Hardcore Predicates for a Diffie-Hellman Problem over Finite Fields

Nelly Fazio[1,2], Rosario Gennaro[1,2], Irippuge Milinda Perera[2], and William E. Skeith III[1,2]

[1]The City College of CUNY
{fazio,rosario,wes}@cs.ccny.cuny.edu
[2]The Graduate Center of CUNY
iperera@gc.cuny.edu

March 9, 2013

Abstract

A long-standing open problem in cryptography is proving the existence of (deterministic) hardcore predicates for the Diffie-Hellman problem defined over finite fields. In this paper we make progress on this problem by defining a very natural variation of the Diffie-Hellman problem over $\mathbb{F}_{p^2}$ and proving the unpredictability of every single bit of one of the coordinates of the secret DH value.

To achieve our result we modify an idea presented at CRYPTO'01 by Boneh and Shparlinski [4] originally developed to prove that the LSB of the Elliptic Curve Diffie-Hellman problem is hard. We extend this idea in two novel ways:

1. We generalize it to the case of finite fields $\mathbb{F}_{p^2}$;
2. We prove that any bit, not just the LSB, is hard using the list decoding techniques of Akavia et al. [1] (FOCS'03) as generalized at CRYPTO'12 by Duc and Jetchev [6].

In the process we prove several other interesting results:

- Our result hold also for a larger class of predicates, called *segment predicates* in [1];
- We extend the result of Boneh and Shparlinski to prove that every bit (and every segment predicate) of the Elliptic Curve Diffie-Hellman problem is hard-core;
- We define the notion of *partial one-way function* over finite fields $\mathbb{F}_{p^2}$ and prove that every bit (and every segment predicate) of one of the input coordinate for these functions is hard-core.

Keywords: Hardcore Bits, Diffie-Hellman Problem, Finite Fields, Elliptic Curves.

1 Introduction

A long-standing open problem in cryptography is proving the existence of (deterministic) hard-core predicates for the Diffie-Hellman problem defined over finite fields. In this paper we make progress on this problem by defining a very natural extension of the Diffie-Hellman problem over $\mathbb{F}_{p^2}$ and proving that a large class of predicates (including every single bit of one of the coordinates) are unpredictable under the assumption that this problem is hard.

In their seminal paper that introduced public-key cryptography [5] Diffie and Hellman defined the following key exchange protocol, which works in arbitrary finite cyclic groups. Let G be such a group, generated by g of order n. Two parties, Alice and Bob, want to establish a secret value. Alice chooses a random value $a \in \mathbb{Z}_n$ and sends the value $A = g^a$ to Bob. Similarly Bob chooses a random value $b \in \mathbb{Z}_n$ and sends the value $B = g^b$ to Alice. At this point they share the common Diffie-Hellman secret value

$$K = g^{ab} = A^b = B^a$$

The *Computational Diffie-Hellman Assumption (CDH)* over the group G informally states that no efficient algorithm can compute $K = g^{ab}$ when given only $g, A = g^a, B = g^b$. The hardness of computing the *entire* value K, however does not rule out an efficient way to compute some of the bits of K, or even just predict them with a probability better than a random guess. This property is very important because without it, Alice and Bob do not have any guarantee about the "pseudo-randomness" of any bit of the secret value K, and those are the properties needed by K in order to be used as a secret key in a subsequent cryptographic scheme. This problem is usually addressed by making a much stronger assumption on the hardness of the Diffie-Hellman problem: the so-called *Decisional Diffie-Hellman Assumption (DDH)* states that the value K is computationally indistinguishable from a random element of G. While the DDH guarantees that the entire value of K is pseudo-random, we know that there are groups G where the DDH is false, even when the CDH is still conjectured to be hard.

Ideally, however, one would like to prove that certain bits (or more generally, certain predicates) of the value K are unpredictable, when given g^a and g^b, simply under the CDH assumption. Such results were established quite early for other conjectured hard-problems (*e.g.*, Blum and Micali's result on the hardness of discrete log bits [3] and Alexi at al. work on the hardness of the RSA input bits [2]). However for the case of the Diffie-Hellman problem no such result has been proven (except for the result by Boneh and Shparlinksi [4] in a slightly different model and which we discuss below). The only hard-core predicates known for the Diffie-Hellman function are the generic "randomized" predicates which work over any computationally hard problem (*e.g.*, the Goldreich-Levin and Naslund hard-core bits [7, 11]).

Hard-Core Predicates. Let $\pi : G \to \{\pm 1\}$ be a predicate[1] defined over G. To prove that π is hard-core for the CDH problem one has to construct a reduction from guessing π better than at random, to solving the CDH problem. More specifically, assume we have an oracle Ω which on input g, g^a, g^b outputs the correct $\pi(g^{ab})$ with probability (taken over the choice of a, b) substantially better than[2] $1/2$, then there is an efficient algorithm A which invokes Ω and solves the CDH problem.

Note that a crucial step of this reduction is to "correct" the answers of the oracle Ω which are guaranteed to be right only slightly more than half of the times. This step requires randomizing the

[1] For reasons that will become clearer in the technical section of the paper, we adopt the convention that predicates map a value to ± 1 instead of $\{0, 1\}$.

[2] Let's assume for now that π is balanced. In the rest of the paper we take into account the possible bias of π.

queries to Ω while still keeping its answers useful to the solution of the underlying CDH problem. This proves somewhat difficult, due to the limited random self-reducibility of the Diffie-Hellman problem.

Randomizing the Problem Representation. Boneh and Shparlinksi in [4] achieved a breakthrough for the Elliptic Curve Diffie-Hellman problem, *i.e.*, the CDH problem defined over the group G of points of an elliptic curve. They were able to prove that the least significant bit of each coordinate of the Diffie-Hellman secret value K is hard-core, when the probability space of the oracle Ω also includes a random choice for the representation of the curve.

More specifically: let p be a prime and let E be an elliptic curve defined over $\mathbb{F}_p$, the finite field with p elements. To represent E we use a short Weierstrass equation $W : y^2 = x^3 + ax + b$, with $a, b \in (\mathbb{F}_p)$ and $4a^3 + 27b^2 \neq 0$. Let $W(E)$ be the set of Weierstrass equations representing E. It is well known that $W(E)$ is defined by the equations W_λ of the form $y^2 = x^3 + \lambda^4 ax + \lambda^6 b$ for $\lambda \in \mathbb{F}_p^\times$. If $Q = (Q_x, Q_y)$ is a point satisfying W then the point $Q_\lambda = (Q_{\lambda,x} = \lambda^2 Q_x, Q_{\lambda,y} = \lambda^3 Q_y)$ satisfies W_λ. Furthermore, the points of E form a group under a certain operation, and the mapping

$$\Phi_\lambda : E \to E \text{ defined as } \Phi_\lambda(Q) = Q_\lambda$$

is an isomorphism with respect to such group operation over E.

Let G be a cyclic subgroup of E generated by a point P. Switching to additive notation for the group operation then the Elliptic Curve CDH (EC-CDH) Assumption says that given W, P, aP, bP it is hard to compute $(ab)P$.

In [4] they prove that if there exists an oracle Ω that works on a random representation of E, *i.e.*, such that

$$\Pr_{\lambda,a,b}[\Omega[\lambda, P, aP, bP] = LSB(\Phi_\lambda((abP)_x))] > 1/2 + \epsilon$$

for a non-negligible value ϵ, then it is possible to solve EC-CDH on any curve (a similar result holds for the y-coordinate of abP).

1.1 Our results

Our main technical contribution is to show that the Boneh-Shparlinski idea of randomizing the representation of the underlying group for the CDH problem, can be also applied to the case of finite fields $\mathbb{F}_{p^2}$.

For a given prime p, there are many different fields $\mathbb{F}_{p^2}$, but they are all isomorphic to each other. Let $h(x) = x^2 + h_1 x + h_0$ be a monic irreducible polynomial of degree 2 in $\mathbb{F}_p$. It is well known that $\mathbb{F}_{p^2}$ is isomorphic to the field $\mathbb{F}_p[x]/(h)$, and therefore elements of $\mathbb{F}_{p^2}$ can be written as linear polynomials: if $g \in \mathbb{F}_{p^2}$ then $g = g_0 + g_1 x$ and addition and multiplication are performed as polynomial operations modulo h. In the following, given $g \in \mathbb{F}_{p^2}$ we denote with g_i the coefficient of the degree-i term.

Let $I_2(p)$ be the set of monic irreducible polynomial of degree 2 in $\mathbb{F}_p$. For $h, \hat{h} \in I_2(p)$ we know that there exists an (easily computable) isomorphism

$$\phi_{h,\hat{h}} : \mathbb{F}_p[x]/(h) \to \mathbb{F}_p[x]/(\hat{h}).$$

Finally, denote with g a generator of the multiplicative group of $\mathbb{F}_{p^2}$ which is known to be cyclic.

