Algorithms for Black-Box Fields and their Application to Cryptography

(extended abstract)

Dan Boneh
dabo@cs.princeton.edu

Richard J. Lipton*
rjl@cs.princeton.edu

Princeton University, Princeton NJ 08544

Abstract. We introduce the notion of a black box field and present several algorithms for manipulating such fields. Black box fields arise naturally in cryptography and our algorithms have several cryptographic implications. First, our results show that any algebraically homomorphic cryptosystem can be broken in sub-exponential time. The existence of such cryptosystems was posed as an open problem in [12]. Second we show that over elliptic (or hyperelliptic) curves the hardness of computing discrete-log implies the security of the Diffie-Hellman protocol. This provable security of the Diffie-Hellman protocol over elliptic curves demonstrates an additional advantage of elliptic curve cryptosystems over conventional ones. Finally, we prove that manipulating black box fields over the rationals is as hard as factoring integers.

1 Introduction

An algebraic structure is often defined as a set of operators acting on some universe. Usually there is no reference as to how the elements in the universe are represented. One can design algorithms for such an abstract algebraic structure by providing the algorithm with oracles for the various operators. We refer to such a representation of an algebraic structure as a *black box representation*. The most widely studied structure given in this fashion is the black box group [3].

In this paper we study *fields* given in a black box representation. We refer to such fields as black box fields, or BBF for short. The definition of black box fields will be given in Section 2. For now we give a high level description. Let K be a BBF. Intuitively speaking, the elements of K are represented as arbitrary binary strings. For an element $x \in K$ we denote by $[x]$ the binary string representing the element x. We refer to $[x]$ as the *black box representation* of x. A black box field algorithm has access to oracles that given $[x]$ and $[y]$ compute the black box representation of the sum $[x+y]$ and product $[xy]$. Similarly, there is an oracle that given $[x], [y]$ will output "true" if and only if $x = y$. Finally, the algorithm is provided with an oracle that given $x \in K$ will output $[x]$.

We will be most interested in the following problem: let p be a prime and $K = \mathbb{F}_p$ be a finite field given as a BBF. Find an algorithm that takes as

* Supported in part by NSF CCR-9304718.

N. Koblitz (Ed.): Advances in Cryptology - CRYPTO '96, LNCS 1109, pp. 283-297, 1996.

input the black box representation of a field element $[\alpha]$ and outputs an integer $0 \leq a < p$ such that $a \equiv \alpha \pmod{p}$. We refer to this problem as the black box field problem, or BBFP for short. A trivial algorithm for this problem is to test all elements of $\mathbb{F}_p$ one by one by using the equality test oracle. The running time of this algorithm is $O(p)$. We are interested in finding algorithms for BBFP whose running time is substantially less than p.

Our main results will be randomized algorithms for BBFP whose expected running time is sub-exponential in $\log p$. Our algorithms are based on a technique due to Maurer [18]. The existence of sub-exponential algorithms for BBFP is surprising when contrasted with a result of Nechaev [25] and Shoup [27]. They considered the equivalent problem to BBFP over groups, i.e. where elements can be added, but not multiplied. They show that for the group $G = \mathbb{Z}/p\mathbb{Z}$ the best algorithm for finding a hidden element must take time $\Omega(\sqrt{|G|})$.

Sub-exponential algorithms for BBFP have several consequences to cryptography. The first application shows that any *algebraically homomorphic* cryptosystem can be broken in sub-exponential time. Such cryptosystems are desirable since they enable non-interactive two-player secure function evaluation [1]. These concepts will be defined in Section 3.1. One may view this result as a general cryptanalytic tool: to show that a cryptosystem can be broken in sub-exponential time it suffices to show that it is algebraically homomorphic.

An important motivation for studying the black box field problem is that algorithms for BBFP can be used to prove the security of the Diffie-Hellman secret key exchange protocol [11]. Proving the equivalence of breaking the Diffie-Hellman protocol and computing discrete-log is one of the oldest problems in public key cryptography. In Section 3.2 we show that the sub-exponential algorithm for BBFP has the following consequence: let G be a group in which the discrete log problem can not be solved in sub-exponential time. Then in the group G the Diffie-Hellman protocol can not be broken in sub-exponential time. The group generated by the points of an elliptic curve over a finite field is an example of a group for which there is no known[2] sub-exponential algorithm for computing discrete log (this is the main motivation for using elliptic curve cryptosystems [14, 23]). Hence, our results show that if computing discrete log in the group of points of an elliptic curve is hard then the Diffie-Hellman protocol in such groups is secure. The existence of such a reduction demonstrates another advantage of elliptic curve cryptosystems.

Finally in Section 6 we consider an equivalent of BBFP over the rationals. We show that solving BBFP over the rationals is as hard as factoring integers. This negative result suggests that performing computations over rational black box fields is much harder than over finite black box fields.

[2] A sub-exponential algorithm is known for the rare event when the curve is supersingular [22].

2 Black-box fields

A black box field is an abstract algebraic construct motivated by cryptographic applications. We begin by giving a precise definition of black box fields (BBF) and the black box field problem (BBFP).

Definition 1. A *black box field* is a six-tuple: (p, n, h, F, G, T) where p is a prime and n is a positive integer representing the encoding length. The functions h, F, G, T are defined as follows:

1. The function $h : \{0,1\}^n \to \mathbb{F}_p$ associates a field element with every n-bit binary string. The function h is surjective, i.e. every field element is represented by at least one binary string.
2. The functions $F, G : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ perform addition and multiplication. They satisfy the following relations: $h(F(x,y)) = h(x) + h(y)$ and $h(G(x,y)) = h(x)h(y)$.
3. The function $T : \{0,1\}^n \times \{0,1\}^n \to \{\texttt{true},\texttt{false}\}$ tests equality of two black box elements: $T(x,y) = \texttt{true}$ if and only if $h(x) = h(y)$.

