Vietnam Journal of Mathematics **39**:3(2011) 267-285

Vietnam Journal of MATHEMATICS

Discrete Logarithms, Diffie-Hellman, and Reductions

Neal Koblitz[1], **Alfred Menezes**[2], **and Igor E. Shparlinski**[3⋆]

[1]*Department of Mathematics, Box 354350, University of Washington, Seattle, WA 98195 U.S.A.*

[2]*Department of Combinatorics & Optimization, University of Waterloo, Waterloo, Ontario N2L 3G1 Canada*

[3]*Department of Computing, Macquarie University, Sydney, NSW 2109, Australia*

Dedicated to Professor Hà Huy Khoái on the occasion of his 65th-birthday

Received November 26, 2010
Revised February 10, 2011

Abstract. We consider the One-Prime-Not-p and All-Primes-But-p variants of the Discrete Logarithm (DL) problem in a group of prime order p. We give reductions to the Diffie-Hellman (DH) problem that do not depend on any unproved conjectures about smooth or prime numbers in short intervals. We show that the One-Prime-Not-p-DL problem reduces to DH in time roughly $L_p(1/2)$; the All-Primes-But-p-DL problem reduces to DH in time roughly $L_p(2/5)$; and the All-Primes-But-p-DL problem reduces to the DH plus Integer Factorization problems in polynomial time. We also prove that under the Riemann Hypothesis, with $\varepsilon \log p$ queries to a yes-or-no oracle one can reduce DL to DH in time roughly $L_p(1/2)$; and under a conjecture about smooth numbers, with $\varepsilon \log p$ queries to a yes-or-no oracle one can reduce DL to DH in polynomial time.

1. Introduction

Almost all commercially deployed public key cryptosystems depend for their

⋆ The third author would like to thank the University of Waterloo for its hospitality and support during the time this work was started. His research was also supported in part by ARC Grant DP0881473, Australia and by NRF Grant CRP2-2007-03, Singapore.

security on the presumed intractability of one of the following two mathematical problems: (a) Integer Factorization and (b) Discrete Logarithm. More concretely, in (a) one assumes that, given an "RSA modulus" N that is the product of two large primes p and q, it is infeasible to find p and q; and in (b) one assumes that if $\mathbb{G}$ is a suitably chosen group of prime order p, it is infeasible to invert the function $x \mapsto xP$, where x is an integer mod p and P is a fixed non-identity group element (here we are using additive notation for the group operation).

However, a matter of great concern since the early days of public key cryptography has been the fact that the security implication goes only one way. That is, an adversary who can factor the RSA modulus or compute discrete logarithms has broken the corresponding cryptosystem. However, there are many other ways to compromise security short of solving the Integer Factorization or Discrete Logarithm problem. In RSA encryption [35], for example, in order to recover plaintext from ciphertext the function that must be inverted is $x \mapsto x^e \bmod N$ (where the encryption exponent e, along with the modulus N, forms the public key). Someone who knows the factorization of $N = pq$ can easily do this, and hopefully someone who does not know the factorization cannot. However, the equivalence of factorization and inversion of the RSA function is a longstanding conjecture — and one that some researchers believe might be false (see [6]). In other words, inverting the RSA encryption function might be strictly easier than factoring N.

Similarly, the classic Diffie-Hellman key exchange [10] in $\mathbb{G}$ works as follows. Alice and Bob each generate a random number x and y, respectively, and keep it secret. Alice sends Bob the group element xP, and he sends her yP; the shared key is then xyP. The problem of determining xyP knowing only P, xP, and yP is called the *Diffie-Hellman problem*. An adversary who can compute discrete logarithms can obviously solve the Diffie-Hellman problem, but the converse is an open conjecture. In contrast to the situation with the RSA problem, though, there is considerable evidence that the two problems Discrete Logarithm and Diffie-Hellman are in fact equivalent; see the survey [29].

There can also be successful attacks that do not even attempt to solve the basic problem of inverting the one-way function. For example, here is a simple "chosen-ciphertext" attack on RSA encryption. Suppose that the adversary Cynthia wants to learn the plaintext x from the ciphertext $y = x^e \bmod N$ that Alice received from Bob. She chooses a random integer r and computes $y' = r^e y \bmod N$. She then tricks Alice into deciphering y' for her. (In some practical settings it is not difficult to obtain the plaintext for a chosen ciphertext provided that it appears innocuous to Alice, i.e., $y' \neq y$.) Cynthia immediately computes the secret message $x = x'/r \bmod N$, where x' is the decryption of y' that Alice gave her. In the case of RSA, a common approach to defending against chosen-ciphertext attack is to append a random padding to the message before exponentiating.

Starting in the 1980's, researchers have used the method of reductions in order to reassure users that no attack on the cryptographic protocol could possibly succeed without the adversary solving a presumably intractable underlying mathematical problem. For example, they might prove that the problem of in-

verting the RSA function reduces in polynomial time to the problem of mounting a successful chosen-ciphertext attack on a particular version of RSA that they've developed. This implies that a successful chosen-ciphertext attacker must also be able to invert the RSA function. This approach to providing assurances of resistance to cryptanalysis is called "provable security."

"Provable security" is controversial; see [15, 16, 17]. Despite the popularity of reductionist arguments, there is considerable doubt about the practical value of the assurances that they provide. More generally, no consensus exists on the real-world value of theoretical results. Some practitioners maintain that the only way to determine whether one can have confidence in a new cryptosystem is to challenge the best cryptanalysts to try to break it, and then wait to see what happens.

The present paper uses the method of reductions to try to shed light on the relationship between the Diffie-Hellman problem and various versions of the Discrete Logarithm problem. We do not, however, make any claims about the practical usefulness of our results. In fact, what is more plausible is the opposite claim — namely, that this work has so little connection with practical cryptography that it is "gentle and clean" in the sense of G. H. Hardy's famous characterization of number theory [12]:

> ...both Gauss and lesser mathematicians may be justified in rejoicing that there is one science at any rate, and that their own, whose very remoteness from ordinary human activities should keep it gentle and clean.

2. Statement of results

Let $\mathbb{G}$ be a group of prime order p, written additively, and let P denote a non-identity element of $\mathbb{G}$. The Discrete Logarithm (DL) problem asks, given $P, Q \in \mathbb{G}$, for the integer x mod p such that $xP = Q$. The Diffie-Hellman (DH) problem asks, given $P, Q, R \in \mathbb{G}$, for the element $S \in \mathbb{G}$ such that $z \equiv xy \pmod{p}$, where $Q = xP$, $R = yP$, $S = zP$. The DH problem trivially reduces to the DL problem, and there has been a long history of efforts to prove that DL reduces to DH (see [29]).

The idea of introducing versions of DL where one is given an oracle to solve DL on groups of a different prime order first appeared in [24], where the authors wanted to express the assumption that intractability of the DL problem "holds even in the presence of oracles breaking the [DL problem] for other groups." The following two problems were first stated in [18].

- The All-Primes-But-p Discrete Logarithm problem. You are given a t-bit prime p, a group $\mathbb{G}$ of order p, and two elements $P, Q \in \mathbb{G}$. You are also given an oracle that, acting as a black box, returns the solution to any discrete logarithm problem in any group of order q, where q is any prime of at most t bits other than p. You must find the discrete logarithm of Q to the base P.