Our first attempt was to use the approach from [4] over $\mathbb{F}_{p^2}$. That is, we hoped to prove that given an oracle Ω which, on input random values g^a, g^b and a random description of $\mathbb{F}_{p^2}$,

outputs $LSB([g^{ab}]_i]$, then we can solve the CDH over $\mathbb{F}_{p^2}$. Unfortunately there are several technical complications with directly applying the approach of [4] to the finite field case, one of them being the fact that representations of an elliptic curve are in bijective correspondence with $\mathbb{F}_p$ allowing them to be represented by a single element of $\mathbb{F}_p$. Conversely the representations of $\mathbb{F}_{p^2}$ are in bijective correspondence with $I_2(p)$ which has $\approx p^2/2$ elements.

A new Diffie-Hellman Problem. To solve these technical problems we had to define the following variant of the CDH problem over $\mathbb{F}_{p^2}$: informally we say that the *Partial-CDH* problem is hard in $\mathbb{F}_{p^2}$ if no efficient algorithm given $g, A = g^a, B = g^b \in \mathbb{F}_p^2$. can compute $K = [g^{ab}]_1 \in \mathbb{F}_p$ (*i.e.*, the coefficient of the degree 1 term of g^{ab}).

We note that the Partial-CDH problem is obviously weaker than the regular CDH problem over $\mathbb{F}_{p^2}$, but that it still allows Alice and Bob to agree on a common secret value in $\mathbb{F}_p$, via the traditional Diffie-Hellman protocol.

Our Main Result. Assuming the hardness of the Partial-CDH problem we prove that for a large class of predicates π (described below – it includes every individual bit of K), the bit $\pi(K)$ is unpredictable given g^a, g^b and a random representation of $\mathbb{F}_{p^2}$. More specifically we prove that if there exists an oracle Ω such that for any $h \in I_2(p)$ it holds that

$$\Pr_{\hat{h},a,b}\left[\Omega[\hat{h}, g, g^a, g^b] = \pi([\phi_{h,\hat{h}}(g^{ab}))]_1\right] > 1/2 + \epsilon$$

for a non-negligible value ϵ, then it is possible to solve Partial-CDH on $\mathbb{F}_p[x]/(h)$.

We may define an analogous problem for the general case of $\mathbb{F}_{p^t}$ with any $t > 1$. The Partial-CDH problem is defined as outputting the coefficient of the term of degree $t-1$. However our hard-core results hold only for the quadratic ($\mathbb{F}_{p^2}$) case. See the conclusion (Section 6) for a discussion.

Our Techniques. To achieve our result we divert from the techniques used in [4] in another fundamental way. To prove that the predicate π is hard-core for the Partial-CDH problem in $\mathbb{F}_{p^2}$ we use the list-decoding approach pioneered by Akavia et al. [1] as extended by Duc and Jetchev in [6] to the case of prediction oracles which also take as input a random representation of the underlying group.

We describe the approach in detail in Section 3. For now we just remind the reader that as defined originally in [1] this approach allows one to prove the security of so-called *segment predicates* which include both the most and least significant bits of the input. In [10] the technique was extended to work for any input bit. So the class of predicates P described above includes every individual bit of the input and also segment predicates as defined in [1].

Additional Results. Because the list-decoding approach works for a larger class of predicates and for larger class of functions we obtain two additional results:

1. In the elliptic curve scenario, we are able to extend the [4] result for EC-CDH to any predicate π as above, not just the LSB.

2. For the finite field case we prove that the predicates π are hard-core for a much larger class of conjectured computationally hard problems. Consider a function $f : \mathbb{F}_{p^2} \to S$ for an arbitrary set S. We say that f is a *Partially One Way, Finite Field Based Function* if the following condition hold:

   - f is "independent" of the representation used for $\mathbb{F}_{p^2}$ (see Section 5.2 for a precise definition);

- no efficient algorithm, given $f(x)$ can compute x_1, *i.e.*, the coefficient of the degree 1 term of x.

Then we can prove that if f is a POWF then it is hard to predict $\pi(x_1)$ better than at random (over a random representation of $\mathbb{F}_{p^2}$) when given only $f(x)$.

Interpretation of Our Results. One way to interpret our results is to think of the group representation as part of the input to the computational hard problem (be it a one-way function, or the CDH problem) being used. This means that our results do not apply to the case when the Diffie-Hellman key exchange protocol is performed over a fixed representation of the finite field (or the elliptic curve). Rather it is necessary for Alice and Bob to choose a random representation (an irreducible polynomial for $\mathbb{F}_{p^2}$ or a Weierstrass equation for the curve E) over which to run the protocol.

1.2 Paper Organization

Section 2 reviews some relevant background, particularly the notion of Fourier transform for codes. In Section 3 we cover the list-decoding approach to prove hard-core predicates [1] and its generalization to the case of elliptic curves from [6].

Sections 4 and 5 present our original results. First, as a warm-up we prove that that every bit of the EC-CDH problem is hard-core. Then we present our main result on the bit security of Partial-CDH over finite fields, and its extension to POWFs.

Finally we conclude in Section 6 with some discussion about our results and a list of interesting problems left open by our work.

2 Background

2.1 Fourier Transforms

Let $\mathbb{Z}_n$ denote the additive group of integers modulo n. For any two functions $f, g : \mathbb{Z}_n \to \mathbb{C}$, their *inner product* is defined as $\langle f, g\rangle = 1/n \sum_{x\in\mathbb{Z}_n} f(x)\overline{g(x)}$. Let $\mathbb{C}(\mathbb{Z}_n)$ denote the vector space formed by all functions $f : \mathbb{Z}_n \to \mathbb{C}$. The ℓ_2-norm of f on $\mathbb{C}(\mathbb{Z}_n)$ is defined as $\|f\|_2 = \sqrt{\langle f, f\rangle}$. Let $\mathbb{T}$ denote the set of complex numbers with unit magnitude. A *character* of $\mathbb{Z}_n$ is a homomorphism $\chi : \mathbb{Z}_n \to \mathbb{T}$, such that $\forall_{x,y\in\mathbb{Z}_n}\chi(x + y) = \chi(x)\chi(y)$. These characters are defined by $\chi_\alpha(x) = \omega_n^{\alpha x}$, where $\alpha \in \mathbb{Z}_n$ and $\omega_n = e^{2\pi i/n}$. The set of all characters form a group $\widehat{\mathbb{Z}}_n$. Since the members of $\widehat{\mathbb{Z}}_n$ are orthogonal and $|\widehat{\mathbb{Z}}_n| = |\mathbb{Z}_n|$, they form an orthogonal basis, termed the *Fourier basis*, for $\mathbb{C}(\mathbb{Z}_n)$. The *Fourier transform* $\widehat{f} : \widehat{\mathbb{Z}}_n \to \mathbb{C}$ of f is defined as $\widehat{f}(\chi) = \langle f, \chi\rangle$. The *Fourier expansion* of f is written as $\sum_{\chi\in\widehat{\mathbb{Z}}_n} \widehat{f}(\chi)\chi$. For $\Gamma \subset \widehat{\mathbb{Z}}_n$ the restriction of f to Γ is the function $f_{|\Gamma} : \mathbb{Z}_n \to \mathbb{C}$ defined by $f_{|\Gamma} = \sum_{\chi\in\Gamma} \widehat{f}(\chi)\chi$. The *Fourier coefficients* of f are the coefficients $\widehat{f}(\chi)$ in the Fourier basis $\widehat{\mathbb{Z}}_n$. The *weight* of a Fourier coefficient is denoted by $|\widehat{f}(\chi)|^2$. Definition 2.1 formalizes the notion of *heavy characters*.

Definition 2.1 (τ-heavy characters). *Let $\tau \in \mathbb{R}^+$ be a threshold and $f : \mathbb{Z}_n \to \mathbb{C}$ be an arbitrary function. We say a character $\chi \in \widehat{\mathbb{Z}}_n$ is τ-heavy if the weight of its corresponding Fourier coefficient is at least τ. The set of all such character is denoted by* $\mathsf{Heavy}_\tau(f)$, i.e.,

$$\mathsf{Heavy}_\tau(f) = \{\chi \in \widehat{\mathbb{Z}}_n : |\widehat{f}(\chi)|^2 \geq \tau\}. \qquad \Diamond$$

2.2 Codes and their Properties

In what follows, we report a few useful known definitions [6] and lemmata [1] about codes over $\mathbb{Z}_n$. As in [6], we will regard $\mathbb{Z}_n$-codes as associating an element $x \in \mathbb{Z}_n$ to a $\mathbb{Z}_n$-codeword C_x, which we will in turn see interchangeably as a function $C_x : \mathbb{Z}_n \to \{\pm 1\}$ or as a length-n sequence of $\{\pm 1\}$.