Notice that an element $x \in \mathbb{F}_p$ can be represented by many different n-bit binary strings since $h^{-1}(x)$ is a set of arbitrary cardinality. Throughout the paper we will use $[x]$ to denote some binary string representing the field element x, i.e. $h([x]) = x$. The functions F and G compute $[x+y]$ and $[xy]$ given $[x]$ and $[y]$. This is consistent with the notation used in the introduction. We usually refrain from mentioning the functions h, F, G, T explicitly. When we say that $\mathbb{F}_p$ is given as a black box field we assume that these functions have already been agreed upon. As an abuse of notation we occasionally write $[x] \in \mathbb{F}_p$ which is to be understood as saying that $[x]$ is a black box representation of the field element $x \in \mathbb{F}_p$.

In Section 3 we give examples where black box fields arise naturally. To familiarize the reader with the concept of a black box field we present a few simple algorithms for such fields. The first is computing the inverse: given $[x] \in \mathbb{F}_p$ we wish to compute $[x^{-1}]$. This can easily be done by observing that $[x^{-1}] = [x^{p-2}]$. Using repeated squaring this requires $O(\log p)$ applications of the multiplication function. Another example is that of computing $[\sqrt{x}]$ given $[x]$ if it exists. An algorithm due to Shanks [6, pp. 32–33] can find square roots in finite fields using only operations which are supported in black box fields. Hence, the algorithm can be applied to $[x]$ and it will output $[\sqrt{x}]$.

Definition 2. Let (p, n, h, F, G, T) be a black box field for some prime p. We denote the map sending x to some $[x]$ by $[]$. The *black box field problem* is the following: find an algorithm A that given p and oracles for $F, G, T, []$ and an element $[\alpha] \in \mathbb{F}_p$ finds α explicitly. Formally, $A^{F,G,T,[]}([\alpha]) = a$ where $a \equiv \alpha \pmod{p}$. The algorithm is said to run in polynomial time if it runs in time $\log^{O(1)} p$. The algorithm is sub-exponential if it runs in sub-exponential time in $\log p$.

The main goal of this paper is to provide algorithms for solving the black box field problem. We first show that it is possible to obtain a small number of bits that uniquely define elements of the finite field $\mathbb{F}_p$.

Conjecture 3. *Let p be a prime and set $k = \lceil 2\log^2 p \rceil$. For an element $x \in \mathbb{F}_p$ define the signature of x as the vector:*

$$sig(x) = \left(\left(\frac{x}{p}\right), \left(\frac{x+1}{p}\right), \ldots, \left(\frac{x+k}{p}\right) \right) \ .$$

where $\left(\frac{x}{p}\right)$ is the Legendre symbol of x over p. Then we conjecture that for sufficiently large p, any two distinct elements x, y in $\mathbb{F}_p$ satisfy $sig(x) \neq sig(y)$.

Problems similar to Conjecture 3 were studied by Davenport [8]. The identity $\left(\frac{x}{p}\right) \equiv x^{\frac{p-1}{2}} \pmod{p}$ shows that given $[x]$ one can compute $\mathrm{sig}(x)$ in a black box field using $O(\log^3 p)$ applications of the oracles. Assuming the conjecture is true, $\mathrm{sig}(x)$ provides enough information to recover x. Unfortunately, there is no known polynomial time algorithm for finding x given $\mathrm{sig}(x)$. In fact, Damgård [7] suggested using this sequence as a pseudo-random sequence.

The argument above shows that one can not hope to obtain an information theoretic lower bound on the number of oracle calls needed to solve BBFP. Assuming conjecture 3 holds, a polynomial number of oracle calls are sufficient to completely constrain the hidden element $[x]$. We note that for the equivalent problem in the black box group $\mathbb{Z}/p\mathbb{Z}$ there is an information theoretic lower bound showing that $\Omega(\sqrt{p})$ oracle calls are needed to obtain enough information about a hidden element (see [27]).

3 Applications

The black box field problem arises naturally in cryptography. In this section we discuss two applications of algorithms for BBFP to cryptography.

3.1 Algebraically homomorphic encryption schemes

Informally a cryptosystem is algebraically homomorphic if given the encryption of two plain-texts x, y one can construct the encryption of the plain-texts $x + y$ and xy in polynomial time. This is captured in the following definition.

Definition 4. Let d, e, k be positive integers denoting the plain-text length, cipher-text length and key-length respectively. Let $\mathbb{Z}_n$ be the ring of integers modulo n where $\lfloor \log_2 n \rfloor = d$. The ring $\mathbb{Z}_n$ constitutes the set of plain-texts. Let (E, D) be an encryption scheme, i.e.

$$D : \mathbb{Z}_n \times \{0,1\}^k \to \{0,1\}^e \quad \text{and} \quad E : \{0,1\}^e \times \{0,1\}^k \to \mathbb{Z}_n$$

where E, D are deterministic polynomial time computable functions. Furthermore, $D(E(x, K_e), K_d) = x$ for some key pair (K_e, K_d) generated by a probabilistic polynomial time key generation algorithm. The encryption scheme is said to be *algebraically homomorphic* if there exist two probabilistic expected polynomial time algorithms $A, M : \{0,1\}^e \times \{0,1\}^e \to \{0,1\}^e$ such that for all $x, y \in \mathbb{Z}_n$ and encryption keys K_e:

$$A(E(x, K_e), E(y, K_e)) = E(x+y, K_e) \quad \text{and} \quad M(E(x, K_e), E(y, K_e)) = E(xy, K_e)$$

As an example we note that for the RSA cryptosystem, given the encryption of two plain-texts x and y one can easily construct the encryption of xy by simply multiplying the two given cipher-texts. RSA is not known to be algebraically homomorphic even though it supports one of the required operations.