- The One-Prime-Not-p Discrete Logarithm problem. You are given a t-bit prime p, a group $\mathbb{G}$ of order p, and two elements $P, Q \in \mathbb{G}$. For a single prime q of your choice that has at most t bits and is not equal to p you are given an oracle that, acting as a black box, returns the solution to any discrete logarithm problem in any group of order q. You must find the discrete logarithm of Q to the base P.

Remark 2.1. In these formulations it is understood that the oracle must be supplied with the group elements and group operation. The latter can be given in the form of either a set of rules or a group-operation oracle.

In [18] the second of these problems was shown to reduce in polynomial time to the Diffie-Hellman problem. However, this result assumed a conjecture about the distribution of primes in short intervals that, although plausible, is out of reach of current techniques of analytic number theory. In this paper we give reductions of these two problems to DH whose analyses require no such assumption; however, the price we have to pay for complete rigor is that we get much weaker results.

For $0 \le \alpha \le 1$ let $L_p(\alpha)$ denote $\exp(c(\log p)^\alpha (\log\log p)^{1-\alpha})$ for some unspecified constant $c > 0$. When we say that some version of the discrete logarithm problem "$L_p(\alpha)$-reduces to DH," we mean that there is a probabilistic algorithm that reduces the DL-type problem to the Diffie-Hellman problem and has rigorously analyzed expected running time bounded by $L_p(\alpha)$ bit operations. Our main goal is to prove the following theorems.

Theorem 2.2. *The One-Prime-Not-p Discrete Logarithm problem $L_p(1/2)$-reduces to the Diffie-Hellman problem.*

Theorem 2.3. *The All-Primes-But-p Discrete Logarithm problem $L_p(2/5)$-reduces to the Diffie-Hellman problem.*

Theorem 2.4. *The All-Primes-But-p Discrete Logarithm problem reduces in polynomial time to the Diffie-Hellman plus Integer Factorization problems.*

Remark 2.5. In most applications, the group $\mathbb{G}$ is chosen so that there is no known subexponential-time algorithm for the DL problem. Even when $\mathbb{G}$ is chosen to be a subgroup of the multiplicative group of a finite field $\mathbb{F}_q$ (as in DSA) or to have an imbedding into the multiplicative group of $\mathbb{F}_{q^k}$ for k small (as in pairing-based protocols), the finite field $\mathbb{F}_q$ or $\mathbb{F}_{q^k}$ is usually chosen to be large enough so that the algorithms that are subexponential in the size of the field have running times that are comparable to those of the squareroot algorithms for the DL in $\mathbb{G}$. It is because the DL problem in the groups $\mathbb{G}$ used in cryptography generally has no known subexponential-time algorithm that $L(\alpha)$-reductions (where $\alpha < 1$) are significant. In contrast, an $L(1/2)$- or $L(2/5)$-reduction would have little significance for the Integer Factorization problem, because integers can be factored in rigorously analyzed time $L(1/2)$ (see [33, 23]) and heuristic time $L(1/3)$ (see [20]).

Finally, we prove two theorems about the oracle complexity of reducing the Discrete Logarithm problem to Diffie-Hellman. Suppose we have an omniscient oracle that gives correct answers to yes-or-no questions — or, equivalently, to queries asking for one bit of data. Informally, the oracle complexity of a problem is the number of queries of such an oracle that are needed in order to solve the problem in polynomial time. Here the problem need only be solved with overwhelming probability, not necessarily 100% of the time. In cryptography the most important oracle complexity result is due to Maurer [27], who proved that factorization of an n-bit integer has oracle complexity εn for any $\varepsilon > 0$. His result is conditional upon the following conjecture about smooth integers in the Hasse interval; we recall that an integer is y-smooth if all of its prime divisors are at most y.

Conjecture 2.6. (Maurer [27]) *For every $0 < \beta < 0.5$ and $c > 1/(0.5 - \beta)$ and for all sufficiently large x, the fraction of $(\log^c x)$-smooth integers in the interval $(x + 1 - \sqrt{x}, x + 1 + \sqrt{x})$ is at least $x^{-1/c-\beta}$.*

Under the same conjecture we prove that the Discrete Logarithm problem in the presence of a Diffie-Hellman oracle also has oracle complexity εn, where n is the bitlength of the group order p. Without using any smoothness conjecture (but under the Riemann Hypothesis) we prove that εn queries to the yes-or-no oracle are enough to reduce the time for DL-to-DH reduction from $L_p(2/3)$ to $L_p(1/2)$ — which, amusingly, also happens to be the fastest known *heuristic* running time for the DL-to-DH reduction *without* the yes-or-no oracle [5]. More precisely, we prove the following theorems:

Theorem 2.7. *Let $n = \lceil \log_2 p \rceil$ be the bitlength of the group order p, and let $\varepsilon > 0$ be arbitrary. Under the Riemann Hypothesis, if one is allowed at most εn queries to an oracle that correctly answers arbitrary yes-or-no questions, then the Discrete Logarithm problem $L_p(1/2)$-reduces to the Diffie-Hellman problem with at most $p^{-\varepsilon/2}$ probability of failure.*

Theorem 2.8. *Let $n = \lceil \log_2 p \rceil$ be the bitlength of the group order p, and let $\varepsilon > 0$ be arbitrary. Under Conjecture 2.6, if one is allowed at most εn queries to an oracle that correctly answers arbitrary yes-or-no questions, then the Discrete Logarithm problem reduces in polynomial time to the Diffie-Hellman problem with at most $p^{-\varepsilon/2}$ probability of failure.*

Our purpose is to contribute to a theoretical understanding of the relations between various types of problems that arise in discrete-logarithm-based cryptography. Since the "holy grail" of a polynomial time reduction from Discrete Logarithm to Diffie-Hellman seems a long way off, it is interesting to investigate what types of additional features might bridge this gap. There are several possible ways to relax the problem: one can ask for only a subexponential-time, rather than polynomial-time reduction; one can give the discrete-logarithm solver access to powerful oracles, such as a one-prime-not-p or any-prime-but-p discrete-logarithm solver, an oracle that answers yes-or-no questions, or an oracle that

factors integers; and one can assume certain number-theoretic conjectures, in other words, one can be satisfied with a heuristic complexity analysis.

The algorithms we present are not designed to be practical; rather, our focus is on rigorously analyzed running times. As Shafi Goldwasser has said:[1]

> There is this tension between having an algorithm that has an analysis and is completely useless in practice, and having an algorithm that has no analysis and is useful in practice.

3. The den Boer – Maurer method

The den Boer – Maurer method (see [9, 26, 28, 29, 40]) for reducing DL to DH uses an auxiliary group defined over $\mathbb{F}_p$, where p is the order of the group $\mathbb{G}$. The idea of the method goes back to the Crypto '88 Rump Session paper [9] by den Boer, who showed that DL reduces to DH if the prime p is such that $p-1$ is smooth (that is, not divisible by any large primes). For den Boer the auxiliary group was simply $\mathbb{F}_p^*$, and in fact the method can be most easily understood in that setting. For simplicity let us assume that $p-1$ is a product of distinct small primes

$$p - 1 = \prod \ell_i.$$

We are given $P, Q \in \mathbb{G}$ and need to find $x \in \mathbb{F}_p$ such that $Q = xP$.

