected number of elliptic curve operations is $\sqrt{\pi 2^{154}/4} \approx 2^{77}$. However, since the GHS attack can efficiently reduce instances of the ECDLP in E155 to instances of the HCDLP in genus 31 hyperelliptic curves over $\mathbb{F}_{2^5}$, we conclude that the ECDLP in E155 is indeed tractable.

Even though the GHS attack only appears to be applicable to an insignificant proportion (2^{32} out of the 2^{156} elliptic elliptic curves over $\mathbb{F}_{2^{155}}$), we feel that caution must be exercised when selecting elliptic curves over $\mathbb{F}_{2^{155}}$ for cryptographic use.

The particular elliptic curve over $\mathbb{F}_{2^{155}}$ included in the IETF standard [21] is $y^2 + xy = x^3 + b$, where

$$b = w^{18} + w^{17} + w^{16} + w^{13} + w^{12} + w^9 + w^8 + w^7 + w^3 + w^2 + w + 1$$

and $\mathbb{F}_{2^{155}} = \mathbb{F}_2[w]/(w^{155} + w^{62} + 1)$. Let $\sigma : \mathbb{F}_{2^{155}} \to \mathbb{F}_{2^{155}}$ be the Frobenius map defined by $x \mapsto x^{2^5}$. The smallest degree factor $f(x)$ of $x^{31} + 1$ over $\mathbb{F}_2$ for which $f(\sigma)(b) = 0$ is $f(x) = (x^{31} + 1)/(x^5 + x^3 + 1)$. It follows from [29, Theorem 6] that the GHS attack reduces the ECDLP in $E(\mathbb{F}_{2^{155}})$ to the HCDLP in the Jacobian of a genus 2^{35} or $2^{35} - 1$ hyperelliptic curve over $\mathbb{F}_{2^5}$. Hence this particular elliptic curve does not succumb to our approach of reducing the ECDLP to the HCDLP over $\mathbb{F}_{2^5}$.

An open question is whether the GHS attack can be applied to *all* elliptic curves over $\mathbb{F}_{2^{155}}$. As shown in [29], except for the Koblitz curves[5], the GHS attack reduces the ECDLP in elliptic curves over $\mathbb{F}_{2^{155}}$ to the HCDLP in Jacobians of genus 15 or 16 curves over $\mathbb{F}_{2^{31}}$. Smart [39] argues that Gaudry's algorithm (with the factor base consisting only a fraction of the prime reduced divisors of degree 1) is infeasible given today's computer technology. However, [39] did not consider (in any

[5]The GHS attack can be proven to fail for Koblitz curves—the attack only yields information about the desired logarithm modulo $\#E(\mathbb{F}_2)$.

detail) the applicability of the other index-calculus methods. In particular, large-prime variants and sieving methods were not considered. While it is likely that the known index-calculus methods are indeed infeasible for this problem, further study and experimentation is needed before this can be concluded with certainty.

Another possibility for attacking the general ECDLP for elliptic curves over $\mathbb{F}_{2^{155}}$, of course, is if the Weil descent methodology can be exploited to yield another way (i.e., different from the GHS attack) of reducing the ECDLP for elliptic curves over $\mathbb{F}_{2^{155}}$ to Jacobians of low genus curves (perhaps not hyperelliptic) for which subexponential-time index-calculus methods can be found. We have no evidence to make a conjecture about the existence of such a possibility, however we would expect that it is much more likely for such a method to exist for elliptic curves over fields $\mathbb{F}_{2^m}$ where m is composite (e.g, $m = 155$), than for elliptic curves over fields $\mathbb{F}_{2^m}$ where m is prime. Some evidence for this is provided by the complete failure of the GHS attack for the ECDLP in elliptic curves over $\mathbb{F}_{2^m}$ where m is prime [29].

7 Conclusions

We have implemented the GHS Weil desent attack and the Gaudry-Enge index-calculus method for the HCDLP. We were successful in solving specific discrete logarithm problems in elliptic curves over $\mathbb{F}_{2^{62}}$, $\mathbb{F}_{2^{93}}$ and $\mathbb{F}_{2^{124}}$. Our experiments, though far from being optimized, indicate that our specific logarithm problem in $\mathbb{F}_{2^{155}}$ is tractable. The ECDLP instance over $\mathbb{F}_{2^{155}}$ is the first concrete instance of the ECDLP which resists all previously known attacks, but which can be solved using the Weil descent attack methodology of Frey.

