# Hard-Core Predicates for a Diffie-Hellman Problem over Finite Fields

N. Fazio[1,2] R. Gennaro[1,2] I.M. Perera[2] W.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

CRYPTO 2013

# Our Results

**Result 1: Bit-security of Diffie-Hellman over Elliptic Curves**

If Diffie-Hellman (DH) problem over elliptic curves (EC) is hard, every bit of the secret Diffie-Hellman value is unpredictable.

**Result 2: Bit-security of (Partial) DH over Finite Fields**

Extension of Result 1 to (partial) DH problem over the finite field $\mathbb{F}_{p^2}$.

**Result 3: Bit-security of Finite Field-based Partial OWF**

Every bit of the input to a finite field-based partial one-way function (FFB-POWF) is unpredictable.

# Our Results

**Result 1: Bit-security of Diffie-Hellman over Elliptic Curves**

If Diffie-Hellman (DH) problem over elliptic curves (EC) is hard, every bit of the secret Diffie-Hellman value is unpredictable.

**Result 2: Bit-security of (Partial) DH over Finite Fields**

Extension of Result 1 to (partial) DH problem over the finite field $\mathbb{F}_{p^2}$.

**Result 3: Bit-security of Finite Field-based Partial OWF**

Every bit of the input to a finite field-based partial one-way function (FFB-POWF) is unpredictable.

# Our Results

**Result 1: Bit-security of Diffie-Hellman over Elliptic Curves**

If Diffie-Hellman (DH) problem over elliptic curves (EC) is hard, every bit of the secret Diffie-Hellman value is unpredictable.

**Result 2: Bit-security of (Partial) DH over Finite Fields**

Extension of Result 1 to (partial) DH problem over the finite field $\mathbb{F}_{p^2}$.

**Result 3: Bit-security of Finite Field-based Partial OWF**

Every bit of the input to a finite field-based partial one-way function (FFB-POWF) is unpredictable.

# One-way Functions and Hard-Core Predicates

## One-way Function

- $f : \mathcal{X} \to \mathcal{Y}$ is a one-way function (OWF) iff
  1. It is easy to compute $f(x)$ given $x \in \mathcal{X}$
  2. It is hard to invert, i.e.,

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_x[f(z) = y \mid y = f(x),\, z = \mathcal{A}(y)] \le \text{negl.}$$

## Hard-Core Predicate for OWF $f$

- $P : \mathcal{X} \to \{0, 1\}$ is a hard-core predicate for $f$ iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_x[\mathcal{A}(f(x)) = P(x)] \le \frac{1}{2} + \text{negl.}$$

# One-way Functions and Hard-Core Predicates

## One-way Function

- $f : \mathcal{X} \to \mathcal{Y}$ is a one-way function (OWF) iff
  1. It is easy to compute $f(x)$ given $x \in \mathcal{X}$
  2. It is hard to invert, i.e.,

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_x[f(z) = y \mid y = f(x),\, z = \mathcal{A}(y)] \leq \text{negl.}$$

## Hard-Core Predicate for OWF $f$

- $P : \mathcal{X} \to \{0,1\}$ is a hard-core predicate for $f$ iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_x[\mathcal{A}(f(x)) = P(x)] \leq \frac{1}{2} + \text{negl.}$$

# Diffie-Hellman Problem and its Hard-Core Predicates

## DH Problem

- DH is hard in a group $\mathbb{G} = \langle g \rangle$ iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_{a,b}\left[\mathcal{A}(\mathbb{G}, g, g^a, g^b) = g^{ab}\right] \leq \text{negl.}$$

## Hard-Core Predicate for DH

- $P : \mathbb{G} \to \{0,1\}$ is a hard-core predicate for DH problem over $\mathbb{G}$ iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_{a,b}\left[\mathcal{A}(\mathbb{G}, g, g^a, g^b) = P(g^{ab})\right] \leq \frac{1}{2} + \text{negl.}$$

# Diffie-Hellman Problem and its Hard-Core Predicates

## DH Problem

- DH is hard in a group $\mathbb{G} = \langle g \rangle$ iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_{a,b}\left[\mathcal{A}(\mathbb{G}, g, g^a, g^b) = g^{ab}\right] \leq \text{negl.}$$

## Hard-Core Predicate for DH

- $P : \mathbb{G} \to \{0, 1\}$ is a hard-core predicate for DH problem over $\mathbb{G}$ iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_{a,b}\left[\mathcal{A}(\mathbb{G}, g, g^a, g^b) = P(g^{ab})\right] \leq \frac{1}{2} + \text{negl.}$$

# Why We Need Hard-Core Predicates

- $f(x), (g^a, g^b)$ could reveal a lot of partial information about $x, g^{ab}$ but not about their hard-core predicates
- Hard-core predicates can be used where *pseudo-randomness* is needed
  - Key exchange, encryption, pseudo-random generators, etc.