We suppose that we have a DH-oracle $\mathcal{O}$ with the property that $\mathcal{O}(aP, bP) = abP$. By repeatedly using the oracle, we can compute expressions of the form $x^k P$ *without knowing* x. For example $x^2 P = \mathcal{O}(Q, Q)$ and $x^5 P = \mathcal{O}(Q, \mathcal{O}(x^2 P, x^2 P))$. The usual repeated-squaring procedure allows one to compute $x^k P$ in polynomially many (in $\log k$) calls to $\mathcal{O}$.

Let g be a generator of $\mathbb{F}_p^*$, and let j be the discrete logarithm of x to the base g in $\mathbb{F}_p^*$. To find x it suffices to find j, and this is done using Pohlig-Hellman [32] and either exhaustive search or baby-step/giant-step. Namely, by the Chinese Remainder Theorem it suffices to find $j \bmod \ell_i$ for each $\ell_i \mid p-1$. Let ℓ be one of the ℓ_i, and set $u = (p-1)/\ell$. Use the oracle to compute the point $Q_1 = x_1 P$, where $x_1 = x^u$. Set $g_1 = g^u$ and $g_2 = g_1^{\lceil\sqrt{\ell}\rceil}$, and compute $g_2^r P$, $1 \le r \le \lceil\sqrt{\ell}\rceil$, and $g_1^s Q_1$, $1 \le s < \lceil\sqrt{\ell}\rceil$. When you have a match, you know that the discrete logarithm of x_1 to the base g_1 in $\mathbb{F}_p^*$ — which is $j \bmod \ell$ – is equal to $r\lceil\sqrt{\ell}\rceil - s$.

4. Proof of Theorem 2.4

Proof. We are given non-identity elements P, Q in a group $\mathbb{G}$ of prime order p, and we must find the value of $x \bmod p$ such that $Q = xP$. We have an

[1] At ECC 2010, 18 October 2010, Redmond, Washington.

oracle for integer factorization, which we use to find the prime factorization $p-1 = \prod \ell_i^{\alpha_i}$. We also have a Diffie-Hellman oracle in the group $\mathbb{G}$ and a discrete logarithm oracle that can be called upon polynomially many times to find discrete logarithms in any group of prime order $\ell \neq p$.

We use the den Boer method. Let g be a fixed generator of $\mathbb{F}_p^*$; it suffices to find the discrete logarithm j of x to the base g in $\mathbb{F}_p^*$. By the Chinese Remainder Theorem, for this it suffices to find $j \bmod \ell_i^{\alpha_i}$ for each prime power factor of $p-1$. Let ℓ^α be one of these prime powers, and let j_s be $j \bmod \ell^s$ for $s = 1, \ldots, \alpha$. We find j_s by induction on s. Set $u_s = (p-1)/\ell^s$ for $1 \leq s \leq \alpha$, and set $R = g^{u_1} P$. Let $\mathbb{H}$ be the subgroup of $\mathbb{F}_p^*$ generated by g^{u_1}, where an element g^{ru_1}, $0 \leq r < \ell$, is represented implicitly by $g^{ru_1}P$, and where the group operation is performed using the DH oracle in $\mathbb{G}$. First, j_1 is clearly equal to the discrete logarithm of the element $x^{u_1}P$ (we can compute this element using the DH oracle without knowing x) to the base R in $\mathbb{H}$. By assumption, this can be computed using the discrete logarithm oracle. Next, for $s = 2, \ldots, \alpha$ suppose that we know j_{s-1}. We use the DH oracle to compute $(g^{-j_{s-1}}x)^{u_s}P$, and then find j_s by asking the discrete logarithm oracle for the discrete logarithm of this element to the base R in $\mathbb{H}$. This discrete logarithm is the s-th base-ℓ digit of $j \bmod \ell^\alpha$. In this way we find $j \bmod \ell^\alpha$. ∎

Remark 4.1. The above proof also shows that, without the Integer Factorization oracle, the All-Primes-But-p Discrete Logarithm problem $L_p(1/2)$-reduces to the Diffie-Hellman problem. This is because there are factoring algorithms with rigorously analyzed $L_p(1/2)$ running time (see [33, 23]). However, in §7. we shall prove a better result (Theorem 2.3) using genus-2 curves.

5. The Maurer-Wolf Genus-2 variant

The version of the den Boer – Maurer method using the jacobians of hyperelliptic curves of genus 2 was first developed in [40]. This variant was a natural generalization of [26] from genus 1 to genus 2.

In cryptography elliptic curves defined over finite fields have often been used as auxiliary groups to achieve objectives that have nothing to do with the curves themselves. The first such use was Lenstra's Elliptic Curve Method (ECM) of factoring integers [19]. Soon after, Goldwasser and Kilian [11] showed how to use elliptic curves to quickly certify primality. Somewhat later, Maurer and Wolf [26, 28, 29] as well as Boneh and Lipton [5] used elliptic curves to provide evidence that the Discrete Logarithm and Diffie-Hellman problems are equivalent.

Results of this type all depend on heuristic analyses of running time; the analyses assume conjectures that are out of reach of current analytic techniques about the distribution of either smooth numbers or prime numbers in short intervals. For example, the Goldwasser-Kilian primality certificate can be rigorously proved to work only for *most* primes; it works for *all* primes if one assumes that primes occur with the expected frequency in intervals of the form $(x, x + c\sqrt{x})$.

Adleman and Huang [2, 3], seeking to put the Goldwasser-Kilian primality certificate on a rigorous foundation, were the first to realize that this deficiency of elliptic curves could be removed by working with the jacobians of genus-2 curves. Analogously, Lenstra, Pila, and Pomerance [21, 22] later developed a genus-2 version of the ECM that has rigorously analyzed running time. In [40], Wolf did something similar for the den Boer – Maurer method of establishing reductions from Discrete Logarithm to Diffie-Hellman problems.

The advantage of genus 2 over genus 1 is that the interval over which the group orders are distributed is larger. That is, for randomly chosen elliptic curves over $\mathbb{F}_p$, the group orders n fall roughly uniformly in the Hasse interval, which extends from $(\sqrt{p}-1)^2$ to $(\sqrt{p}+1)^2$. If x denotes the beginning of this interval, then we can write $x \le n \le x+(4+o(1))x^{1/2}$. Similarly, the group orders of the jacobians of genus-2 curves fall roughly uniformly in the genus-2 Hasse-Weil interval, which extends from $(\sqrt{p}-1)^4$ to $(\sqrt{p}+1)^4$. Again letting x denote the beginning of this interval, in the genus-2 case we have $x \le n \le x+(8+o(1))x^{3/4}$. From the standpoint of the study of the distribution of prime and smooth numbers, intervals of the form $(x, x+cx^{1/2})$ are too short — almost nothing can be said about the abundance or even the existence of the desired numbers in such an interval (see [38]) — whereas intervals of the form $(x, x+cx^{3/4})$ are plenty long.

In the den Boer algorithm the Diffie-Hellman oracle was needed only to compute the implicit representations of powers of x — that is, to compute x^kP knowing xP but not x. In the elliptic and hyperelliptic curve versions due to Maurer and Wolf, one needs to compute expressions of the form $f(x)P$, where $f(x)$ is a rational function in the unknown x. Note that since $x^{-1}P = x^{p-2}P$, inverting is a special case of raising to the k-th power. The implicit representation of any $f(x)$ can be obtained by a combination of inverting, multiplying, and raising to powers (which require the use of the Diffie-Hellman oracle), along with addition/subtraction and constant multiplication (which do not). Also note that there is a probabilistic algorithm that in expected polynomial time can find the implicit representation of a squareroot of $f(x)$ whenever $f(x)$ is a square in $\mathbb{F}_p$. For example, if $p \equiv 3 \pmod 4$, then one need only compute $f(x)^{(p+1)/4}P$, which is the desired squareroot. In the general case one uses the algorithm of Shanks [36], which involves various exponentiations and comparisons that can all be performed using implicit representations with the help of the Diffie-Hellman oracle.

