On the Static Diffie-Hellman Problem on Elliptic Curves over Extension Fields

Robert Granger

Claude Shannon Institute
School of Computing, Dublin City University
Glasnevin, Dublin 9, Ireland
rgranger@computing.dcu.ie

Abstract. We show that for any elliptic curve $E(\mathbb{F}_{q^n})$, if an adversary has access to a Static Diffie-Hellman Problem (SDHP) oracle, then by making $O(q^{1-\frac{1}{n+1}})$ SDHP oracle queries during an initial learning phase, for fixed $n > 1$ and $q \to \infty$ the adversary can solve *any* further instance of the SDHP in heuristic time $\tilde{O}(q^{1-\frac{1}{n+1}})$. This reduces the security of elliptic curves defined over $\mathbb{F}_{p^2}$ and $\mathbb{F}_{p^4}$ proposed by Galbraith, Lin and Scott at EUROCRYPT 2009, should these curves be used in any protocol where a user can be made to act as a proxy SDHP oracle. Our algorithm also solves the *Delayed Target DHP* as defined by Freeman; we provide further algorithms for solving the *Delayed Target DLP*, the *One-More DHP* and *One-More DLP* as studied by Koblitz and Menezes in the context of jacobians of hyperelliptic curves of small genus. We correct an oversight in their analysis, and show that for *any* group in which index calculus can be effectively applied, the above problems have a natural relationship, and will *always* be easier than the DLP.

Keywords: Static Diffie-Hellman problem, elliptic curves.

1 Introduction

In recent years, there has been a steadily growing appreciation of the existence of an apparent separation between the difficulty of solving a single instance of some hard problem, and the difficulty of breaking a cryptosystem whose security is related to or depends upon it, when used in practice. In 2004 for example, Brown and Gallant showed that given access to a *Static Diffie-Hellman Problem* (SDHP) oracle, in the best case one can compute a static Diffie-Hellman secret in a group of order n with only $O(n^{1/3})$ SDHP oracle queries and $O(n^{1/3})$ group operations [6]. For cryptographically interesting elliptic curves, i.e., those for which generic attacks are the best known, this result is in stark contrast to the time required to compute discrete logarithms, namely $O(n^{1/2})$. In the protocols [15, 17] and [7] for instance, a user can be made to act as a proxy SDHP oracle, thus rendering such systems vulnerable to this attack. So while solving the SDHP in this case may still be hard, it has lower complexity than the best

discrete logarithm problem (DLP) - and hence Diffie-Hellman problem (DHP) - algorithms.

In 2006 Cheon rediscovered the Brown-Gallant algorithm when the requisite information is provided in the guise of the Strong Diffie-Hellman Problem [8]. Cheon also extended the attack to utilise divisors of $n+1$ as well as of $n-1$, as with the Brown-Gallant algorithm; indeed both algorithms can be regarded as instances of the well-known reduction from the DLP to the DHP due to den Boer, Maurer, Wolf *et al.,* (see [38] for a survey), but with restricted access to a DHP oracle. Incidently Cheon's break of the Strong DHP does not in itself reveal any weakness in the protocols that depend upon it, since the reduction given in security proofs had until then been in the wrong direction. In the case of Boneh-Boyen signatures [4], in 2009 Jao and Yoshida proved the reduction in the reverse direction, thus providing a proof of security, and an attack on the scheme with complexity $O(n^{2/5+\epsilon})$, when $O(n^{1/5+\epsilon})$ signature queries are performed [30].

For the RSA problem, in 2007 Joux, Naccache and Thomé showed that with initial subexponential access to an e-th root oracle an attacker can later compute the e-th root of any element with complexity lower than that required to factor the modulus [34]. This algorithm was then adapted to solve the SDHP in finite fields [32], with similar efficiency improvements.

It is therefore natural to ask whether initial access to an SDHP oracle can aid in solving later SDHP instances faster than solving the DLP in the context of elliptic curves? Depending on the divisors of $n \pm 1$, one can always use the Brown-Gallant-Cheon algorithm to reduce the computational complexity at the cost of making SDHP oracle queries. However for elliptic curves defined over extension fields $\mathbb{F}_{q^n}$, we present a family of algorithms which for fixed $n > 1$ and $q \to \infty$ requires potentially far fewer oracle queries and far less computation. We also present a subexponential SDHP algorithm for a family of finite fields. In all cases, the resulting SDHP algorithm is nearly a square root faster than the fastest known DLP algorithm.

As exemplified in the case of the Strong DHP and Cheon's algorithm, there has also been a growing appreciation of the fact that in some security proofs, the reduction to a hard problem is often in one direction only, which means algorithms that solve the resulting hard problem do not result in a break of the system. Koblitz and Menezes have studied this situation in detail [35] and have given examples of groups for which the expected relations between the difficulty of such problems arising from a variety of protocols do not appear to behave as one expects (see also [36, 37]). We point out a simple oversight in their analysis and show that in the context of *any* group in which index calculus is effective, i.e., provides the best known algorithm for the given problems, which includes jacobians of hyperelliptic curves and elliptic curves over extension fields, the aforementioned problems do indeed have natural relationships. Not only does this alter their conclusions significantly, but this enables us to argue that there exist numerous cryptographically interesting groups for which some cryptographically interesting problems will *always* be easier than the DLP. Hence should reductions

in the reverse direction be found, the security assurances provided by these groups will most likely be weaker than desired.

The sequel is organised as follows. In §2 we recall the SDHP. In §3 we motivate our main idea, present our basic algorithm, and analyse asymptotic variants of it. Then in §4 we detail existing proposals in the literature that are vulnerable to our attack. In §5 we give a full account of our experimental implementation at the 128-bit security level for extension degrees $n = 2, 3, 4$ and 5, and assess their impact on the above curves. In §6 we present algorithms for three other problems which arise in cryptographic protocols and analyse their impact, and make some concluding remarks.

2 The Static Diffie-Hellman Problem

Let $\mathbb{G}$ be a cyclic group of prime order p, and let g be fixed generator of $\mathbb{G}$. The classical Diffie-Hellman problem in $\mathbb{G}$ can be stated as follows [12]:

Problem 1. (DHP): Given g and random g^x and g^y, find g^{xy}.