Definition 2.2 (ϵ-concentrated function). *We say a function $f : \mathbb{Z}_n \to \{\pm 1\}$ is* Fourier ϵ-concentrated *if there exist a size $poly(n, 1/\epsilon)$, $\epsilon > 0$, set of characters $\Gamma \subset \widehat{\mathbb{Z}}_n$ such that $\left\| f - f_{|\Gamma} \right\|_2 \leq \epsilon$. We say a function is* Fourier concentrated *if it is ϵ-concentrated for every $\epsilon > 0$.* $\Diamond$

Definition 2.3 (ϵ-concentrated code). *We say a code $\mathcal{C} = \{C_x : \mathbb{Z}_n \to \{\pm 1\}\}$ is* ϵ-concentrated *if all its codewords C_x are Fourier ϵ-concentrated. We say a code is* Fourier concentrated *if it is ϵ-concentrated for every $\epsilon > 0$.* $\Diamond$

Definition 2.4 (Code recoverability). *We say a code $\mathcal{C} = \{C_x : \mathbb{Z}_n \to \{\pm 1\}\}$ is* recoverable *if there exists an algorithm that, given as input a threshold τ and a character $\chi \in \widehat{\mathbb{Z}}_n$, produces a list of all elements x associated with codewords C_x for which χ is a τ-heavy coefficient, that is, $\{x \in \mathbb{Z}_n : \chi \in \mathsf{Heavy}_\tau(C_x)\}$, in time polynomial in $\log n$ and $1/\tau$.* $\Diamond$

The following two results appear in [1]. Lemma 2.5 shows that, in a concentrated code $\mathcal{C}$, any noisy version $\tilde{C}_x$ of codeword C_x share at least one heavy coefficient with C_x. Theorem 2.6 shows that one can efficiently learn all the heavy characters of any function when given query access to it. Therefore having query access to $\tilde{C}_x$ (which in our case is obtained by querying the prediction oracle Ω), one can learn at least one heavy coefficient of C_x, and that if the code is also recoverable, then one can recover x.

Lemma 2.5 ([1], Lem.1). *Let $f, g : \mathbb{Z}_n \to \{\pm 1\}$ such that f is Fourier concentrated and, for some $\epsilon > 0$,*

$$\Pr_{x \in \mathbb{Z}_n} [f(x) = g(x)] \geq \mathsf{maj}_f + \epsilon$$

where maj_f denotes the bias of the function f, i.e., *$\mathsf{maj}_f = \mathsf{max}_{\{b=\pm 1\}} \Pr_{x \in \mathbb{Z}_n} [f(x) = b]$. Then there exist a threshold τ such that $1/\tau$ is polynomial in ϵ and $\log n$, and there exists a character $\chi \neq 0$ heavy for f and g: $\chi \in \mathsf{Heavy}_\tau(f) \wedge \mathsf{Heavy}_\tau(g)$.* $\Diamond$

Theorem 2.6 ([1], Thm.6). *There exists a randomized learning algorithm over $\mathbb{Z}_n$ that, given query access to a function $w : \mathbb{Z}_n \to \{\pm 1\}$, $\tau > 0$ and $0 < \delta < 1$, returns a list of $O(1/\tau)$ characters containing $\mathsf{Heavy}_\tau(w)$ with probability at least $1 - \delta$. The probability is taken over the random coins of the algorithm, whose running time is*

$$\tilde{O}\left(\log(n) \ln^2 \frac{(1/\delta)}{\tau^{5.5}} \right).$$

$\Diamond$

An overview of the above learning algorithm [1] is provided in Appendix A.

3 Hardcore Predicates by List Decoding

In this Section we review the work of Akavia et al. [1] on how to prove that certain predicates are hard-core for a one-way function f using list decoding of a particular error-correcting code. We

also summarize the extensions by Duc and Jetchev [6] to the case of elliptic-curve based one-way functions.

Let $f : \mathbb{Z}_n \to S$ be a one-way function and let $y = f(x)$ for $x \in \mathbb{Z}_n$. Let also $\pi : \mathbb{Z}_n \to \{\pm 1\}$ denote a predicate (with the convention that a 0 bit is encoded as -1). Finally we denote with β_π the bias of the predicate π, *i.e.*, $\beta_\pi = \mathsf{max}_{\{b=\pm 1\}} \Pr_x[\pi(x) = b]$

The goal is to prove that π is a hard-core predicate for the function f. The proof goes as usual by contradiction by assuming that there exists an oracle Ω which when queried on $f(x)$ returns a bit b which is equal to $\pi(x)$ with probability $\beta_\pi + \epsilon$ for a non-negligible ϵ, and then using Ω to invert f, *i.e.*, find x given y.

To achieve this goal, Akavia et al. in [1] define a *multiplication code*

$$\mathcal{C} \;=\; \{\; C_x : \mathbb{Z}_n \to \{\pm 1\} \;\}_{x \in \mathbb{Z}_n} \;\text{ where }\; C_x(\lambda) = \pi(\lambda \cdot x)$$

In order for their proof to work this code needs the following properties:

- *Accessibility:* Given $y = f(x)$, it must be possible to obtain a "noisy" version $\tilde{C}_x$ of the codeword C_x, *i.e.*, one that agrees with the correct one with probability $\beta_\pi + \epsilon$ for a non-negligible ϵ. In [1] this is done by assuming that the one-way function has some homomorphic property, i.e given $y = f(x)$ and $\lambda \in \mathbb{Z}_n$ it is possible to compute $y_\lambda = f(x \cdot \lambda)$ (modular exponentiation has this property). Then by querying Ω on y_λ one gets the desired accessibility property;

- *Concentration:* Every codeword C_x must be a Fourier concentrated function. Remember that according to the definition above this means that for every ϵ there exists a polynomial (in $\log n$ and ϵ^{-1}) set Γ of Fourier characters, such that $||C_x - C_{x,\Gamma}|| \leq \epsilon$ (where $C_{x,\Gamma}$ is the restriction of C_x to the Fourier characters in Γ);

- *Recoverability:* There exists an algorithm that on input a Fourier character χ and a threshold τ, outputs a list L_χ containing all the values $x \in \mathbb{Z}_n$ such that χ is τ-heavy for C_x. The algorithm runs in polynomial (in $\log n$ and τ^{-1}) time, which in particular means that the size of L_χ is also "small";

Concentration and recoverability depends on the choice of the predicate π. In [1] the notion of *segment predicates* is defined and shown to be sufficient for the purpose. Later Morillo and Rafols in [10] prove that any individual input bit yields a concentrated and recoverable code (we review this in Appendix B. Therefore in the following we assume π to be one of such predicates.

If the code $\mathcal{C}$ has the above properties then it is possible to prove that π is a hard-core predicate. Assume we have an oracle Ω which when queried on $f(x)$ returns a bit b which is equal to $\pi(x)$ with probability $\beta_\pi + \epsilon$ where $\epsilon = 1/poly(\ell)$ (where $\ell = |n|$). We need to show how to use Ω to invert f.

Intuitively the inversion works as follows. On input $y = f(x)$, the oracle Ω allows us to access a "noisy" $\tilde{C}_x$ version of C_x, *i.e.*, such that $\Pr_\lambda[C_x(\lambda) = \tilde{C}_x(\lambda)] > \beta_\pi + \epsilon$. By applying Lemma 2.5 we know that there exists a threshold τ which is polynomial in ϵ and at least one Fourier character χ which is τ-heavy for both C_x and $\tilde{C}_x$. Using the learning algorithm described in Theorem 2.6 we obtain a list containing all the τ-heavy Fourier characters for $\tilde{C}_x$; for each such character we use the recovery property to create a polynomial size list of possible pre-images for y which because of Lemma 2.5 must necessarily include x. The correct x can be identified by evaluating the OWF f over all the possible candidates and comparing with y. Details can be found in [1] (in any case, in Sections 4 and 5 we present the details of this algorithm as they apply to our specific results).

3.1 Accessibility via Elliptic Curve Isomorphisms

Taking the [1] result as a starting point, and using techniques first developed in [4], Duc and Jetchev [6] show how to obtain the accessibility property in a different way, when the one-way function is defined over the group G of points of an elliptic curve. Their result does not require the one-way function f to have some homomorphic property; on the other hand it requires the oracle to work over a *random* description of the elliptic curve.

Let p be a prime and let E be an elliptic curve defined over $\mathbb{F}_p$. To represent E we use a short Weirstrass equation $W : y^2 = x^3 + ax + b$, with $a, b \in (\mathbb{F}_p)$ and $4a^3 + 27b^2 \neq 0$. Let $W(E)$ be the set of Weirstrass equations representing E: so $W \in W(E)$. It is well known that $W(E)$ is defined by the equations W_λ of the form $y^2 = x^3 + \lambda^4 ax + \lambda^6 b$ for $\lambda \in \mathbb{F}_p^\times$. If $Q = (Q_x, Q_y)$ is a point satisfying W then the point $Q_\lambda = (Q_{\lambda,x} = \lambda^2 Q_x, Q_{\lambda,y} = \lambda^3 Q_y)$ satisfies W_λ. It is not hard to see that the mapping

$$\Phi_\lambda : E \to E \text{ defined as } \Phi_\lambda(Q) = Q_\lambda$$

is an isomorphism with respect to the group operation over E.

