On the Bit Security of Elliptic Curve Diffie–Hellman

Barak Shani
Department of Mathematics, University of Auckland, New Zealand

Abstract

This paper gives the first bit security result for the elliptic curve Diffie–Hellman key exchange protocol for elliptic curves defined over prime fields. About $5/6$ of the most significant bits of the x-coordinate of the Diffie–Hellman key are as hard to compute as the entire key. A similar result can be derived for the $5/6$ lower bits. The paper also generalizes and improves the result for elliptic curves over extension fields, that shows that computing one component (in the ground field) of the Diffie–Hellman key is as hard to compute as the entire key.

Keywords: hidden number problem, bit security, elliptic curve Diffie–Hellman

1. INTRODUCTION

The notion of *hardcore functions* goes back almost to the invention of public key cryptography. Loosely speaking, for a one-way function f, a function b is a *hardcore function for* f if given $f(x)$ it is hard to compute $b(x)$ (while given x, computing $b(x)$ is easy).

The main interest is in functions b that output some bits of x, which gives this research field the name *bit security*. That is, while computing x from $f(x)$ is computationally hard by definition, one tries to assess the hardness of computing partial information about x. This can be done by providing an (efficient) algorithm that computes $b(x)$, or more commonly by reducing the problem of computing x to computing $b(x)$. That is, one provides an (efficient) algorithm that inverts f given an algorithm that computes b on f.

For popular candidates for one-way functions, such as the RSA function ($RSA_{N,e}(x) = x^e \bmod N$) and discrete exponentiation in a subgroup of prime order ($EXP_g(x) = g^x$; g has prime order), all single-bit functions are known to be hardcore. This result, which is standard these days, took more than 15 years to achieve, where year after year small improvements were made. An important aspect to consider is the success in computing $b(x)$. The mentioned result applies to every algorithm that computes $b(x)$ with a non-negligible success over a trivial guess. See [12] for a survey on hardcore functions which presents the developments over the years.

 This article is a minor revision of the version published by Springer-Verlag available at DOI:10.1007/978-3-662-54365-8_15.

The notion of a hardcore function can be generalized to suit the Diffie–Hellman key exchange protocol. Let $(G, \cdot)$ be a group and let $g \in G$. For a function b, given g^u and g^v, we consider the hardness of computing $b(s)$ for (the Diffie–Hellman key) $s = g^{uv}$. Proving bit security for Diffie–Hellman key exchange has known less success than the aforementioned results. For $G = \mathbb{Z}_p^*$, the multiplicative group of integers modulo a prime p, the $\sqrt{\log p} + \log\log p$ most (and least) significant bits of s are hard to compute as s itself [9] (see also [14]; a similar result holds for twice as many consecutive inner bits, as a consequence of [20, Section 5.1]). For $G = \mathbb{F}_{p^m}^*$, the multiplicative group of a finite extension field, represented as a vector space over $\mathbb{F}_p$, computing a single component of s is as hard to compute as s itself [26], which follows from the fact that a single component of a product st is linear in all of the components of s. Moreover, using this linearity, a result in a similar fashion to the case of $G = \mathbb{Z}_p^*$ can be obtained from [23] for a single component (see also [17]). These results need – essentially – a perfect success in computing the partial information.

The case of the elliptic curve Diffie–Hellman key exchange protocol has known even fewer results, mainly because of the inherent nonlinearity of the problem. For elliptic curves over prime fields there are no known (non-trivial) results. For the group of elliptic curve points over an extension field of degree 2, computing a single component of the x-coordinate of s is as hard to compute as s itself [15, Remark 3.1]. This result requires perfect success in computing the component. We mention that for the case of elliptic curves over prime fields it is claimed in [7] that computing the top $(1-\epsilon)$ fraction of bits of the x-coordinate of s, for $\epsilon \approx 0.02$, is as hard as computing all of them, but a proof is not provided, probably since it is a weak result, as the authors mentioned. Obtaining bit security results for elliptic curve Diffie–Hellman keys has been an open problem for almost 20 years (Section 5 in [6, 12]).

Some results on hardness of bits, related to the elliptic curve Diffie–Hellman protocol, were given by Boneh and Shparlinski [8] and by Jetchev and Venkatesan [16] (building on [8] and assuming the generalized Riemann hypothesis). These results differ from ours in two aspects. They do not provide hardness of bits for the elliptic curve Diffie–Hellman protocol for a single fixed curve. Furthermore, the techniques used to achieve these results are very different from ours, as they reduce the problem to an easier linear problem, while we keep working with the non-linear addition law.

In this paper we study the bit security of the elliptic curve Diffie–Hellman key exchange protocol. Our main result is Theorem 2, where we show that about $5/6$ of the most significant bits of the x-coordinate of the Diffie–Hellman key are as hard to compute as the entire key. As above, this result holds if one assumes a perfect success in computing these bits. This result directly follows from the solution to the *elliptic curve hidden number problem* given in Theorem 1. This solution is based on the ideas behind the solution to the *modular inversion hidden number problem* given in [7] and follows the formal proof given by Ling, Shparlinski, Steinfeld and Wang [18] (earlier ideas already appear in [2, 3]).

Additional results are given in Section 6. In Section 6.1 we show how to derive the same result for

the least significant bits. Section 6.2 addresses the case of elliptic curves over extension fields. This problem was first studied by Jao, Jetchev and Venkatesan [15]. We present a general approach to this problem and improve the known result to hold for both coordinates of the Diffie–Hellman key and to any constant extension degree.

As the literature on the elliptic curve hidden number problem is very minimal and incomplete, short discussions – some of which are quite trivial – appear throughout the paper in order to give a complete and comprehensive study of the problem. We hope that this work will initiate the study of bit security of elliptic curve Diffie–Hellman key exchange that will lead to improvements either in the number of hardcore bits or in the required success probability for computing them.

2. Mathematical Background

Throughout the paper $p > 3$ is an m-bit prime number and $\mathbb{F}_p$ is the field with p elements represented by $\{-\frac{p-1}{2}, \ldots, \frac{p-1}{2}\}$. For $k > 0$ and $x \in \mathbb{F}_p$, we denote by $\mathrm{MSB}_k(x)$ any $h \in \mathbb{F}_p$ such that $|x-h| \leq \frac{p}{2^{k+1}}$.[1] We have $h = \mathrm{MSB}_k(x) = x - e$ for $|e| \leq \frac{p}{2^{k+1}}$, which we loosely call *noise*.

2.1 Elliptic curves

Throughout the paper E is an elliptic curve over $\mathbb{F}_p$, given in a short Weierstrass form

$$y^2 = x^3 + ax + b, \quad a, b \in \mathbb{F}_p \quad \text{and} \quad 4a^3 + 27b^2 \neq 0\,.$$

A point $P = (x, y) \in \mathbb{F}_p^2$ that satisfies this equation is a point on the curve E. We denote the x-coordinate (resp. y-coordinate) of a given point P by x_P or P_x (resp. y_P or P_y). The set of points on E, together with the *point at infinity* O, is known to be an abelian group. Hasse's theorem states that the number of points $\#E$ on the curve $E(\mathbb{F}_p)$ satisfies

$$|\#E - p - 1| \leq 2\sqrt{p}\,.$$

The (additive) inverse of a point $Q = (x_Q, y_Q)$ is $-Q = (x_Q, -y_Q)$. For an integer n we denote by $[n]P$ the successive n-time addition of a point P; $[-n]P = [n](-P)$. Addition of points $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$, where $P \neq \pm Q$, is given by the following formula. Let $s = s_{P+Q} = \frac{y_P - y_Q}{x_P - x_Q}$, then

$$(P+Q)_x = s^2 - x_P - x_Q \quad \text{and} \quad (P+Q)_y = -(y_P + s((P+Q)_x - x_P))\,.$$

[1] The function MSB_k is standard and thought of as providing the k most significant bits of x. It differs from the classical definition of most-significant-bits functions by (at most) 1 bit. For broad discussions see [4, Section 5], [5, Section 3] and [20, Section 5.1].

2.2 Lattices

Let $B = \{b_1, \ldots, b_r\}$ a set of linearly independent vectors in the Euclidean space $\mathbb{R}^s$, for some integers $r \leq s$. The set $L = \left\{\sum_{i=1}^{r} n_i b_i \mid n_i \in \mathbb{Z}\right\}$ is called an *r-dimensional lattice* and B is a *basis for* L. The (Euclidean) norm of a vector $v \in \mathbb{R}^s$ is denoted by $\|v\|$.

For a lattice L in $\mathbb{R}^s$ and a real number $\gamma \geq 1$, the *γ-shortest vector problem* (γ-SVP) is to find a non-zero lattice vector $v \in L$ with norm not larger than γ times the norm of the shortest non-zero vector in L. In other words, $\|v\| \leq \gamma \min\{\|u\| \mid 0 \neq u \in L\}$.

This problem is a fundamental problem in lattice cryptography. References to surveys and state-of-the-art algorithms for γ-SVP are given in Section 1.2 in the work of Ling, Shparlinski, Steinfeld and Wang [18], and like their work our result uses the γ-SVP algorithms of Schnorr [21] and Micciancio–Voulgaris [19].

3. Hidden Number Problems

The *hidden number problem* was introduced by Boneh and Venkatesan [9] in order to study bit security of the Diffie–Hellman key exchange protocol in the multiplicative group of integers modulo a prime p. This problem is formulated as follows.

> HNP: Fix a prime p, an element $g \in \mathbb{Z}_p^*$ and a positive number k. Let $\alpha \in \mathbb{Z}_p^*$ be a hidden number and let $\mathcal{O}_{\alpha,g}$ be an oracle that on input x computes the k most significant bits of $\alpha g^x \bmod p$. That is, $\mathcal{O}_{\alpha,g}(x) = \mathrm{MSB}_k(\alpha \cdot g^x \bmod p)$. The goal is to recover the hidden number α, given query access to the oracle $\mathcal{O}_{\alpha,g}$.

Various natural variants of this problem can be considered, such as changing the group the elements are taken from and the function the oracle is simulating. Moreover, one can consider oracles with different probability of producing the correct answer. The survey [25] covers many of these generalizations as well as different applications.

The elliptic curve equivalent, known as the *elliptic curve hidden number problem*, is formulated as follows for $\psi \in \{x, y\}$.

> EC-HNP$_\psi$: Fix a prime p, an elliptic curve E over $\mathbb{F}_p$, a point $R \in E$ and a positive number k. Let $P \in E$ be a hidden point and let $\mathcal{O}_{P,R}$ be an oracle that on input t computes the k most significant bits of the ψ-coordinate of $P + [t]R$. That is, $\mathcal{O}_{P,R}(t) = \mathrm{MSB}_k((P + [t]R)_\psi)$. The goal is to recover the hidden point P, given query access to the oracle $\mathcal{O}_{P,R}$.

The elliptic curve hidden number problem, to the best of our knowledge, was first considered (more generally, and only for the x-coordinate) by Boneh, Halevi and Howgrave-Graham [7], and besides

being mentioned in the surveys [24, 25] there is no other literature about it.[2] We remark that there are no known solutions to this problem, even for large k's (except, of course, of trivial cases, i.e., $k \geq \log p - O(\log \log p)$).

A related[3] non-linear problem is the *modular inversion hidden number problem*, which was introduced by Boneh, Halevi and Howgrave-Graham [7]. It is formulated as follows.

> MIHNP: Fix a prime p and positive numbers k, d. Let $\alpha \in \mathbb{Z}_p$ be a hidden number and let $t_1, \ldots, t_d \in \mathbb{Z}_p \setminus \{-\alpha\}$ chosen independently and uniformly at random. The goal is to find the secret number α given the d pairs $\left(t_i, \text{MSB}_k\left(\frac{1}{\alpha+t_i}\right)\right)$.

We now explain the relation between the elliptic curve hidden number problem and bit security of the elliptic curve Diffie–Hellman key exchange protocol.

Remark 1. Given an elliptic curve E over a field $\mathbb{F}_q$, a point $Q \in E$ and the values $[a]Q$ and $[b]Q$, the Diffie–Hellman key P is the value $P = ECDH_Q([a]Q, [b]Q) = [ab]Q$. Suppose one has an oracle that on input $[u]Q$ and $[v]Q$ outputs some partial information on $[uv]Q$. Then, one can choose an integer t and calculate $[t]Q$, and by adding $[t]Q$ and $[a]Q$, one gets $[a]Q + [t]Q = [a+t]Q$. Querying the oracle on $[b]Q$ and $[a+t]Q$, one gets partial information on $[(a+t)b]Q = [ab]Q + [tb]Q = P + [t]([b]Q) = P + [t]R$, for $R = [b]Q$. Repeating for several t's, if it is possible to solve the elliptic curve hidden number problem, one can find the Diffie–Hellman key $P = [ab]Q$.