In Diffie-Hellman (DH) key agreement between two parties, Alice chooses a random secret $x \in \mathbb{Z}/p\mathbb{Z}$ and computes g^x, while Bob chooses a random secret $y \in \mathbb{Z}/p\mathbb{Z}$ and computes g^y, which are then exchanged. Upon receipt each party computes the shared secret g^{xy} by exponentiating the other party's group element by their own secret. A fundamental security requirement of DH key agreement is that the DHP should be hard.

Should Alice for any reason repeatedly reuse the same secret, $x = d$ say, then the resulting problem is a strict subproblem of the DHP. This problem is referred to as the Static DHP, which we state as follows:

Problem 2. ($SDHP_d$): Given fixed g and g^d, and random g^y, find g^{dy}.

Note that this situation need not just arise as an efficiency measure during multiple DH key agreements (Alice need only compute g^d once and reuse this value for multiple key agreements), but may also arise in text-book El-Gamal encryption [15], Ford-Kaliski key retrieval [17] and Chaum-Van Antwerpen's undeniable signatures [7].

As in [6] we define an oracle for solving the SDHP as follows:

Definition 1. *($SDHP_d$ Oracle). Let $\mathbb{G}$ be a cyclic group of prime order p, written additively. Let $d \in \mathbb{Z}/p\mathbb{Z}$ be arbitrary. An $SDHP_d$ oracle (with respect to $\mathbb{G}$) computes the function $\delta : \mathbb{G} \to \mathbb{G}$ defined by:*

$$\delta(X) = dX.$$

We now consider how to solve the SDHP when $\mathbb{G} = E(\mathbb{F}_{q^n})$.

3 An SDHP Algorithm for $E(\mathbb{F}_{q^n})$

In this section we motivate and present our algorithm for solving the SDHP in the present context.

The key observation in [34] is that if one is able to define a suitable 'factor base' in the group under consideration, i.e., a relatively small subset of group elements over which a non-negligible proportion of all group elements can be 'factored' via the group operation, then it is possible to solve the SDHP with input an arbitrary group element, given knowledge of the action of the SDHP oracle on the factor base elements alone. This follows from the simple fact that if in an additively written group $\mathbb{G}$ we have $R = P_1 + \cdots + P_n$, with P_i in some factor base $\mathcal{F}$, then

$$\mathrm{SDHP}_d(R) = dR = dP_1 + \cdots + dP_n.$$

Note that if an arbitrary group element R is not expressible over the factor base, then by adding random elements $P_r \in \mathcal{F}$ to R and testing expressibility, one can produce an element $R + P_r$ which factors over $\mathcal{F}$, thus permitting the SDHP to be solved as before. Therefore a good factor base over which a non-negligible proportion of elements may be expressed, combined with randomisation, enables one to solve the SDHP for arbitrary group elements.

Observe that in contrast to the DHP, for the SDHP one does not need to know d in order to compute it's action on an arbitrary element of $\mathbb{G}$, and the implicit information 'leaked' via the SDHP oracle calls enables one to solve the SDHP using the above observation more readily than one is able to factor a modulus in RSA, or solve the DLP in $\mathbb{F}_p$, for example, for which there is no such information. This same idea is also applied in both [32] and [35].

When $\mathbb{G}$ is the multiplicative group of a finite field, the problem of how best to construct a factor base, and how to express arbitrary elements over such a factor base is well studied [33, 31, 32]. For finite fields there exists a natural notion of size for elements, or equivalently a norm function, given by either the absolute value of an element for prime fields, or the degree of an element for extension fields, or a combination of both depending on the algorithm being used to generate multiplicative relations. A norm function imbues a notion of smoothness for a group and those elements of small norm generate more group elements than those elements of larger norm, hence the best choice for a factor base is those elements of norm up to some bound.

In the context of elliptic curves over prime fields, there does not appear to be a utilisable notion of norm that enables the selection of a factor base that generates a higher proportion of group elements than any other, nor a means by which to factor elements over one should one be chosen. It is precisely this issue that has so far precluded the discovery of a successful native index calculus

algorithm for computing discrete logarithms on such curves[1], which is why they are so attractive from a security perspective.

For elliptic curves over extension fields, the story is very different. While the 'Weil descent' methodology [19, 25, 28] has proven successful for solving or weakening the DLP in some cases, this involves mapping to a generally larger group, which although possessing a natural factor base, does not allow the requisite SDHP oracle queries to be made on the preimages of the factor base elements, since in general such preimages will not exist. There does however exist a notion of smoothness for such elliptic curves, as remarkably discovered by Gaudry [24].

3.1 Gaudry's insight

Developing upon an intriguing idea due to Semaev [43], in 2004 Gaudry showed how to define a useful factor base for $E(\mathbb{F}_{q^n})$, over which elements can be 'factored', or more properly, decomposed, which leads to a an index calculus algorithm for computing logarithms over these curves. For fixed $n > 1$ and $q \to \infty$, the algorithm has complexity $O(q^{2-\frac{2}{n}})$, which is much faster than the Pollard rho complexity $O(q^{n/2})$.

We begin by recalling Semaev's *Summation Polynomials*.

Definition 2. *For* $\mathrm{char}(\mathbb{F}_q) > 3$ *let* E *be an elliptic curve defined over* $\mathbb{F}_{q^n}$ *by the equation* $y^2 = x^3 + ax + b$. *The summation polynomials* f_n *of* E *are defined by the following recurrence, with initial values for* $n = 2$ *and* 3 *given by* $f_2(X_1, X_2) = X_1 - X_2$, *and*

$$f3(X_1, X_2, X_3) = (X_1 - X_2)^2 X_3^2 - 2((X_1 + X_2)(X_1X_2 + a) + 2b)X_3 + ((X_1X_2 - a)^2 - 4b(X_1 + X_2)),$$

and for $n \geq 4$ *and* $1 \leq k \leq n - 3$,

$$f_n(X_1, \ldots, X_n) = Res_X(f_{n-k}(X_1, \ldots, X_{n-k-1}, X), f_{k+2}(X_{n-k}, \ldots, X_n, X)).$$

While this definition may appear rather mysterious, Semaev derived the above formulae by insisting that f_n satisfies the following property, which relates f_n to the addition law on E.