Boneh and Shparlinski were the first to note that this isomorphism gives raise to a natural extension of the prediction oracle Ω, by requiring that the input distribution for Ω also include λ. Following this idea, in [6] the oracle takes as input $f(Q)$ where f is a one-way function defined over the group E, and *also* a value λ (*i.e.*, a representation W_λ of E). The oracle returns a bit b such that $b = \pi(Q_{\lambda,x})$ with probability $\beta_\pi + \epsilon$ (for a non-negligible ϵ) where the probability is taken not only over the choice of Q (and the internal random choices of Ω) but *also* over the choice of $\lambda \in \mathbb{F}_p^\times$.

As defined, the prediction oracle Ω gives noisy access to the *quadratic* codeword $C_Q(\lambda) = \pi(\lambda^2 Q_x)$, which would complicate matters (in particular it makes it hard to prove concentration and recovery, see [6] for a discussion). To apply the techniques of [1] we need noisy access to the multiplication code defined as

$$C_Q : \mathbb{F}_p \to \{\pm 1\} \text{ defined as } C_Q(\lambda) = \pi(\lambda Q_x)$$

Following Boneh and Shparlinski again [4], Duc and Jetchev defined a modified oracle Ω' which queries Ω if λ is a square in $\mathbb{F}_p^\times$, otherwise tosses a β_π-biased coin. It is not hard to see that if Ω had advantage ϵ, then Ω' has advantage $\epsilon/2$ (see [4]).

Using Ω', the generic approach on [1] shows that π is a hardcore predicate for *any* one-way function f defined over E, provided that the output of f does not depend on the Weirstrass equation used to describe E (in other words that the function f is defined over the group of points, irrespective of its representation). Duc and Jetchev call such functions *Elliptic Curve Based OWFs (ECB-OWF)* and discuss the application of their result to bilinear pairings defined over elliptic curves, which are indeed a conjectured example of ECB-OWF.

4 Hardcore Predicates for the Diffie-Hellman Problem over Elliptic Curves

In this Section we show our first original result: if the Diffie-Hellman problem over elliptic curves is hard, then every bit – and every segment predicate – of a secret Diffie-Hellman value is unpredictable. This generalizes the result of Boneh and Shparlinski [4] which holds only for the least significant bit.

For a security parameter ℓ, consider an instance generator $\mathcal{E}$ which on input ℓ outputs E_ℓ an elliptic curve defined over $\mathbb{F}_{p_\ell}$ where p_ℓ is a ℓ-bit prime, such that G_ℓ is a cyclic subgroup of E_ℓ

(under the standard group operation defined over the curve points) generated by a point P_ℓ. In the following we will drop the ℓ suffix, when it is clear from the context. We also use the additive notation for the group operation over E, therefore every point $Q \in G$ can be written as $Q = aP$ for some $a \in [1 \ldots |G|]$.

Assumption 4.1. *We say that the Diffie-Hellman problem over $\mathcal{E}$ is hard if for every polynomial time machine A we have that the following probability*

$$\Pr_{a,b}\left[E_\ell \leftarrow \mathcal{E}(1^\ell)\ ;\ A[\ E_\ell,\ P_\ell,\ aP_\ell,\ bP_\ell\ ] = (ab)P_\ell\ \right]$$

is negligible in ℓ. $\Diamond$

For every point $Q \in E$ we denote with Q_x the x-coordinate of Q. As before we denote with $W(E)$ the set of short Weirstrass equations describing a curve E; recall that each $W \in W(E)$ can be uniquely associated with a $\lambda \in \mathbb{F}_p^\times$ which gives rise to the isomorphism Φ_λ defined in the previous Section.

Let $B_k : \mathbb{F}_p \to \{\pm 1\}$ denote the least significant bit predicate and let β_k be the bias of B_k.

We are now ready to state our first main Theorem. Intuitively it says that under Assumption 4.1 every bit of the binary expansion of the x-coordinate of $(ab)P$ is unpredictable (*e.g.*, pseudorandom) for a random representation of the curve E.

Theorem 4.2. *Under Assumption 4.1, for any polynomial time machine Ω we have that*

$$\Pr\left[\ \Omega[\ \lambda,\ aP,\ bP\ ] = B_k[\Phi_\lambda[(ab)P]_x\right] - \beta_k$$

must be negligible, where the probability is taken over the choices of $\lambda \in \mathbb{F}_p^\times$ and $a, b \in [1 \ldots |G|]$. $\Diamond$

The proof appears in Appendix C. Here we give an intuitive explanation of it. The crucial observation is that the techniques of Duc and Jetchev [6] apply not just to ECB-OWFs but to any computation which "respects" the isomorphism Φ_λ defined by a change in the Weirstrass representation of the curve. The Diffie-Hellman problem is one such problem since applying the Diffie-Hellman transform to $\Phi_\lambda(aP), \Phi_\lambda(bP)$ yields the value $\Phi_\lambda(abP)$ – indeed this is at the basis of the [4] result. Therefore an oracle Ω contradicting Theorem 4.2 on input aP, bP and a curve W_λ defined by a parameter $\lambda \in \mathbb{F}_p^\times$ would output a bit equal to $B_k[\lambda^2[(ab)P]_x \bmod p]$ with non-negligible advantage. This allows us to construct a multiplication code with the required properties and apply the [1] framework to prove that the predicate is hard-core.

We note here that the extension to segment predicates follow from using the concentration and recoverability argument for those predicates presented in [1].

5 Hard-Core Predicates for Diffie-Hellman over Finite Fields

In this Section we state and prove our main result: after defining a natural (though weaker) variation of the Diffie-Hellman problem over finite fields $\mathbb{F}_{p^t}$ for $t > 1$, we prove that in the case of quadratic extensions ($t = 2$), this problem admits a large class of hard-core predicates, including every single bit of one of the coordinates of the secret value.

For a given prime p, there are many different fields $\mathbb{F}_{p^2}$, but they are all isomorphic to each other. Let $h(x) = x^2 + h_1 x + h_0$ be a monic irreducible polynomial of degree 2 in $\mathbb{F}_p$. It is well

known that $\mathbb{F}_{p^2}$ is isomorphic to the field $\mathbb{F}_p[x]/(h)$, and therefore elements of $\mathbb{F}_{p^2}$ can be written as linear polynomials: if $g \in \mathbb{F}_{p^2}$ then $g = g_0 + g_1 x$ and addition and multiplication are performed as polynomial operations modulo h. In the following given $g \in \mathbb{F}_{p^2}$ we denote with g_i the coefficient of the degree-i term.

Let $I_2(p)$ be the set of monic irreducible polynomial of degree 2 in $\mathbb{F}_p$. For $h, \hat{h} \in I_2(p)$ we know that there exists an (easily computable) isomorphism

$$\phi_{h,\hat{h}} : \mathbb{F}_p[x]/(h) \to \mathbb{F}_p[x]/(\hat{h}).$$

Finally denote with g a generator of the multiplicative group of $\mathbb{F}_{p^2}$ which is known to be cyclic.

A new Diffie-Hellman Problem. Denote with g the generator of the multiplicative group of $\mathbb{F}_{p^2}$ which is known to be cyclic. We define the following variant of the CDH problem over $\mathbb{F}_{p^2}$: informally we say that the *Partial-CDH* problem is hard in $\mathbb{F}_{p^2}$ if no efficient algorithm given $g, A = g^a, B = g^b \in \mathbb{F}_p^2$. can compute $K = [g^{ab}]_1 \in \mathbb{F}_p$ (*i.e.*, the coefficient of the degree 1 term of g^{ab}) for any representation of $\mathbb{F}_{p^2}$.

More formally, for a security parameter ℓ, consider an instance generator $\mathcal{F}$ which on input 1^ℓ outputs p_ℓ an ℓ-bit prime. Let $g_{(\ell)}$ be a generator of the multiplicative group of the finite field $\mathbb{F}_{p_\ell^2}$. In the following we will drop the ℓ suffix, when it is clear from the context.

Assumption 5.1. *We say that the Partial Diffie-Hellman problem over $\mathcal{F}$ is hard if for every polynomial time machine A we have that for all $h_\ell \in I_2(p_\ell)$ the following probability*

$$\Pr\left[p_\ell \leftarrow \mathcal{F}(1^\ell)\ ;\ a, b \leftarrow [1 \dots p_\ell^2 - 1]\ ;\ A[\ p_\ell,\ h_\ell,\ g_{(\ell)},\ g_{(\ell)}^a,\ g_{(\ell)}^b\ ] = [g_{(\ell)}^{ab}]_1\ \right]$$

is negligible in ℓ. ◊

Note that A gets as input a representation h_ℓ of the field, and that A's advantage must be negligible for all representations.