In the proof below we use the fact that one can get $\text{MSB}_k(x_P)$ for the secret point P. This can be easily justified by taking $t = 0$ in EC-HNP, or equivalently querying the oracle from Remark 1 on $[a]Q$ and $[b]Q$. Moreover,

Remark 2. Similar to HNP [9, Section 4.1] and MIHNP [7, Section 2.1], EC-HNP can be self-randomized. Indeed, given $\{(Q_i, \mathcal{O}((P+Q_i)_\psi))\}_{1 \leq i \leq n}$, for an oracle $\mathcal{O}$, choose $1 \leq i_0 \leq n$, and define a new secret $P' := P + Q_{i_0}$. Let $Q'_i := Q_i - Q_{i_0}$, then we have $P + Q_i = P' + Q'_i$, and so $\mathcal{O}((P' + Q'_i)_\psi) = \mathcal{O}((P+Q_i)_\psi)$. If one can find P', recovering $P = P' - Q_{i_0}$ is easy. This shows that given $\{(Q_i, \mathcal{O}((P+Q_i)_\psi))\}_i$, one can randomize the secret P as well as the 'multipliers' Q_i. Alternatively, if access to the oracle is still provided, one can query on $t_{i_0} + t_i$ to receive $\mathcal{O}((P' + Q_i)_\psi)$, as well as taking the approach of [9, Section 4.1]. This self-randomization allows us to assume without loss of generality that R in EC-HNP is a generator for $\langle Q \rangle$.

[2] In [15] (a variant of) this problem is studied for elliptic curves over extension fields.

[3] We show below that the technique used to solve this problem also applies to EC-HNP. In addition, [24] reveals that obtaining bit security results for the elliptic curve Diffie–Hellman scheme has been a primary motivation for studying this problem.

4. MAIN RESULTS

The main result is Theorem 2, which gives the first bit security result for prime-field elliptic curve Diffie–Hellman key exchange. This result follows from the following theorem, which shows how to recover the secret point in EC-HNP$_x$ given a γ-SVP algorithm.

Theorem 1. *Let E be an elliptic curve over a prime field $\mathbb{F}_p$, let n be an integer and k a real number. Let an unknown $P = (x_P, y_P) \in E \setminus \{O\}$ and a known generator $R \in E \setminus \{O\}$ be points on the curve. Let $\mathcal{O}$ be a function such that $\mathcal{O}(t) = MSB_k((P + [t]R)_x)$, and denote $Q_i := [t_i]R$. Then, given a γ-SVP algorithm, there exists a deterministic polynomial-time algorithm that recovers the unknown x_P with $2n + 1$ calls to $\mathcal{O}$ and a single call to the γ-SVP algorithm on a $(3n + 3)$-dimensional lattice with polynomially bounded basis, except with probability*

$$\mathcal{P}_1 \leq \frac{8^n(6\eta\Delta + 1)^{6n+3}}{(p - 2\sqrt{p} - 2)^n} + \frac{16(6\eta\Delta + 1)^6}{p - 2\sqrt{p} - 2} + \frac{2n + 3}{p - 2\sqrt{p}}$$

over the choices of $x_{Q_1}, \ldots, x_{Q_n}$, when it returns no answer or a wrong answer, where $\eta = 2\gamma\sqrt{3n + 1}$ and $\Delta = \lceil \frac{p}{2^{k+1}} \rceil$.[4] *If the correct x-coordinate x_P has been recovered, the algorithm determines which of the two candidates $\pm y_P$ is the correct y-coordinate, except with probability*

$$\mathcal{P}_2 \leq \frac{(16\Delta)^n}{(p - 2\sqrt{p} - 2)^n}$$

over the choices of $x_{Q_1}, \ldots, x_{Q_n}$.

Remark 3. In the theorem, as in the corollary below, R is taken to be a generator of E in order to give precise bounds on the probabilities. Both results hold even if R is not a generator of E, as long as it generates a "large enough" subgroup. The size of the subgroup appears in the denominator of the probabilities bounds (see footnote 7), and so the results also hold if the subgroup's order is greater than $p/poly(\log(p))$, for example. For substantially smaller subgroups, one would need to adjust the value for k.

The following corollary shows that one can solve EC-HNP$_x$ given an oracle for $k > (\frac{5}{6} + \epsilon)m$ most significant bits (where m is the bit length of p, and for any constant ϵ). Similar to Ling et al. [18], we consider two different SVP approximation algorithms to show the influence of ϵ on the running time and the minimum allowed value for p.

Corollary 1. *Fix $0 < \delta \leq 3\epsilon < 1/2$. Let $n_0 = \lceil \frac{1}{6\epsilon} \rceil$, p be an m-bit prime, E be an elliptic curve over $\mathbb{F}_p$ and $k > (5/6 + \epsilon)m$. There exist deterministic algorithms A_i, for $i = 1, 2$, that solve EC-HNP$_x$*

[4] As the matter of exact precision is not important, we set Δ to be an integer.

(with MSB$_k$ and a generator R) for $m \geq m_i$, with probability at least $1 - p^{-\delta}$ over the choices of $x_{Q_1}, \ldots, x_{Q_{n_0}}$ where

$$m_1 = \lceil c_1 \epsilon^{-1} \log \epsilon^{-1} \rceil \quad \textit{and} \quad m_2 = \lceil c_2 \epsilon^{-2} \frac{(\log \log \epsilon^{-1})^2}{\log \epsilon^{-1}} \rceil \,,$$

for some absolute effectively computable constants c_1, c_2, and their running time is T_i where

$$T_1 = (2^{\epsilon^{-1}} m)^{O(1)} \quad \textit{and} \quad T_2 = (\epsilon^{-1} m)^{O(1)} \,.$$

As a consequence, following Remark 1, we get a hardcore function for the elliptic curve Diffie–Hellman problem and the following bit security result for elliptic curve Diffie–Hellman key exchange.

Theorem 2. *Fix $0 < \delta \leq 3\epsilon < 1/2$. Let p be an m-bit prime, E be an elliptic curve over $\mathbb{F}_p$, a point $P \in E \setminus \{O\}$ of order at least $p/poly(\log(p))$ and $k > (5/6 + \epsilon)m$. Given an efficient algorithm to compute MSB$_k$ $\big(([ab]P)_x\big)$ from $[a]P$ and $[b]P$, there exists a deterministic polynomial-time algorithm that computes $[ab]P$ with probability at least $1 - p^{\delta}$.*

In a nutshell, the approach of solving non-linear problems like MIHNP and EC-HNP is to form some polynomials with desired small roots, and use a lattice basis reduction algorithm to find some of these roots. The polynomials' degree, the number of their monomials, and subsequently the dimension of the lattice play a main role in the quality of the result one can obtain.

4.1 Our approach

The first obstacle in approaching EC-HNP is the nonlinearity (over the ground field) of the addition rule. This can be easily overcome by the "linearization" approach of Boneh et al. [7], which we adopt, but at the cost of not being able to use Babai's algorithm for closest lattice point [1]. This prevents non-linear problems, like MIHNP and EC-HNP, of achieving results as good as the result for the linear HNP.

The second obstacle in approaching EC-HNP$_x$ (and similarly EC-HNP$_y$) is that while one only gets partial information of x_P, the formula for $(P + Q)_x$ also involves (the unbounded unknown) y_P. Similar to the approach of [7], one can isolate this unknown in one equation, and substitute to all of the other equations, hence 'losing' one equation. Doing so will impose an extra bounded unknown in each equation, as well as many additional monomials, coming from the noise term of the equation we use to eliminate y_P.[5] This will therefore result in a significantly large dimension of the lattice one constructs.[6] Instead, we show how one can combine two correlated equations to eliminate y_P. This helps us to

[5] Alternatively, once y_P is isolated, one can square both sides of the equation to eliminate y_P using the elliptic curve equation. While this allows us to keep all initial equations, doing so will result in polynomials of a larger degree with many more monomials.

[6] We speculate that this is the reason why [7] can only rigorously solve EC-HNP$_x$ given $(1 - \epsilon)$ fraction of the bits, for $\epsilon \approx 0.02$.

define one bounded unknown (twice as large) while keeping the number of monomials relatively small. Taking this approach we form new equations from pairs of initial equations, causing a 'loss' of about half of the equations.

Formally, we proceed as follows.

4.1.1 Eliminating y_P

For some integer t consider the pair $Q = [t]R, -Q = [-t]R \in E$, and suppose $P \neq \pm Q$. Let $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$, therefore $-Q = (x_Q, -y_Q)$, and write $s_{P+Q} = \frac{y_P - y_Q}{x_P - x_Q}$ and $s_{P-Q} = \frac{y_P - y_{-Q}}{x_P - x_{-Q}} = \frac{y_P + y_Q}{x_P - x_Q}$. The following operations take place in $\mathbb{F}_p$.

$$\begin{aligned}(P+Q)_x + (P-Q)_x &= s_{P+Q}^2 - x_P - x_Q + s_{P-Q}^2 - x_P - x_Q \\ &= \left(\frac{y_P - y_Q}{x_P - x_Q}\right)^2 + \left(\frac{y_P + y_Q}{x_P - x_Q}\right)^2 - 2x_P - 2x_Q \\ &= 2\left(\frac{y_P^2 + y_Q^2}{(x_P - x_Q)^2} - x_P - x_Q\right) = 2\left(\frac{x_Q x_P^2 + (a + x_Q^2)x_P + a x_Q + 2b}{(x_P - x_Q)^2}\right).\end{aligned} \tag{1}$$

4.1.2 Constructing polynomials with small roots

Write $h_0 = \text{MSB}_k(x_P) = x_P - e_0$, $h = \text{MSB}_k((P+Q)_x) = (P+Q)_x - e$ and $h' = \text{MSB}_k((P-Q)_x) = (P-Q)_x - e'$. Letting $\widetilde{h} = h + h'$ and $\widetilde{e} = e + e'$ and plugging $x_P = h_0 + e_0$ in (1) we get

$$\begin{aligned}\widetilde{h} + \widetilde{e} &= (P+Q)_x + (P-Q)_x \\ &= 2\left(\frac{x_Q(h_0 + e_0)^2 + (a + x_Q^2)(h_0 + e_0) + a x_Q + 2b}{(h_0 + e_0 - x_Q)^2}\right).\end{aligned}$$

Multiplying by $(h_0 + e_0 - x_Q)^2$ and rearranging we get that the following bivariate polynomial

$$\begin{aligned}F(X,Y) = \; & X^2 Y + (\widetilde{h} - 2x_Q)X^2 + 2(h_0 - x_Q)XY + 2[\widetilde{h}(h_0 - x_Q) - 2h_0 x_Q - a - x_Q^2]X \\ & + (h_0 - x_Q)^2 Y + [\widetilde{h}(h_0 - x_Q)^2 - 2h_0^2 x_Q - 2(a + x_Q^2)h_0 - 2a x_Q - 4b]\end{aligned}$$

satisfies $F(e_0, \widetilde{e}) \equiv 0 \pmod{p}$.

Repeating with n different Q_i leads to n polynomials of the form

$$F_i(X,Y) = X^2 Y + A_i X^2 + A_{0,i} XY + B_i X + B_{0,i} Y + C_i\,, \tag{2}$$

that satisfy $F_i(e_0, \widetilde{e_i}) \equiv 0 \pmod{p}$. Our aim is to find "small" roots for F_i; if one of these roots satisfies $X = e_0$, we can substitute in h_0 and recover x_P.

We start with a simple argument that shows that indeed we expect to solve EC-HNP$_x$ with more than the top $5/6$ fraction of the bits. The argument is identical to the argument given in [7, Section 3.1].