Theorem 1. *Let* E *be an elliptic curve over a field* k, $n \geq 2$ *and* f_n *its* n*-th summation polynomial. Let* $x_1, \ldots, x_n$ *be* n *elements of an algebraic closure* $\hat{k}$ *of* k. *Then* $f_n(x_1, \ldots, x_n) = 0$ *iff there exists an* n*-tuple* $(y_1, \ldots, y_n)$ *of elements in* $\hat{k}$ *such that for all* i, $P_i = (x_i, y_i)$ *is a point on* E *and*

$$P_1 + \cdots + P_n = \mathcal{O}.$$

[1] There are of course attacks that apply to a very small minority of elliptic curves [39, 20, 44, 42, 41], though these are well understood and are easily avoided, or in the case of pairing-based cryptography, which relies on curves which are susceptible to [39, 20], are employed.

One can therefore see immediately that f_n provides an encoding for all sets of n points on a given curve whose sum is the identity element. For an elliptic curve E over a prime field $\mathbb{F}_p$, Semaev proposed setting the factor base to be the set be all points on E whose abscissa have magnitude less than $p^{1/n}$. Then one computes random multiples of some base point P, say $R_i = r_i P$, and attempts to write each such R_i as a sum of n points in the factor base. To do this one need only solve

$$f_{n+1}(x_1, \ldots, x_n, x_{R_i}) = 0. \tag{1}$$

By symmetry, one expects this to be possible for a proportion $1/n!$ of points R_i, and when $O(p^{1/n})$ points that decompose have been found (the approximate size of the factor base) one can obtain their logarithms w.r.t. P via a sparse linear algebra elimination, which has complexity $O(p^{2/n})$. Finding the logarithm of an arbitrary group element is then easy. Therefore, if finding small roots of (1) were possible, for fixed $n \geq 5$ and $p \to \infty$ this algorithm would be faster than Pollard rho.

Unfortunately, finding such small roots, at least for more than two variables [9], appears hard. Gaudry's insight was to observe that for elliptic curves over extension fields, if one uses a factor base consisting of points whose abscissae form a one-dimensional subvariety of $E(\mathbb{F}_{q^n})$, then the Weil restriction of scalars of equation (1) from $\mathbb{F}_{q^n}$ to $\mathbb{F}_q$ forms an algebraic system of n equations in n indeterminates, which is nearly always zero-dimensional and which can be solved via elimination theory, Using a 'double large prime variation' [26] this leads to a DLP algorithm with complexity $O(q^{2-\frac{2}{n}})$. We are now ready to present the basic version of our algorithm, in which we detail how this Weil restriction approach works.

3.2 Basic SDHP Algorithm

Let E be an elliptic curve whose field of definition is $\mathbb{F}_{q^n}$. We define a factor base $\mathcal{F}$ à la Gaudry as follows:

$$\mathcal{F} = \{P = (x, y) \in E(\mathbb{F}_{q^n}) \mid x \in \mathbb{F}_q\}.$$

On heuristic grounds, one expects $|\mathcal{F}| \approx q$, see [24]. For each $P \in \mathcal{F}$ we make an oracle call to the SDHP-oracle, to give $\mathrm{SDHP}_d(P) = dP$.

For an arbitrary point $R \in E(\mathbb{F}_{q^n})$, the goal is to find dR. We attempt write R as a sum of n elements of $\mathcal{F}$, i.e.,

$$R = P_1 + \cdots + P_n.$$

By symmetry, one expects the proportion of elements expressible in such a way to be approximately $1/n!$. To perform this decomposition one uses Semaev's summation polynomial f_{n+1}, and attempts to solve

$$f_{n+1}(x_1, \ldots, x_n, x_R) = 0 \in \mathbb{F}_{q^n}. \tag{2}$$

Note that the expression on the left of equation (2) involves the defining coefficients of the curve E, and the abscissa x_R, all of which are in $\mathbb{F}_{q^n}$. Assume that the extension $\mathbb{F}_{q^n}/\mathbb{F}_q$ has the polynomial basis $\{1, t, \ldots, t^{n-1}\}$. Then each one of the n coefficients of powers of t must be zero. Since each of the n abscissae x_i are in $\mathbb{F}_q$, equation (2) defines a variety with n equations in n indeterminates over $\mathbb{F}_q$, which we solve via a Grobner basis computation, see §5.

If there is a solution $(x_1, \ldots, x_n)$ to the system (2), then one needs to compute all 2^n possible combinations $\pm P_1 \pm \cdots \pm P_n$ for the corresponding ordinates in order to find the correct combination which sums to R. Then the solution to the SDHP for R is immediate:

$$\mathrm{SDHP}_d(R) = dR = dP_1 + \cdots + dP_n,$$

where all the terms on the right hand side are already known, due to the oracle queries on $\mathcal{F}$.

If a solution does not exist, then one adds to R a random element $P_r \in \mathcal{F}$ (or any linear combination thereof) and attempts to decompose this point once again. One expects this to succeed after approximately $n!$ attempts. When it does we have the following equation:

$$R + P_r = P_1 + \cdots + P_n,$$

which implies that

$$\mathrm{SDHP}_d(R) = dR = dP_1 + \cdots + dP_n - dP_r,$$

where again all the terms on the right hand side are already known. Hence our SDHP_d instance is solved.

3.3 Discussion

Our first observation is that the above algorithm and this discussion of it are entirely heuristic; however we believe that the algorithm and its complexity can be made completely rigorous using the results of Diem [10, 11], should one choose to do so, see §3.4.

Our second observation - which is fundamental to the complexity of the algorithm - is that in contrast to the DLP, there is no linear algebra elimination, since only a single relation is sought. So once the initial oracle querying phase is complete, the complexity of the algorithm depends only on the problem of computing one relation. We therefore analyse this cost now.

For $n \geq 3$, Semaev's summation polynomials $\{f_n\}$ are symmetric and are of degree 2^{n-2} in each variable. Hence equation (2) is of degree 2^{n-1} each variable. In order to simplify the system greatly, it pays to express f_{n+1} in terms of the elementary symmetric functions $e_1, \ldots, e_n$ in the variables $x_1, \ldots, x_n$. We then have a system of n equations in the n indeterminates $e_1, \ldots, e_n$ each of which again has degree bounded by 2^{n-1} in each variable. In order to solve this system, we perform a Gröbner basis computation.