Algebraically homomorphic encryption schemes have several applications which make them desirable. Most importantly, they enable two players two perform non-interactive secure function evaluation. See [1] for the appropriate definitions. The existence of such functions was posed as an open question in [12, pp. 6–7]. Unfortunately our sub-exponential algorithm for BBFP shows that any algebraically homomorphic encryption scheme can be broken in sub-exponential time.

Theorem 5. *Suppose that BBFP in a finite field of size p can be solved in time $T_{BBF}(p)$. Then any algebraically homomorphic encryption scheme (D, E) over a plain-text ring of size n can be broken in expected time*

$$O(T_{BBF}(n) + \exp((1 + o(1))\sqrt[3]{\log n \log^2 \log n}))$$

Proof. To simplify the exposition we assume that n is square free. This restriction can be easily lifted using methods of Pohlig and Hellman [24]. Since one can factor integers in expected $\exp((1 + o(1))\sqrt[3]{\log n \log^2 \log n})$ time (see [16]) it is possible to factor the plain-text ring into a direct product of finite fields: $\mathbb{Z}_n = \prod_{i=1}^{s} \mathbb{F}_{p_i}$ where the p_i are distinct primes.

Let K_e, K_d be some encryption/decryption key pair. Given $E(x, K_e)$ we wish to find x in the required time bound. For each p_i we define the black box field (p_i, e, h, A, M, T) as follows: $h(x) = D(x, K_d) \pmod{p_i}$. Notice that for $a \in \mathbb{F}_{p_i}$ the string $[a]$ can be any string w satisfying $w = E(a', K_e)$ where $a \equiv a' \pmod{p_i}$. The addition and multiplication functions A, M are simply the corresponding functions used in Definition 4. Hence A, M can be computed in expected polynomial time. To implement the equality testing oracle T observe that $a' \equiv b' \pmod{p_i}$ if and only if $(a' - b') * n/p_i \equiv 0 \pmod{n}$. Thus, for $a, b \in \mathbb{F}_{p_i}$ given $[a] = E(a', K_e)$ and $[b] = E(b', K_e)$ testing if $a = b$ is done by testing if $E((a'-b')*n/p_i, K_e) = E(0, K_e)$. The string $E((a'-b')*n/p_i, K_e)$ can be computed from $[a], [b]$ in expected polynomial time. We have thus shown that (p_i, e, h, A, M, T) is a black box field and the functions A, M, T can be computed in probabilistic polynomial time.

Given $E(x, K_e)$ we use the algorithm for BBFP in each of the black box fields $\mathbb{F}_{p_i}$ to recover $x \mod p_i$. Using chinese remaindering we can now recover x. The total expected running time is $\sum_{i=0}^{s} O(T_{BBF}(p_i)) < O(T_{BBF}(n))$. □

Theorem 5 can be generalized to work for more general finite commutative rings. We do not pursue these generalizations here. An immediate corollary of Theorem 5 is that the sub-exponential algorithm for BBFP (Theorem 8) enables one to break any algebraically homomorphic encryption scheme in sub-exponential time. This fact may also be viewed as a general tool for cryptanalysis. A cryptanalyst which is faced with a new cryptosystem might try to prove that it is algebraically homomorphic. If he succeeds then our techniques immediately give a method for breaking the system[3].

3.2 The Diffie-Hellman protocol

The Diffie-Hellman secret key exchange protocol [11] is one of the oldest public key protocols. The protocol enables two parties to perform a secret key exchange. We briefly describe the protocol using an arbitrary finite cyclic groups G. Say Alice and Bob wish to perform a secret key exchange. They agree ahead of time on some generator $g \in G$ of the group G. The generator g is made public. Both Alice and Bob secretly pick random integers $0 < a, b < |G|$. Alice sends to Bob the value g^a and Bob sends to Alice the value g^b. Both Alice and Bob can now compute the value $g^{ab} = (g^a)^b = (g^b)^a$ which is used as their secret key.

A passive eavesdropper, Eve, who listens in on the conversation hears g^a and g^b. To discover the secret key, Eve has to compute g^{ab}. Thus, we define the Diffie-Hellman function as:

$$\mathrm{DH}_g(g^a, g^b) = g^{ab} \quad .$$

Note that the cyclic group G is implicit in this notation. In their original paper, Diffie and Hellman claimed that for $G = \mathbb{Z}_p^*$ computing the function $\mathrm{DH}_g(x, y)$ is hard. Many other types of groups have been suggested by various authors. Examples include the multiplicative group of residues modulo a composite number [20, 19], elliptic curves over finite fields [14, 23], the Jacobian of a hyperelliptic curve over a finite field [13] and the class group of imaginary quadratic fields [4]. In all these groups the function $\mathrm{DH}_g(x, y)$ is believed to be hard to compute.

A long standing open problem in cryptography is the question of whether computing $\mathrm{DH}_g(x, y)$ is as hard as computing discrete-log for the group G. The discrete log-function is defined as $\mathrm{Dlog}_g(g^a) = a$ where a is an integer in the range 0 to $|G| - 1$. It is not difficult to see that an oracle for $\mathrm{Dlog}_g(z)$ enables one to compute $\mathrm{DH}_g(x, y)$ in polynomial time. The hard question is whether the converse holds: given an oracle for computing $\mathrm{DH}_g(x, y)$ can one compute $\mathrm{Dlog}_g(z)$ in polynomial time (in $\log |G|$)? Surprisingly, there have been very few results on this problem [10, 18, 30]. Recently Maurer [18] obtained a beautiful result showing that given a polynomial number of advice bits which depend only on $|G|$ one can compute $\mathrm{Dlog}_g(x)$ in polynomial time given an oracle for $\mathrm{DH}_g(x, y)$. Unfortunately, computing these advice bits takes exponential time.