4.2 A simple heuristic argument

The solutions to the system of the n polynomials in (2) can be represented by a lattice of dimension $4n+3$, as follows. The lattice is spanned by the rows of a matrix M of the following structure

$$M = \begin{pmatrix} E & R \\ 0 & P \end{pmatrix}$$

where E and P are diagonal square matrices of dimensions $3n+3$ and n, respectively, and R is a $(3n+3) \times n$ matrix. Each of the first $3n+3$ rows of M is associated with one of the terms in (2), and each of the last n columns is associated with one of these equations. For example, for $n=2$ we get the matrix (m is the bit size of p and k the number of bits we get)

$$M = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & C_1 & C_2 \\
0 & 2^{k-m} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & B_{0,1} & 0 \\
0 & 0 & 2^{k-m} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & B_{0,2} \\
0 & 0 & 0 & 2^{k-m} & 0 & 0 & 0 & 0 & 0 & B_1 & B_2 \\
0 & 0 & 0 & 0 & 2^{2(k-m)} & 0 & 0 & 0 & 0 & A_{0,1} & 0 \\
0 & 0 & 0 & 0 & 0 & 2^{2(k-m)} & 0 & 0 & 0 & 0 & A_{0,2} \\
0 & 0 & 0 & 0 & 0 & 0 & 2^{2(k-m)} & 0 & 0 & A_1 & A_2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 2^{3(k-m)} & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2^{3(k-m)} & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p
\end{pmatrix}.$$

For $e_0, \widetilde{e}_i$, the last n columns give us equations over the integers:

$$e_0^2 \widetilde{e}_i + A_i e_0^2 + A_{0,i} e_0 \widetilde{e}_i + B_i e_0 + B_{0,i} \widetilde{e}_i + C_i - k_i p = 0\,.$$

For the corresponding solution vector

$$\mathbf{v} := \langle 1, \widetilde{e}_1, \ldots, \widetilde{e}_n, e_0, e_0\widetilde{e}_1, \ldots, e_0\widetilde{e}_n, e_0^2, e_0^2\widetilde{e}_1, \ldots, e_0^2\widetilde{e}_n, k_1, \ldots, k_n \rangle\,,$$

we get that $\mathbf{v}M =$

$$\langle 1, \frac{\widetilde{e}_1}{2^{m-k}}, \ldots, \frac{\widetilde{e}_n}{2^{m-k}}, \frac{e_0}{2^{m-k}}, \frac{e_0\widetilde{e}_1}{2^{2(m-k)}}, \ldots, \frac{e_0\widetilde{e}_n}{2^{2(m-k)}}, \frac{e_0^2}{2^{2(m-k)}}, \frac{e_0^2\widetilde{e}_1}{2^{3(m-k)}}, \ldots, \frac{e_0^2\widetilde{e}_n}{2^{3(m-k)}}, 0, \ldots, 0 \rangle\,.$$

Therefore, $\mathbf{v}M$ is a lattice point with $3n+3$ non-zero entries, all of which are smaller than 1, so its Euclidean norm is smaller than $\sqrt{3n+3}$.

The determinant of the lattice is $\frac{p^n}{2^{(m-k)(6n+3)}}$. We apply the heuristic for short lattice vectors and expect that $\mathbf{v}M$ is the shortest vector if $\sqrt{3n+3} \ll \sqrt{4n+3}\left(2^{(k-m)(6n+3)}p^n\right)^{1/(4n+3)}$. Substituting $p = 2^{m+O(1)}$ and ignoring lower terms we get $2^k \gg 2^{5/6m}$, and so we expect that $\mathbf{v}M$ is the shortest

lattice vector when we get more than $\frac{5}{6}m$ bits. Therefore, this becomes a problem of recovering the shortest lattice vector.

Boneh et al. [7] suggest using Coppersmith's method [10] and construct a lattice that leads to a smaller bound on the number of bits one needs in order to recover the secret element in this kind of non-linear problems. This approach has to assume linear independence of the equations involved, and therefore does not provide a proof, but only a heuristic. Since the aim of this paper is to prove bit security, we do not follow this path.

We now turn to a complete formal proof of Theorem 1. It follows the same arguments as in the proof of Theorem 1 in [18], where necessary adaptations have been made.

5. PROOFS

The proof of Theorem 1 is very technical. The algorithm of recovering x_P appears in Algorithm 1, but we first lay the groundwork, so that the probability analysis that appears after the algorithm could be understood. We first give an overview of the key points of the proof.

Overview

In the algorithmic part:

- Using $\mathcal{O}$, we construct the polynomial relations (as in (2) above)

  $$F_i(X,Y) = X^2Y + A_iX^2 + A_{0,i}XY + B_iX + B_{0,i}Y + C_i$$

  for which $F_i(e_0, \widetilde{e}_i) \equiv 0 \pmod{p}$.

- Using these relations, we construct a lattice (see (4)), such that the vector

  $$\mathbf{e} := (\Delta^3, \Delta^2 e_0, \Delta^2\widetilde{e}_1, \ldots, \Delta^2\widetilde{e}_n, \Delta e_0^2, \Delta e_0\widetilde{e}_1, \ldots, \Delta e_0\widetilde{e}_n, e_0^2\widetilde{e}_1, \ldots, e_0^2\widetilde{e}_n)$$

  is a short lattice vector.

- We run a γ-SVP algorithm on the lattice to receive a short lattice vector

  $$\mathbf{f} := (\Delta^3 f_0', \Delta^2 f_0, \Delta^2 f_1 \ldots, \Delta^2 f_n, \Delta f_{0,0}, \Delta f_{0,1}, \ldots, \Delta f_{0,n}, f_{00,1}, \ldots, f_{00,n})\,.$$

  As $\mathbf{e}$ and $\mathbf{f}$ are two short lattice vectors, we expect them to be a (scalar) multiple of each other.

- Supposing this is the case, the scalar f_0' is found by observing the first coordinate of $\mathbf{e}$ and $\mathbf{f}$. We then compute $e_0 = f_0/f_0'$ provided $f_0' \neq 0$.

- From the relation $h_0 = x_P - e_0$ we derive $x_P = h_0 + e_0$.

The second part of the proof analyzes the success probability of the algorithm, as follows:

- If $e_0 \neq f_0/f_0'$ or $f_0' = 0$ the algorithm fails.
- To derive the probability of these events we form a certain family of low-degree polynomials (see (12)), for which we are interested in their set of zeros. The number of polynomials in the family is a function of $\Delta = \lceil \frac{p}{2^{k+1}} \rceil$, and so a function of k.
- Claim 1 shows that if $y_P \neq 0$, then the polynomials are not identically zero.
- We show that these events occur if the points x_{Q_i} are roots of some of these polynomials. Thus, we derive an exact expression of the probability of these events to hold.

The last part of the proof shows how one can determine the correct value for y_P using a consistency check with all of the given values.

5.1 Proof of Theorem 1

Assume without loss of generality $3\eta\Delta \leq 3\eta\Delta^3 < p$, as otherwise the bound on the probability makes the claim trivial, and that the unknown P is chosen uniformly at random (see Remark 2). Throughout, unless stated otherwise, i, j are indices such that $1 \leq i \leq n$ and $0 \leq j \leq n$. Set $t_0 = 0$, choose $t_i \in [1, \#E-1]$ independently and uniformly at random, and query the oracle $\mathcal{O}$ on $\pm t_j$ to get the $2n+1$ values $\mathcal{O}(\pm t_j)$ denoted by $h_0 = \text{MSB}_k(P_x) = x_P - e_0$, $h_i = \text{MSB}_k((P+Q_i)_x) = (P+Q_i)_x - e_i$ and $h_{i'} = \text{MSB}_k((P-Q_i)_x) = (P-Q_i)_x - e_{i'}$, for some integers $-\Delta \leq e_j, e_{i'} \leq \Delta$. Denote $\widetilde{h}_i = h_i + h_{i'}$ and $\widetilde{e}_i = e_i + e_{i'}$, and suppose $P \neq \pm Q_i$.

The following has been shown in Section 4.1.2. For every $1 \leq i \leq n$, one has

$$\begin{aligned}\widetilde{h}_i + \widetilde{e}_i &= h_i + e_i + h_{i'} + e_{i'} = (P+Q_i)_x + (P-Q_i)_x \\ &\equiv 2\left(\frac{x_{Q_i}(h_0+e_0)^2 + (a+x_{Q_i}^2)(h_0+e_0) + ax_{Q_i} + 2b}{(h_0+e_0-x_{Q_i})^2}\right) \pmod{p}.\end{aligned}$$

Consider the polynomials

$$F_i(X,Y) := X^2Y + A_iX^2 + A_{0,i}XY + B_iX + B_{0,i}Y + C_i\,,$$

where (all congruences hold mod p)

$$\begin{aligned} &A_i \equiv \widetilde{h}_i - 2x_{Q_i}\,, && A_{0,i} \equiv 2(h_0 - x_{Q_i})\,, \\ &B_i \equiv 2[\widetilde{h}_i(h_0 - x_{Q_i}) - 2h_0x_{Q_i} - a - x_{Q_i}^2]\,, && B_{0,i} \equiv (h_0 - x_{Q_i})^2, \text{ and} \\ &C_i \equiv \widetilde{h}_i(h_0 - x_{Q_i})^2 - 2((h_0^2+a)x_{Q_i} + (a+x_{Q_i}^2)h_0 + 2b)\,. \end{aligned}$$

It holds that $F(e_0, \widetilde{e}_i) \equiv 0 \pmod{p}$ for every $1 \leq i \leq n$. As $e_0, \widetilde{e}_i$ are relatively small, one hopes that

finding a *small solution* to one of these polynomials would allow to recover e_0 and subsequently P. To achieve this goal, we use these relations to construct a lattice and apply the γ-SVP algorithm.

Formally, we start by 'balancing' the coefficients (as lattice basis reduction algorithms work better where all the coefficients are of similar size). For every $1 \le i \le n$, set

$$\begin{aligned} &a_i \equiv \Delta^{-1} A_i \pmod p\,, \quad a_{0,i} \equiv \Delta^{-1} A_{0,i} \pmod p\,, \\ &b_i \equiv \Delta^{-2} B_i \pmod p\,, \quad b_{0,i} \equiv \Delta^{-2} B_{0,i} \pmod p\,, \text{ and} \\ &c_i \equiv \Delta^{-3} C_i \pmod p\,. \end{aligned} \tag{3}$$

The vector

$$\mathbf{e} = (\Delta^3, \Delta^2 e_0, \Delta^2 \widetilde{e}_1, \ldots, \Delta^2 \widetilde{e}_n, \Delta e_0^2, \Delta e_0 \widetilde{e}_1, \ldots, \Delta e_0 \widetilde{e}_n, e_0^2 \widetilde{e}_1, \ldots, e_0^2 \widetilde{e}_n)$$

belongs to the lattice L consisting of solutions

$$\mathbf{x} = (x_0', x_0, x_1, \ldots, x_n, x_{0,0}, x_{0,1}, \ldots, x_{0,n}, x_{00,1}, \ldots, x_{00,n}) \in \mathbb{Z}^{3n+3}$$

of the congruences

$$\begin{aligned} &c_i x_0' + b_i x_0 + b_{0,i} x_i + a_i x_{0,0} + a_{0,i} x_{0,i} + x_{00,i} \equiv 0 \pmod p,\ 1 \le i \le n\,, \\ &x_0' \equiv 0 \pmod{\Delta^3}\,, \\ &x_j \equiv 0 \pmod{\Delta^2} \quad 0 \le j \le n\,, \text{ and} \\ &x_{0,j} \equiv 0 \pmod{\Delta} \quad 0 \le j \le n\,. \end{aligned}$$

The lattice L is generated by the rows of a $(3n+3) \times (3n+3)$ matrix M of the following structure:

$$M = \begin{pmatrix} \mathbf{\Delta^2} & 0 & M_1 \\ 0 & \mathbf{\Delta} & M_2 \\ 0 & 0 & P \end{pmatrix} \tag{4}$$

where $\mathbf{\Delta^2}$, $\mathbf{\Delta}$ and P are diagonal square matrices of dimensions $n+2$, $n+1$ and n, respectively, such that the diagonal of P consists of the prime p, the matrix $\mathbf{\Delta}$ consists of Δ and the matrix $\mathbf{\Delta^2}$ of Δ^2, except of the first diagonal entry which is Δ^3; and the matrices M_1 and M_2 are of dimensions $(n+2) \times n$ and $(n+1) \times n$ respectively, given by

$$M_1 = \begin{pmatrix} -C_1 & -C_2 & \ldots & -C_n \\ -B_1 & -B_2 & & -B_n \\ -B_{0,1} & 0 & & 0 \\ 0 & -B_{0,2} & & \\ \vdots & 0 & \ddots & \\ & \vdots & & \\ 0 & 0 & & -B_{0,n} \end{pmatrix}, \qquad M_2 = \begin{pmatrix} -A_1 & -A_2 & \ldots & -A_n \\ -A_{0,1} & 0 & & 0 \\ 0 & -A_{0,2} & & \vdots \\ \vdots & 0 & \ddots & \\ & \vdots & & \\ 0 & 0 & & -A_{0,n} \end{pmatrix}.$$

As $|\widetilde{e}_i| = |e_i + e_{i'}| \le 2\Delta$ for every $1 \le i \le n$, we have

$$\|\mathbf{e}\| \le \sqrt{3\Delta^6 + 12n\Delta^6} = \sqrt{3+12n}\Delta^3 \le 2\Delta^3\sqrt{3n+1}\,.$$