In practice our experiments showed that the Gröbner basis w.r.t the lexicographic ordering always satisfies the so-called shape lemma, i.e., it is of the following form:

$$e_1 - g_1(e_n), e_2 - g_2(e_n), \ldots, e_{n-1} - g_{n-1}(e_n), g_n(e_n), \tag{3}$$

where $g_i(e_n)$ is a univariate polynomial in e_n for each i. In general the degree of the univariate polynomial in e_n that we obtain will be $2^{n(n-1)}$ and indeed in our experiments this is borne out. The complexity of Faugere's algorithm F4 [16] to compute this basis is therefore at least

$$\tilde{O}(Poly(2^{n(n-1)})).$$

Since this is doubly exponential in n, this makes the algorithm practical only for very small values of n. However for fixed n and $q \to \infty$, this is polynomial in $\log q$.

To find whether or not the system has roots $e_1, \ldots, e_n \in \mathbb{F}_q$, one extracts the linear factors of the univariate polynomial $g_n(e_n)$ using the Cantor-Zassenhaus algorithm and then substitutes each $\mathbb{F}_q$ root e_n into $g_i(e_n)$ to find $e_{n-1}, \ldots, e_1$. For each such vector of $\mathbb{F}_q$ roots $(e_1, \ldots, e_n)$ one tests whether the polynomial

$$p(x) = x^n - e_1 x^{n-1} + e_2 x^{n-2} - \cdots - (-1)^n e_n \tag{4}$$

splits over $\mathbb{F}_q$. If it does then these roots are the abscissae of points in $E(\mathbb{F}_q)$, and there exists a linear combination

$$\epsilon_1 P_1 + \cdots + \epsilon_n P_n \tag{5}$$

with $\epsilon_i \in \{-1, 1\}$ which sums to R. This step is also polynomial in $\log q$.

On average one expects to have to perform $n!$ such decompositions in order to find a relation. Therefore the complexity of the our basic SDHP algorithm for fixed $n > 1$ and $q \to \infty$ is polynomial in $\log q$. This gives the following heuristic result.

Heuristic Result 1. *For any elliptic curve $E(\mathbb{F}_{q^n})$, by making $O(q)$ queries to an SDHP oracle during an initial learning phase, for fixed $n > 1$ and $q \to \infty$, an adversary can solve any further instance of the SDHP in time $\tilde{O}(1)$.*

Note that prior to the learning phase, the adversary needs to construct the factor base by testing whether a given abscissa $x \in \mathbb{F}_q$ gives a point lying on E or not. We incorporate this computation into the learning phase, since it has the same complexity of $\tilde{O}(q)$. The solving phase then has complexity $\tilde{O}(1)$. It is of course possible to balance the cost of these phases, which we now consider.

3.4 Balancing the setup and relation-finding costs

To balance the cost of the oracle querying phase and the relation finding phase, one needs to reduce the size of the factor base by some proportion. To this end,

Let $|\mathcal{F}| = q^\alpha$, with $0 < \alpha \leq 1$. Then given the decomposition of a random point $R \in E$ as a sum of points whose abscissa are in $\mathbb{F}_q$, the probability that each abscissa is in $\mathcal{F}$ is $q^{\alpha-1}$. Assuming these events are independent, the probability that all n abscissae are in $\mathcal{F}$ is $q^{n(1-\alpha)}$. Hence in order to obtain one relation, one expects to have to perform $1/q^{n(1-\alpha)} = q^{n(1-\alpha)}$ successful decompositions.

Asymptotically for fixed $n > 1$ and $q \to \infty$ one can regard the cost of a decomposition as unital (modulo some log factors) and hence to balance the two stages α must satisfy:

$$q^\alpha = q^{n(1-\alpha)},$$

and so $\alpha = n/(n+1) = 1 - \frac{1}{n+1}$. This gives the following heuristic result as stated in the abstract.

Heuristic Result 2. *For any elliptic curve $E(\mathbb{F}_{q^n})$, by making $O(q^{1-\frac{1}{n+1}})$ queries to an SDHP oracle during an initial learning phase, for fixed $n > 1$ and $q \to \infty$, an adversary can solve any further instance of the SDHP in time $\tilde{O}(q^{1-\frac{1}{n+1}})$.*

Observe that there is no possibility (nor necessity) for considering so-called large primes, i.e., those with absicissa in $\mathbb{F}_q$ but not lying in $\mathcal{F}$, since there is no linear algebra elimination step on the single relation. If we compare the above complexity to that obtained by Gaudry for the DLP - $O(q^{2-\frac{2}{n}})$, which uses a double large-prime variant - we see that our algorithm for solving the SDHP is nearly a square root faster. Intuitively this difference in complexity arises from there not being a linear algebra step in the solution of the SDHP.

We note that Diem has given a rigorous algorithm that is essentially equivalent to Gaudry's DLP algorithm above [10], which for fixed $n \geq 2$ solves the DLP on any elliptic curve over $\mathbb{F}_{q^n}$ in proven expected time $q^{2-2/n}(\log q)^{O(1)}$. We believe his treatment can be adapted *mutatis mutandis* to transform the above two heuristic results into theorems, though since it is not the primary focus of this paper, we have not verified this here.

Observe that in practice the limiting factor is not the decompositions, but the oracle queries, since these would typically be performed on a single server, whereas the former can be easily distributed. One can therefore reduce the number of such queries below the above threshold, at the cost of needing to perform more decompositions. Such a trade-off is easily optimised, based on the amount of computing power available, but will nevertheless require an exponential number of oracle queries, for fixed n and $q \to \infty$.

3.5 Subexponential SDHP algorithm via Diem's algorithm

Diem has also proven the following remarkable result. For $n \to \infty$ and assuming $n = O(\sqrt{\log q})$, the DLP over any elliptic curve $E(\mathbb{F}_{q^n})$ can be solved in expected time $q^{O(1)} = e^{O(\log (q^n)^{2/3})}$, see [11]. Thus for a family of finite fields, any elliptic curve DLP can be solved using a *native* subexponential index calculus algorithm.

While Diem is not precise in his analysis of the exponents in the complexity of the constituent parts of the algorithm, it is clear that since for the SDHP

there is no linear algebra step, one expects a similar improvement over the DLP algorithm in this context to the fixed n case, i.e., nearly square root, and that this also can be rigorously proven. This therefore provides an SDHP algorithm that requires a subexponential number of oracle queries. We leave it as an open problem to find the precise complexity of Diem's algorithm, and the resulting complexity of our SDHP algorithm in this context.

4 Potentially Vulnerable Curves

In this section we consider curves in the literature that are potentially vulnerable to our attack.