Now suppose that a genus-2 curve C is defined by

$$v^2 = g(u) \tag{1}$$

where $g \in \mathbb{F}_p[u]$ has degree 5 (see [7, 30]).

Suppose that the elements $P, Q \in \mathbb{G}$ are an instance of a Discrete Logarithm problem; in other words, we are asked to find x such that $Q = xP$. We outline how to go from the implicit representation Q of x to an element of the jacobian group $\mathcal{J}_C$ of the curve C. We first go from Q to the implicit representation (Q', R') of a point on the curve with coordinates $(x+r, y)$, where r is a random

integer mod p that we select. Namely, for each random choice of r we compute $Q' = rP + Q$ and then test whether $g(x+r)$ is a square in $\mathbb{F}_p$ — which we do by computing $g(x+r)^{(p-1)/2}P$ and checking whether or not it equals P. If not, then we choose another r and try again. Once we find $Q' = (x+r)P$ such that $g(x+r)$ is a square in $\mathbb{F}_p$, we use the Diffie-Hellman oracle along with Shanks' algorithm to compute an implicit representation $R' = yP$ of y for which $y^2 = g(x+r)$. Then (Q', R') is an implicit representation of a point on the curve.

Recall that elements of the jacobian group $\mathcal{J}_C$ are represented by reduced divisors $\mathrm{div}(a, b)$, where $a, b \in \mathbb{F}_p[u]$ satisfy $\deg b < \deg a \leq 2$, and a is monic and divides $b^2 - g$. The points (x, y) on the curve C correspond to the elements of $\mathcal{J}_C$ for which a is monic of degree 1; that is, we set $a = u - x$, $b = y$ in that case.

We shall say that an element $\mathrm{div}(a, b)$ with $a = a_0 + a_1 u + a_2 u^2$ and $b = b_0 + b_1 u$ is represented *implicitly* by the 5-tuple $(a_0 P, a_1 P, a_2 P, b_0 P, b_1 P) \in \mathbb{G}^5$. Thus, for example, the element of $\mathcal{J}_C$ corresponding to the point $(x + r, y) \in C$ is represented implicitly by $(-Q', P, O, R', O)$, where O is the identity of $\mathbb{G}$. Let B be this element of $\mathcal{J}_C$.

The group operation on the jacobian $\mathcal{J}_C$ (see [7, 30]) can be expressed in terms of rational functions of the components of the reduced divisors. This means that even when those components are represented implicitly, they can be composed with the help of the Diffie-Hellman oracle.

The generalization of den Boer's method to hyperelliptic curves works as follows. One first constructs a curve for which $\mathcal{J}_C$ has a Discrete Logarithm problem that is relatively easy to solve by purely generic means (such as exhaustive search or baby-step/giant-step). This means that its order n should be t-smooth for fairly small t, and in practice it is also useful for n to be squarefree, so that the jacobian group is cyclic.

Given an instance $P, Q \in \mathbb{G}$ of a Discrete Logarithm problem, one constructs an implicitly represented point on C and then the corresponding implicitly represented element B of $\mathcal{J}_C$, as above. One also constructs an explicit element of $\mathcal{J}_C$ that generates the group and computes its implicit representation $A \in \mathbb{G}^5$. Working with the implicit representatives of elements and using a generic discrete logarithm algorithm, one finds the discrete logarithm of B to the base A. Since the element A is known explicitly, once one has the discrete logarithm one can construct B explicitly, thereby getting the x-coordinate $x + r$ of the point of C that gave the element B of $\mathcal{J}_C$. Since r is known, this reveals x.

6. Almost-smooth numbers in the Hasse-Weil interval

In order to prove Theorems 2.2 and 2.3, we have to construct a genus-2 curve over $\mathbb{F}_p$ whose jacobian group $\mathcal{J}_C$ has a suitable order n. Unlike in the den Boer case, where the group order is $p-1$, or in the elliptic curve case, where the group order also has magnitude close to p, the two-dimensional abelian variety $\mathcal{J}_C$ has

group order $n \approx p^2$. More precisely, $(\sqrt{p}-1)^4 \le n \le (\sqrt{p}+1)^4$. This means that it would do us no good to find curves for which n is prime (as was done in Adleman-Huang primality testing [2, 3]). Rather, in order to use the oracle that's available to us in the One-Prime-Not-p or All-Primes-But-p problem, we need a subgroup of $\mathcal{J}_C$ of prime order q having no greater bitlength than p. Thus, we want n to be the product of a prime q no greater than p and a smooth number s.

We shall want $s = n/q$ to be as smooth as possible, in part so that we can factor n relatively quickly and in part (in the case of Theorem 2.2, where we can use the Discrete Logarithm oracle for only one prime $q \ne p$) so that discrete logs can be found relatively quickly in subgroups of prime order dividing s. In the case of factoring, the best rigorously analyzed result is Corollary 1.2 of [21], which states that all prime factors of n that are less than y can be found in time $L_y(2/3)\log^2 n$. For the discrete log we have no such luck, because we need to use generic algorithms when we have implicitly represented group elements — we cannot do better than $\sqrt{y}$.

Let $\mathcal{P}(x)$ denote the set of prime numbers in the interval $[1, x]$; let $\mathcal{S}(x, y)$ denote the set of y-smooth numbers in the interval $[1, x]$; and let $\mathcal{S}^*(x, y)$ denote the set of squarefree numbers in $\mathcal{S}(x, y)$. Thus, $\mathcal{S}^*(x_2, y) \setminus \mathcal{S}^*(x_1, y)$ denotes the set of y-smooth squarefree integers in the interval $(x_1, x_2]$. We further denote $\pi(x) = \#\mathcal{P}(x)$, $\psi(x, y) = \#\mathcal{S}(x, y)$, and $\psi^*(x, y) = \#\mathcal{S}^*(x, y)$. Let $\rho(u)$ be the *Dickman – de Bruijn function* defined recursively by $\rho(u) = 1$ for $0 \le u \le 1$ and

$$\rho(u) = 1 - \int_1^u \frac{\rho(v-1)}{v} dv, \qquad u > 1.$$

We shall need the following well-known estimates for $\rho(u)$, $\psi(x, y)$, and $\psi^*(x, y)$.

Lemma 6.1. (a) (see Corollary 2.3 of [13]) *For $u \ge 1$ one has*

$$\rho(u) = \exp\bigl(-u(\log u + \log\log u) + O(1)\bigr). \tag{2}$$

(b) (see Theorem 1.1 of [13]) *Let $x \ge y \ge 2$, and set $u = \log x/\log y$. For any fixed $\varepsilon > 0$ one has*

$$\psi(x, y) = x\rho(u)\left(1 + O\left(\frac{\log(1+u)}{\log y}\right)\right) \tag{3}$$

uniformly in the range $y \ge \exp\bigl((\log\log x)^{(5/3)+\varepsilon}\bigr)$.

(c) (see Theorem 3 of [14]) *For $x \ge y \ge 2$ one has*

$$\psi^*(x, y) = \left(\frac{6}{\pi^2} + o(1)\right)\psi(x, y) \tag{4}$$

as $\log\log x/\log y \longrightarrow 0$.