We are now ready to state our main Theorem. We show when given an oracle Ω which predicts the kth bit of the degree-1 coefficient of the Diffie-Hellman secret with non-negligible advantage, where the probability is taken over the input pair, *as well as the representation of the field*, then one can efficiently solve the Partial Diffie-Hellman problem with non-negligible probability.

Theorem 5.2. *Under Assumption 5.1, for any polynomial time machine Ω we have that for all $h \in I_2(p)$*

$$\Pr\left[\hat{h} \leftarrow I_2(p)\ ;\ a, b \leftarrow [1 \dots p^2 - 1]\ ;\ \Omega[\ h,\ \hat{h},\ g^a,\ g^b\ ] = B_k[\phi_{h,\hat{h}}(g^{ab})]_1] - \beta_k\right]$$

must be negligible. ◊

The proof of Theorem 5.2 appears in the next Section. Here we give an informal intuition of the proof.

Our goal is to construct a code similar to that of [6], which must be accessible by querying Ω over many different representation of the field. For an element $\alpha \in \mathbb{F}_{p^2}$, and a fixed $h \in I_2(p)$, a natural definition for a codeword is as follows:

$$C_\alpha(\hat{h}) = B_k([\phi_{h,\hat{h}}(\alpha)]_1) \tag{1}$$

This code is accessible using Ω, however it is defined over $I_2(p)$, and it is not immediately seen to be a multiplication code like the ones used in [1, 6]. Note, however, that the predicate B_k is evaluated *only on the first coordinate of* $\phi_{h,\hat{h}}(\alpha)$. In this case it holds that that $[\phi_{h,\hat{h}}(\alpha)]_1 = \lambda\alpha_1$ for some $\lambda \in \mathbb{F}_p^\times$ (see Lemma 5.3 below). Consider then the following multiplication code over $\mathbb{F}_p$: for $\alpha \in \mathbb{F}_{p^2}$ and for $\lambda \in \mathbb{F}_p^\times$, we set

$$C_\alpha(\lambda) = B_k(\lambda\alpha_1) \tag{2}$$

extended with $C_\alpha(0) = -1$. We stress that in light of Lemma 5.3, the above code is conceptually the same as equation (1) in that codewords are obtained by evaluating a predicate over all possible representations of elements. We've simply restricted attention to the degree-1 coordinate. Therefore the multiplication is accessible via Ω and then the proof follows closely the one in [1, 6].

Remark 5.1 (List of candidate solutions). The list-decoding algorithm of [1] appled to the code above returns a polynomial size list of possible candidates for α_1. In our reduction $\alpha = g^{ab}$ and therefore it will be sufficient to output a random element of the list to contradict Assumption 5.1. Differently than in Theorem 4.2 we will not be able to apply Shoup's "self-corrector" in this case to identify the correct solution with high probability, as we have only a single coordinate for g^{ab}.

Remark 5.2 (Segment Predicates). While Theorem 5.2 is stated only for the predicate B_k, it holds for any predicate π such that the corresponding code C_α can be proven to be concentrated and recoverable; in particular the segment predicates defined in [1].

5.1 Proof of Theorem 5.2

We start with a Lemma that gives a simple characterization of the isomorphisms between two different representations of the field $\mathbb{F}_{p^2}$. When describing such maps, it will be convenient for us to view them as matrices in $GL_2(\mathbb{F}_p)$.

Lemma 5.3. *For any $h \in I_2(p)$ there exists a unique function $L_h : \mathbb{F}_p \times \mathbb{F}_p^\times \to I_2(p)$ which takes a pair (a, b) to the polynomial $\hat{h} = L_h(a, b)$ such that the matrix $\left(\begin{smallmatrix} 1 & a \\ 0 & b \end{smallmatrix}\right)$ defines an isomorphism $\mathbb{F}_p[x]/(h) \to \mathbb{F}_p[x]/(\hat{h})$. Moreover, for any $\hat{h} \in I_2(p)$, $L_h^{-1}(\hat{h})$ represents the complete set of isomorphisms from $\mathbb{F}_p[x]/(h) \to \mathbb{F}_p[x]/(\hat{h})$ using the above matrix identification.* $\Diamond$

Proof. First note that any isomorphism of fields must send the unit element to itself (and thus fix the entire base field $\mathbb{F}_p$). Thus, when viewing such an isomorphism as a linear transformation, the first basis element $\binom{1}{0}$ must be fixed, which determines the first column of the matrix as $\binom{1}{0}$. Since clearly we must have $b \neq 0$ if the map is to represent an isomorphism, the completeness would follow immediately, once we establish the existence and uniqueness of the map L_h. We define L_h as follows. For $a, b \in \mathbb{F}_p$ with $b \neq 0$, let $L_h(a, b)(x) = \frac{h(a+bx)}{b^2}$. To make the notation less cumbersome, we'll fix a, b in what follows, and refer to this polynomial more simply as $L_h(x)$. To see that this definition is as desired, note that to specify a homomorphism ϕ from $\mathbb{F}_p[x]/(h)$ to another field K of characteristic p it is both necessary and sufficient to choose $\phi(x) = \overline{x} \in K$ such that $h(\overline{x}) = 0$ in K. The matrix corresponding to (a, b) sends $x \mapsto a + bx$, and indeed, $a + bx$ is a root of h in the ring $\mathbb{F}_p[x]/(L_h)$ by construction. However, it remains to show that $L_h \in I_2(p)$, as well as the uniqueness of L_h. Towards the first goal: it is an elementary fact that since h was irreducible over $\mathbb{F}_p$, so is $h(a + bx)$, and hence L_h. It is easy to verify additionally that L_h is monic, and has degree 2, so that $L_h \in I_2(p)$. Thus, by the above remarks, the mapping defined by $x \mapsto a + bx$ is an isomorphism

$\mathbb{F}_p[x]/(h) \to \mathbb{F}_p[x]/(L_h)$ as desired. The fact that L_h so constructed is unique (within $I_2(p)$) follows easily as well, since if $h(a+bx)$, and hence $L_h(x)$, are elements of an ideal (h') for some other $h' \in I_2(p)$, then L_h, h' are associates, and thus $L_h = h'$ since both are monic. ■

Remark 5.3. We actually know a little more about the distribution; in particular, we have $\left|L_h^{-1}(\hat{h})\right| = 2$ for any $\hat{h} \in I_2(\mathbb{F}_p)$. This follows at once from the fact that every isomorphism has a (unique) matrix representation as above, and that $\mathrm{Gal}(\mathbb{F}_{p^2}/\mathbb{F}_p) \cong \mathbb{Z}_2$ (so that there are precisely two isomorphisms between any two representations $\mathbb{F}_p[x]/(h), \mathbb{F}_p[x]/(\hat{h})$).

Proof Sketch (Theorem 5.2). Suppose that the theorem were false, and that an oracle Ω with an advantage that is not negligible exists. Now consider another oracle Ω' that takes as input a base representation $h \in I_2(p)$, a Diffie-Hellman pair g^a, g^b as well as an element of $\lambda \in \mathbb{F}_p$ (instead of $\hat{h} \in I_2(p)$), which works as follows. The oracle selects $a \xleftarrow{\$} \mathbb{F}_p$, and constructs an isomorphism $\hat{h}$ from the matrix $\left(\begin{smallmatrix} 1 & a \\ 0 & \lambda \end{smallmatrix}\right)$ as described in Lemma 5.3. Ω' then returns the output of $\Omega(h, \hat{h}, g^a, g^b)$. One can then show that

$$\left| \Pr_{\substack{\lambda \leftarrow \mathbb{F}_p \\ a,b \leftarrow [1..p^2-1]}} \left[\Omega'(h, \lambda, g^a, g^b) = B_k(\lambda[g^{ab}]_1) \right] - \beta_k \right|$$

is also not a negligible function. At this point, the proof follows closely that of Theorem 4.2. To begin, observe that we can, for any element $\alpha \in \mathbb{F}_{p^2}$, construct the following encoding of $[\alpha]_1$ in its base polynomial representation as an element of $\mathbb{F}_p[x]/(h))$:

$$C_\alpha : \mathbb{F}_p \to \{\pm 1\} \text{ defined as } C_\alpha(\lambda) = B_k(\lambda[\alpha]_1)$$

where $[\alpha]_1$ is taken under the representation determined by h. The fact that this code is concentrated and recoverable follows immediately from the proof of Theorem 4.2. The argument for accessibility is the same, but with the added simplification that we no longer need to restrict to squares in $\mathbb{F}_p$.