At EUROCRYPT 2009, Galbraith, Lin and Scott proposed the use of special elliptic curves over $\mathbb{F}_{p^2}$ and $\mathbb{F}_{p^4}$ [21], which possess efficiently computable homomorphisms that permit a particularly efficient version of Gallant-Lambert-Vanstone point multiplciation method [23]. As well as the single bit speed-up of Pollard rho available on these curves, both the GHS attack [25] and Gaudry's attack [24] are considered, and appropriate recommendations are made in light of these. In particular, for curves over $\mathbb{F}_{p^2}$, neither of these attacks is faster than Pollard rho, and so the use of these curves may be considered 'risk free'. For curves over $\mathbb{F}_{p^4}$, in light of the latter attack the authors recommend that primes of length 80 bits should be used to achieve 128-bit security, rather than of length 64 bits, although it is stated that this is a very conservative choice, since Gaudry's algorithm requires expensive computations, and so potentially smaller primes could be used. Similarly Hankerson, Karabina and Menezes have considered the GLS point multiplication method over binary fields of the form $\mathbb{F}_{q^2}$ [27].

Prior to our attack, the only potential weakness of cryptographically interesting curves over $\mathbb{F}_{p^2}$ would be due to the Brown-Gallant-Cheon attack. In the best case (from an adversary's perspective), should the group order ± 1 be divisible by an integer of size $O(p^{1/3})$, then the SDHP secret can be computed in time $\tilde{O}(p^{2/3})$. Such a condition can be easily avoided should this attack be a concern. For the curves considered in [27], the Weil descent method is analysed and it is shown that the proportion of susceptible curves is negligible and can be provably avoided with a feasible computation. However, regardless of the divisibility properties of the group order ± 1, the balanced SDHP algorithm from §3.4 achieves a complexity of $\tilde{O}(p^{2/3})$ (and similarly for the binary curves). Assuming that point decompositions over the factor base can be computed efficiently, this attack may therefore pose a real threat.

For curves over $\mathbb{F}_{p^4}$, our attack has complexity $\tilde{O}(p^{4/5})$, which is much faster than Gaudry's attack on the DLP, which has complexity $\tilde{O}(p^{3/2})$. Again assuming that point decompositions can be performed efficiently, curves over degree 4 extensions may also be vulnerable.

Also of interest but lesser so are the legacy curves which until recently formed part of the Oakley Key Determination Protocol, a part of IPSEC. These are the 'Well Known Groups' 3 and 4 [29] which are elliptic curves defined over the fields

$\mathbb{F}_{2^{155}}$ and $\mathbb{F}_{2^{185}}$, and which have been the target of numerous attempted attacks via the Weil descent method [45, 25, 22, 40], since their inception.

In the following section we address whether our attack poses a threat to these systems in practice.

5 Experimental Results

In this section we detail the results of an implementation of our SDHP algorithm using the computational algebra system MAGMA [5] (V2.16-5), which was run on an Intel Xeon running at 3.16GHz with 32G of memory. We considered two sets of curves. The first set consisted of four randomly produced curves of prime order, each of which were 256 bits in length, for fields of the form $\mathbb{F}_{p^2}$, $\mathbb{F}_{p^3}$, $\mathbb{F}_{p^4}$ and $\mathbb{F}_{p^5}$, see §5.1 and §5.2. These curves were chosen in order to measure how vulnerable the curves proposed in [21] are to our algorithm. We also provide estimates for solving the DLP on these curves via Pollard rho and the state of the art index calculus algorithms.

The second set consisted of four randomly produced curves of order $4 \cdot p$ with p of bitlength 256 over the binary fields $\mathbb{F}_{2^{ln}}$, for $n = 2, 3, 4$ and 5, so that ln is as close to 256 as possible. The reason for implementing the attack on these curves was twofold: firstly to assess the security of the curves proposed in [27]; and secondly to compare the efficiency of the attack with the prime field case, with a view to assessing whether it is possible to break the SDHP on the Oakley curves, see §5.3.

While our implementations in all cases are clearly sub-optimal, our goal was to provide a proof-of-concept implementation. Our results give a reasonable indication of what can be achieved in practice, and indeed provide an upper bound for the time required to solve the SDHP in each case. With a tailored and optimised low-level implementation our attack times could no doubt be improved significantly.

5.1 Large prime characteristic

For each of $n = 2, 3, 4$ and 5 we used curves of the form

$$E(\mathbb{F}_{p^n}) : y^2 = x^3 + ax + b,$$

for a and b randomly chosen elements of $\mathbb{F}_{p^n}$, such that $\#E(\mathbb{F}_{p^n})$ was a prime of bitlength 256.

For $n = 2, 3$ and 4 we computed the symmetrised summation polynomials f_3, f_4 and f_5 respectively, and all experiments were completed within two hours. For the computation of f_6, we surprisingly ran out of memory, and so instead independently symmetrised the two f_4 polynomials used in the resultant computation to reduce the number of terms, and substituted x_R into this partially symmmetrised version of f_6. One can extract the elementary symmetric polynomials from these two independent sets by appropriately recombining them. The

resulting Gröbner basis computation eventually exhausted the available memory and so the $n = 5$ experiments were unfortunately not completed. Without an accurate idea of how long the Gröbner basis computation might take were we to have sufficient available memory, we consider finding relations for curves over these fields to be impractical given our resources at the present time. Note however that for prime base fields, we know of no proposals in the literature for the use of degree five extension fields for elliptic curve cryptography. We therefore include results only for $n = 2, 3$ and 4, in Table 1.

Table 1. Data for testing and decomposing points for elliptic curves over extension fields. Times are in seconds.

n	$\log p$	$\#f_{n+1}$	$\#$ symf_{n+1}	T(GB)	T(roots)
2	128	13	5	0.001	0.009
3	85.3	439	43	0.029	0.027
4	64	54777	1100	5363	3.68

The column titles in the table denote respectively: the degree of the extension field; the size of the prime base field in bits; the number of monomials in f_{n+1}; the number of monomials in f_{n+1} once symmetrised; the average time required to perform a Gröbner basis computation; and the average time required to find the points that sum to the point being decomposed respectively.

As per §3.3 the last of these consists of the extraction of the degree one factors of the polynomial $g_n(e_n)$ and then substitutes the roots into the remaining polynomials $g_i(e_n)$ in equation (3). This is followed by the desymmetrisation factorisation (equation (4)) and then computation of the correct linear combination of factor base elements that sum to P (equation (5)).