The next theorem is the central element in our proof of Theorems 2.2 and 2.3.

Theorem 6.2. *For any fixed $\varepsilon > 0$, for x sufficiently large, and for*

$$y \geq \exp\left((\log\log x)^{(5/3)+\varepsilon}\right), \tag{5}$$

we have

$$\#\left\{qs \in [x^2, x^2 + x^{3/2}] \ : \ q \in \mathcal{P}(x), \ s \in \mathcal{S}^*(2x, y)\right\} \ \geq \ C\rho(u)\frac{x^{3/2}}{\log x}, \tag{6}$$

where $C > 0$ is an absolute constant.

We shall need the following result of Matomäki [25]:

Lemma 6.3 (Matomäki). *There exist absolute constants $c, c_4, c_6 > 0$ and two functions $A_4(x, z)$ and $A_6(x, z)$ such that*

$$\int_x^{2x} |A_i(x, z)|^i dz \leq c_i x^{2/3}, \qquad i = 4, 6, \tag{7}$$

and for all sufficiently large $x \geq 2$ and for $z \in [x, 2x]$ one has

$$\pi\left(z + \frac{z}{4\sqrt{x}}\right) - \pi(z) \ \geq \ \frac{z}{4\sqrt{x}\,\log z}(c + A_4(x, z) + A_6(x, z)). \tag{8}$$

Corollary 6.4. *There exists an absolute constant $c_0 > 0$ such that for all sufficiently large $x \geq 2$ one has*

$$\pi\left(z + \frac{z}{4\sqrt{x}}\right) - \pi(z) \ \geq \ c_0\frac{z}{4\sqrt{x}\,\log z} \tag{9}$$

for all $z \in [x, 2x]$ that are outside of at most $O(x^{2/3})$ intervals of the form $[n, n + 1) \subset [x, 2x]$ for n an integer.

Proof. The corollary follows easily from Lemma 6.3. Namely, note that for x fixed, $|A_4(x, z)|^4 + |A_6(x, z)|^6$ cannot be greater than $(c/3)^6 = \min((c/3)^6, (c/3)^4)$ (where without loss of generality we're supposing that $c < 3$) for all z on more than $\gamma x^{2/3}$ intervals of the form $[n, n + 1) \subset [x, 2x]$, where $\gamma = (c_4 + c_6)(3/c)^6$, because that would contradict (7). It follows that for all $n \in [x, 2x]$, except for at most $\gamma x^{2/3}$ of them, there is a value $z \in [n, n + 1)$ such that $|A_i(x, z)|^i \leq (c/3)^i$ for $i = 4, 6$, and so, by (8), the inequality (9) holds with $c_0 = c/3$ for such z. Now for fixed x, as $z \in [x, 2x]$ increases from n to $n+1$ the left-hand side of (9) either remains constant or increases by 1, whereas the right-hand side of (9) increases by less than $c_0/(4x^{1/2}\log x)$. For large x this means that, setting $c_0 = c/4$, we have (9) holding for all $z \in [n, n + 1)$ except for at most $\gamma x^{2/3}$ values of n, as claimed. ∎

Proof. We now prove Theorem 6.2. Without loss of generality we suppose that $y < x/2$. In that case for $s \in \mathcal{S}^*(2x, y)$ and $qs \in [x^2, x^2 + x^{3/2}]$ we have $q \geq$

$x/2 > y$, and so the different products qs are pairwise distinct. The number on the left in (6) is at least equal to

$$\begin{aligned}\sum_{\substack{s\in\mathcal{S}^*(2x,y)\\ s\ge x+\sqrt{x}}} &\left(\pi\left(\frac{x^2+x^{3/2}}{s}\right)-\pi\left(\frac{x^2}{s}\right)\right)\\ &\ge \sum_{s\in\mathcal{S}^*(2x,y)\setminus\mathcal{S}^*(3x/2,y)}\left(\pi\left(\frac{x^2+x^{3/2}}{s}\right)-\pi\left(\frac{x^2}{s}\right)\right).\end{aligned} \tag{10}$$

We fix $s \in \mathcal{S}^*(2x,y)\setminus\mathcal{S}^*(3x/2,y)$ and set $z = x^2/s$. We apply Corollary 6.4 with x replaced by $x/2$, that is, $x \ge 4$, $z \in [x/2, x]$, and $4\sqrt{x}$ is replaced by $\sqrt{8x}$ in (9). Note that $x^{3/2}/s = z/\sqrt{x} > z/\sqrt{8x}$, and so Corollary 6.4 can be used to bound each term in (10) from below. Also since $s \le 2x$, it is easy to see that there are at most 4 different values of s that give z-values in the same interval $[n, n+1)$. Hence, the bound in Corollary 6.4 can be applied for all but $O(x^{2/3})$ values of $s \in \mathcal{S}^*(2x,y)\setminus\mathcal{S}^*(3x/2,y)$, giving

$$\pi\left(\frac{x^2+x^{3/2}}{s}\right)-\pi\left(\frac{x^2}{s}\right) \ \ge\ c_0\frac{x^2/s}{\sqrt{8x}\log(x^2/s)} \ \ge\ C_0\frac{\sqrt{x}}{\log x} \tag{11}$$

for those values of s, where C_0 is an absolute constant.

Meanwhile, Lemma 6.1(b)-(c) shows that for $y \ge \exp\left((\log\log x)^{(5/3)+\varepsilon}\right)$ we have the estimate

$$\psi^*(2x,y)-\psi^*(3x/2,y) = \left(\frac{3}{\pi^2}+o(1)\right)x\rho(u), \tag{12}$$

for the number of terms in (10); and by Lemma 6.1(a) we also have $\rho(u) = x^{-o(1)}$. We recall that there are at most $O(x^{2/3})$ values of $s \in \mathcal{S}^*(2x,y)\setminus\mathcal{S}^*(3x/2,y)$ for which (11) does not apply, and so we subtract $O(x^{2/3})$ from the right side of (12). We still obtain the estimate $\left(\frac{3}{\pi^2}+o(1)\right)x\rho(u)$ for the number of s that remain. Hence, the summation (10) is bounded from below by

$$\sum_{s\in\mathcal{S}^*(2x,y)\setminus\mathcal{S}^*(3x/2,y)}\left(\pi\left(\frac{x^2+x^{3/2}}{s}\right)-\pi\left(\frac{x^2}{s}\right)\right) \ge C\rho(u)\frac{x^{3/2}}{\log x}$$

for some absolute constant $C > 0$, as claimed. ∎

7. Proof of Theorems 2.2–2.3

The idea of the proof of both Theorems 2.2 and 2.3 is to find a genus-2 curve C over $\mathbb{F}_p$ for which the order of the jacobian $\mathcal{J}_C(\mathbb{F}_p)$ has been factored and is equal to the product of a prime q and an $L_p(\alpha)$-smooth squarefree integer s (where $\alpha \in (0,1)$ is appropriately chosen). Recall that the notation $L_x(\alpha)$

includes an unspecified constant in the exponent, and so equalities of the form $y = L_p(\alpha)$ should be interpreted as an order of magnitude statement, with lower order terms neglected. Thus, for example, we shall use the identities $L_x(\alpha)L_x(\beta) = L_x(\max(\alpha, \beta))$ and $L_{L_x(\alpha)}(\beta) = L_x(\alpha\beta)$, where lower order terms in the exponent are neglected. Another useful identity is the following:

Lemma 7.1. *For $y = L_x(\alpha)$ and $u = \log x/\log y$ we have $1/\rho(u) = L_x(1-\alpha)$.*

Proof. This follows from Lemma 6.1(a), since neglecting lower order terms we see that $-\log\rho(u)$ is asymptotically equal to

$$u\log u \sim \frac{\log x}{c\log^{\alpha} x\log\log^{1-\alpha} x}(\log\log x - \alpha\log\log x) = \frac{1-\alpha}{c}\log^{1-\alpha} x\log\log^{\alpha} x,$$

as claimed. ∎

We need the following lemma, which is a slightly weaker version of Theorem 1.1 in [22].