[3] This does not hold for probabilistic encryption schemes. The reason is that given two cipher-texts one can not test if they are the encryption of the same plain-text. As a result, the equality testing oracle can not be implemented.

We show how algorithms for BBFP can be used to reduce $\mathrm{Dlog}_g(z)$ to $\mathrm{DH}_g(x,y)$. First we state an observation due to Pohlig and Hellman [24] which shows that the ability to compute discrete log in groups of prime order is sufficient for computing discrete log in arbitrary groups.

Lemma 6. *Let G be a finite cyclic group for which the factorization of $|G|$ is known. Suppose that for any element $g \in G$ of prime order p one can compute $Dlog_g(x)$ in time $T(p)$. Then one can compute $Dlog_h(x)$ for any element $h \in G$ in time $T(|G|)\log^{O(1)}|G|$.*

The main connection between BBFP and the security of the Diffie-Hellman protocol is explained in the following theorem. The theorem shows that if BBFP can be solved quickly then an algorithm for breaking Diffie-Hellman will give an algorithm for computing discrete-log.

Theorem 7. *Suppose BBFP over the field $\mathbb{F}_p$ can be solved in time $T_{BBF}(p)$. Let G be some finite cyclic group for which the factorization of $|G|$ is known. Suppose that $\mathrm{DH}_g(x,y)$ can be evaluated in time $T_{DH}(|G|)$ for any $g \in G$ of prime order. Then for any $g \in G$ the function $Dlog_g(x)$ can be computed in time*

$$T_{BBF}(|G|) \cdot T_{DH}(|G|) \cdot \log^{O(1)}|G|$$

Proof. Let g be a generator of G and $x = g^a$. We wish to compute $\mathrm{Dlog}_g(x)$. First we show that when G has prime order p the theorem is immediate. The ability to evaluate $\mathrm{DH}_g(x,y)$ transforms G into the black box field $\mathbb{F}_p$. An element $b \in \mathbb{F}_p$ is represented as $[b] \doteq g^b$. Two elements can be added $[b+c] = g^b g^c$ and multiplied $[bc] = \mathrm{DH}_g(g^b, g^c)$. Since $[a] = x$ the algorithm for BBFP will output a on input x. The algorithm for solving BBFP runs in time $T_{BBF}(|G|)$ and therefore makes at most $T_{BBF}(|G|)$ calls to the multiplication oracle. Each call requires a calculation of the Diffie-Hellman function which takes time $T_{DH}(|G|)$. The total running time is $O(T_{BBF}(|G|) \cdot T_{DH}(|G|))$.

When $|G|$ is not prime we make use of Lemma 6. Let p be a prime dividing $|G|$ and let $h = g^{|G|/p}$. The group generated by h has prime order. By assumption, $\mathrm{DH}_h(x,y)$ can be evaluated in time $T_{DH}(p)$. Hence, by the previous paragraph, $\mathrm{Dlog}_h(y)$ can be computed in time $T_{BBF}(p) \cdot T_{DH}(p)$. Lemma 6 can now be applied to show that $\mathrm{Dlog}_g(x)$ can be computed in time $T_{BBF}(|G|) \cdot T_{DH}(|G|) \cdot \log^{O(1)}|G|$ for any $g \in G$. □

We note that when $|G|$ is square free the theorem can be strengthened to obtain a more direct reduction from computing discrete-log to breaking Diffie-Hellman. Namely, for a fixed g an algorithm for $\mathrm{DH}_g(x,y)$ can be converted to an algorithm for $Dlog_g(x)$.

Theorem 7 shows that algorithms for BBFP can be used to reduce computing discrete-log to breaking Diffie-Hellman. We present two algorithms for BBFP in the next section. In Section 5 we show several groups for which these algorithms imply the security of the Diffie-Hellman protocol. These results are summarized in Theorem 13.

4 Algorithms for black-box fields

In this section we describe two sub-exponential algorithms for BBFP. The results depend on an assumption which is often used in computational number theory:

Smoothness assumption

Let $L_\alpha(p)$ be the function $L_\alpha(p) = \exp\left(\log^\alpha p \, \log\log^{1-\alpha} p\right)$. For an integer x let $d(x)$ be the largest prime divisor of x. If $d(x) < \alpha$ we say that x is α-smooth. The assumption we need is that integers chosen uniformly in the range $[p+1-2\sqrt{p}, p+1+2\sqrt{p}]$ satisfy

$$\Pr_x[d(x) < L_\alpha(p)] > \frac{1}{L_{1-\alpha}(p)^{1-\alpha+o(1)}}$$

for any fixed $\alpha > 0$. When the integer x is chosen in the range $[1, p]$ this assumption is known to be true (see [5, 9]). The assumption when $\alpha = \frac{1}{2}$ is necessary for the running time analysis of the elliptic curve factoring method of Lenstra [15].

4.1 A sub-exponential algorithm for BBFP

Theorem 8. *Let K be a finite field of size p given as a black box field. Under the smoothness assumption, BBFP can be solved using $O(\log p)$ space and expected time*

$$L_{\frac{1}{2}}(p)^{2+o(1)} = \exp\left((2+o(1))\sqrt{\log p \, \log\log p}\right) .$$

The proof of Theorem 8 uses a technique similar to the one used by Maurer [18]. Before proving Theorem 8 we state some simple facts regarding elliptic curves. We denote by $E_{a,b}(p)$ the set of points $(x, y) \in \mathbb{F}_p^2$ on the curve $y^2 = x^3 + ax + b$ plus a point O called the point at infinity. It is well known [28] that there is a natural Abelian group structure defined on the points in $E_{a,b}(p)$. Given two points $P_1, P_2 \in E_{a,b}(p)$ we denote their sum in this group by $P_1 + P_2$.