As one can see, symmetrisation reduces the size of the system greatly. Note that the only setup cost comes from computing f_{n+1} and its symmetrisation; the final two columns give the average decomposition cost per input point, which for $n = 2$ and 3 is over 1000 inputs includes both those that do decompose over $\mathcal{F}$, as well as those that do not.

For $n = 4$, since the computation is significantly more costly, we report the time for one input point only; note that the input system for the Gröbner basis computation always has the same form but with different coefficients, and hence one expects this part of the computation to be very consistent. With regards to the root finding time, the three stages described above took $3.68s, 0.00s$ and $0.04s$ respectively, and so the dominant cost is the initial factorisation, which is necessary whether an input point decomposes or not. Hence we estimate the average time over uniformly chosen input points to be $\approx 3.68 + 0.04/4! \approx 3.68s$, since a point decomposes with probability $1/4!$.

5.2 Upper bounds on attack times

From the data in Table 1 and the time required to compute a scalar multiplication, one can compute an upper bound on the time required to carry out the attack in §3.4. Setting $|\mathcal{F}| = p^\alpha$, a minimising α balances the two stages of the attack, namely the oracle calls, and the relation finding stage. We ignore the cost of constructing the factor base since this only involves a handful of field operations and a Legendre symbol computation. A more careful version of the argument of §3.4 leads to the following equation:

$$p^{n(1-\alpha)} \cdot n! \cdot (T(\text{GB}) + T(\text{roots})) = p^\alpha \cdot T(\text{scalar}),$$

where $T(\text{scalar})$ denotes the average cost of a scalar multiplication. With our implementation the latter costs approximately $0.008s$, $0.011s$ and $0.012s$ on the curves defined over $\mathbb{F}_{p^2}$, $\mathbb{F}_{p^3}$ and $\mathbb{F}_{p^4}$ respectively. Table 2 details the resulting values of alpha for $n = 2, 3, 4$ and the corresponding attack times.

Table 2. Attack data for our implementation. Times are in seconds.

n	α	Attack time	Pollard rho
2	0.6701 (2/3)	$2^{79.8}$	$2^{111.3}$
3	0.7645 (3/4)	$2^{59.7}$	$2^{111.4}$
4	0.8730 (4/5)	$2^{50.5}$	$2^{111.4}$

The Pollard rho attack times have been estimated as $\sqrt{\pi \cdot 2^{256}/2}$ group operations, where the cost of a group operation has been estimated using the $T(\text{scalar})$ times above, assuming use of the double and add algorithm.

We have incorporated the speed-up afforded by performing random walks on equivalence classes of points [14, 47] when the set of points $\{\pm\psi^i(P) : 0 \leq i < m\}$ for a given point P are deemed to be equivalent, where ψ is the homomorphism from [21]. This results in the three curves have virtually identical security.

Pollard rho however is not the fastest asymptotic DLP algorithm in this context. In the basic index calculus one finds $O(p)$ relations with a linear algebra cost of $O(p^2)$. Assuming the decomposition cost is sufficiently small, one can reduce the size of the factor base to balance the cost the cost of the two stages, to $O(p^{2-\frac{2}{n+1}})$, which is originally due to Harley. In addition, one can also use single and double large prime variations [46, 26], resulting in complexities of $O(p^{2-\frac{2}{n+1/2}})$ and $O(p^{2-\frac{2}{n}})$ respectively.

Our implementation allows one to give upper bounds for the attack times for each of these approaches, and consequently provides information regarding what size of p should be chosen to provide 128 bit security, for each n, subject to our attack implementation. This security level is the length of time required to compute 2^{128} basic group operations.

Note that in the double large prime variation, for the most interesting case $n = 4$ the number of relations required is $O(p^{3/2})$. With our decomposition implementation, the time for the relation generation stage is $p^{3/2} \cdot 4! \cdot 5366.68s \approx 2^{113.0}s$, which is comparable to Pollard rho. Hence for this security level, p of length 64 bits would appear to be secure.

However, in a real attack the decomposition time could clearly be improved, necessitating increasing p accordingly to compensate. Furthermore, since the relation generation stage is more costly than the linear algebra, to balance the two stages of the algorithm one would need to increase the factor base size marginally; how this affects the expected runtime we leave as an open problem.

5.3 Characteristic two

For each of $n = 2, 3, 4$ and 5 we used curves of the form

$$E(\mathbb{F}_{2^{ln}}) : y^2 + xy = x^3 + b,$$

for b a randomly chosen element of $\mathbb{F}_{2^{ln}}$, such that $\#E(\mathbb{F}_{2^{ln}})$ was a four times a prime of bitlength 256. Note that this is the form of the Oakley curves [29].

Note that the base fields $\mathbb{F}_{2^l}$ in each case are not necessarily of prime extension degree over $\mathbb{F}_2$. Since our focus was to compare the effect of characteristic for fields of a given size with particular small extension degrees, we disregard any possible DLP weaknesses due to Weil descent for these example curves.

For these curves the summation polynomials are surprisingly simple, and very sparse, making their computation easy, in contrast to the prime base field case. Observe that as a result the size of the f_i and their symmetrisation is much smaller than before, faciliting a much faster Gröbner basis computation for $n = 4$.

Unfortunately, while promising, for $n = 5$ we also had insufficient memory to complete a decomposition, but expect that if both this computation and the prime field computation completed, this one would be far quicker and would require far less memory. For $n = 2, 3$ and 4 the time for a scalar multiplication was $0.014s$. Table 3 details our results.

Table 3. Attack data for testing and decomposing points for elliptic curves over binary extension fields. Times are in seconds.

n	l	$\#f_{n+1}$	$\#$ symf_{n+1}	Time GB	Time roots	α	Attack time
2	129	5	3	0.000	0.008	0.6672 (2/3)	$2^{80.9}$
3	86	24	6	0.005	0.008	0.7572 (3/4)	$2^{60.0}$
4	65	729	39	247	0.88	0.8575 (4/5)	$2^{50.6}$
5	52	148300	638	N/A	N/A	N/A	N/A

Note that despite the α values being smaller for binary fields - due to faster decompositions - the attack times are slightly higher, because the fields are 258 and 260 bits in size, as opposed to 256.

Whether the Oakley curves are immune to this attack remains unclear, but is a question certainly worthy of further investigation if resources are available. Incidentally, the field over which the Group 4 curve is defined has already been shown to be weak in the sense of [40], but only by a small margin in comparison with Pollard rho.