Lemma 7.2. *Let p be a sufficiently large prime. Then for all but at most $28p^{1/2}$ integers n in the interval $[p^2-0.5p^{3/2}, p^2+0.5p^{3/2}]$ there are at least $p^{9/2}(\log p)^{-3}$ polynomials $g \in \mathbb{F}_p[u]$ of degree 5 such that the curve $C:\ v^2 = g(u)$ is of genus 2, and its jacobian $\mathcal{J}_C$ has order n.*

Proof. We now prove Theorems 2.2 and 2.3.

Let y have order $L_p(\alpha)$, where $\alpha \in (0,1)$ will be chosen later so as to optimize the total running time. We use Lemma 7.2 in conjunction with Theorem 6.2 with x set equal to $\sqrt{p^2-0.5p^{3/2}}$. We can define $u = \log p/\log y$ rather than $\log x/\log y$, since the difference does not affect any of the estimates below.

Combining Lemma 7.2 with Theorem 6.2, since there are p^6 degree-5 polynomials over $\mathbb{F}_p$, we see that it takes $O(\log^4 p/\rho(u))$ random polynomials $g \in \mathbb{F}_p[u]$ in order to find one for which the corresponding $\mathcal{J}_C$ has order n of the form qs, where q is a prime $\le x$ and s is y-smooth and squarefree. Since $\log^4 p$ is much smaller than $1/\rho(u)$, by Lemma 7.1 we can estimate the number of trial curves by $L_p(1-\alpha)$. For each such curve, a theorem of Pila [31] ensures that the order n of $\mathcal{J}_C$ can be computed in polynomial time. According to Corollary 1.2 of [21], there is an $L_y(2/3)$-algorithm that finds and factors the y-smooth part s of n (that is, it finds all prime factors of n that are $\le y$). (Note that the $\log^2 n$-factor in Corollary 1.2 of [21] can be incorporated into $L_y(2/3)$ because $(\log y)^{2/3}$ is large compared to $\log\log n$ when $y = L_p(\alpha)$ with $\alpha > 0$.) Since $y = L_p(\alpha)$, it follows that $L_y(2/3) = L_p(2\alpha/3)$. Then in deterministic polynomial time one can check whether or not $n/s = q$ is prime (see [4]).

Finally, in Theorem 2.2 we need to compute discrete logarithms by exhaustive search or baby-step/giant-step in subgroups of order a prime divisor of s; either method takes time of order $y = L_p(\alpha)$. In Theorem 2.2 the discrete logarithm in the subgroup of order q is supplied by the "prime-not-p" DL oracle, and in

Theorem 2.3 that oracle supplies the discrete logarithms in the order-ℓ subgroups for all prime divisors ℓ of n.

To summarize, the running time is dominated by three steps:

- search for the curve — $L_p(1-\alpha)$ trials;
- factorization of the group order of each trial jacobian — time $L_p(2\alpha/3)$;
- (Theorem 2.2 only) discrete logarithm computations on the jacobian — time $L_p(\alpha)$.

Then the running time of the entire algorithm is the largest of these three estimates, and we optimize by choosing α in Theorem 2.2 so that $\max(1-\alpha, 2\alpha/3, \alpha)$ is minimal, and in Theorem 2.3 so that $\max(1-\alpha, 2\alpha/3)$ is minimal. The former choice is $\alpha = 1/2$ and the latter is $\alpha = 3/5$ (which gives a running time of $L_p(1-3/5) = L_p(2/5)$). This gives the desired results. ∎

Remark 7.3. In §4. we saw that a factoring oracle makes a dramatic difference in the efficiency of reduction of All-Primes-But-p-DL to the Diffie-Hellman problem. However, the proof of Theorem 2.2 suggests that in the case of One-Prime-Not-p-DL a factoring oracle would probably not help speed up the reduction. It should also be noted that the $L_p(2/3)$-reduction from DL to DH of Maurer-Wolf (see Theorem 6 of [29]) would not be sped up by a factoring oracle. In these cases the time needed to compute the discrete logarithm of an implicitly represented element is much greater than the time needed to factor.

8. Proof of Theorems 2.7–2.8

Proof. We prove Theorem 2.7. The approach is similar to the one taken by Maurer-Wolf [29] in establishing a rigorously analyzed $L_p(2/3)$ DL-to-DH reduction and taken by us in proving Theorems 2.2 and 2.3 in §7.. Namely, we use the jacobians of genus-2 curves, whose group orders, according to [22] (see Lemma 7.2), are distributed fairly uniformly on an interval of the form $(x, x+x^{3/4})$ (where $x = p^2 - 0.5p^{3/2}$). The most important difference with those proofs is that we no longer need a *subexponentially* small probability that the number of elements of a jacobian group has the desired smoothness. Rather, because of our access to the yes-or-no oracle we can use the following recent result of Soundararajan [38], giving an *exponentially* small lower bound $x^{-\varepsilon}$ on the probability that an integer in the interval is $L_p(1/2)$-smooth.

Lemma 8.1. (Soundararajan [38]) *Assume the Riemann Hypothesis, let x be large, suppose that $x \ge y \ge \exp(5\sqrt{\log x \log\log x})$, and set $u = \log x/\log y$. Then there is an absolute constant B and constants C_δ depending only on δ such that for $z = Bu\sqrt{x}/\rho(u/2)$ and for any $\delta > 0$*

$$\psi(x+z, y) - \psi(x, y) \ge C_\delta z x^{-\delta}. \tag{13}$$

We apply Lemma 8.1 with

$$x = p^2 - 0.5p^{3/2}, \qquad y = \exp(5\sqrt{\log x \log\log x}),$$

that is, $y = L_p(1/2)$. We find that

$$u = \left(\frac{1}{5} + o(1)\right)\sqrt{\log x/\log\log x}$$

and, by Lemma 6.1(a), we have

$$\frac{1}{\rho(u/2)} = \exp\left(\left(\frac{1}{20} + o(1)\right)\sqrt{\log x \log\log x}\right).$$

Thus, z in the lemma has order of magnitude $L_x(1/2)\sqrt{x}$. Note that the same result applies when z is replaced by a larger interval length (perhaps changing the constant C_δ); in particular, we can take z of magnitude $x^{3/4}$. We conclude that a number in the interval $(p^2-0.5p^{3/2}, p^2+0.5p^{3/2})$ has probability $\geq C_\delta p^{-2\delta}$ of being y-smooth, $y = L_p(1/2)$.