Run the γ-SVP algorithm and denote the vector it outputs by

$$\mathbf{f} = (\Delta^3 f'_0, \Delta^2 f_0, \Delta^2 f_1 \ldots, \Delta^2 f_n, \Delta f_{0,0}, \Delta f_{0,1}, \ldots, \Delta f_{0,n}, f_{00,1}, \ldots, f_{00,n})\,, \tag{5}$$

where $f'_0, f_j, f_{0,j}, f_{00,i} \in \mathbb{Z}$. Notice that

$$\|\mathbf{f}\| \le \gamma\|\mathbf{e}\| \le 2\gamma\Delta^3\sqrt{3n+1} = \eta\Delta^3 \text{ for } \eta = 2\gamma\sqrt{3n+1}\,,$$

and also that

$$\begin{aligned}
&|f'_0| \le \|\mathbf{f}\|\Delta^{-3} \le \eta\,,\\
&|f_j| \le \|\mathbf{f}\|\Delta^{-2} \le \eta\Delta\,,\\
&|f_{0,j}| \le \|\mathbf{f}\|\Delta^{-1} \le \eta\Delta^2\,, \text{ and}\\
&|f_{00,i}| \le \|\mathbf{f}\| \le \eta\Delta^3\,.
\end{aligned}$$

As $\mathbf{e}, \mathbf{f}$ are both short lattice vectors, we expect them to be scalar multiples of each other. Therefore, let

$$\mathbf{d} = f'_0\mathbf{e} - \mathbf{f} = (0, \Delta^2 d_0, \Delta^2 d_1, \ldots, \Delta^2 d_n, \Delta d_{0,0}, \Delta d_{0,1}, \ldots, \Delta d_{0,n}, d_{00,1}, \ldots, d_{00,n})\,,$$

where

$$\begin{aligned}
&d_0 = f'_0 e_0 - f_0\,, && |d_0| = |f'_0 e_0 - f_0| \le \eta|e_0| + |f_0| \le \eta\Delta + \eta\Delta = 2\eta\Delta\,,\\
&d_i = f'_0\widetilde{e}_i - f_i\,, && |d_i| = |f'_0\widetilde{e}_i - f_i| \le \eta|\widetilde{e}_i| + |f_i| \le \eta 2\Delta + \eta\Delta = 3\eta\Delta\,,\\
&d_{0,0} = f'_0 e_0^2 - f_{0,0}\,, && |d_{0,0}| = |f'_0 e_0^2 - f_{0,0}| \le \eta|e_0|^2 + |f_{0,0}| \le \eta\Delta^2 + \eta\Delta^2 = 2\eta\Delta^2\,,\\
&d_{0,i} = f'_0 e_0\widetilde{e}_i - f_{0,i}\,, && |d_{0,i}| = |f'_0 e_0\widetilde{e}_i - f_{0,i}| \le \eta|e_0\widetilde{e}_i| + |f_{0,i}| \le \eta 2\Delta^2 + \eta\Delta^2 = 3\eta\Delta^2\,, \text{ and}\\
&d_{00,i} = f'_0 e_0^2\widetilde{e}_i - f_{00,i}\,, && |d_{00,i}| = |f'_0 e_0^2\widetilde{e}_i - f_{00,i}| \le \eta|e_0^2\widetilde{e}_i| + |f_{00,i}| \le \eta 2\Delta^3 + \eta\Delta^3 = 3\eta\Delta^3\,.
\end{aligned} \tag{6}$$

Notice that if $f'_0 \neq 0$ and also one of the coordinates of $\mathbf{d}$ (except of the first one) is zero, we can recover some previously unknown information. More precisely, suppose $f'_0 \neq 0$, then

$$\text{If } d_0 = 0, \text{ then } e_0 = f_0/f'_0\,; \tag{7}$$

$$\text{If } d_i = 0, \text{ then } \widetilde{e}_i = f_i/f'_0\,, \quad 1 \le i \le n\,; \tag{8}$$

$$\text{If } d_{0,0} = 0, \text{ then } e_0^2 = f_{0,0}/f'_0\,; \tag{9}$$

$$\text{If } d_{0,i} = 0, \text{ then } e_0\widetilde{e}_i = f_{0,i}/f'_0\,, \quad 1 \le i \le n\,; \tag{10}$$

$$\text{If } d_{00,i} = 0, \text{ then } e_0^2\widetilde{e}_i = f_{00,i}/f'_0\,, \quad 1 \le i \le n\,. \tag{11}$$

As $\widetilde{e}_i = e_i + e_{i'}$ it is unclear how to use these values in general to recover the secret x_P. We therefore focus on e_0, from which we derive x_P. Although there are several ways to recover e_0 from these equations, for the sake of the proof we only focus on (7), thus in case $f'_0 \neq 0$ we take $h_0 + f_0/f'_0$ as the candidate for x_P, and if $f'_0 = 0$, we fail. We remark that a more involved approach can be taken (to determine e_0 and in the case $f'_0 = 0$), using the consistency check in Appendix A.

A pseudocode for the algorithm that recovers x_P is the following.

Algorithm 1: Find x_P

1: Construct a lattice, generated by the rows of the matrix M as in (4).
2: Run the γ-SVP algorithm on the lattice to get the vector $\mathbf{f}$ as in (5).
3: **if** $f'_0 \neq 0$ **then**
  return $h_0 + f_0/f'_0$
  else
  Fail

Probability of failure

We now define the following events:

(E-1) $y_P = 0$;

(E-2) $d_0 \neq 0$ and (E-1) does not hold;

(E-3) $f'_0 = 0$ and (E-1) and (E-2) do not hold.

It is clear that if none of the events hold, one can recover x_P. The requirement $y_P \neq 0$ will be made clear in Claim 1 below.

As there are at most 3 values for $x_P \in \mathbb{F}_p$ that satisfy the equation $x_P^3 + ax_P + b \equiv 0 \pmod{p}$, and since P is assumed to be chosen uniformly at random, the probability that (E-1) holds satisfies

$$\Pr[\text{(E-1)}] \leq \frac{3}{\#E - 1} \leq \frac{3}{p - 2\sqrt{p}}\,.$$

In order to derive a bound on the probability of the other events we form some useful equations. As

$$c_i\Delta^3 + b_i\Delta^2 e_0 + b_{0,i}\Delta^2\widetilde{e}_i + a_i\Delta e_0^2 + a_{0,i}\Delta e_0\widetilde{e}_i + e_0^2\widetilde{e}_i \equiv 0 \pmod{p},\ 1 \leq i \leq n\,,$$

and

$$c_i\Delta^3 f'_0 + b_i\Delta^2 f_0 + b_{0,i}\Delta^2 f_i + a_i\Delta f_{0,0} + a_{0,i}\Delta f_{0,i} + f_{00,i} \equiv 0 \pmod{p},\ 1 \leq i \leq n\,,$$

we get (by the definition of $\mathbf{d}$)

$$b_i\Delta^2 d_0 + b_{0,i}\Delta^2 d_i + a_i\Delta d_{0,0} + a_{0,i}\Delta d_{0,i} + d_{00,i} \equiv 0 \pmod{p},\ 1 \leq i \leq n\,,$$

and therefore (using (3) above)

$$B_i d_0 + B_{0,i} d_i + A_i d_{0,0} + A_{0,i} d_{0,i} + d_{00,i} \equiv 0 \pmod{p},\ 1 \le i \le n\,.$$

Multiplying by $(x_P - x_{Q_i})^2$ and using the definitions for $A_i, A_{0,i}, B_i$ and $B_{0,i}$ we get

$$\begin{aligned}(x_P - x_{Q_i})^2\Big(&2[\widetilde{h}_i(h_0 - x_{Q_i}) - 2h_0 x_{Q_i} - a - x_{Q_i}^2]d_0 + (h_0^2 - 2h_0 x_{Q_i} + x_{Q_i}^2)d_i \\ &+ (\widetilde{h}_i - 2x_{Q_i})d_{0,0} + 2(h_0 - x_{Q_i})d_{0,i} + d_{00,i}\Big) \equiv 0 \pmod{p},\ 1 \le i \le n\,,\end{aligned}$$

which simplifies, as a polynomial in x_{Q_i}, to

$$U_i x_{Q_i}^4 - V_i x_{Q_i}^3 + W_i x_{Q_i}^2 + Y_i x_{Q_i} + Z_i \equiv 0 \pmod{p},\ 1 \le i \le n\,, \tag{12}$$

where (all congruences hold mod p)

$$\begin{aligned}
U_i &\equiv d_i - 2d_0\,,\\
V_i &\equiv 2(2x_P - 2e_0 - \widetilde{e}_i)d_0 + (4x_P - 2e_0)d_i + 2d_{0,0} + 2d_{0,i}\,,\\
W_i &\equiv 2(3x_P^3 - 6e_0x_P - 3\widetilde{e}_i x_P + e_0\widetilde{e}_i - 3a)d_0 + (6x_P^2 - 6e_0x_P + e_0^2)d_i + (6x_P - \widetilde{e}_i)d_{0,0}\\
&\quad + (6x_P - 2e_0)d_{0,i} + d_{00,i}\,,\\
Y_i &\equiv 2(3\widetilde{e}_i x_P^2 - 2e_0\widetilde{e}_i x_P + 2ax_P - 2ae_0 - 4b)d_0 - 2(2x_P^3 - 3e_0x_P^2 + e_0^2x_P)d_i\\
&\quad + (2\widetilde{e}_i x_P + 2a)d_{0,0} - (6x_P^2 - 4e_0x_P)d_{0,i} - 2x_P d_{00,i}\,,\ \text{and}\\
Z_i &\equiv 2(-\widetilde{e}_i x_P^3 + e_0\widetilde{e}_i x_P^2 + ax_P^2 - 2ae_0x_P + 4bx_P - 4be_0)d_0 + (x_P^4 - 2e_0x_P^3 + e_0^2x_P^2)d_i\\
&\quad + (-\widetilde{e}_i x_P^2 + 2ax_P + 4b)d_{0,0} + (2x_P^3 - 2e_0x_P^2)d_{0,i} + x_P^2 d_{00,i}\,.
\end{aligned} \tag{13}$$

We now show that if for some $1 \le i \le n$ the left hand side of (12) is the constant zero polynomial, then $d_0 = 0 = d_{0,0}$. We conclude that if $d_0 \ne 0$ or $d_{0,0} \ne 0$, then the left hand side of (12) is a non-constant polynomial in x_{Q_i} (of degree at most 4) for every $1 \le i \le n$.

Claim 1. Let $1 \le i \le n$, and assume $y_P \ne 0$. The left hand side of (12) is constant if and only if $d_0 = d_{0,0} = d_i = d_{0,i} = d_{00,i} = 0$.

Proof. The first implication is clear from (13). Suppose that the left hand side of (12) is constant for some $1 \le i \le n$. Then $U_i \equiv V_i \equiv W_i \equiv Y_i \equiv Z_i \equiv 0 \pmod{p}$. One can express the latter as a system of 5 equations in the 5 variables $d_0, d_i, d_{0,0}, d_{0,i}$ and $d_{00,i}$. A non-zero solution exists if and only if the system is singular. We show that the system is nonsingular if and only if $y_P \ne 0$, which completes the proof.

We use the first 4 equations to eliminate $d_i, d_{0,i}, d_{00,i}$ and remain with the "global" variables $d_0, d_{0,0}$. One then has

$$-2(2x_P^3 + 3e_0x_P^2 + 2ax_P + ae_0 + 2b)d_0 + (3x_P^2 + a)d_{0,0} \equiv 0 \pmod{p}\,,$$

which simplifies to

$$-4y_P d_0 - 2e_0(3x_P^2 + a)d_0 + (3x_P^2 + a)d_{0,0} \equiv 0 \pmod{p}.$$

If $3x_P^2 + a \equiv 0 \pmod{p}$, then $y_P d_0 \equiv 0 \pmod{p}$. Otherwise, one can express $d_{0,0}$ in terms of d_0. Plugging this value, with the other recovered variables, to the last equation, one gets

$$(x_P^6 + 2ax_P^4 + 2bx_P^3 + a^2x_P^2 + 2abx_P + b^2)d_0 \equiv y_P^4 d_0 \equiv 0 \pmod{p}.$$

In both cases, since $y_P \neq 0$, we have $d_0 \equiv d_{0,0} \equiv d_i \equiv d_{0,i} \equiv d_{00,i} \equiv 0 \pmod{p}$, and since all of these values are of size smaller than p (as we suppose $3\eta\Delta < 3\eta\Delta^3 < p$), the claim follows. ∎

We use this claim to bound the probabilities of (E-2) and (E-3), which will prove the first claim in the theorem. The probability of events (E-2) and (E-3) is taken over the choice of the points Q_i for $1 \leq i \leq n$. That is, we consider the number of n-tuples

$$(x_{Q_1}, \ldots, x_{Q_n}) \in \left(E_x \setminus \{x_P\}\right)^n$$

such that (E-2) holds or (E-3) holds, where $E_x := \{z \in \mathbb{F}_p \mid \exists Q \in E, Q_x = z\}$.[7] Note that $\#E - 1 \leq 2|E_x| \leq \#E + 2$.