As in Theorem 4.2, we will be able to efficiently construct a list of candidates for $[g^{ab}]_1$. As mentioned, we unfortunately will not be able to apply Shoup's "self-corrector" in this case, as we have only a single coordinate. Nevertheless, we still obtain a contradiction by guessing a random element of the list as the value of $[g^{ab}]_1$, since the list is of polynomial size. ■

5.2 Finite Field Based OWF's

The work of [6] introduces "elliptic curve based one way functions", and goes on to prove interesting hardness results for this *entire class* of functions. Loosely speaking, elliptic curve based OWF's are one way functions which are well defined on isomorphism classes of curves, and do not depend on any specific representation. Similarly, we consider *finite field based OWF's*, which are those that do not depend on the isomorphism class. When considering only prime-order fields $\mathbb{F}_p$, this concept is somewhat trivial, since once you fix a bit representation for integers, there are no non-trivial isomorphisms. However, the situation becomes far more interesting when one considers field extensions. Even with a fixed representation for integers, there are many different representations of even a quadratic extension (see Lemma 5.3). As demonstrated in [6] for the case of elliptic curves, having a one way function which is well defined on many different representations may give rise to a number of hardness results that apply to the entire class of functions. We demonstrate similar results, showing that for quadratic extensions, an efficient oracle that predicts the k-th bit of the

input *over a random representation* will imply an efficient procedure that can "partially" invert the function (i.e. if f is the one way function, given $f(x)$, it computes x_1, the degree-1 coordinate of x).

In order to define a function f on a finite field, we first define the function on a particular "base" representation F. Then, to define f on any other isomorphic copy F', we wish to simply compute $f \circ \psi$, where $\psi : F' \to F$ is an isomorphism. The following definition guarantees that f is well defined on isomorphism classes of finite fields.

Definition 5.4. *Let $F \cong \mathbb{F}_{p^t}$ be a concrete representation of a finite field. A function $f : F \to Y$ is said to be* finite field based *if for any $F' \cong F$ and any two isomorphisms $\psi, \psi' : F' \to F$, we have $f \circ \psi = f \circ \psi'$.* $\Diamond$

Remark 5.4. Note that any function f satisfying Definition 5.4 is actually defined on a quotient space, $F/\sim$, where $\alpha \sim \alpha'$ if and only if α, α' have the same minimal polynomial over $\mathbb{F}_p$. Furthermore, any function which is well defined on $F/\sim$ will satisfy the definition. Thus, an equivalent definition would be to require that $f(\alpha)$ depends only the minimal polynomial of α. (This follows from the fact that the Galois group acts transitively on the roots of irreducible polynomials.)

We now define a natural relaxation of the notion of one-way function over finite fields, where it is assumed to be hard to output the maximal degree coordinate of the input. While this definition makes sense for the general case p^t for $t > 1$, we only consider the case of quadratic extensions.

Consider the instance generator $\mathcal{F}$ which on input a security parameter 1^ℓ, outputs p_ℓ (an ℓ-bit prime), and a function $f_\ell : \mathbb{F}_{p_\ell^2} \to S_\ell$, where S_ℓ is an arbitrary set. We drop the suffix ℓ when clear from the context.

Definition 5.5. *We say that $\mathcal{F}$ is partially one-way if for any efficient algorithm A the following probability*

$$\Pr\left[p_\ell, f_\ell \leftarrow \mathcal{F}(1^\ell)\ ;\ (x \leftarrow \mathbb{F}_{p_\ell^2}\ ;\ A[h_\ell, f_\ell(x)] = x_{t-1}\right]$$

is negligible in ℓ, for all $h_\ell \in I_2(p_\ell)$. $\Diamond$

Again, note that A takes as input a representation of the field, but the probability must be negligible for all representations.

In the case of quadratic extensions, we can obtain results similar to what was shown in [6] for elliptic-curve based OWF. In particular, the existence of a noisy oracle which works with non-negligible probability over the point, as well as the representation of the field, will give rise to an efficient procedure which "partially" inverts f contradicting Definition 5.5. More formally, we have the following.

Theorem 5.6. *Suppose that f is a finite field based, partially one way, function, and fix a base representation $\mathbb{F}_{p^2} = \mathbb{F}_p[x]/(h)$ for some $h \in I_2(p)$. Then for any probabilistic polynomial time machine Ω, it must be that*

$$\left| \Pr_{\substack{\hat{h} \in_U I_2(p) \\ R \in_U F}} \left[\Omega(h, f(R)) = B_k(\phi_{h,\hat{h}}(R)_1)\right] - \beta_k \right|$$

is negligible. $\Diamond$

The proof is a combination of the proofs of Theorems 4.2 and 5.2 and will be presented in the final paper.

Remark 5.5. We note that the Diffie-Hellman problem does *not* satisfy the above definition: apart from the fact that the domain is actually two (or three) field elements, the value g^{ab} is not independent of the representation. However, if one modifies the usual Diffie-Hellman problem to report the minimal polynomial of g^{ab} instead, then the definition is satisfied (with the aforementioned caveat regarding the input coming from a product space). We also remark that the minimal polynomial is efficiently computable; see for example the work of [13]. Finally, we note that the equivalence classes under $\sim$ are small – for $\mathbb{F}_{p^t}$, each class has size t. Since t is usually a small constant (in our case, it is 2), the aforementioned conversion in which one "throws away" some information by only considering the minimal polynomial will not affect the problem's computational character.

6 Conclusions and Future Work

We presented a relaxed variant of the Diffie-Hellman problem over finite fields of the form $\mathbb{F}_{p^t}$ for $t > 1$ and proved that for the case of quadratic extensions $\mathbb{F}_{p^2}$, this problem admits several hard-core predicates (including every single bit of one coordinate of the secret Diffie-Hellman value) over a random representation of the field. These are the first results known for hard-core predicates for the CDH problem over finite fields. We extended this result to a larger class of computationally hard problems (which we called partially one-way, finite field based functions) over such finite fields.

We also proved that the same class of predicates is hard-core for the Elliptic Curve Diffie-Hellman, over a random representation of the underlying elliptic curve, thereby extending the Boneh-Shparlinski result [4] which worked only for the least significant bit.

Our results can be interpreted as "augmenting" the input to the computational hard problem (being it a one-way function, or the CDH problem) with a random description of the underlying group being used.

Our work leaves several open questions. Perhaps the most natural is to extend the results to $\mathbb{F}_{p^t}$ for $t > 2$. In the case of $t = 2$, the isomorphisms from one representation to another amounted in some sense, to a linear change of variables: $x \mapsto a + bx$. This made the set of isomorphisms between representations easy to analyze, and enabled us to show that when restricting attention to the coefficient of x, each of these maps acts by translation for some $\lambda \in \mathbb{F}_p^\times$. For $t > 2$, this is *not* the case, and thus our original techniques must be augmented somehow. Perhaps one can find a large (enough) number of representations for which the isomorphisms have the required properties as a linear map.

Other natural questions include the study of the hardness of the Partial-CDH problem in $\mathbb{F}_{p^t}$ for $t > 1$. While it seems quite a reasonable assumption to make, the ultimate goal would be to reduce it to the "full" CDH over another platform. In particular is it possible to reduce Partial-CDH over $\mathbb{F}_{p^t}$ to the regular CDH problem over $\mathbb{F}_p$? A related question is if we can use the hardness of Partial-DH over, say, $\mathbb{F}_{p^2}$ to prove the unpredictability of a predicate for the traditional CDH problem over $\mathbb{F}_p$.

Finally it is our hope that the techniques presented in this paper could eventually lead to the proof that CDH over $\mathbb{F}_p$ does have a (deterministic) hard-core predicate.

Acknowledgments.

The authors would like to thank Adi Akavia and Dimitar Jetchev for several useful discussions and clarifications.

Nelly Fazio's research is sponsored in part by NSF CAREER award #1253927, and by PSC-CUNY award 64578-00 42 (jointly funded by The Professional Staff Congress and The City University of New York). Nelly Fazio and William E. Skeith III are sponsored in part by NSF award #1117675. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. Nelly Fazio and Rosario Gennaro are supported in part by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] A. Akavia, S. Goldwasser, and S. Safra. Proving hard-core predicates using list decoding. In *IEEE Symposium on Foundations of Computer Science—FOCS*, pages 146–157, 2003.

[2] W. Alexi, B. Chor, O. Goldreich, and C. Schnorr. Rsa and rabin functions: Certain parts are as hard as the whole. *SIAM Journal on Computing*, 17(2):194–209, 1988.

[3] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal on Computing*, 13(4):850–864, 1984.

[4] D. Boneh and I. E. Shparlinski. On the unpredictability of bits of the elliptic curve diffie-hellman scheme. In *Advances in Cryptology—CRYPTO*, pages 201–212, 2001.