Using Lemma 7.2 and the same argument as in §7., we see that for a randomly chosen degree-5 polynomial $g(u)$ there is probability at least $c_\delta p^{-2\delta}$ (where c_δ is chosen slightly smaller than C_δ to allow for the $\log p$-factors in Lemma 7.2) that $v^2 = g(u)$ is the equation of a genus-2 curve C for which the number of elements of the jacobian is y-smooth, $y = L_p(1/2)$. Once we find such a polynomial $g(u)$, the Maurer-Wolf method allows us to find discrete logarithms in $\mathbb{G}$ in time $L_p(1/2)$ using the DH oracle. Note, however, that we cannot use the Maurer-Wolf method in quite the same form that was described in §5., because that requires the jacobian group $\mathcal{J}_C$ to be of squarefree order (and hence cyclic), and the y-smooth numbers provided by Lemma 8.1 are not necessarily squarefree. (We do not know of any way to prove Lemma 8.1 with $\psi(x+z,y) - \psi(x,y)$ replaced by $\psi^*(x+z,y) - \psi^*(x,y)$; for example, Lemma 6.1(c) does not imply such a result.) We proceed by first selecting 6 elements uniformly at random from $\mathcal{J}_C$. Since $\mathcal{J}_C$ has rank at most 4, it follows from a result of Pomerance [34] that the 6 chosen elements form a generating set for $\mathcal{J}_C$ with probability at least 0.29. Indeed, by [34, Corollary 2] the expected value of the first i such that the first through i-th elements chosen uniformly at random from a finite abelian group $\mathcal{G}$ of rank r generate $\mathcal{G}$ is less than $r + \sigma$ where $\sigma = 2.11458\ldots$. Thus, denoting by ρ_i the probability that i is the smallest integer such that the first through i-th elements chosen generate $\mathcal{J}_C$, we have

$$\sum_{i=r}^{\infty} i\rho_i \leq r + \sigma$$

where $r \leq 4$ is the rank of $\mathcal{J}_C$. Using $\sum_{i=r}^{\infty} \rho_i = 1$ we derive

$$\sum_{i=r+1}^{\infty} (i-r)\rho_i \leq \sigma.$$

Therefore

$$\sum_{i=r}^{r+2} \rho_i = 1 - \sum_{i=r+3}^{\infty} \rho_i \geq 1 - \frac{1}{3} \sum_{i=r+3}^{\infty} (i-r)\rho_i \geq 1 - \sigma/3 > 0.29.$$

Thus, $r+2 \leq 6$ randomly chosen elements from $\mathcal{J}_C$ form a generating set with probability at least 0.29.

After using the factorization of $\#\mathcal{J}_C$ to determine the orders of the 6 elements, the Pohlig-Hellman method [32] and baby-step/giant-step search can be used in a standard manner (see [39]) to express the implicitly represented element B in terms of the 6 chosen elements.

So it remains to show how at most εn queries to the omniscient yes-or-no oracle can be used to locate such a polynomial $g(u)$, with at most $p^{-\varepsilon/2}$ probability of failure.

We use essentially the same procedure as Maurer [27]; the method is due to Chor and Goldreich [8]. We choose some fixed $\mathbb{F}_p$-basis $\alpha_0, \alpha_1, \ldots, \alpha_5$ of $\mathbb{F}_{p^6}$ and let the polynomial $g(u) = a_0 + a_1 u + \cdots + a_5 u^5$ correspond to the element $\sum a_i \alpha_i$ of the field $\mathbb{F}_{p^6}$. We also list the field elements in some order, say, lexicographically with a_i regarded as the i-th base-p digit of the order of $\sum a_i \alpha_i$ in the list.

The algorithm starts by randomly choosing two elements $s, t \in \mathbb{F}_{p^6}$, which map the ordering of elements to a new ordering according to the rule $w \mapsto sw+t$; that is, the elements of $\mathbb{F}_{p^6}$ are now listed in the order $\{sw_1+t, sw_2+t, \ldots\}$, where $\{w_1, w_2, \ldots\}$ is the lexicographical ordering with respect to the basis $\{\alpha_i\}$. When we say "first," we mean with respect to this new ordering. The algorithm then asks the yes-or-no oracle for the successive bits of the index of the first element $sw + t \in \mathbb{F}_{p^6}$ that corresponds to a polynomial $g(u)$ that gives a jacobian whose number of elements is y-smooth, $y = L_p(1/2)$, if the index of that element is $< 2^{\lfloor \varepsilon n \rfloor}$. If there are more than $\lfloor \varepsilon n \rfloor$ bits in that integer, the oracle is instructed to output 0. The algorithm fails if it outputs 0.

According to a fundamental result of Chor and Goldreich [8], the "pairwise random" sequence $sw + t$ behaves randomly enough to get a good sample and have probability of success not very different from that afforded by a fully random sequence. Namely, their result tells us that the probability that none of the first k elements sampled leads to a "good" $g(u)$ is no greater than $(1 - c_\delta p^{-2\delta})/(kc_\delta p^{-2\delta}) < p^{2\delta}/(kc_\delta)$. We apply this with $k = 2^{\lfloor \varepsilon n \rfloor} \geq 0.25 p^\varepsilon$ and $\delta = \varepsilon/8$. Then the probability of failure is at most $p^{\varepsilon/4}/(p^\varepsilon c_{\varepsilon/8}) < p^{-\varepsilon/2}$ for large p. This completes the proof of Theorem 2.7.

The proof of Theorem 2.8 is similar, except that we use elliptic curves rather than genus-2 curves, Proposition (2.7) of [19] instead of Lemma 7.2, and Conjecture 2.6 instead of Lemma 8.1. We omit the details, which are essentially the same as for Theorem 2.7. ∎

Remark 8.2. It would seem natural to try to improve the estimate $L_p(1/2)$ in Theorems 2.7 by using higher genus curves. Indeed, the Hasse-Weil interval for the jacobian of a genus-g curve is of the form $[x, x + cx^\beta]$, where $\beta = 1 - 1/(2g)$

approaches 1 as $g \longrightarrow \infty$. However, for $\alpha < 1/2$ nothing has been proved about the abundance of $L_x(\alpha)$-smooth numbers in an interval of that form, even for β close to 1 (see [38]). One would need such a result in order to lower the time estimate in Theorem 2.7.

9. Conclusion

There are at least three ways to compare different versions of problems:

(a) formal reductions allowing the use of unproved conjectures, such as those having to do with the distribution of primes and smooth numbers;

(b) mathematically rigorous formal reductions;

(c) ignoring reductions entirely and trusting only one's gut instincts in a real-world setting.

It's interesting to compare and contrast the conclusions that follow from (a), (b), and (c) using Shoup's language of relativized results (see [37]). We have the following table of running times "relative to DH" (that is, in the presence of a Diffie-Hellman oracle).

Table 1 Relativized running times with a DH oracle

	DL	One-Prime-Not-p DL	All-Primes-But-p-DL
(a) heuristic	$L_p(1/2)$	polynomial time	polynomial time
(b) rigorous	$L_p(2/3)$	$L_p(1/2)$	$L_p(2/5)$

We also showed that $\varepsilon \log p$ yes-or-no oracle queries brings the lower-left entry in the table from $L_p(2/3)$ to $L_p(1/2)$ (under the Riemann Hypothesis, with exponentially small probability of failure) and brings the upper-left entry from $L_p(1/2)$ to polynomial time (with exponentially small probability of failure). In addition, we saw that a factoring oracle helps only the lower-right entry in the table, where it reduces the complexity from $L_p(2/5)$ to polynomial time.