# Why We Need Hard-Core Predicates

- $f(x), (g^a, g^b)$ could reveal a lot of partial information about $x, g^{ab}$ but not about their hard-core predicates
- Hard-core predicates can be used where *pseudo-randomness* is needed
  - Key exchange, encryption, pseudo-random generators, etc.

# Known Hard-Core Predicates

## Specific Hard-Core Predicates

- MSB of DL over $\mathbb{F}_p$ is hard-core - *Blum and Micali (1984)*
- LSB of RSA is hard-core - *Alexi et al. (1988)*
- Each bit of DL modulo Blum integer is hard-core - *Håstad et al. (1993)*
- Every bit of RSA is hard-core - *Håstad and Näslund (1998)*
- LSB of EC-based DH secret is hard-core (in a modified model) - *Boneh and Shparlinski (2001)*

## General Hard-Core Predicates

- Every OWF $f$ can be modified to obtain a OWF $g$ having a specific hard-core bit - *Goldreich and Levin (1989)*

# Known Hard-Core Predicates

## Specific Hard-Core Predicates

- MSB of DL over $\mathbb{F}_p$ is hard-core - *Blum and Micali (1984)*
- LSB of RSA is hard-core - *Alexi et al. (1988)*
- Each bit of DL modulo Blum integer is hard-core - *Håstad et al. (1993)*
- Every bit of RSA is hard-core - *Håstad and Näslund (1998)*
- LSB of EC-based DH secret is hard-core (in a modified model) - *Boneh and Shparlinski (2001)*

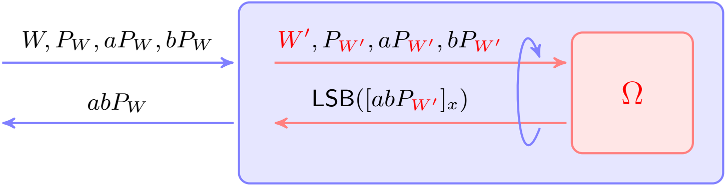
## General Hard-Core Predicates

- Every OWF $f$ can be modified to obtain a OWF $g$ having a specific hard-core bit - *Goldreich and Levin (1989)*

# The Result of Boneh and Shparlinski (2001)

## Highlights

- EC-based DH is hard $\rightarrow$ LSB of DH secret is hard-core
- Given $\Omega$ predicting LSB of DH secret over a random representation of the curve, recover the entire DH secret



# The Result of Akavia et al. (2003)

## Highlights

- A framework for proving that a predicate $\pi$ is hard-core for a OWF $f$

## Approach

1. Define a multiplication code

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

2. Use the oracle that predicts $\pi(x)$ from $f(x)$ to construct a noisy version of $C_x$
3. Use list-decoding techniques to find a small set of candidates for $x$

# The Result of Akavia et al. (2003)

## Highlights

- A framework for proving that a predicate $\pi$ is hard-core for a OWF $f$

## Approach

1. Define a multiplication code

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

2. Use the oracle that predicts $\pi(x)$ from $f(x)$ to construct a noisy version of $C_x$
3. Use list-decoding techniques to find a small set of candidates for $x$

# The Result of Akavia et al. (2003)

## Highlights

- A framework for proving that a predicate $\pi$ is hard-core for a OWF $f$

## Approach

1. Define a multiplication code

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

2. Use the oracle that predicts $\pi(x)$ from $f(x)$ to construct a noisy version of $C_x$
3. Use list-decoding techniques to find a small set of candidates for $x$

# The Result of Akavia et al. (2003)

## Highlights

- A framework for proving that a predicate $\pi$ is hard-core for a OWF $f$

## Approach

1. Define a multiplication code

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

2. Use the oracle that predicts $\pi(x)$ from $f(x)$ to construct a noisy version of $C_x$
3. Use list-decoding techniques to find a small set of candidates for $x$