Probability of event (E-2). Assume (E-2) holds, that is $d_0 \neq 0$ and $y_P \neq 0$, and fix some values of $d_j, d_{0,j}$ for $0 \leq j \leq n$ and $d_{00,i}$ for $1 \leq i \leq n$. Let us consider the number of n-tuples

$$(x_{Q_1}, \ldots, x_{Q_n}) \in \left(E_x \setminus \{x_P\}\right)^n$$

satisfying (12).

Since $d_0 \neq 0$ Claim 1 shows that the left hand side of (12) is nonconstant for all $1 \leq i \leq n$. Thus, as all the relations in (12) are satisfied, there are at most 4 values x_{Q_i} that satisfy each relation, and so there are at most 4^n n-tuples that satisfy these n non-constant polynomials.

From (6) above we get: as $d_0 \neq 0$ it can take at most $4\eta\Delta$ values, each d_i can take at most $6\eta\Delta + 1$ values, $d_{0,0}$ can take at most $4\eta\Delta^2 + 1$ values, each $d_{0,i}$ can take at most $6\eta\Delta^2 + 1$ values, and each $d_{00,i}$ can take at most $6\eta\Delta^3 + 1$ values. Therefore, there are at most

$$4^n 4\eta\Delta(6\eta\Delta + 1)^n(4\eta\Delta^2 + 1)(6\eta\Delta^2 + 1)^n(6\eta\Delta^3 + 1)^n <$$
$$4^n 4\eta\Delta(6\eta\Delta + 1)^n(4\eta\Delta + 1)^2(6\eta\Delta + 1)^{2n}(6\eta\Delta + 1)^{3n} < 4^n(6\eta\Delta + 1)^{6n+3}$$

n-tuples $(x_{Q_1}, \ldots, x_{Q_n})$ for which event (E-2) happens. Denote them by $\mathcal{Q}$. The probability that $d_0 \neq 0$ (given $y_P \neq 0$) satisfies

$$\Pr[\text{(E-2)}] \leq \frac{|\mathcal{Q}|}{\left|E_x \setminus \{x_P\}\right|^n} < \frac{4^n(6\eta\Delta + 1)^{6n+3}}{\left(\frac{1}{2}(\#E - 1) - 1\right)^n} \leq \frac{8^n(6\eta\Delta + 1)^{6n+3}}{(p - 2\sqrt{p} - 2)^n}.$$

[7] In the case that R is not a generator of E, one would define $E_x := \{z \in \mathbb{F}_p \mid \exists Q \in \langle R \rangle, Q_x = z\}$. Proving the theorem for *any* R boils down to proving that the roots of (12) are not restricted to E_x.

Probability of event (E-3). Assume (E-3) holds, that is $f'_0 = 0, d_0 = 0$ and $y_P \neq 0$. We may suppose that for all the n-tuples in $\mathcal{Q}$ event (E-3) holds, and thus consider the remaining n-tuples which are not in $\mathcal{Q}$. We first notice that $d_{0,0} = 0$. Indeed, if $d_{0,0} \neq 0$, then by Claim 1 the left hand side of (12) is nonconstant for all $1 \leq i \leq n$. In that case, the only n-tuples that satisfy (12) are in $\mathcal{Q}$. We therefore have $f_0 = f'_0 e_0 - d_0 = 0 = f'_0 e_0^2 - d_{0,0} = f_{0,0}$.

Consider the set $S = \{i \in \{1, \ldots, n\} \mid d_i = d_{0,i} = d_{00,i} = 0\}$. Let $l = |S|$, and notice that if $l = n$ then $f_0 = f_i = f_{0,0} = f_{0,i} = f_{00,i} = 0$, and since $f'_0 = 0$ by assumption then $\mathbf{f} = 0$. As $\mathbf{f}$ is a non-zero vector by construction, $l < n$.

Fix some values of $d_i, d_{0,i}, d_{00,i}$ for $1 \leq i \leq n$. We now consider the number of n-tuples

$$(x_{Q_1}, \ldots, x_{Q_n}) \notin \mathcal{Q}$$

satisfying (12). If $i \in S$ then the left hand side of (12) is the constant zero, and so there are $|E_x| - 1$ possible values for x_{Q_i} satisfying (12). If $i \notin S$ then either $d_i \neq 0$ or $d_{0,i} \neq 0$ or $d_{00,i} \neq 0$ and by Claim 1 the left hand side of (12) is nonconstant, so there are at most 4 solutions x_{Q_i} to the corresponding equation in (12).

Overall, there are at most $4^{n-l}(|E_x| - 1)^l$ n-tuples $(x_{Q_1}, \ldots, x_{Q_n}) \notin \mathcal{Q}$ that satisfy (12). The possible values for each $d_i, d_{0,i}, d_{00,i}$ for each $i \notin S$ are given above. So overall there are at most

$$4^{n-l}(|E_x| - 1)^l (6\eta\Delta + 1)^{n-l} (6\eta\Delta^2 + 1)^{n-l} (6\eta\Delta^3 + 1)^{n-l} <$$
$$4^{n-l}(|E_x| - 1)^l (6\eta\Delta + 1)^{n-l} (6\eta\Delta + 1)^{2(n-l)} (6\eta\Delta + 1)^{3(n-l)} = 4^{n-l}(|E_x| - 1)^l (6\eta\Delta + 1)^{6(n-l)}$$

n-tuples $(x_{Q_1}, \ldots, x_{Q_n}) \notin \mathcal{Q}$ for which event (E-3) happens. Denote them by $\mathcal{Q}'$. Over these tuples (not in Q), the probability that $f'_0 = 0$ (given $d_0 = 0$ and $y_P \neq 0$) is bounded by

$$\frac{|\mathcal{Q}'|}{\left|E_x \setminus \{x_P\}\right|^n} \leq \sum_{l=0}^{n-1} \left(\frac{4(6\eta\Delta + 1)^6}{|E_x| - 1} \right)^{n-l} \leq \sum_{l=1}^{n} \left(\frac{4(6\eta\Delta + 1)^6}{\frac{1}{2}(\#E - 1) - 1} \right)^{l} =$$
$$\sum_{l=1}^{n} \left(\frac{1}{2} \frac{16(6\eta\Delta + 1)^6}{\#E - 3} \right)^{l} \leq \sum_{l=1}^{n} \left(\frac{1}{2} \right)^{l} \left(\frac{16(6\eta\Delta + 1)^6}{p - 2\sqrt{p} - 2} \right)^{l}.$$

If $\frac{16(6\eta\Delta+1)^6}{p-2\sqrt{p}-2} < 1$, then the latter is smaller than $\frac{16(6\eta\Delta+1)^6}{p-2\sqrt{p}-2}$. In any case we get that this probability is bounded by

$$\frac{16(6\eta\Delta + 1)^6}{p - 2\sqrt{p} - 2}.$$

We finally get that the probability that event (E-3) happens satisfies

$$\Pr[\text{(E-3)}] \leq \frac{|\mathcal{Q}|}{\left|E_x \setminus \{x_P\}\right|^n} + \frac{|\mathcal{Q}'|}{\left|E_x \setminus \{x_P\}\right|^n} < \frac{8^n (6\eta\Delta + 1)^{6n+3}}{(p - 2\sqrt{p} - 2)^n} + \frac{16(6\eta\Delta + 1)^6}{p - 2\sqrt{p} - 2}.$$

Notice that the probability that $Q_i = \pm P$ for some $1 \leq i \leq n$ is

$$\frac{2}{\#E - 1} \leq \frac{2}{p - 2\sqrt{p}} .$$

Thus, the probability that $Q_i = \pm P$ for any $1 \leq i \leq n$ is bounded by

$$\frac{2n}{p - 2\sqrt{p}} .$$

This concludes the first claim in the theorem.

Now suppose x_P has been recovered. To determine which of the two values $\pm\sqrt{x_P^3 + ax_P + b}$ is the correct y-coordinate of P, we run the consistency check, which is presented in Appendix A, on both candidates. It is clear that the correct candidate will pass the test. If both candidates pass the consistency check then we cannot determine the point P. We analyze the probability of the event in which the incorrect candidate $-P = (x_P, -y_P)$ passes the test.

We consider how many Q_i lead the system to be consistent with both $\pm y_P$. Recall that

$$h_i + e_i = \left(\frac{y_{Q_i} - y_P}{x_{Q_i} - x_P}\right)^2 - x_P - x_{Q_i} = \frac{x_P x_{Q_i}^2 + (a + x_P^2)x_{Q_i} + ax_P + 2b - 2y_{Q_i}y_P}{(x_{Q_i} - x_P)^2} .$$

If $-P$ passes the test, then there exist $\bar{e}_i$ with $|\bar{e}_i| \leq \Delta$ such that $h_i = (P - Q_i)_x - \bar{e}_i$, for all $1 \leq i \leq n$. We therefore have

$$h_i + \bar{e}_i = \left(\frac{y_{Q_i} + y_P}{x_{Q_i} - x_P}\right)^2 - x_P - x_{Q_i} = \frac{x_P x_{Q_i}^2 + (a + x_P^2)x_{Q_i} + ax_P + 2b + 2y_{Q_i}y_P}{(x_{Q_i} - x_P)^2} .$$

Subtracting one from the other and multiplying by $(x_P - x_{Q_i})^2$ we get

$$(e_i - \bar{e}_i)(x_P - x_{Q_i})^2 = -4y_P y_{Q_i} .$$

Squaring both sides and rearranging results in

$$(e_i - \bar{e}_i)^2(x_P - x_{Q_i})^4 - 16y_P^2(x_{Q_i}^3 + ax_{Q_i} + b) \equiv 0 \pmod{p} .$$

This is a non-constant polynomial in x_{Q_i} of degree 4 and therefore for every $\bar{e}_i$ there are at most 4 values for x_{Q_i} that satisfy this equation. Since there are at most 2Δ possible values for each $\bar{e}_i$, and since we can form n such equations,[8] we conclude that the probability that the point $(x_P, -y_P)$ passes the consistency check is bounded by

$$\frac{4^n(2\Delta)^n}{(|E_x| - 1)^n} \leq \frac{(16\Delta)^n}{(p - 2\sqrt{p} - 2)^n} .$$

This concludes the proof.

[8] Notice that we can also form n equations from the values $h_{i'}$. For each i each solution x_{Q_i} should satisfy an additional equation $(e_{i'} - \bar{e}_{i'})(x_P - x_{Q_i})^2 = 4y_P y_{Q_i}$. However, adding the two equations results in the condition $e_i + e_{i'} - \bar{e}_i - \bar{e}_{i'} = 0$. While this condition can be always satisfied (e.g. $\bar{e}_{i'} = e_i, \bar{e}_i = e_{i'}$), the probability it holds depends on the model for the oracle, i.e. how the noise terms $e_i, e_{i'}$ are generated.

5.2 Proof of Corollary 1

Consider the bounds on $\mathcal{P}_1$ and $\mathcal{P}_2$ in Theorem 1. One needs $1 - \mathcal{P}_1 - \mathcal{P}_2 \geq 1 - p^{-\delta}$, therefore $\mathcal{P}_1 + \mathcal{P}_2 \leq p^{-\delta}$, for the claim to hold. As $\mathcal{P}_2$ is smaller than the first bound on $\mathcal{P}_1$ in Theorem 1 we get that $\mathcal{P}_1 + \mathcal{P}_2$ is bounded by

$$2\frac{8^n(6\eta\Delta+1)^{6n+3}}{(p-2\sqrt{p}-2)^n} + \frac{16(6\eta\Delta+1)^6}{p-2\sqrt{p}-2} + \frac{2n+3}{p-2\sqrt{p}}\,. \tag{14}$$

It is sufficient to bound the latter by $p^{-\delta}$.

Consider the third term in (14). For the claim to hold, one needs

$$\frac{2n_0+3}{p-2\sqrt{p}} < \frac{1}{p^{\delta}}\,,$$

from which it is easy to derive the minimal p (thus the minimal bit size m of p) for the condition to hold. We therefore let δ' such that $p^{-\delta'} = p^{-\delta} - \frac{2n_0+3}{p-2\sqrt{p}}$ (assuming the later is positive) and bound each of the other terms in (14) by $\frac{p^{-\delta'}}{2}$. Notice that $\delta' > \delta$.