Meanwhile, a guts-trusting practical person would reject the entire "relative to DH" way of thinking, because the only way a DH would ever be computed is by first computing the DL. Once that's rejected, he/she would say that neither the ability to factor quickly nor the ability to find discrete logarithms in groups of order $q \neq p$ would be of any help whatsoever.

A somewhat extreme viewpoint would be that, like most of theoretical cryptography, relativized results proved by mathematically rigorous reductions, such as those in the present paper, live in a Platonic realm that is orthogonal to the domain of real-world cryptography. According to this view — which we neither subscribe to nor can easily refute — such results have as little relevance to

the practical cryptographer as Michael Jordan's basketball prowess has to the inhabitants of Flatland [1].

Acknowledgments

We wish to thank Glyn Harman for calling our attention to Matomäki's result in [25].

References

1. E. A. Abbott, *Flatland: A Romance of Many Dimensions*, London: Seeley and Co., 1884.
2. L. M. Adleman and M.-D. Huang, Recognizing primes in random polynomial time, *Proc. 19th ACM Symp. Theory Comput.*, 1987, pp. 462-469.
3. L. M. Adleman and M.-D. Huang, *Primality Testing and Abelian Varieties over Finite Fields*, LNM 1512, Springer-Verlag, 1992.
4. M. Agrawal, N. Kayal, and N. Saxena, Primes is in P, *Ann. Math.*, **160** (2004), pp. 781-793.
5. D. Boneh and R. Lipton, Algorithms for black-box fields and their application to cryptography, *Advances in Cryptology — CRYPTO '96*, LNCS 1109, Springer-Verlag, 1996, pp. 283-297.
6. D. Boneh and R. Venkatesan, Breaking RSA may not be equivalent to factoring, *Advances in Cryptology — Eurocrypt 1998*, LNCS 1233, Springer-Verlag, 1998, pp. 59-71.
7. D. G. Cantor, Computing in the Jacobian of a hyperelliptic curve, *Math. Comp.*, **48** (1987), pp. 95-101.
8. B. Chor and O. Goldreich, On the power of two-point based sampling, *J. Complexity*, **5** (1989), pp. 96-106.
9. B. den Boer, Diffie-Hellman is as strong as discrete log for certain primes, *Advances in Cryptology — Crypto '88*, LNCS 403, Springer-Verlag, 1988, pp. 530-539.
10. W. Diffie and M. Hellman, New directions in cryptography, *IEEE Trans. Inform. Theory*, **22** (1976), pp. 644-654.
11. S. Goldwasser and J. Kilian, Almost all primes can be quickly certified, *Proc. 18th ACM Symp. Theory Comput.*, 1986, pp. 316-329.
12. G. H. Hardy, *A Mathematician's Apology*, Cambridge Univ. Press, 1940.
13. A. Hildebrand and G. Tenenbaum, Integers without large prime factors, *J. Théor. Nombres Bordeaux*, **5** (1993), pp. 411-484.
14. A. Ivić and G. Tenenbaum, Local densities over integers free of large prime factors, *Quart. J. Math.*, **37** (1986), pp. 401-417.
15. N. Koblitz, The uneasy relationship between mathematics and cryptography, *Notices of the AMS*, **54** (2007), pp. 972-979.
16. N. Koblitz and A. J. Menezes, Another look at "provable security," *J. Cryptology*, **20** (2007), pp. 3-37.
17. N. Koblitz and A. J. Menezes, Another look at "provable security." II, *Progress in Cryptology — Indocrypt 2006*, LNCS 4329, Springer-Verlag, 2006, pp. 148-175.
18. N. Koblitz and A. Menezes, Intractable problems in cryptography, *Proc. 9th International Conf. Finite Fields and Their Applications*, *Contemporary Math.*, **518** (2010), pp. 279-300.
19. H. W. Lenstra, Jr., Factoring integers with elliptic curves, *Ann. Math.*, **126** (1987), pp. 649-673.

20. A. K. Lenstra and H. W. Lenstra, Jr., eds., *The Development of the Number Field Sieve*, LNM 1554, Springer-Verlag, 1993.
21. H. W. Lenstra, Jr., J. Pila, and C. Pomerance, A hyperelliptic smoothness test. I, *Phil. Trans. R. Soc. Lond.*, (A) **345** (1993), pp. 397-408.
22. H. W. Lenstra, Jr., J. Pila, and C. Pomerance, A hyperelliptic smoothness test. II, *Proc. Lond. Math. Soc.*, (3) **84** (2002), pp. 105-146.
23. H. W. Lenstra, Jr. and C. Pomerance, A rigorous time bound for factoring integers, *J. Am. Math. Soc.*, **5** (1992), pp. 483-516.
24. T. Malkin, R. Moriarty, and N. Yakovenko, Generalized environmental security from number theoretic assumptions, *Theory of Cryptography — TCC 2006*, LNCS 3876, Springer-Verlag, 2006, pp. 343-359.
25. K. Matomäki, Large differences between consecutive primes, *Quart. J. Math.*, **58** (2007), pp. 489-518.
26. U. Maurer, Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms, *Advances in Cryptology — CRYPTO '94*, LNCS 839, Springer-Verlag, 1994, pp. 271-281.
27. U. Maurer, On the oracle complexity of factoring integers, *Computational Complexity*, **5** (1995), pp. 237-247.
28. U. Maurer and S. Wolf, The relationship between breaking the Diffie-Hellman protocol and computing discrete logarithms, *SIAM J. Computing*, **28** (1999), pp. 1689-1721.
29. U. Maurer and S. Wolf, The Diffie-Hellman protocol, *Designs, Codes and Cryptography*, **19** (2000), pp. 147-171.
30. A. Menezes, Y.-H. Wu, and R. Zuccherato, An elementary introduction to hyperelliptic curves, Appendix to N. Koblitz, *Algebraic Aspects of Cryptography*, Springer-Verlag, 1998, pp. 155-178.
31. J. Pila, Frobenius maps of abelian varieties and finding roots of unity in finite fields, *Math. Comp.*, **55** (1990), pp. 745-763.
32. S. Pohlig and M. Hellman, An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance, *IEEE Trans. Inf. Th.*, **24** (1978), pp. 106-110.
33. C. Pomerance, Fast, rigorous factorization and discrete logarithm algorithms, in D. S. Johnson, T. Nishizeki, A. Nozaki, and H. S. Wilf, eds., *Discrete Algorithms and Complexity*, Academic Press, 1987, pp. 119-143.
34. C. Pomerance, The expected number of random elements to generate a finite abelian group, *Periodica Mathematica Hungarica*, **43** (2001), pp. 191-198.
35. R. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public key cryptosystems, *Communications of the ACM*, **21** (2) (1978), pp. 120-126.
36. D. Shanks, Five number-theoretic algorithms, *Cong. Numer.*, **7** (1972), pp. 51-70.
37. V. Shoup, OAEP reconsidered, *Advances in Cryptology — CRYPTO 2001*, LNCS 2139, Springer-Verlag, 2001, pp. 239-259.
38. K. Soundararajan, Smooth numbers in short intervals, arXiv:1009.1591v1, 8 Sept. 2010.
39. E. Teske, The Pohlig-Hellman method generalized for group structure computation, *J. Symbolic Computation*, **27** (1999), pp. 521-534.
40. S. Wolf, Diffie-Hellman and discrete logarithms, Diploma Thesis, Department of Computer Science, ETH Zürich, 1995.