While the GHS attack is only known to apply to an insignificant proportion of all elliptic curves over $\mathbb{F}_{2^{155}}$, our results provide some evidence that elliptic curves over $\mathbb{F}_{2^{155}}$ should be used with caution and preferably avoided altogether.

We emphasize that our computational results cannot be extended to solve cryptographically interesting instances of the ECDLP for elliptic curves over fields $\mathbb{F}_{2^m}$ where $m \in [160, 600]$ is prime, since the GHS attack is ineffective in these cases [29].

Acknowledgements

The authors would like to thank Steven Galbraith and Nigel Smart for some helpful comments, and also Florian Hess for help with his KASH program for performing the GHS attack. We are indebted to the CSE at the University of Illinois at Urbana-Champaign for granting us access to their Linux Cluster.

References

[1] L. Adleman, J. DeMarrais and M. Huang, "A subexponential algorithm for discrete logarithms over the rational subgroup of the jacobians of large genus hyperelliptic curves over finite fields", *Algorithmic Number Theory*, LNCS **877**, 1994, 28-40.

[2] E. Bach and J. Shallit, *Algorithmic Number Theory*, MIT Press, 1996

[3] M. Bauer, "A subexponential algorithm for solving the discrete logarithm problem in the Jacobian of high genus hyperelliptic curves over arbitrary finite fields, preprint, 1999.

[4] D. Bernstein, "How to find small factors of integers", preprint, 2000.

[5] D. Cantor, "Computing in the jacobian of a hyperelliptic curve", *Mathematics of Computation*, **48** (1987), 95-101.

[6] D. Cantor and H. Zassenhaus, "A new algorithm for factoring polynomials over finite fields", *Mathematics of Computation*, **36** (1981), 587-592.

[7] A. Enge, "Computing discrete logarithms in high-genus hyperelliptic jacobians in provably subexponential time", *Mathematics of Computation*, to appear.

[8] A. Enge and P. Gaudry, "A general framework for subexponential discrete logarithm algorithms", *Acta Arithmetica*, to appear.

[9] A. Enge and A. Stein, "Smooth ideals in hyperelliptic function fields", *Mathematics of Computation*, to appear.

[10] R. Flassenberg and S. Paulus, "Sieving in function fields", *Experimental Mathematics*, **8** (1999), 339-349.

[11] G. Frey, "How to disguise an elliptic curve (Weil descent)", Talk at ECC '98, Waterloo, 1998. Slides available from http://www.cacr.math.uwaterloo.ca/conferences/1998/ecc98/slides.html

[12] G. Frey, "Applications of arithmetical geometry to cryptographic constructions", *Proceedings of the Fifth International Conference on Finite Fields and Applications*, Springer-Verlag, 2001, 128-161.

[13] 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** (1994), 865-874.

[14] S. Galbraith and N. Smart, "A cryptographic application of Weil descent", *Codes and Cryptography*, LNCS **1746**, 1999, 191-200.

[15] R. Gallant, R. Lambert and S. Vanstone, "Improving the parallelized Pollard lambda search on binary anomalous curves", *Mathematics of Computation*, **69** (2000), 1699-1705.

[16] P. Gaudry, "An algorithm for solving the discrete log problem on hyperelliptic curves", *Advances in Cryptology – Eurocrypt 2000*, LNCS **1807**, 2000, 19-34.

[17] P. Gaudry, F. Hess and N. Smart, "Constructive and destructive facets of Weil descent on elliptic curves", preprint, January 2000. Available from http://ultralix.polytechnique.fr/Labo/Pierrick.Gaudry/papers.html

[18] J. Hafner and K. McCurley, "A rigorous subexponential algorithm for computation of class groups", *Journal of the American Mathematical Society*, **2** (1989), 837-850.

[19] R. Harley, Fact sheet for solution of ECC2-108K ECDLP challenge, http://cristal.inria.fr/~harley/ecdl7/factsheet.html

[20] F. Hess, KASH program for performing the GHS attack, 2000.

[21] Internet Engineering Task Force, *The OAKLEY Key Determination Protocol*, IETF RFC 2412, November 1998.

[22] A. Joux and R. Lercier, "Improvements on the general number field sieve for discrete logarithms in finite fields", *Mathematics of Computation*, to appear.

[23] E. Kaltofen and A. Lobo, "Distributed matrix-free solution of large sparse linear systems over finite fields", *Algorithmica*, **24** (1999), 331-348.

[24] N. Koblitz, "CM-curves with good cryptographic properties", *Advances in Cryptology – Crypto '91*, LNCS **576**, 1992, 279-287.

[25] B. LaMacchia and A. Odlyzko, "Solving large sparse linear systems over finite fields", *Advances in Cryptology – Crypto '90*, LNCS **537**, 1991, 109-133.