Plugging $p = 2^{m+O(1)}$ and $\Delta = 2^{m-k+O(1)}$ in the first term (14), and since $k > (5/6+\epsilon)m$, we have

$$\begin{aligned}\frac{2\cdot 8^n(6\eta\Delta+1)^{6n+3}}{(p-2\sqrt{p}-2)^n} &= \frac{2^{3n+1}(2^{O(1)}\eta 2^{m-k+O(1)}+1)^{6n+3}}{(2^{m+O(1)}-2^{m/2+O(1)}-2)^n} = \eta^{6n+3}2^{(6n+3)(m-k+O(1))-(m+O(1))n} \\ &\leq \eta^{6n+3}2^{(6n+3)(m/6-m\epsilon+O(1))-(m+O(1))n} = 2^{(6n+3)(\log\eta-m\epsilon)+m/2+O(n)}\,.\end{aligned}$$

The latter is smaller than $\frac{p^{-\delta'}}{2} = 2^{-\delta'(m-1+O(1))}$ if $(6n+3)(\log\eta - \epsilon m) + m/2 + O(n) \leq -\delta'(m + O(1))$, which simplifies to (for some sufficiently large absolute constant C_0)

$$(6n+3)(\epsilon - m^{-1}(\log\eta + C_0)) \geq \delta' + \frac{1}{2} > \delta + \frac{1}{2}\,. \tag{15}$$

Using $3\epsilon \geq \delta$ and $n \geq n_0$, it is easy to verify that (for a sufficiently large absolute constant C_1)

$$m > \epsilon^{-1}(2\log\eta + C_1) \tag{16}$$

implies (15).

Similarly, to show that the second term in (14) is bounded by $\frac{p^{-\delta'}}{2}$ one gets the condition (for some sufficiently large absolute constant C_2)

$$6(\epsilon - m^{-1}(\log\eta + C_3)) \geq \delta' > \delta\,,$$

which can be shown to hold when (for a sufficiently large absolute constant C_3)

$$m > (6\log\eta + C_3)(6\epsilon - \delta)^{-1}\,.$$

The latter is implied by (15), therefore by (16), provided C_0 is large enough.

For A_1 we apply the 1-SVP algorithm (with running time $\tilde{O}(2^{2d})$) of Micciancio and Voulgaris [19] to a lattice of dimension $d = 3n_0 + 3$, which gives $\eta = 2\sqrt{3n_0 + 1}$. For A_2, we use the $2^{O(d(\log\log d)^2/\log d)}$-SVP algorithm (with running time $\tilde{O}(d)$) of Schnorr [21] for the dimension $d = 3n_0 + 3$, which gives $\eta = 2^{n_0+2}\sqrt{3n_0 + 1}$. Using $n_0 = \lceil \frac{1}{6\epsilon} \rceil$, the bounds m_i follow.

6. Additional results

The techniques presented in the previous sections can be used to show some additional results. Considering EC-HNP with the LSB_k function, similar results can be derived for the least significant $5/6$ bits of the x-coordinate as we show in Section 6.1. In Section 6.2 we consider the Diffie–Hellman key exchange protocol in elliptic curves over extension fields $\mathbb{F}_q$ for $q = p^d$ with $d > 1$ (an example of such useful curves are the GLS curves [11], for which $d = 2$), and the problem of recovering the Diffie–Hellman key given ability to compute one component of the key. That is, we consider the problem of recovering a Diffie–Hellman key $[ab]P \in E(\mathbb{F}_q)$ from an algorithm that takes $P, [u]P, [v]P$ and outputs a single component in $\mathbb{F}_p$ of $([uv]P)_\psi$ for a coordinate $\psi \in \{x, y\}$. We give a general method to solve this problem, discuss several approaches, and improve the known result to any constant extension degree.

6.1 EC-HNP with least significant bits

We first revise the approach for HNP, where one transforms HNP with the LSB_k function to HNP with MSB_k. As we allow k to take any (positive) real value, we define LSB_k by $\mathrm{LSB}_k(x) := x \pmod{\lceil 2^k \rceil}$. In other words, $\mathrm{LSB}_k(x)$ gives x mod l for $2 \le l = \lceil 2^k \rceil \le p$, not necessarily a power of 2.

Let $h = \mathrm{LSB}_k(st) = (st \mod p) \mod l = (st - qp) - el$ for some q and $|e| < \frac{p}{2l} \le \frac{p}{2^{k+1}}$. For $u = l^{-1} \in \mathbb{Z}_p^*$ we have

$$\overline{h} := hu = (st - qp - el)u = (sut - q'p) - e \equiv sut - e = \mathrm{MSB}_k(sut) \pmod{p}.$$

This shows that for HNP the solution for MSB_k also gives a solution to LSB_k, as one can reduce the latter to the former. In contrast, for EC-HNP it is not clear how to transform EC-HNP with LSB_k to EC-HNP with MSB_k. Instead, as we now show, one can "linearize" the problem and then solve it with a similar approach to the solution for EC-HNP$_x$ with the MSB_k function.

Let $h = \mathrm{LSB}_k((P + Q)_x) = (P + Q)_x \mod l = (s_{P+Q}^2 - x_P - x_Q - qp) - le$ for some q and

$|e| < \frac{p}{2l} \leq \frac{p}{2^{k+1}}$. For $u = l^{-1} \in \mathbb{Z}_p^*$ we have (where the operations are in $\mathbb{F}_p$)

$$\begin{aligned}\overline{h} := hu &= \left(\left(\frac{y_P - y_Q}{x_P - x_Q}\right)^2 - x_P - x_Q - qp - le\right) u \\ &= u\left(\left(\frac{y_P - y_Q}{x_P - x_Q}\right)^2 - x_P - x_Q\right) - q'p - e \equiv u\left(\left(\frac{y_P - y_Q}{x_P - x_Q}\right)^2 - x_P - x_Q\right) - e\,.\end{aligned}$$

Now let $h_0 = \mathrm{LSB}_k(x_P) = x_P - le_0$ and $h' = \mathrm{LSB}_k((P-Q)_x) = (P-Q)_x \mod l = (s_{P-Q}^2 - x_P - x_Q - rp) - le'$ for some r and $|e_0|, |e'| < \frac{p}{2l} \leq \frac{p}{2^{k+1}}$. Then

$$\overline{h'} := h'u \equiv u\left(\left(\frac{y_P + y_Q}{x_P - x_Q}\right)^2 - x_P - x_Q\right) - e' \pmod p\,.$$

Letting $\widetilde{h} = \overline{h} + \overline{h'}$ and $\widetilde{e} = e + e'$ and plugging $x_P = h_0 + le_0$ in (1) above we get

$$\begin{aligned}\widetilde{h} + \widetilde{e} &= u\left((P+Q)_x + (P-Q)_x\right) \\ &\equiv 2u\left(\frac{x_Q(h_0 + le_0)^2 + (a + x_Q^2)(h_0 + le_0) + ax_Q + 2b}{(h_0 + le_0 - x_Q)^2}\right) \pmod p\,.\end{aligned}$$

Multiplying by $(h_0 + le_0 - x_Q)^2$ results in a bivariate polynomial in $e_0, \widetilde{e}$ of degree 3, similar to (2) above, and the rest of the solution follows.

Remark 4. Nguyen and Shparlinski used continued fractions to generalize this conversion for inner bits [20, Section 5.1]. That is, they showed how to convert a hidden number problem with inner bits to hidden number problem with most significant bits (MSB). When converting to MSB, the noise level in this technique grows by a factor of at least 2.[9] Again, it is unclear how to convert EC-HNP with inner bits to EC-HNP with MSB, but we can form "linear" equations as above. Nevertheless, to achieve the noise level coming from the most significant $5/6$ bits, would require to start with $2 \cdot 5/6$ fraction of inner bits. As this value exceeds 1, this problem is trivial.

6.2 Bit security of elliptic curve Diffie–Hellman over extension fields

The field $\mathbb{F}_q = \mathbb{F}_{p^d}$ is a d-dimensional vector space over $\mathbb{F}_p$. We fix a basis $\{\mathbf{b_1}, \ldots, \mathbf{b_d}\}$ for $\mathbb{F}_q$, and represent points $\mathbf{x} \in \mathbb{F}_q$ with respect to that basis: for $\mathbf{x} = \sum_{i=1}^d x^i \mathbf{b_i}$ we write $\mathbf{x} = (x^1, \ldots, x^d)$. We consider $E(\mathbb{F}_q)$, the group of elliptic curve points over $\mathbb{F}_q$.

The elliptic curve hidden number problem in this setting can be formulated as the problem of computing a secret point $P \in E$, given partial information on a specific component. For example, one gets $\mathrm{MSB}_k\left((P+Q)_x^i\right)$ for some component $1 \leq i \leq d$ and some number k.

[9] Depends whether the bits are consecutive or not.

A natural and important question is whether the ability to recover one component – that is, when $k = \log p$ in the example above – allows to recover the entire secret point. This question is natural as extension fields provide additional algebraic structure that can be exploited, and it is important since it potentially allows to prove stronger results than in prime-field cases as one component represents only a fraction of $1/d$ of all bits. For example, for HNP over $\mathbb{F}_q^*$, Verheul [26] used the fact that a single component of the product st is linear in all of the components of s, to give a reduction from computing the entire secret to computing any component. In particular it shows that for fields of characteristic 2, single bits are as hard to compute as the entire secret point.

This problem, in the elliptic curve context, was studied by Jao, Jetchev and Venkatesan (JJV) [15]. They consider the following hidden number problem for elliptic curves, which they call *multiplier elliptic curve hidden number problem*: Given an oracle $\mathcal{O}$ that computes a single component of the x-coordinate of the map $r \to [r]P$, that is $\mathcal{O}(r) = ([r]P)_x^i$, recover the point P. Similar to the direction taken in Remark 1, one can easily see that solving this problem will give a bit security result for elliptic curve Diffie–Hellman key exchange.

The algorithm given by JJV to this problem is polynomial in $\log(p)$ but not in d, and therefore suits problems where one fixes the degree d and let $\log p$ grow. That is, for extension fields $\mathbb{F}_{p^d}$ of a constant degree. However, there is a drawback in JJV's approach that leads them to prove that their algorithm gives a polynomial time solution only for degrees $d = 2, 3$. We explain this drawback below. We also show that the solution for $d = 3$ has a minor mistake and is incomplete.

The approach presented here overcomes this drawback, and therefore gives a complete solution to any constant extension degree. Moreover, the solution holds for the y-coordinate as well. Our solution is based on (a generalization of) the algorithm given by JJV.

The following section generalizes the method taken in [15, Section 3.3], and thus unifies the different approaches one can take to solve elliptic curve hidden number problems. Afterwards, we state the new result and compare our approach with the previous one.

In a nutshell, the essence of the solution is to construct a system of (small degree) polynomials with small number of variables, for which either $\mathbf{x_P} = (x_P^1, \ldots, x_P^d)$ or $\mathbf{y_P} = (y_P^1, \ldots, y_P^d)$ or $(\mathbf{x_P}, \mathbf{y_P})$ is a simultaneous solution, which will result in some small number of candidates for P. JJV suggest using a Gröbner basis algorithm to find the solutions to the system. Approaching such elimination problem, the polynomials' degree, the number of variables in the system and the number of solutions to the polynomial system play a main role in the complexity of this task.

6.2.1 A general method

The method presented in this section applies for both EC-HNP considered in this paper and the multiplier version considered in [15]. To make this section more accessible, we first give a few concrete

approaches to the problem. Fix a point $Q \in E$, let $P = (\mathbf{x}, \mathbf{y}) \in E$ be a variable, and let $\psi \in \{x, y\}$. The following facts give rise to three approaches.

natural: Both point addition $(P+Q)_\psi$ and point multiplication $([r]P)_\psi$ can be represented by rational functions in $\mathbf{x}, \mathbf{y}$; the former has small degree, and the degree of the latter is polynomial in r.

JJV: Point multiplication $([r]P)_x$ can be represented by a rational function only in $\mathbf{x}$ using the elliptic curve division polynomials, where its degree is polynomial in r.[10]

our: The function $(P+Q)_x + (P-Q)_x$ and the function $(P+Q)_y - (P-Q)_y$ can be represented by small-degree rational functions in $\mathbf{x}$.

The JJV approach is taken by Jao, Jetchev and Venkatesan [15]. This paper shows that $(P+Q)_x + (P-Q)_x$ is a rational function of degree 2, and similarly one can derive that $(P+Q)_y - (P-Q)_y$ is a rational function of degree 3 (see below).