Fact 9. *Let $E_{a,b}$ be an elliptic curve over the black box field $\mathbb{F}_p$. Let $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ be two points on $E_{a,b}$. Let $P_3 = P_1 + P_2 = (x_3, y_3)$. Then given $[x_1], [y_1], [x_2], [y_2]$ it is possible to compute $[x_3], [y_3]$ in polynomial time in $\log p$.*

Proof. The values x_3, y_3 are algebraic expressions in x_1, y_1, x_2, y_2, a, b. See [28, pp. 58–59] for a list of these expressions. Thus, $[x_3], [y_3]$ can be computed from $[x_1], [y_1], [x_2], [y_2]$ using the addition and multiplication oracles. □

Throughout the section we use $[P]$ to denote the point $P = (x, y) \in E_{a,b}(p)$ whose coordinates are given as black box elements, i.e. $[P] = ([x], [y])$. Fact 9 shows that given points $[P_1] = ([x_1], [y_1])$ and $[P_2] = ([x_2], [y_2])$ it is possible to construct $[P_1 + P_2] = ([x_3], [y_3])$ in polynomial time. Similarly, using repeated doubling one can construct the point $[kP]$ given a point $[P]$ and an integer k.

Fact 10. *Let $E_{a,b}$ be an elliptic curve over $\mathbb{F}_p$. Then the group $E_{a,b}$ can be generated by two points. Furthermore, two random points P_1, P_2 generate the group $E_{a,b}$ with probability at least $\Omega(1/\log^2 p)$.*

Proof. The structure of the group $E_{a,b}$ is known to be $E_{a,b} \simeq \mathbb{Z}_n \times \mathbb{Z}_m$ for some m, n where $m|n$ (see [28]). Thus $E_{a,b}$ can be generated by two points. The number of pairs which generate $\mathbb{Z}_n \times \mathbb{Z}_m$ is lower bounded by $\Omega(\varphi(nm)^2)$. The result now follows since always $\varphi(x)/x > \Omega(1/\log x)$. □

Proof of Theorem 8. The basic idea is to generate random elliptic curves $E_{a,b}$ over $\mathbb{F}_p$ until a curve with a smooth order is found. It is well known that the number of points on a random elliptic curve over $\mathbb{F}_p$ is approximately uniformly distributed in the range $[p + 1 - 2\sqrt{p},\ p + 1 + 2\sqrt{p}]$ (see [15]). Hence, by the smoothness assumption, after an expected $\exp\left((\frac{1}{2} + o(1))\sqrt{\log p \log\log p}\right)$ tries we will have generated a curve $E_{a,b}$ such that the largest prime divisor of $|E_{a,b}|$ is less than $\exp\left(\sqrt{\log p \log\log p}\right)$.

Let $E_{a,b}$ be an elliptic curve over $\mathbb{F}_p$ for which the largest prime divisor of $|E_{a,b}|$ is less than d for some $d > 0$. The number of points on the curve $E_{a,b}$ can be found in polynomial time using Schoof's algorithm [26]. We show that using this curve it is possible to solve the black box field problem over $\mathbb{F}_p$ in time $O(d^2 \log p)$. This will prove the theorem. Note that since the prime factors of $|E_{a,b}|$ are small, they can be found in the required time bound.

By Fact 10 we can find a pair of points P_1, P_2 which generate the group $E_{a,b}$ by picking them at random. If it so happens that P_1, P_2 do not generate $E_{a,b}$ then the algorithm will fail and we will know that P_1, P_2 were poorly chosen. Fact 10 shows that in an expected polynomial number of attempts the pair P_1, P_2 will work.

Let $P_1 = (x_1, y_1), P_2 = (x_2, y_2)$ be two points which generate $E_{a,b}$. Given a black box representation $[x]$ of some field element $x \in \mathbb{F}_p$ we construct x explicitly, i.e. find an integer w such that $w \equiv x \pmod{p}$. This is done in three steps:

1. Embed $[x]$ in the curve $E_{a,b}$, i.e. find $[y] \in \mathbb{F}_p$ so that the point $[P] = ([x], [y])$ is on $E_{a,b}$.
2. Find integers α and β such that $[P] = [\alpha P_1 + \beta P_2]$. Such integers exist since P_1, P_2 generate $E_{a,b}$.
3. Calculate $\alpha P_1 + \beta P_2$ explicitly. The x coordinate of the resulting point, which is an explicit field element, is the required value.

We now explain how to carry out steps 1 and 2. In step 1 we are looking for $[y] \in \mathbb{F}_p$ such that $y^2 = x^3 + ax + b$. Given $[x]$ one can clearly construct $[x^3 + ax + b]$. The required $[y]$ can be found by taking the square root of the element $[x^3 + ax + b]$. This is possible since in Section 2 we saw that given $[z] \in \mathbb{F}_p$ there is an algorithm for constructing $[\sqrt{z}]$ in random polynomial time. If the above element does not have a square root in $\mathbb{F}_p$ we run through the entire computation using $[\hat{x}] = [x + r]$ for some randomly chosen r. After a constant number of tries it will be possible to embed $[\hat{x}]$ in the curve.

Next we explain how to find α and β as required in step 2. Recall that $E_{a,b} \simeq \mathbb{Z}_n \times \mathbb{Z}_m$ where $n = q_1^{\gamma_1} \cdots q_r^{\gamma_r}$, $m|n$ and all the primes q_i are less than d. To find α, β we use a simple generalization of the Pohlig-Hellman algorithm [24] for discrete log in groups of smooth order. For simplicity we assume that all the

γ_i are equal to 1. The Pohlig-Hellman algorithm generalizes to the case where $\gamma_i > 1$ as well. The method for finding α, β is to construct $\alpha, \beta \pmod{q_i}$ for all $i = 1, \ldots, r$ and then use chinese remaindering to find α, β.