6 Other Cryptographically Relevant Assumptions

The SDHP algorithm presented in this paper also solves the *Delayed Target DHP* (DTDHP), as defined by Freeman [18], which may be phrased as follows: A solver is given $X \in \mathbb{G}$ and initial access to an SDHP oracle for the element X; when the SDHP is removed, the solver is given a random element $Y \in \mathbb{G}$ and must solve the DHP for input (X, Y).

Koblitz and Menezes studied this problem in the context of jacobian's of hyperelliptic curves of small genus [35], along with several other problems, including the *Delayed Target DLP* (DTDLP), the *One-More DHP* (1MDHP) and the *One-More DLP* (1MDLP). In the DTDLP, rather than given access to an SDHP oracle, the solver is given access to a discrete logarithm oracle but the problem is otherwise identical. In the one-more versions the solver is supplied with a challenge oracle that outputs random elements of the group, as well as an SDHP and a DLP oracle respectively. This time however the solver chooses an integer t and must solve t instances of the SDHP or DLP, but is only allowed to use the SDHP or DLP oracle at most $t - 1$ times.

The 1MDHP was first formulated in [3] while the 1MDLP was first formulated in [1] and [2]. Using their given context as an example Koblitz and Menezes argue that the two pairs of problems - the DTDHP and DTDLP, and the 1MDHP and 1MDLP - should each be incomparable to one another. However, we point out that their analysis of the DTDHP and 1MDHP contains an error, since it only considers the impact of the Brown-Gallant-Cheon algorithm and not the index calculus methods they have used for studying the DTDLP and 1MDLP. Had they done so then for jacobians of curves of genus ≥ 3, the complexity for the delayed target problems would be identical, and similarly for the one-more variants.

Indeed, taking the basic SDHP algorithm presented in §3.2, one sees that by changing the SDHP oracle calls to DLP oracle calls, one obtains an otherwise unaltered algorithm and hence their complexities are the same. Similarly any variation in factor base size will give rise to algorithms of the same complexity; the oracle calls themselves are not relevant to the structure of the algorithm, so it should be clear that for any group in which one can identify and use a factor base to generate relations, the DTDHP and DTDLP will have identical complexities, *relative to the algorithm under consideration.* Exceptions to this statement arise when other algorithms such as the BGC algorithm apply for just one of the problems, or are generic, i.e., do not exploit any representational properties of the group.

Similarly for the one-more problem variants, in our context we have the following simple algorithm. We choose the same factor base as in §3.2, and

perform $|\mathcal{F}|$ oracle calls on its elements. Then for each of the $|\mathcal{F}|+1$ challenge elements, we solve the appropriate problem exactly as before. The only difference between the one-more and the delayed target problems is that for the one-more variants we must solve $|\mathcal{F}|+1$ such challenges, and not just one. If we perform the analysis of §3.4 once more we find that the optimal size of $\mathcal{F}$ is given by $\alpha = 1$, exactly as in §4.5 of [35]. Again either oracle can be used for a given relation and so the 1MDHP and 1MDLP have the same complexity, and the same argument as before implies that this observation holds for all groups in which index calculus is effective, i.e., is the best available algorithm.

So we have the rather curious situation that even though one can not necessarily find a natural reduction between two problems, the presence of an index calculus algorithm implies that any representation-specific algorithm that solves one problem, can be used to solve the other automatically. Note that the BGC is generic and hence is not representation-specific.

One should perhaps therefore conclude that these pairs of problems do have a very natural relationship, for elliptic curves over extension fields and for jacobians of hyperelliptic curves of small genus, and indeed any group that permits index calculus. Furthermore, due to the algorithms specified, one can also conclude that in the prescence of index calculus algoithms, all four of these problems (as well as the SDHP) are easier to solve than the DLP, whenever index calculus is effective.

Acknowledgements

The author would like to thank both Steven Galbraith and Alfred Menezes for their comments on an earlier draft of this article.

References

1. M. Bellare, C. Namprempre, D. Pointcheval and M. Semanko, *The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme,* Journal of Cryptology, 16, pp. 185-215, 2003.
2. M. Bellare and A. Palacio, *GQ and Schnorr identification schemes: proofs of security against impersonation under active and concurrent attacks,* Advances in Cryptology - CRYPTO 2002, LNCS 2442, pp. 149-162, Springer-Verlag, 2002.
3. A. Boldyreva, *Efficient threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme,* PKC 2003, LNCS 2567, pp. 31-46, Springer-Verlag, 2003.
4. D. Boneh and X. Boyen, *Short signatures without random oracles,* Advances in Cryptology - EUROCRYPT 2004, LNCS 3027, pp. 56-73, Springer-Verlag, 2004.
5. Wieb Bosma, John Cannon and Catherine Playoust. *The Magma algebra system I: The user language.* J. Symbolic Comput., 24(3-4):235-265, 1997.
6. D.R.L. Brown and R.P. Gallant, *The Static Diffie-Hellman Problem,* `eprint.iacr.org`, Cryptology ePrint Archive, Report 2004/306, 2004.
7. D. Chaum and H. van Antwerpen, *Undeniable signatures,* Advances in Cryptology - Crypto 1989, LNCS 435, pp. 212-217, Springer-Verlag, 1989.