To unify these different approaches, we let an unknown $P \in E$ and a known $R \in E$ be points and consider a family of functions $f_t : E \to \mathbb{F}_q$; in the first approach above $f_t(P) = (P + [t]R)_\psi$ or $f_t(P) = ([t]P)_\psi$, in the second $f_t(P) = ([t]P)_x$ and in the third $f_t(P) = (P+[t]R)_\psi \pm (P+[-t]R)_\psi$. We also consider an oracle $\mathcal{O} : [0, \#E - 1] \to \mathbb{F}_p$ that outputs a single component of $f_t(P)$, that is $\mathcal{O}(t) = \mathcal{O}_P(t) = f_t^i(P)$. It is clear that in these three approaches given an oracle for a component of the Diffie–Hellman key, one can form $\mathcal{O}$.

Suppose we can represent $f(P) = f_t(P)$ as a rational function, that is

$$f(P) = \frac{R_1(x_P^1, \ldots, x_P^d, y_P^1, \ldots, y_P^d)}{R_2(x_P^1, \ldots, x_P^d, y_P^1, \ldots, y_P^d)},$$

where R_1, R_2 are polynomials in $\mathbb{F}_q[z_1, \ldots, z_{2d}]$.[11] Rewrite

$$\frac{R_1(z_1, \ldots, z_{2d})}{R_2(z_1, \ldots, z_{2d})} = \frac{R_1^1 \mathbf{b}_1 + \ldots + R_1^d \mathbf{b}_d}{R_2^1 \mathbf{b}_1 + \ldots + R_2^d \mathbf{b}_d},$$

where for $1 \le j \le d$ each polynomial $R_1^j(z_1, \ldots, z_{2d}), R_2^j(z_1, \ldots, z_{2d})$ has coefficients in $\mathbb{F}_p$. We "rationalize" the denominator (by multiplying the numerator and denominator by a polynomial such that the denominator belongs to $\mathbb{F}_p[z_1, \ldots, z_{2d}]$) to express

$$\frac{R_1(z_1, \ldots, z_{2d})}{R_2(z_1, \ldots, z_{2d})} = r^1(z_1, \ldots, z_{2d})\mathbf{b}_1 + \ldots + r^d(z_1, \ldots, z_{2d})\mathbf{b}_d,$$

where r^j are rational functions with coefficients in $\mathbb{F}_p$. We assume to have access to component i of $f_t(P)$, that is, we have

$$\mathcal{O}(t) = f_t^i(P) = r_t^i(x_P^1, \ldots, x_P^d, y_P^1, \ldots, y_P^d) = \frac{r_{t,1}^i(x_P^1, \ldots, x_P^d, y_P^1, \ldots, y_P^d)}{r_{t,2}^i(x_P^1, \ldots, x_P^d, y_P^1, \ldots, y_P^d)}.$$

[10] Also $([r]P)_y$ can be represented by a rational function using the division polynomials, in variables $\mathbf{x}, \mathbf{y}$.

[11] This is the general method. In JJV and our approaches we have functions only in $\mathbf{x} = (x^1, \ldots, x^d)$.

Multiplying by $r^i_{t,2}$ and rearranging we get that the following polynomials

$$g_t(z_1, \ldots, z_{2d}) := r^i_{t,1}(z_1, \ldots, z_{2d}) - r^i_{t,2}(z_1, \ldots, z_{2d}) f^i_t(P)$$

are polynomials in $\mathbb{F}_p[z_1, \ldots, z_{2d}]$, for which the point $P = (x^1_P, \ldots, x^d_P, y^1_P, \ldots, y^d_P)$ is a simultaneous solution, and so if one can find the solutions to this system, then one can recover P.

Notice that in the rationalization step one multiplies the denominator by all of its $d-1$ conjugates (to admit the norm). Thus, if the denominator's degree is l, one would multiply in general by a polynomial of degree $l(d-1)$. Hence, each of the polynomials g_t is in general of degree at least d. Furthermore, we need at least $2d$ equations to solve a system with $2d$ variables. Elimination techniques can then be used to reduce the number of variables. This elimination process is polynomial in terms of the ground field, that is in $\log(p)$, but not necessarily in the degrees. The current algorithms to solve systems of $2d$ equations, all of degree at least d, do not run in time polynomial in d. Therefore, this method is efficient if one fixes the extension degree d, as then the affect of d on the elimination process running time is constant.

6.2.2 Improved results

As noted above, JJV suggest computing a Gröbner basis for the ideal generated by $\{g_t\}$, and range over its roots, one of which is P. We remark that the analysis of time complexity of current Gröbner bases algorithms is complex and depends on the number of solutions to the system, the degree of the polynomials and number of variables. Therefore, it seems desirable to take approaches in which the resulting polynomials have fewer variables, like JJV or our approaches described above, and that are of minimal degree.

We start by explaining the drawback in the approach of JJV. As identified in [15], there are two complexities in their approach: as the degree of the division polynomials grows exponentially in the size of the multiplier r, one can only write down the explicit algebraic expressions for small multipliers; as a consequence, it is not clear that by considering only small multipliers r, this hidden number problem has a unique solution, or a small set of solutions.[12]

Specifically, this leads the paper [15] to give precise statements only for degrees 2 and 3 (Propositions 3.1 & 3.2), but to leave the constant degree case (Section 3.3) with a description of a general approach, and so a proof of bit security cannot be derived in this case.

In contrast, our approach overcomes these complexities. Indeed, as the degree of the polynomials is independent of the size of the 'multipliers' Q, the points Q are not restricted to any (short) interval.

[12] For comparison, it is easy to show that restricting to small multipliers in HNP in $\mathbb{F}^*_p$ yields exponentially many solutions.

As already mentioned in [15], in the case of random multipliers, it is easy to argue for uniqueness.[13] Indeed, in our case we get

$$(P+Q)_x + (P-Q)_x = 2\left(\frac{\mathbf{x}_Q\mathbf{x}_P^2 + (\mathbf{a}+\mathbf{x}_Q^2)\mathbf{x}_P + \mathbf{a}\mathbf{x}_Q + 2\mathbf{b}}{(\mathbf{x}_P - \mathbf{x}_Q)^2}\right) = \frac{R_1(x_P^1,\ldots,x_P^d)}{R_2(x_P^1,\ldots,x_P^d)},$$

and by the aforementioned method we have

$$g_Q(x^1,\ldots,x^d) := r^i_{Q,1}(x^1,\ldots,x^d) - r^i_{Q,2}(x^1,\ldots,x^d)\left((P+Q)^i_x + (P-Q)^i_x\right),$$

for low degree polynomials $r^i_{Q,1}, r^i_{Q,2}$. Standard arguments (like the root counting above) can be used to show that for uniform and independent Q's, a sufficiently large system $\{g_Q\}$ is expected to have a unique (simultaneous) root. The same arguments hold for the y-coordinate, where one takes the third-degree polynomial

$$(P+Q)_y - (P-Q)_y = 2\mathbf{y}_Q\left(\frac{\mathbf{x}_P^3 + 3\mathbf{x}_Q\mathbf{x}_P^2 + 3\mathbf{a}\mathbf{x}_P + \mathbf{a}\mathbf{x}_Q + 4\mathbf{b}}{(\mathbf{x}_P - \mathbf{x}_Q)^3}\right).$$

This overcomes the difficulty [15] has in proving their result for every constant extension degree. We therefore get the following results (for a more structured proof see [22]).

Proposition 1. *Let E be an elliptic curve over an extension field $\mathbb{F}_{p^d}$. There exists an algorithm, polynomial in $\log p$, that solves EC-HNP given an oracle that outputs a complete component of either the x or y coordinates.*

Corollary 2. *For an elliptic curve defined over a constant-degree extension field, computing a single component of the Diffie–Hellman key (for either the x or y coordinates) is as hard as computing the entire key.*

We now give a comparison between JJV's approach and our approach. As explained, both approaches have systems with d variables, therefore both need d queries to the oracle. We therefore focus on the polynomials' degree and subsequently the running time.

6.2.3 Comparison of approaches

We show that the new approach leads to a more efficient algorithm. In JJV's approach the polynomials g_r are of different form (given by different division polynomials) and their degree can be as large as $d(r^2-1)+1$, for each multiplier r. On the other hand, the polynomials g_Q in our approach are of the same form and of degree at most $2d$ ($3d$ for the y-coordinate). Moreover, for both the x-coordinate and the y-coordinate our approach results in polynomials only in $\mathbf{x}$, that is, in d variables.

[13] We note that the multipliers here and in [15] have different context, as the elliptic curve hidden number problem is defined differently. However, the arguments for uniqueness stay the same.

We remark that in practice it is suggested to use Gröbner bases algorithms with more equations than information theoretically needed (to reduce the system to dimension zero). While in our approach each new equation is of a fixed degree, in the JJV approach this involves equations of higher degrees, and so it is not clear that adding new equations to the system will indeed result in a better running time.

Specifically, following the approach of JJV for degrees $d = 2, 3$ (even without our uniqueness argument) one can see that our approach results in smaller lists of candidates for P than those coming from using division polynomials (see [15, Sections 3.1 & 3.2] for the exact details). Indeed, for $d = 2$, JJV's approach ends with a univariate polynomial of degree 7 (therefore there are at most 7 $\mathbb{F}_p$-solutions), while our approach leads to a polynomial of degree 4; for $d = 3$, JJV's approach results in two bivariate polynomials of degree $10, 25$ in each variable, and using the resultant one forms a univariate polynomial of degree at most 500, while our approach leads to two bivariate polynomials, both of degree 6, and so with at most 72 $\mathbb{F}_p$-solutions.

Correction. We finish with a couple of remarks regarding the solution for $d = 3$ in [15, Section 3.2]. As mentioned above, JJV take the resultant of two bivariate polynomials of degree $10, 25$ in each variable. First, as we show in Appendix B, this resultant is a univariate polynomial of degree at most 500, not 250 as written there. More importantly, while the resultant's degree is bounded by a constant value, in general it can also be identically zero, which will then not yield a constant-sized set of possible solutions (as the zero polynomial is satisfied by every point). This point is important, especially because the authors identify a problem with showing uniqueness of the solution, or the existence of a small set of solutions. However, the paper [15] does not treat this point.

7. Comments

It is desirable to get bit security results also in the case of an imperfect oracle. The main obstacle in achieving such a result is that the lattice constructed by the algorithm has to be of an exact shape, which will not be achieved in general if some equations are not of the right form. It should be noted that like other problems (see for example [9, Section 4.1] for HNP) one can consider an imperfect oracle which is very likely to answer all the queries correctly, when its inputs are random. In addition, one can consider the approach suggested in [13] for imperfect oracles.

A natural question is whether a similar strong bit security result can be shown for the y-coordinate of the elliptic curve Diffie–Hellman key. Unfortunately, the trick presented in this paper, using 2 correlated equations to eliminate one variable, seems out of reach when one works with the y-coordinate. We remark that one can still get some results using the approaches described in Section 4.1, but they ought to be weak results.

Moreover, while Weierstrass equations are normally used to represent elliptic curves, Edwards

curves are also of interest. The y-coordinate in Edwards curves is considered analogous to the x-coordinate in Weierstrass curves. One therefore expects to have analogous equations for $(P+Q)_y + (P-Q)_y$ and for the y-coordinate of point multiplication, i.e. $([r]P)_y$. It is of interest to get solutions for the elliptic curve hidden number problem using Edwards curves as well.

Acknowledgements Many thanks to my supervisor Steven Galbraith for his help and guidance.

REFERENCES

[1] Babai, L. (1986) "On Lovász' Lattice Reduction and the Nearest Lattice Point Problem," in Combinatorica, **6**(1), 1–13.

[2] Blackburn, S.R., Gomez-Perez, D., Gutierrez, J, and Shparlinski, I.E. (2003) "Predicting the Inversive Generator," in Paterson, K.G. (ed.) Cryptography and Coding. LNCS, vol. 2898, pp. 264–275. Springer, Heidelberg.

[3] Blackburn, S.R., Gomez-Perez, D., Gutierrez, J, and Shparlinski, I.E. (2005) "Predicting Nonlinear Pseudorandom Number Generators," in Mathematics of Computation, **74**(251), 1471–1494.

[4] Blake, I.F., and Garefalakis, T. (2004) "On the Complexity of the Discrete Logarithm and Diffie–Hellman Problems," in Journal of Complexity, **20**(2-3), 148–170.

[5] Blake, I.F., Garefalakis, T., and Shparlinski, I.E. (2006) "On the Bit Security of the Diffie–Hellman Key," in Applicable Algebra in Engineering, Communication and Computing, **16**(6), 397–404.

[6] Boneh, D. (1998) "The Decision Diffie–Hellman problem," in Buhler, J.P. (ed.) Algorithmic Number Theory – ANTS-III. LNCS, vol. 1423, pp. 48–63. Springer, Heidelberg.