[26] A. Lenstra and M. Manasse, "Factoring with two large primes", *Advances in Cryptology – Eurocrypt '90*, LNCS **473**, 1991, 72-82.

[27] A. Menezes, T. Okamoto and S. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field", *IEEE Transactions on Information Theory*, **39** (1993), 1639-1646.

[28] A. Menezes, P. van Oorschot and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.

[29] A. Menezes and M. Qu, "Analysis of the Weil descent attack of Gaudry, Hess and Smart", *Topics in Cryptology – CT-RSA 2001*, LNCS **2020**, 2001, 308-318.

[30] A. Menezes, Y. Wu and R. Zuccherato, "An elementary introduction to hyperelliptic curves", appendix in *Algebraic Aspects of Cryptography* by N. Koblitz, Springer-Verlag, 1998, 155-178.

[31] V. Müller, A. Stein and C. Thiel, "Computing discrete logarithms in real quadratic congruence function fields of large genus", *Mathematics of Computation*, **68** (1999), 807-822.

[32] P. van Oorschot and M. Wiener, "Parallel collision search with cryptanalytic applications", *Journal of Cryptology*, **12** (1999), 1-28.

[33] S. Paulus and A. Stein, "Comparing real and imaginary arithmetics for divisor class groups of hyperelliptic curves", *Algorithmic Number Theory*, LNCS **1423**, 1998, 576-591.

[34] S. Pohlig and M. Hellman, "An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance", *IEEE Transactions on Information Theory*, **24** (1978), 106-110.

[35] J. Pollard, "Monte Carlo methods for index computation mod p", *Mathematics of Computation*, **32** (1978), 918-924.

[36] T. Satoh and K. Araki, "Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves", *Commentarii Mathematici Universitatis Sancti Pauli*, **47** (1998), 81-92.

[37] I. Semaev, "Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p", *Mathematics of Computation*, **67** (1998), 353-356.

[38] N. Smart, "The discrete logarithm problem on elliptic curves of trace one", *Journal of Cryptology*, **12** (1999), 193-196.

[39] N. Smart, "How secure are elliptic curves over composite extension fields?", *Advances in Cryptology – Eurocrypt 2001*, LNCS **2045**, 2001.

[40] J. Solinas, "Efficient arithmetic on Koblitz curves", *Designs, Codes and Cryptography*, **19** (2000), 195-249.

[41] E. Teske, "Speeding up Pollard's rho method for computing discrete logarithms", *Algorithmic Number Theory*, LNCS **1423**, 1998, 541-554.

[42] D. Wiedemann, "Solving sparse linear equations over finite fields", *IEEE Transactions on Information Theory*, **32** (1986), 54-62.

[43] M. Wiener and R. Zuccherato, "Faster attacks on elliptic curve cryptosystems", *Selected Areas in Cryptography*, LNCS **1556**, 1999, 190-200.

A Elliptic Curve and Hyperelliptic Curve Selection

This section describes how the elliptic curve E124 was selected, and how a random instance of the ECDLP in E124 was generated and reduced to an instance of the HCDLP in C124. The other elliptic curves and hyperelliptic curves listed in Table 1 were generated in an analagous manner.

Elliptic Curve Generation. Let $n = 31$, and $q = 2^4$. Let a be an arbitrary element of trace 1 in $\mathbb{F}_{2^{124}}$. The order of 2 modulo n is $t = 5$. The elliptic curve E124 was chosen by selecting random elements $b \in B$ (where B is defined in Theorem 1(ii)) until the number of $\mathbb{F}_{2^{124}}$-rational points on $y^2 + xy = x^3 + ax^2 + b$ is twice a prime. By Theorem 1 we know that $m(b) = t + 1 = 6$ and hence the GHS attack will reduce any instance of the ECDLP in E124 to an instance of the HCDLP in a genus 31 or 32 hyperelliptic curve over $\mathbb{F}_{2^4}$.