Let q_i be a prime dividing n and let $k = n/q_i$. Notice that the points kP_1, kP_2 have order at most q_i. Therefore, since the point kP is in the group generated by kP_1, kP_2 there must exist integers $0 \leq \alpha_i, \beta_i < q_i$ such that

$$[kP] = [\alpha_i kP_1 + \beta_i kP_2]$$

These integers can be found in time $O(q_i^2)$ by trying all possible pairs α_i, β_i in the range $[0, q_i]$.

As was stated above, the $(\alpha_i, \beta_i) \pmod{q_i}$ can be combined using chinese remaindering to obtain integers α, β such that $P = \alpha P_1 + \beta P_2$. Since $|E_{a,b}|$ has at most $O(\log p)$ prime factors and they are all less than d the running time of this procedure is at most $O(d^2 \log p)$. □

4.2 A two step algorithm

Before we present the results of this section we have to explain the notion of an algebraic algorithm. Intuitively, a function $f : \mathbb{F}^n \to \mathbb{F}$ can be computed by an algebraic algorithm if it can be computed efficiently even when the input is given as black box field elements. This means that the function can be computed by only applying elementary arithmetic operations and equality tests to its input.

Definition 11. Let $\mathbb{F}_p$ be a finite field and $f : \mathbb{F}_p^n \to \mathbb{F}_p$ some function on n inputs. We say that f can be computed in time $T(p)$ by an *algebraic algorithm* if there exists a Turing machine M satisfying the following property: for any black box representation (p, n, h, F, G, T) of the field $\mathbb{F}_p$ the machine M given access to the oracles F, G, T and $[]$ will run in time at most $T(p)$ and will output $[f(x_1, \ldots, x_n)]$ on input $[x_1], \ldots, [x_n]$.

in Section 2 we noted that the functions $f(x) = x^{-1} \pmod{p}$ and $f(x) = \sqrt{x} \pmod{p}$ can be computed by random polynomial time *algebraic* algorithms.

We improve the algorithm of the previous section by improving the brute force search step. The improvement relies on a certain additional assumption. Namely, we assume that the Diffie-Hellman function over the group of points of an elliptic curve can be computed by an $L_{1/3}(p)$ *algebraic* algorithm. Though this assumption is likely to be false (there are no sub-exponential algebraic algorithms for computing discrete log) the theorem is quite useful. It provides a more efficient reduction from computing discrete-log to breaking Diffie-Hellman.

Theorem 12. *Suppose* $\mathrm{DH}_g(P_1, P_2)$ *can be computed in time* $L_{1/3}(p)$ *by an* algebraic *algorithm where* $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$ *and* g *are points on an elliptic curve* $E_{a,b}(p)$. *Then under the smoothness assumption, BBFP in a finite field of size* p *can be solved in expected time*

$$L_{\frac{1}{3}}(p)^{2+o(1)} = \exp\left((2 + o(1))\sqrt[3]{\log p \, \log\log^2 p}\right) .$$

Proof sketch. First generate a curve $E_{a,b}$ over $\mathbb{F}_p$ with n points such that the largest prime divisor of n is $L_{2/3}(p)$. By the smoothness assumption this can be done in expected $L_{1/3}(p)^{1/3+o(1)}$ time. The prime factors of n can be found using the elliptic curve factoring algorithm [15] in time $L_{1/3}(p)^{1+o(1)}$. Let q be a prime divisor of n. The algebraic algorithm for computing $\mathrm{DH}_g(P_1, P_2)$ can be used to transform the subgroup of points on $E_{a,b}$ of order q into a black box field. Therefore the brute force search in the algorithm of Theorem 13 can be done in time $L_{1/2}(q)^2 < L_{1/3}(p)^2$. The entire algorithm takes $L_{1/3}(p)^{2+o(1)}$ time. □

Theorem 12 is a rare example where the elliptic curve method produces an algorithm with a running time of $L_{1/3}(p)$. Such running times are usually associated with the number field sieve [16].

5 Security of the Diffie-Hellman protocol

We use the algorithms for BBFP of Section 4 to derive the security of the Diffie-Hellman protocol over various groups. Theorem 7 shows that if discrete-log can not be computed in time $T_{DL}(|G|)$ in a group G then the Diffie-Hellman protocol can not be broken in time $O(T_{DL}(|G|)/T_{BBF}(|G|))$. Thus, the faster we can solve BBFP the more secure the Diffie-Hellman protocol is.

Since the algorithms for BBFP take sub-exponential time the above approach can only be used in groups for which there is no sub-exponential time algorithm for discrete log. There are several groups which are suspected to have such a property. The groups are all constructed using a prime modulus p.

1. Let $E_{a,b}$ be the group of points on an elliptic curve modulo p. The best algorithm for discrete log in such groups is Shank's baby-step-giant-step [21] algorithm. There is no known sub-exponential algorithm for discrete-log which works in all groups $E_{a,b}$ (such algorithms exist when $|E_{a,b}|$ is a smooth number or when $E_{a,b}$ is supersingular [22]).
2. Let J be the Jacobian of a hyper-elliptic curve of genus g. When the genus g is fixed, there is no known sub-exponential algorithm for discrete log which works for all J. Adleman, DeMarrais and Huang [2] show that when the genus is at least $\log p$ the index calculus method can be adapted to give an $L_{\frac{1}{2}}(p^{2g+1})^{1+o(1)}$ algorithm for discrete log in J.
3. Let g be an element of $\mathbb{Z}_p^*$ which generates a subgroup of order $L_{1/3}(p)$. Let H_g be this subgroup. Discrete log in H_g is not known to be solvable in sub-exponential time in $|H_g|$. The digital signature standard [31] relies on the exponential time hardness of discrete log in H_g for its security.
4. Finally we mention that all sub-exponential algorithms for group $\mathbb{Z}_p^*$ rely on the index calculus method and therefore require sub-exponential space. There is no known sup-exponential time and polynomial space algorithm for computing discrete log in $\mathbb{Z}_p^*$.