[7] Boneh, D., Halevi, S., and Howgrave-Graham, N. (2001) "The Modular Inversion Hidden Number Problem," in Boyd, C. (ed.) Advances in Cryptology – ASIACRYPT 2001. LNCS, vol. 2248, pp. 36–51. Springer, Heidelberg.

[8] Boneh, D., and Shparlinski, I.E. (2001) "On the Unpredictability of Bits of the Elliptic Curve Diffie–Hellman Scheme," in Kilian, J. (ed.) Advances in Cryptology – CRYPTO 2001. LNCS, vol. 2139, pp. 201–212. Springer, Heidelberg.

[9] Boneh, D., and Venkatesan, R. (1996) "Hardness of Computing the Most Significant Bits of Secret Keys in Diffie–Hellman and Related Schemes," in Koblitz, N. (ed.) Advances in Cryptology – CRYPTO 1996. LNCS, vol. 1109, pp. 129–142. Springer, Heidelberg.

[10] Coppersmith, D. (1997) "Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities," in Journal of Cryptology, **10**(4), 233–260.

[11] Galbraith, S.D., Lin, X. and Scott, M. (2011) "Endomorphisms for Faster Elliptic Curve Cryptography on a Large Class of Curves," in Journal of Cryptology, **24**(3), 446–469.

[12] González Vasco, M.I., and Näslund, M. (2001) "A Survey of Hard Core Functions," in Lam, K.-Y., Shparlinski, I., Wang, H., Xing, C. (eds.) Proc. Workshop on Cryptography and Computational Number Theory 1999. Progress in Computer Science and Applied Logic, vol. 20, pp. 227–255. Birkhäuser, Basel.

[13] González Vasco, M.I., Näslund, M., and Shparlinski, I.E. (2004) "New Results on the Hardness of Diffie–Hellman Bits," in Bao, F., Deng, R., Zhou, J. (eds.) Public Key Cryptography – PKC 2004. LNCS, vol. 2947, pp. 159–172. Springer, Heidelberg.

[14] González Vasco, M.I., and Shparlinski, I.E. (2001) "On the Security of Diffie-Hellman Bits," in Lam, K.-Y., Shparlinski, I., Wang, H., Xing, C. (eds.) Proc. Workshop on Cryptography and Computational Number Theory 1999. Progress in Computer Science and Applied Logic, vol. 20, pp. 257–268. Birkhäuser, Basel.

[15] Jao, D., Jetchev, D., and Venkatesan, R. (2007) "On the Bits of Elliptic Curve Diffie–Hellman Keys," in Srinathan, K., Pandu Rangan, C., Yung, M. (eds.) Progress in Cryptology – INDOCRYPT 2007. LNCS, vol. 4859, pp. 33–47. Springer, Heidelberg.

[16] Jetchev, D., and Venkatesan, R. (2008) "Bits Security of the Elliptic Curve Diffie–Hellman Secret Keys," in Wagner, D. (ed.) Advances in Cryptology – CRYPTO 2008. LNCS, vol. 5157, pp. 75–92. Springer, Heidelberg.

[17] Li, W.-C. W., Näslund, M., and Shparlinski, I.E. (2002) "Hidden Number Problem with the Trace and Bit Security of XTR and LUC," in Yung, M. (ed.) Advances in Cryptology – CRYPTO 2002. LNCS, vol. 2442 , pp. 433–448. Springer, Heidelberg.

[18] Ling, S., Shparlinski, I.E., Steinfeld, R., and Wang, H. (2012) "On the Modular Inversion Hidden Number Problem," in Journal of Symbolic Computation, **47**(4), 358–367.

[19] Micciancio, D., and Voulgaris, P. (2013) "A Deterministic Single Exponential Time Algorithm for Most Lattice Problems Based on Voronoi Cell Computations," in SIAM Journal on Computing, **42**(3), 1364–1391.

[20] Nguyen, P.Q., and Shparlinski, I.E. (2002) "The Insecurity of the Digital Signature Algorithm with Partially Known Nonces," in Journal of Cryptology, **15**(3), 151–176.

[21] Schnorr, C.P. (1987) "A Hierarchy of Polynomial Time Lattice Basis Reduction Algorithms," in Theoretical Computer Science, **53**(2-3), 201–224.

[22] Shani, B. (2017) "On the Bit Security of Elliptic Curve Diffie–Hellman," in Fehr, S. (ed.) Public Key Cryptography – PKC 2017. LNCS, vol. 10174, pp. 361–387. Springer, Heidelberg.

[23] Shparlinski, I.E. (2001) "Sparse Polynomial Approximation in Finite Fields," in Proc. 33rd ACM Symposium on Theory of Computing – STOC 2001, pp. 209–215. ACM, New York.

[24] Shparlinski, I.E. (2002) "Playing "Hide-and-Seek" in Finite Fields: Hidden Number Problem and its Applications," in Proc. 7th Spanish Meeting on Cryptology and Information Security, pp. 49–72.

[25] Shparlinski, I.E. (2005) "Playing "Hide-and-Seek" with Numbers: The Hidden Number Problem, Lattices and Exponential Sums," in Garrett, P., Lieman, D. (eds.) Public-Key Cryptography; Proceedings of Symposia in Applied Mathematics, vol. 62, AMS, pp. 153–177.

[26] Verheul, E.R. (2000) "Certificates of Recoverability with Scalable Recovery Agent Security," in Imai, H., Zheng, Y. (eds.) Public Key Cryptography – PKC 2000. LNCS, vol. 1751, pp. 258–275. Springer, Heidelberg.

A. Consistency check – filtering impossible secrets.

We introduce a test that takes a candidate P' for the secret point P, and determines whether P' is not the secret. That is, after running the test, P' is either guaranteed not to be P or it is potentially the secret point P. We give a bound on the probability that the outcome of the test is inconclusive, for $P' \neq P$ (it is clear that if $P' = P$ the test is inconclusive). Specifically, given the candidate for x_P from Theorem 1, one can test which value (if any) is the correct y-coordinate y_P. Moreover, one can test whether $y_P \neq 0$ or $P \neq \pm Q_i$.

Given a candidate $P' = (x_{P'}, y_{P'})$, the consistency check goes over the pairs $(Q, h = \mathrm{MSB}_k((P + Q)_x))$ and checks if these values are consistent with the problem's settings. That is, we use h to derive a candidate $\bar{e}$ for the noise e, and check if $|\bar{e}| \leq \Delta$. Formally, using $h_0 = x_P - e_0$ we compute

$$\bar{e}_0 := x_{P'} - h_0 \mod p\,,$$

and check if $|\bar{e}_0| \leq \Delta$. If so then for every $1 \leq i \leq n$ using $h_i = \mathrm{MSB}_k((P + Q_i)_x)$ we compute

$$\bar{e}_i := \left(\frac{y_{P'} - y_Q}{x_{P'} - x_Q}\right)^2 - x_{P'} - x_Q - h_i \mod p\,,$$

and check if $|\bar{e}_i| \leq \Delta$. We do the same process with $h_{i'}$. If at any point this inequality does not hold, we can stop the test and determine that $P' \neq P$. Otherwise, P' passes the consistency check and is potentially the secret point P.

For completeness, we analyze the probability (over the samples Q_i) of the event in which a candidate $P' \neq P$ passes the consistency check. Hence, suppose that $P' = (x_{P'}, y_{P'})$ passed the consistency check.

Probability of $x_{P'} \neq x_P$. Given $h_i, h_{i'}$, from Section 4.1.2 above we have

$$h_i + h_{i'} = 2\left(\frac{x_P x_{Q_i}^2 + (a + x_P^2)x_{Q_i} + ax_P + 2b}{(x_P - x_{Q_i})^2}\right) - e_i - e_{i'}\,.$$

Since P' passed the consistency check there exist $|\bar{e}_i|, |\bar{e}_{i'}| \leq \Delta$ such that

$$h_i + h_{i'} = 2\left(\frac{x_{P'} x_{Q_i}^2 + (a + x_{P'}^2)x_{Q_i} + ax_{P'} + 2b}{(x_{P'} - x_{Q_i})^2}\right) - \bar{e}_i - \bar{e}_{i'}\,.$$

Subtracting these two equations and multiplying by $(x_P - x_{Q_i})^2(x_{P'} - x_{Q_i})^2$ we get

$$\begin{aligned}
&(e_i + e_{i'} - \bar{e}_i - \bar{e}_{i'})(x_P - x_{Q_i})^2(x_{P'} - x_{Q_i})^2 = \\
&2\Big((x_P x_{Q_i}^2 + (a + x_P^2)x_{Q_i} + ax_P + 2b)(x_{P'} - x_{Q_i})^2 \\
&-\ (x_{P'} x_{Q_i}^2 + (a + x_{P'}^2)x_{Q_i} + ax_{P'} + 2b)(x_P - x_{Q_i})^2\Big)\,.
\end{aligned}$$

By rearranging we get a polynomial in x_{Q_i} of degree 4. By simple algebra one can check that this polynomial is identically zero if and only if $x_{P'} = x_P$ (thus $e_i + e_{i'} - \bar{e}_i - \bar{e}_{i'} = 0$). We assume $x_{P'} \neq x_P$. Therefore for every $\bar{e}_i, \bar{e}_{i'}$ there are at most 4 values for x_{Q_i} that satisfy this equation. Since there are $2\Delta + 1$ possible values for each $\bar{e}_i, \bar{e}_{i'}$ we conclude that the probability that $x_{P'} \neq x_P$ is bounded by

$$\frac{4^n(2\Delta + 1)^{2n}}{(|E_x| - 1)^n} \leq \frac{2^n(4\Delta + 2)^{2n}}{(p - 2\sqrt{p} - 2)^n}\,.$$

Probability of $x_{P'} = x_P$ and $y_{P'} \neq y_P$. The probability that $P' = (x_P, -y_P)$ passes the consistency check, is analyzed at the end of the proof of Theorem 1, and shown to be bounded by

$$\frac{4^n(2\Delta)^n}{(|E_x| - 1)^n} \leq \frac{(16\Delta)^n}{(p - 2\sqrt{p} - 2)^n}\,.$$

Remark 5. Although the aim of this paper is to give a bit security result and not a practical algorithm, for completeness purposes we consider a matter of practice. In the case in which the value $d_0 \neq 0$, the recovered value $e := f_0/f_0' \neq e_0$, and therefore $x_{P'} := h + e \neq x_P$. Running the consistency check on P' might reveal that indeed $P' \neq P$. One can derive from equations (8)-(11) other candidates for e_0 and subsequently candidates for x_P, and apply the consistency check on them. If none of these

candidates pass the consistency check, then one can test P' where $y_{P'} = 0$ and $P' = \pm Q_i$. We analyze the probability that there exists $P' \neq P$ that is consistent with all $2n+1$ samples.

We use the analysis above which shows that the probability that a candidate P' with $x_{P'} \neq x_P$ passes the test with the $2n$ equations is bounded by

$$\frac{(4\Delta+2)^{2n}}{(|E_x|-1)^n} \leq \frac{2^n(4\Delta+2)^{2n}}{(p-2\sqrt{p}-2)^n}\,.$$

We also have $x_{P'} - \bar{e}_0 = h_0 = x_P - e_0$, so $x_{P'} = x_P - e_0 + \bar{e}_0$ can take 2Δ values. Thus, the probability that any P' with $x_{P'} \neq x_P$ passes the consistency check is bounded by

$$\frac{2^{n+1}\Delta(4\Delta+2)^{2n}}{(p-2\sqrt{p}-2)^n}\,.$$

With the above bound for $y_{P'} \neq -y_P$ we get that the probability that there exists $P' \neq P$ that passes the consistency check is bounded by

$$\frac{2^{n+1}\Delta(4\Delta+2)^{2n}}{(p-2\sqrt{p}-2)^n} + \frac{(16\Delta)^n}{(p-2\sqrt{p}-2)^n}\,.$$

B. Resultant's degree

Claim 2. Let $p, q \in k[x,y]$ be two polynomials with

$$\deg_x p = n_x\,,\ \deg_y p = n_y\,,$$
$$\deg_x q = m_x\,,\ \deg_y q = m_y\,.$$

Then the degree (in x) of the resultant of p and q in variable y is at most $m_y n_x + n_y m_x$.

Proof. The Sylvester matrix of p and q with respect to y is a $(m_y+n_y)\times(m_y+n_y)$ matrix. The first m_y rows, coming from the coefficients of p, contain polynomials in x of degree at most n_x. Similarly, the last n_y rows contain polynomials in x of degree at most m_x. The resultant of p and q in variable y is given by the determinant of this matrix, which is formed by summing products of an entry from each row. The first m_y rows contribute at most $m_y n_x$ to the degree of x, and the last n_y rows contribute at most $n_y m_x$. ∎