[5] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[6] A. Duc and D. Jetchev. Hardness of computing individual bits for one-way functions on elliptic curves. In *Advances in Cryptology—CRYPTO*, pages 832–849, 2012.

[7] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *ACM Symposium on Theory of Computing—STOC*, pages 25–32, 1989.

[8] E. Kushilevitz and Y. Mansour. Learning decision trees using the fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993.

[9] Y. Mansour. An $O(n^{\log\log n})$ learning algorithm for DNF under the uniform distribution. *Journal of Computer and System Sciences*, 50(3):543–550, 1995.

[10] P. Morillo and C. Ràfols. The security of all bits using list decoding. In *Public Key Cryptography—PKC*, pages 15–33, 2009.

[11] M. Näslund. All bits in ax + b mod p are hard. In *Advances in Cryptology—CRYPTO*, pages 114–128, 1996.

[12] V. Shoup. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology—EUROCRYPT*, pages 256–266, 1997.

[13] V. Shoup. Efficient computation of minimal polynomials in algebraic extensions of finite fields. In *Proceedings of the 1999 international symposium on Symbolic and algebraic computation*, pages 53–58. ACM, 1999.

A Learning Heavy Fourier Coefficients

In [8], Kushilevitz and Mansour presented an algorithm to compute the heavy Fourier coefficients of any given function $f : \{0,1\}^k \to \{\pm 1\}$. Their algorithm proceeds in k stages, where each stage i maintains a list of prefixes $\alpha^i = \alpha_1 \dots \alpha_i$ that can be extended into elements α whose associated character χ_α is heavy. To progress from a stage to the next, the algorithm employs a test to determine for which choice of $\alpha_{i+1} \in \{0,1\}$ it is the case that $\alpha^i \alpha_{i+1}$ can still be extended into a heavy character. This approach clearly would not work for functions over $\mathbb{Z}_n$ (even for $k = 1$), since in that case iterating through all the possible $\alpha_1 \in \mathbb{Z}_n$ would require time exponential in $\log n$. For the case when n is a power of 2, Mansour [9] proposed an extension to the techniques of [8] that, given query access to a polynomial P, computes the heavy coefficients of P.

A solution for boolean $\mathbb{Z}_n$-functions for arbitrary n was devised in [1]. We report below a short overview of this algorithm, and refer the reader to [1] for the full details and proof of correctness. Say that a contiguous interval $J \subset \mathbb{Z}_n$ is "far from heavy" if even stretching J by a small extent, the aggregate weight of the covered coefficients falls below the threshold τ. At a high level, the algorithm of [1] results from three main ingredients: (1) a binary search-like *splitting* algorithm that, starting from a partition of $\mathbb{Z}_n$ into contiguous intervals, proceeds iteratively by splitting each interval in two halves and discarding those that are far from heavy, and eventually outputs a "small" collection of singleton that contains (with high probability) all the heavy coefficients of f; (2) an efficient *distinguishing* procedure that efficiently determines (with high probability) if a given interval in the collection is far from heavy; and (3) a *sieving* process that, given in input the (somewhat short) list of singletons produced by the splitting algorithm, singles out (with high probability) the heavy coefficients in the list.

Splitting algorithm. Given a threshold τ, this algorithm initially splits $\mathbb{Z}_n$ into a collection $Coll_0$ of intervals of size $s_0 = n/\ell_0$, where $\ell_0 = O(1/\tau)$. At each round i, the splitting algorithm starts with a collection $Coll_i$ of intervals, each of size $s_i = n/\ell_i$ (with $\ell_i = 2\ell_{i-1}$ for $i \geq 1$) and produces a collection $Coll_{i+1}$ of length-s_{i+1} intervals. In particular, each interval $J_j^{\ell_i} \in Coll_i$ is split into two sub-intervals, and the distinguishing procedure is run on each of them to determine whether it is "far from" containing a heavy coefficient and can thus be safely discarded, or it may plausibly contain a heavy coefficient, and should therefore be inserted in the next-round collection $Coll_{i+1}$. After $O(\log(n/\ell_0))$ rounds, the resulting collection will consist of a "somewhat short" list of singletons, which is the output of the splitting algorithm. By the property of the distinguishing procedure, this list will include, with high probability, all the heavy coefficients for f.

Distinguishing procedure. This procedure receives in input an interval $J \subset \mathbb{Z}_n$, and seeks to estimate the aggregate weight of the coefficients that fall within J, *i.e.*, $\mathbf{wt}(J) \doteq \sum_{\alpha \in J} |\hat{f}(\chi_\alpha)|^2$. Since computing this exactly is inefficient, one settles for computing $\mathbf{est}(J) \doteq \sum_{\alpha \in \mathbb{Z}_n} c_\alpha |\hat{f}(\chi_\alpha)|^2$ for some "J-decaying" coefficients c_α that are close to 0 for $\alpha \notin J$ and close to 1 for $\alpha \in J$. Next, one seeks a function h whose Fourier coefficients' weight match the c_α coefficients, *i.e.*, $|\hat{h}(\chi_\alpha)|^2 = c_\alpha$. In this way, one can compute $\mathbf{est}(J)$ as $\sum_{\alpha \in \mathbb{Z}_n} |\hat{h}(\chi_\alpha)\hat{f}(\chi_\alpha)|^2 = \sum_{\alpha \in \mathbb{Z}_n} |\widehat{h \star f}(\chi_\alpha)|^2 = \|\widehat{h \star f}\|^2$, where $\star$ is the convolution operator. Since $\|\widehat{h \star f}\|^2 = \|h \star f\|^2$ by Parseval's identity, we see that

$\mathbf{est}(J) = \|h \star f\|^2$, for a suitable choice of h, which turns out to be the character χ_{-midJ} opposite to the midpoint $midJ$ of the interval J. At this point, one can get a handle on $\mathbf{est}(J)$ based on the identity $\|h \star f\|^2 = \mathbf{E}_{x \in \mathbb{Z}_n}[\mathbf{E}_{y \in \mathbb{Z}_n}[h(y)f(x-y)]$, which can be efficiently estimated via Chernoff-like bounds. (Refer to §7.2.3 in [1] for more details.)

Sieving process. This process receives in input a good collection of singletons output by the splitting algorithm. To further reduce the size of this list, the sieving process estimates (up to an error of $\tau/4$) the weight of each singleton, and discards all those of estimated weight is less than $3\tau/4$. This produces a shorter list (of length $O(1/\tau)$) that contains all heavy characters with probability at least $1 - \delta$.

Below we report the pseudocode description of the splitting algorithm (Algorithm 1) and the distinguish procedure (Algorithm 2) as reported in [6].

Algorithm 1: Splitting algorithm

Input: A noisy codeword $f \in \mathbb{Z}_n \to \{\pm 1\}$, $\tau > 0$ and $< \delta < 1$
Output: A list L of singletons such that $|L| = O(1/\tau)$ and $\mathsf{Heavy}_\tau(f) \subset L$ with probability $1 - \delta$ over the random coin of the algorithm

$\ell_0 = 1/\sqrt{\tau}$
$Coll_0 = \{[0, \lfloor n/\ell_0 \rfloor - 1], [\lfloor n/\ell_0 \rfloor, 2\lfloor n/\ell_0 \rfloor - 1], \ldots, [(\ell_0 - 1)\lfloor n/\ell_0 \rfloor, \ell_0 \lfloor n/\ell_0 \rfloor - 1], [\ell_0 \lfloor n/\ell_0 \rfloor, n-1]\}$
for $i = 1, \ldots, \log_2 n - 1$ **do**
    $Coll_i = \emptyset$; $\ell_i = 2\ell_{i-1}$
    for $J_j^{\ell_i} \in Coll_i$ **do**
        Parse $J_j^{\ell_i}$ as $[a, b]$
        Run the distinguishing procedure on $[a, \frac{a+b}{2} - 1]$
        If the decision id **yes**, $Coll_{i+1} \leftarrow Coll_{i+1} \cup [a, \frac{a+b}{2} - 1]$
        Run the distinguishing procedure on $[\frac{a+b}{2}, b]$
        If the decision is **yes**, $Coll_{i+1} \leftarrow Coll_{i+1} \cup [\frac{a+b}{2}, b]$
    end
end
return $\alpha : [\alpha, \alpha] \in Coll_{\lfloor \log_2 n \rfloor}$

Algorithm 2: Distinguishing procedure

Input: A function $f \in \mathbb{Z}_n \to \{\pm 1\}$ and $J_j^\ell = [a, b]$ with $b - a = n/2^\ell$
Output: **yes** or **no** determining whether J_j^ℓ should be kept or discarded (all intervals containing τ-heavy Fourier coefficients should be kept)

Let $\epsilon \leftarrow \delta\tau^{1.5}/\log n$
Let $m_2 \leftarrow \Theta(\ln(1/\epsilon)/\tau^2)$ and $m_1 \leftarrow \Theta(\ln(m_2/\epsilon)/\tau^2)$
Select $x_1, \ldots, x_{m_2}$ at random from $\mathbb{Z}_n$
for $r = 1, \ldots, m_2$ **do**
    Choose $y_1^{(r)}, \ldots, y_{m_1}^{(r)} \in_U \{0, \ldots, \ell - 1\}$ (independently and uniformly at random)
end
Compute $\mathbf{est}(J_j^\ell) \leftarrow \frac{1}{m_2} \sum_{r=1}^{m_2} \frac{1}{m_1} \sum_{s=1}^{m_1} \chi_{-midJ}(y_s^{(r)}) f(x_r - y_s^{(r)})$, where $midJ = \lfloor \frac{a+b}{2} \rfloor$
return ***yes*** *if* $\mathbf{est}(J_j^\ell) \geq \tau/8$ *or* ***no*** *otherwise*

B Fourier Concentration of $C_Q(\lambda)$ for Linear λ

It is difficult to prove the Fourier concentration of the elliptic curve multiplication code $C_Q(\lambda) = B_k(\lambda Q_x)$ when λ is quadratic. However, when λ is linear, the Fourier concentration of C_Q can easily be proven as Duc and Jetchev shows in [6]. We provide the intuition of this proof in this section.