The algorithms for BBFP can be used to derive hardness results for breaking the Diffie-Hellman protocol for the groups mentioned above. These results are summarized in the following theorem.

Theorem 13. *Let p be a prime. Under the smoothness assumption we obtain the following results:*

1. *If discrete log can not be computed in $L_{1/2}(p)^{2+o(1)}$ time in the groups $E_{a,b}, J, H_g$ then the Diffie-Hellman protocol can not be broken in time $L_{\frac{1}{2}}(p)$ in these groups.*
2. *If discrete log can not be computed in $L_{1/3}(p)^{2+o(1)}$ time in the groups $E_{a,b}, J, H_g$ then the Diffie-Hellman protocol can not be broken by an* algebraic *algorithm in time $L_{1/3}(p)$ in these groups.*

Proof. First, we may assume that the factorization of $|G|$ is known. This factorization can be found using the number field sieve algorithm whose conjectured running time is $L_{1/3}(p)^{1+o(1)}$ [16]. Part (1) follows by combining Theorem 7 and Theorem 8. Part (2) follows by combining Theorem 7 and Theorem 12. □

Note that part (2) of the theorem relies on a weaker assumption than part (1). The conclusion is also weaker since we can only prove that $\mathrm{DH}_g(x, y)$ can not be computed by an *algebraic* algorithm.

6 Black-box fields in characteristic 0

In this section we consider the black box field problem over the field of rational numbers $\mathbb{Q}$. We are provided with a black box representation of the rationals which supports the axioms of Definition 1. The black box field problem over $\mathbb{Q}$ can be stated as follows: given a black box representation of an integer $[x]$ between 1 and N find the integer x. The algorithm is said to be polynomial if it runs in time polynomial in $\log N$. We prove the following negative result:

Theorem 14. *BBFP over $\mathbb{Q}$ can not be solved in polynomial time, unless factoring integers is easy.*

By "factoring integers is easy" we mean that a non-negligible fraction of the n-bit RSA composites can be factored in polynomial time. We refer to [17] for the precise definition. To prove Theorem 14 we first recall the notion of the straight line complexity of a polynomial. Let $f(x) \in \mathbb{Q}[x]$ be some polynomial. A straight line computation of f is a sequence of polynomials $g_1, \ldots, g_m$ such that $g_m = f$ and each g_i is one of the following: (i) g_i is the constant 1 or the variable x; (ii) $g_i = g_j \circ g_k$ where $\circ$ is one of $+ - *$ and $j, k < i$. The length of the computation is m. The *straight line complexity* of $f(x)$, denoted by $L(f)$, is the length of the shortest such computation. The following theorem can be easily derived from a result due to Lipton [17].

Theorem 15. *Let $\{f_k(x)\}$ be a sequence of polynomials over $\mathbb{Q}[x]$ such that for any k, the polynomial $f_k(x)$ has at least $2^k/k^e$ integer roots for some integer $e > 0$. Then for any $d > 0$ and sufficiently large k we have $L(f_k) > k^d$, unless factoring is easy.*

Proof of Theorem 14. In a black box field the only operations allowed are additions, multiplications and comparisons. Thus, a computation in a black box field can be viewed as an algebraic decision tree [29]. That is, given an input x_0 the computation proceeds as follows: at every internal node v in the tree some polynomial $f_v(x_0)$ is evaluated. If $f_v(x_0) = 0$ the computation branches to the left child of v otherwise the computation branches to the right child. The leaves of the tree are labeled with the output.

Assume towards a contradiction that for any N there is an algorithm which solves BBFP over $\mathbb{Q}$ in time $\log^{O(1)} N$. That is, given the black box representation of an integer $0 < x \leq N$ the algorithm will find x. The algorithm defines an algebraic decision tree with N leaves. Since the algorithm is polynomial, the depth of the tree is at most $\log^d N$ for some $d > 0$. Furthermore, the straight line complexity of every polynomial f_v in the tree must satisfy $L(f_v) < \log^e N$ for some $e > 0$.

Consider the path $v_0, \ldots, v_m$ from the root v_0 to a leaf v_m in which every v_i is the right child of v_{i-1}. For $i = 0, \ldots, m$ let Z_{v_i} be the set of integer roots of f_{v_i} in the range $[1, N]$. An input x will not reach the leaf v_m if it is contained in some Z_{v_i}. Since only one integer in the range $[1, N]$ is allowed to reach the leaf v_m we know that $|\bigcup_{i=1}^{m} Z_{v_i}| = N - 1$. Hence, there must exist an i such that $|Z_{v_i}| > (N-1)/m$. Since $m < \log^d N$ we conclude that the polynomial f_{v_i} has at least $N/\log^d N$ integer roots.

Applying this argument for $N = 2^k$ where $k = 1, 2, \ldots$ we obtain a sequence of polynomials f_k where each f_k has at least $2^k/k^d$ integer roots and $L(f_k) < k^e$. This sequence contradicts the statement of Theorem 15. □

7 Conclusions and open problems

We defined a new problem which we call the black box field problem or BBFP for short. We demonstrated several applications of BBFP to cryptography: (1) efficient algorithms for BBFP provide reductions from computing discrete-logs to breaking the Diffie-Hellman protocol. (2) algorithms for BBFP can be used as a cryptanalytic tool to break algebraically homomorphic cryptosystems. These applications demonstrate the importance of this problem.

We described two sub-exponential algorithms for solving BBFP. The first solves BBFP in time $L_{1/2}(p)$. The second solves BBFP in time $L_{1/3}(p)$ under certain (strong) assumptions. These algorithms were used to show that over elliptic curves the hardness of computing discrete-log implies the security of the Diffie-Hellman protocol. These results demonstrate an advantage of elliptic-curve cryptosystems over conventional ones. In addition we noted that for small sub-groups of $\mathbb{Z}_p^*$, such as the ones used by DSS [31], our results show that the hardness of discrete-log implies the security of the Diffie-Hellman protocol. We have also shown that assuming factoring integers is hard, BBFP can not be solved in polynomial time over the rationals.