8. J. Cheon, *Security analysis of the Strong Diffie-Hellman problem,* Advances in Cryptology - Eurocrypt 2006, LNCS 4004, pp. 1-11, Springer-Verlag, 2006.
9. D. Coppersmith. *Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known*, Advances in Cryptology - EUROCRYPT 1996, LNCS 1070, pp. 178-189, Springer-Verlag, 1996.
10. C. Diem, *On the discrete logarithm problem in class groups of curves,* Mathematics of Computation, to appear.
11. C. Diem, *On the discrete logarithm problem in elliptic curves,* Preprint, 2009.
12. W. Diffie and M.E. Hellman, *New directions in cryptography,* IEEE Trans. Inform. Theory 22 (6), pp. 644-654, 1976.
13. *Digital Signature Standard (DSS)*, FIPS PUB 186-2, 2000.
14. I. Duursma, P. Gaudry and F. Morain. *Speeding up the Discrete Log Computation on Curves with Automorphisms*, Advances in Cryptology - ASIACRYPT99, LNCS 1716, pp. 103-121, Springer-Verlag, 1999.
15. T. El-Gamal, *A public-key cryptosystem and a signature scheme based on discrete logarithms,* Advances in Cryptology - Crypto 1984, LNCS 196, pp. 10-18, Springer-Verlag, 1985.
16. J.C. Faugère, *A new efficient algorithm for computing Grbner bases (F4),* Journal of Pure and Applied Algebra, 139 (1-3):61–88, 1999.
17. W. Ford and B. Kaliski, *Server-assisted generation of a strong secret from a password,* 9th international workshop on enabling technologies - WET ICE 2000, IEEE Press, 2000.
18. D. Freeman, *Pairing-based identification schemes,* technical report HPL-2005-154, Hewlett-Packard Laboratories, 2005.
19. 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`
20. G. Frey and H.G. Rück, *A remark concerning m-divisibility and the discrete logarithm problem in the divisor class group of curves,* Math. Comp., 62, pp. 865-874, 1994.
21. S.D. Galbraith, X. Lin and M. Scott, *Endomorphisms for Faster Elliptic Curve Cryptography on a Large Class of Curves,* Advances in Cryptology - EUROCRYPT 2009,LNCS 5479, pp. 518-535, Springer-Verlag, 2009.
22. S.D. Galbraith, F. Hess, N.P. Smart, *Extending the GHS Weil Descent Attack,* Advances in Cryptology EUROCRYPT 2002, LNCS 2332, pp. 29-44, Springer-Verlag, 2004.
23. R.P. Gallant, R.J. Lambert and S.A. Vanstone, Faster Point Multiplication on Elliptic Curves with Efficient Endomorphisms, Advences in Cryptology - Crypto 2001, LNCS 2139, pp. 190-200, Springer-Verlag, 2001.
24. P. Gaudry, *Index calculus for abelian varieties of small dimension and the elliptic curve discrete logarithm problem,* Journal of Symbolic Computation 44, pp. 160-1702, 2009.
25. P. Gaudry, F. Hess and N.P. Smart, *Constructive and destructive facets of Weil descent on elliptic curves,* Journal of Cryptology, (15), pp. 19-46, 2002.
26. P. Gaudry, E. Thomé, N. Thériault and C. Diem. *A Double Large Prime Variation for Small Genus Hyperelliptic Index Calculus,* Math. Comp. **76**, No. 257 (2007), pp. 475-492.
27. D. Hankerson, K. Karabina and A.J. Menezes, *Analyzing the Galbraith-Lin-Scott point multiplication method for elliptic curves over binary fields,* IEEE Transactions on Computers, 58, 1411-1420, 2009.

28. F. Hess. *The GHS Attack Revisited,* Advances in Cryptology - EUROCRYPT 2003, LNCS 2656, pp. 374-387, Springer-Verlag, 2003.
29. IETF, The Oakley Key Determination Protocol, IETF RFC 2412, November 1998.
30. D. Jao and K. Yoshida, *Boneh-Boyen Signatures and the Strong Diffie-Hellman Problem,* Pairing 2009, LNCS 5671, pp. 1-16, Springer-Verlag, 2009.
31. A. Joux and R. Lercier *The Function Field Sieve in the Medium Prime Case,* Advances in Cryptology - EUROCRYPT 2006, LNCS 4004, pp. 254-270, Springer-Verlag, 2006.
32. A. Joux, R. Lercier, D. Naccache and E. Thomé, *Oracle-Assisted Static Diffie-Hellman Is Easier Than Discrete Logarithms.* 12th IMA International Conference, Cryptography and Coding 2009, LNCS 5921, pp. 351-367, Springer-Verlag, 2009.
33. A. Joux, R. Lercier, N.P. Smart and F. Vercauteren, *The Number Field Sieve in the Medium Prime Case,* Advances in Cryptology - CRYPTO 2006, LNCS 4117, pp. 326-344, Springer-Verlag, 2006.
34. A. Joux, D. Naccache and E. Thomé, *When e-th roots become easier than factoring,* Advances in Cryptology - ASIACRYPT 2007, LNCS 4833, pp. 13-28, Springer-Verlag, 2007.
35. N. Koblitz and A.J. Menezes, *Another look at non-standard discrete log and Diffie-Hellman problems,* Journal of Mathematical Cryptology, Volume 2, Issue 4, pp. 311326, December, 2008.
36. N. Koblitz and A.J. Menezes, *Intractable problems in cryptography,* Proceedings of Fq9, to appear.
37. N. Koblitz and A.J. Menezes, *The brave new world of bodacious assumptions in cryptography,* Notices of the AMS, 57, 357-365, 2010.
38. U.M. Maurer and S. Wolf. *The Diffie-Hellman Protocol.* Designs, Codes, and Cryptography, volume 19, p. 2000, 1999.
39. A.J. Menezes, T. Okamoto and S.A. Vanstone, *Reducing elliptic curve logarithms to a finite field,* IEEE Trans. Info. Theory, 39, pp.1639-1646, 1993.
40. A. Menezes, E. Teske and A. Weng, *Weak Fields for ECC,* Topics in Cryptology – CT-RSA, 2004, LNCS 2964, pp. 366-386, Springer-Verlag, 2004.
41. T. Satoh and K. Araki, *Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves,* Comm. Math. Univ. Sancti Pauli, 47, pp. 81-92, 1998.
42. I.A. Semaez, *Evaluation of discrete logarithms on some elliptic curves,* Math. Comp., 67, pp.353-356, 1998.
43. I. Semaev, *Summation Polynomials and the discrete logarithm problem on elliptic curves,* `eprint.iacr.org`, Cryptology ePrint Archive, Report 2004/031, 2004.
44. N.P. Smart, *The discrete logarithm problem on elliptic curves of trace one,* Journal of Cryptology, 12, pp. 141-151, 1999.
45. N.P. Smart, *How Secure are Elliptic Curves over Composite Extension Fields?,* Advances in Cryptology - EUROCRYPT 2001, LNCS 2045, pp. 30-39, Springer-Verlag, 2001.
46. N. Thériault. *Index calculus attack for hyperelliptic curves of small genus*, Advances in Cryptology - ASIACRYPT 2003, LNCS 2894, pp. 75-92, SpringerVerlag, 2003.
47. M.J. Wiener and R.J. Zuccherato. *Faster Attacks on Elliptic Curve Cryptosystems*, Selected Areas in Cryptography - SAC 1998, LNCS 1556, pp. 190-200, SpringerVerlag, 1998.