First, the authors note that if C_Q is ϵ-concentrated in $\Gamma = \{\chi_\alpha\}$, where $\alpha \in \mathbb{F}_p$, then B_k is ϵ-concentrated in the set $\Gamma_Q = \{\chi_\beta : \beta \equiv \alpha \cdot Q_x \bmod p\}$. This follows from the fact that C_Q is a multiplicative code. Therefore, proving the the Fourier concentration of C_Q translates to proving the Fourier concentration of B_k. So, they analyze the Fourier coefficients of $B_k : \mathbb{F}_p \to \{\pm 1\}$.

In order to simplify the computation, the Duc and Jetchev borrow an important result from Morillo and Ràfols [10]. In [10], the authors notice that if x is an integer then $B_k(x) + B_k(x + 2^k)$ is a constant function. Although this fails when x is an integer mod p, one can still have reasonable control over the coefficients. Formally, let $g(x)$ be defined as

$$g(x) = \frac{B_k(x) + B_k(x + 2^k)}{2}.$$

Then, the Fourier transform of $B_k(x)$ relates to the Fourier transform of $g(x)$ as

$$\widehat{g}(\alpha) = \frac{\omega_p^{2^k \alpha} + 1}{2} \widehat{B_k}(\alpha), \quad \alpha \in \mathbb{Z}_p.$$

Next, by using the computations from [10], Duc and Jetchev obtain the following characterization of the asymptotic upper-bound of $|\widehat{B_k}(\alpha)|$. We refer the reader to [6] for the detailed analysis.

$$|\widehat{B_k}(\alpha)|^2 < \mathcal{O}\left(\frac{1}{\lambda_{\alpha,k}^2 \mu_{\alpha,k}^2}\right).$$

Now, one can easily pick the heavy Fourier coefficients and hence show that B_k is Fourier concentrated as follows: 1) pick $(\lambda_{\alpha,k}, \mu_{\alpha,k})$ in the box $[0, 1/\tau] \times [0, 1/\tau]$ for some $0 < \tau < 1$ such that $1/\tau = poly(\log p)$, 2) show that the function $f_u(\lambda) = B_k(\lambda u \bmod p)$ is τ-concentrated for all $u \in \mathbb{F}_p^\times$.

C Proof of Theorem 4.2

Proof Sketch. Assume for the sake of contradiction that there exists an oracle Ω such that the quantity

$$\Pr\left[\ \Omega[\ \lambda,\ aP,\ bP,\ ] = B_k[\Phi_\lambda[(ab)P]_x]\right] - \beta_k$$

is larger than a non-negligible quantity ϵ.

From this oracle we build a modified oracle Ω' which queries Ω if λ is a square in $\mathbb{F}_p^\times$, otherwise tosses a β_k-biased coin. It is not hard to see [4] that if Ω had advantage ϵ, then Ω' has advantage $\epsilon/2$. We now show how to use Ω' to break Assumption 4.1.

Let E be an elliptic curve defined by an equation $W \in W(E)$ over $\mathbb{F}_p$ and let G be a cyclic subgroup of $|E|$ generated by the point P. Given aP, bP we want to compute $Q = (ab)P$ with non-negligible probability.

Consider the codeword

$$C_Q : \mathbb{F}_p \to \{\pm 1\} \ \text{ defined as } \ C_Q(\lambda) = B_k(\lambda Q_x)$$

The following properties hold for C_Q

- *Accessibility* The oracle Ω' gives us access to $\tilde{C}_Q$ a noisy version of this codeword defined as

$$\tilde{C}_Q = \Omega'(\lambda, aP, bP)$$

Because Ω' has advantage $\epsilon/2$ we know that $Prob_\lambda[C_Q(\lambda) = \tilde{C}_Q(\lambda)] > \beta_k + \epsilon/2$.

- *Concentration* The codeword C_Q is a Fourier concentrated function. Indeed for a threshold τ the τ-heavy characters of C_Q must belong to the set

$$\Gamma_{Q,\tau} = \{\chi_\beta : \beta = \alpha Q_x \bmod p \text{ for } \alpha \in \Gamma_\tau\}$$

where Γ_τ is a set of size $O(\tau^{-2})$ containing the τ-heavy coefficients of the function B_k. We refer the reader to [6, 10] for a proof of this statement and also the definition of Γ_τ which shows that the elements of Γ_τ can be easily enumerated. See also Appendix B.

- *Recoverability* Given a Fourier character χ_β we want to find a set L_β containing all the points Q such that χ_β is τ-heavy for C_Q. If χ_β is τ-heavy for C_Q then $\chi_\beta \in \Gamma_{Q,\tau}$ and therefore $Q_x = \beta\alpha^{-1} \bmod p$ for $\alpha \in \Gamma_\tau$, therefore

$$L_\beta = \{Q : Q_x = \beta\alpha^{-1} \bmod p \text{ for } \alpha \in \Gamma_\tau\}$$

By applying Lemma 2.5 we know that there exists a threshold τ which is polynomial in ϵ and at least one Fourier character χ which is τ-heavy for both C_Q and $\tilde{C}_Q$.

We then invoke Theorem 2.6 and use the [1] learning algorithm to learn a polynomial-size list L_Q of all the τ-heavy Fourier characters for $\tilde{C}_Q$. For each such character $\chi_\beta \in L_Q$ we use the recovery property to create a polynomial size list L_β of possible values for Q. Let $L = \cup_{\chi_\beta \in L_Q} L_\beta$; this is a polynomial-size set and because of Lemma 2.5 it must necessarily include Q.

More specifically, on input E, P, aP, bP and with access to Ω, the following algorithm produces a polynomial size list of points in E which is guaranteed to contain Q with probability $1 - \delta$:

1. Let τ be the threshold determined by Lemma 2.5 ; note that τ^{-1} is polynomial in $\ell = |p|$, since ϵ^{-1} is.

2. Learn the polynomial-size set L_Q containing all τ-heavy Fourier characters of $\tilde{C}_Q$, using the learning algorithm in [1], which is correct with probability $1 - \delta$ This algorithms uses oracle Ω' to obtain the required query access to $\tilde{C}_x$. By applying Lemma 2.5 we know that there exists at least one Fourier character χ which is τ-heavy for C_Q and $\chi \in L_Q$.

3. Use the recovery algorithm to construct a polynomial-size list of candidates values for Q. For each $\chi_\beta \in L_Q$ let

$$L_\beta = \{R \in E : \chi_\beta \text{ is } \tau\text{-heavy for } C_R\} = \{R \in E : R_x = \beta\alpha^{-1} \bmod p \text{ for } \alpha \in \Gamma\}$$

Let $L = \cup_{\chi_\beta \in L_Q} L_\beta$. Note that L's size is polynomial in ℓ and that $Q \in L$ with probability $1 - \delta$.

The algorithm runs in polynomial-time, since the learning algorithm of [1] is efficient and all the enumerations in the algorithm are over lists of polynomial size.

At this point to contradict Assumption 4.1 it would be sufficient to choose a random point in L. The probability to select the correct point Q is $1/|L|$ and therefore the algorithm outputs the correct Q with probability $(1-\delta)/|L|$ which is non-negligible since $|L|$ is polynomial-size.

Another option is to use the above algorithm as a subroutine in Shoup's "self-corrector" for the Diffie-Hellman problem (Theorem 7 in [12]). Shoup shows how an algorithm A that runs in time T_A and produces a list of m points, which contains the correct Diffie-Hellman value with probability $> 7/8$ can be easily converted into an algorithm B that output only the correct Diffie-Hellman value with overwhelming probability and runs in time $T_A\ell + poly(m, \ell)$. ■