The elements of $\mathbb{F}_{2^{124}}$ are represented as binary polynomials modulo the irreducible polynomial $z^{124} + z^{19} + 1$. The defining equation for the elliptic curve E124 is $y^2 + xy = x^3 + ax^2 + b$ where $a = z^{105}$ and

$$\begin{aligned} b \;=\; & z^{108}+z^{106}+z^{102}+z^{101}+z^{99}+z^{93}+z^{87}+z^{85}+z^{75}+z^{70}+z^{68}+z^{67}+z^{66}+z^{64}+z^{62}+z^{59}+ \\ & z^{58}+z^{56}+z^{55}+z^{54}+z^{53}+z^{51}+z^{50}+z^{49}+z^{48}+z^{46}+z^{45}+z^{44}+z^{42}+z^{41}+z^{40}+z^{33}+z^{32}+ \\ & z^{29}+z^{27}+z^{24}+z^{23}+z^{22}+z^{20}+z^{18}+z^{16}+z^{15}+z^{14}+z^{9}+z^{8}+z^{7}+z^{6}+z^{3}+z^{2}+z. \end{aligned}$$

The number of $\mathbb{F}_{2^{124}}$-rational points on E124 is $2r$, where

$$r = 10633823966279326985483775888689817121$$

is prime.

ECDLP Instance Generation. We selected two points P, Q from E124($\mathbb{F}_{2^{124}}$) *verifiably at random* as follows. We first defined 124-bit integers m_1 and m_2 to be the 124 rightmost bits of the 160-bit outputs of the SHA-1 cryptographic hash function with inputs the strings “” and “a”, respectively[6]. We identify a 124-bit integer $c = c_{123}2^{123} + c_{122}2^{122} + \cdots + c_0$ with the element $c_{123}z^{123} + c_{122}z^{122} + \cdots + c_0$ of $\mathbb{F}_{2^{124}}$. Then, for each $i \in \{1, 2\}$, we define n_i to be the smallest integer $\geq m_i$ for which the field element corresponding to n_i is the x-coordinate of some point of order r in E124($\mathbb{F}_{2^{124}}$); for such an n_i we arbitrarily select one of the two possible y-coordinates. In this way, we derive the following two points:

$$\begin{aligned} P &= (19166289931116350914892435465096922889, \\ & \quad\; 39549266381157102372793271078772986 63), \\ Q &= (14152416137154867042654754006541690809, \\ & \quad\; 15733241592903071723351565426494711869). \end{aligned}$$

The ECDLP challenge is to find the integer $l \in [0, r-1]$ such that $Q = lP$. Note that since P and Q were (pseudo)randomly generated, the discrete logarithm l is not known a priori by us.

[6]These two strings are commonly used as inputs to generate test vectors for hash functions. For example, see Table 9.6 of [28].

HCDLP Instance Generation. Hess's KASH program [20] for the Weil restriction represents elliptic curve points as zero divisors. For technical reasons, he excludes the point at infinity from occurring in the support of the divisors. Thus, instead of representing an elliptic curve point P by a zero divisor $(P) - (\infty)$, we represent P by the equivalent zero divisor $(P + R) - (R)$, where R is an arbitrary point on the curve. We arbitrarily selected the following point of order r:

$$\begin{aligned} R &= (11949386922129241854287919257049811485, \\ &\quad\ 13819702817838731027194193290120801107). \end{aligned}$$

Let $P_1 = P{+}R$, $P_2 = Q{+}R$ and $P_3 = R$. Hess's KASH program was used to reduce $(\text{E124}, P_1, P_2, P_3)$ to $(\text{C124}, D_1, D_2, D_3)$, where C124 is a genus-31 hyperelliptic curve over $\mathbb{F}_{2^4}$ and D_1, D_2, D_3 are divisors in $J_{\text{C124}}(\mathbb{F}_{2^4})$. The elements of $\mathbb{F}_{2^4}$ are represented as binary polynomials modulo the irreducible polynomial $w^4 + w + 1$. The Weierstrass equation for the hyperelliptic curve C124 is $v^2 + h(u)v = f(u)$, where

$$\begin{aligned} f(u) &= w^6u^{63} + w^{14}u^{60} + w^6u^{56} + w^6u^{48} + 1, \\ h(u) &= w^3u^{31} + w^9u^{30} + wu^{28} + w^{11}u^{24} + w^{12}u^{16} + w^{12}. \end{aligned}$$

The divisors D_1, D_2 and D_3 are:

$$\begin{aligned} D_1 = \operatorname{div}(&u^{31}{+}w^6u^{30}{+}w^4u^{29}{+}w^5u^{28}{+}w^{10}u^{27}{+}w^3u^{26}{+}w^{14}u^{25}{+}w^4u^{24}{+}w^{14}u^{23}{+}u^{22}{+}w^5u^{21}{+} \\ &w^9u^{20}{+}w^{14}u^{19}{+}w^4u^{18}{+}w^{14}u^{17}{+}w^{12}u^{16}{+}w^6u^{15}{+}w^{14}u^{14}{+}w^7u^{13}{+}w^7u^{12}{+}w^2u^{11}{+}w^7u^{10}{+} \\ &w^{13}u^9{+}w^7u^8{+}u^7{+}w^9u^6{+}w^{14}u^5{+}w^3u^4{+}w^2u^3{+}w^{10}u^2{+}w^9u{+}1,\ u^{30}{+}w^8u^{29}{+}wu^{28}{+}w^8u^{27}{+} \\ &w^{14}u^{26}{+}w^5u^{24}{+}w^{10}u^{23}{+}w^4u^{22}{+}w^8u^{21}{+}w^9u^{19}{+}w^2u^{18}{+}w^3u^{16}{+}w^5u^{15}{+}w^{13}u^{14}{+}w^{11}u^{13}{+} \\ &w^7u^{12}{+}u^{11}{+}w^8u^{10}{+}u^9{+}w^2u^8{+}w^6u^7{+}u^6{+}wu^5{+}w^9u^4{+}w^{13}u^3{+}w^2u{+}w^7), \end{aligned}$$

$$\begin{aligned} D_2 = \operatorname{div}(&u^{31}{+}w^{12}u^{30}{+}w^3u^{29}{+}w^8u^{28}{+}w^{12}u^{27}{+}w^{14}u^{26}{+}w^{13}u^{25}{+}w^9u^{24}{+}w^7u^{23}{+}w^{12}u^{22}{+}u^{20}{+} \\ &w^3u^{18}{+}w^{12}u^{17}{+}u^{16}{+}w^{12}u^{15}{+}w^3u^{14}{+}w^9u^{13}{+}w^6u^{12}{+}w^9u^{11}{+}w^7u^{10}{+}w^2u^9{+}w^8u^8{+} \\ &w^{11}u^7{+}w^9u^6{+}w^{12}u^5{+}w^{10}u^4{+}w^{11}u^3{+}w^{11}u^2{+}w^{11}u{+}1,\ w^{14}u^{29}{+}w^6u^{28}{+}u^{27}{+}w^{11}u^{26}{+} \\ &w^{11}u^{25}{+}w^4u^{24}{+}w^{14}u^{22}{+}w^5u^{21}{+}w^3u^{20}{+}w^{14}u^{19}{+}w^5u^{18}{+}w^2u^{17}{+}w^8u^{15}{+}u^{14}{+}w^4u^{13}{+} \\ &w^7u^{12}{+}w^{10}u^{11}{+}w^6u^{10}{+}w^4u^9{+}w^2u^8{+}w^{14}u^7{+}wu^6{+}w^{11}u^4{+}w^{11}u^3{+}w^2u^2{+}w^9u{+}w^6), \end{aligned}$$

$$\begin{aligned} D_3 = \operatorname{div}(&u^{31}{+}w^{14}u^{30}{+}w^5u^{28}{+}u^{27}{+}w^8u^{26}{+}w^{11}u^{25}{+}w^{13}u^{24}{+}w^2u^{23}{+}w^5u^{22}{+}w^9u^{21}{+}w^7u^{20}{+} \\ &w^{12}u^{19}{+}w^4u^{18}{+}w^9u^{17}{+}w^{13}u^{16}{+}w^4u^{15}{+}w^{13}u^{14}{+}u^{12}{+}wu^{11}{+}w^3u^{10}{+}w^6u^9{+}w^8u^8{+}w^7u^7{+} \\ &w^{14}u^6{+}u^5{+}w^5u^4{+}w^9u^2{+}w^7u{+}w^9,\ w^7u^{30}{+}w^3u^{29}{+}w^4u^{28}{+}wu^{27}{+}w^6u^{26}{+}w^7u^{25}{+}wu^{23}{+} \\ &w^6u^{22}{+}w^7u^{21}{+}w^9u^{19}{+}w^9u^{18}{+}w^2u^{16}{+}w^5u^{15}{+}w^2u^{13}{+}w^5u^{12}{+}u^{11}{+}w^6u^{10}{+}u^9{+}w^2u^8{+} \\ &w^5u^7{+}w^7u^6{+}w^2u^5{+}w^9u^4{+}w^2u^3{+}w^7u^2{+}w^3u{+}w^{13}). \end{aligned}$$

Our task is to solve the following logarithm problem in $J_{\text{C124}}(\mathbb{F}_{2^4})$: find the integer $l \in [0, r-1]$ such that $(D_2 - D_3) = l(D_1 - D_3)$.

ECDLP and HCDLP Solutions. Our implementation of the Enge-Gaudry algorithm obtained

$$l = 28969719448201630335077609980735 4482.$$

Finally, we verified that $Q = lP$ on E124.