The problem of solving BBFP over a finite field in polynomial time is still open. We briefly describe a promising approach. Schoof's algorithm [26] for

counting the number of points on elliptic curves over $\mathbb{F}_p$ can be made to work over black box fields. Given a curve $y^2 = x^3 + [a]x + [b]$ over $\mathbb{F}_p$ the algorithm will output an explicit integer which is the number of $\mathbb{F}_p$-points on the curve. Given a black box element $[j]$ we construct the curve $y^2 = x^3 + [a]x + [b]$ whose j-invariant is $[j]$. The number of points on this curve provides a lot of information about j. Currently it is not known how to reconstruct j from this information. Progress in this direction will be of great interest and is likely to yield a polynomial time algorithm for BBFP.

Acknowledgments

The authors wish to thank Ueli Maurer and Stefan Wolf for helpful comments on this manuscript.

References

1. M. Abadi, J. Feigenbaum, "Secure circuit evaluation: a protocol based on hiding information from an oracle", J. Cryptology, No. 2, 1990, pp. 1–12.
2. L. Adleman, J. DeMarrais, Ming-Deh Huang, "A sub-exponential algorithm for discrete logarithm over the rational subgroup of the Jacobian of large genus hyperelliptic curves over finite fields", Proceedings of ANTS, 1994.
3. L. Babai, E. Szemerédi, "On the complexity of matrix group problems I", Proceedings FOCS 1984, pp. 229–240.
4. J. Buchmann, H. Williams, "A key exchange system based on imaginary quadratic fields", Journal of cryptography, vol. 1, no. 2, pp. 107–118, 1988.
5. E. Canfield, P. Erdös, C. Pomerance, "On a problem of Oppenheim concerning "Factorisatio Numerorum", J. Number Theory 17, 1983, pp. 1–28.
6. H. Cohen, "A course in computational algebraic number theory", Springer-Verlag, 1991.
7. I. Damgård, "On the randomness of Legendre and Jacobi sequences", Proceedings of Crypto 1988, pp. 163–172.
8. H. Davenport, "On the distribution of quadratic residues (mod p)", J. London Math. Soc., 8, 1933, pp. 46–52.
9. N. DeBruijn, "On the number of positive integers $\leq x$ and free of prime factors $> y$", Indag. Math. 38, 1966, pp. 239–247.
10. B. den Boer, "Diffie-Hellman is as strong as discrete log for certain primes", Proceedings of Crypto 1988, pp. 530–539.
11. W. Diffie, M. Hellman, "New directions in cryptography", IEEE Transactions on Information Theory, vol. 22, no. 6, pp. 644–654, 1976.
12. J. Feigenbaum, N. Merrirr, "Open Questions and summary of discussions', Proceedings of DIMACS workshop on Distributed Computing and Cryptography, Vol. 2, 1989.
13. N. Koblitz, "A family of Jacobians suitable for discrete log cryptosystems", Proceedings of Crypto 88, pp. 94–99.
14. N. Koblitz, "Elliptic curve cryptosystems", Math. of comp., Vol. 48, 1987, pp. 203–209.

15. H. Lenstra Jr., "Factoring integers with elliptic curves", Annals of Math. 126, 1987, pp. 649–673.
16. A. Lenstra, H. Lenstra Jr., M Manasse, J. Pollard, "The number field sieve", Proceedings of STOC 1990, pp. 564–572.
17. R. Lipton, "Straight line complexity and integer factorization", First algorithmic number theory symposium, 1994.
18. U. Maurer, "Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms", Proceedings of Crypto 1994, pp. 271–281.
19. U. Maurer, Y. Yacobi, "Non-interactive public-key cryptography", EUROCRYPT 91, Lecture notes in computer science, Springer-Verlag, vol. 547, pp. 498–507, 1991.
20. K. McCurley, "A key distribution system equivalent to factoring", Journal of cryptography, vol. 1, no. 2, 1988, pp. 95–105.
21. K. McCurley, "The discrete logarithm problem", In cryptology and computational number theory, AMS lecture notes, C. Pomerance editor, 1989.
22. A. Menezes, S. Vanstone, "Reducing elliptic curve logarithms to logarithms in a finite field", Proceedings of STOC 1991, pp. 80–89.
23. V. Miller, "Use of elliptic curves in cryptography", Proceedings of Crypto 1985, pp. 417–426.
24. S. Pohlig, M. Hellman, "An improved algorithm for computing discrete logarithms over $GF(p)$ and its cryptographic significance", IEEE Trans. Inform. Theory, Vol. 24, 1978, pp. 106–110.
25. V. Nechaev, "Complexity of a determinate algorithm for the discrete logarithm", Mathematical Notes, Vol. 55, No. 2, pp. 165-172, 1994.
26. R. Schoof, "Elliptic Curves Over Finite Fields and the Computation of Square Roots mod p", Math. of Comp., Vol. 44, no. 170, 1985, pp. 483–494.
27. V. Shoup, "Lower bounds for discrete logarithms and related problems", Manuscript, 1995.
28. J. Silverman, "The arithmetic of elliptic curves", Springer-Verlag, 1986.
29. M. Steele, A. Yao, "Lower bounds for algebraic decision trees", J. of alg., Vol. 3, 1982, pp. 1–8.
30. S. Wolf, "Diffie-Hellman and Discrete Logarithms", Thesis ETH Zurich, 1995.
31. Specifications for the digital signature standard, National Institute for Standards and Technology, Federal Information Processing Standard Publication XX, draft, August 1991.