# More details on Akavia et al. (2003)

- $\mathcal{C} = \{C_x : \mathbb{Z}_n \to \{\pm 1\} \mid x \in \mathbb{Z}_n\}$ where $C_x(\lambda) = \pi(\lambda \cdot x)$
- An oracle $\Omega$ predicting $\pi(x)$ given $f(x)$

It should be shown that $\mathcal{C}$ meets the following properties

Accessible Given $f(x)$, it is possible to get a "noisy" $\tilde{C}_x$ of $C_x$

- They assume $f$ is homomorphic
  i.e., given $\lambda$ and $f(x)$ it is possible to compute $f(\lambda x)$
- Noisy access to $C_x(\lambda)$ is obtained by querying the oracle on $f(\lambda x)$

Concentrated Every codeword $C_x$ is a Fourier concentrated function

Recoverable Given a frequency (character) $\chi$, $\exists$ a poly time algorithm that finds all values $x$ such that $\chi$ is "heavy" for $C_x$

Fourier-Learnable It is possible to efficiently learn all the heavy coefficients of $C_x$ given query access to its noisy version.

# More details on Akavia et al. (2003)

- $\mathcal{C} = \{C_x : \mathbb{Z}_n \to \{\pm 1\} \mid x \in \mathbb{Z}_n\}$ where $C_x(\lambda) = \pi(\lambda \cdot x)$
- An oracle $\Omega$ predicting $\pi(x)$ given $f(x)$

It should be shown that $\mathcal{C}$ meets the following properties

**Accessible** Given $f(x)$, it is possible to get a "noisy" $\tilde{C}_x$ of $C_x$

- They assume $f$ is homomorphic
  i.e., given $\lambda$ and $f(x)$ it is possible to compute $f(\lambda x)$
- Noisy access to $C_x(\lambda)$ is obtained by querying the oracle on $f(\lambda x)$

**Concentrated** Every codeword $C_x$ is a Fourier concentrated function

**Recoverable** Given a frequency (character) $\chi$, $\exists$ a poly time algorithm that finds all values $x$ such that $\chi$ is "heavy" for $C_x$

**Fourier-Learnable** It is possible to efficiently learn all the heavy coefficients of $C_x$ given query access to its noisy version.

# More details on Akavia et al. (2003)

- $\mathcal{C} = \{C_x : \mathbb{Z}_n \to \{\pm 1\} \mid x \in \mathbb{Z}_n\}$ where $C_x(\lambda) = \pi(\lambda \cdot x)$
- An oracle $\Omega$ predicting $\pi(x)$ given $f(x)$

It should be shown that $\mathcal{C}$ meets the following properties

**Accessible** Given $f(x)$, it is possible to get a "noisy" $\tilde{C}_x$ of $C_x$

- They assume $f$ is homomorphic
  i.e., given $\lambda$ and $f(x)$ it is possible to compute $f(\lambda x)$
- Noisy access to $C_x(\lambda)$ is obtained by querying the oracle on $f(\lambda x)$

**Concentrated** Every codeword $C_x$ is a Fourier concentrated function

**Recoverable** Given a frequency (character) $\chi$, $\exists$ a poly time algorithm that finds all values $x$ such that $\chi$ is "heavy" for $C_x$

**Fourier-Learnable** It is possible to efficiently learn all the heavy coefficients of $C_x$ given query access to its noisy version.

# More details on Akavia et al. (2003)

- $\mathcal{C} = \{C_x : \mathbb{Z}_n \to \{\pm 1\} \mid x \in \mathbb{Z}_n\}$ where $C_x(\lambda) = \pi(\lambda \cdot x)$
- An oracle $\Omega$ predicting $\pi(x)$ given $f(x)$

It should be shown that $\mathcal{C}$ meets the following properties

**Accessible** Given $f(x)$, it is possible to get a "noisy" $\tilde{C}_x$ of $C_x$

- They assume $f$ is homomorphic
  i.e., given $\lambda$ and $f(x)$ it is possible to compute $f(\lambda x)$
- Noisy access to $C_x(\lambda)$ is obtained by querying the oracle on $f(\lambda x)$

**Concentrated** Every codeword $C_x$ is a Fourier concentrated function

**Recoverable** Given a frequency (character) $\chi$, $\exists$ a poly time algorithm that finds all values $x$ such that $\chi$ is "heavy" for $C_x$

**Fourier-Learnable** It is possible to efficiently learn all the heavy coefficients of $C_x$ given query access to its noisy version.

# More details on Akavia et al. (2003)

- $\mathcal{C} = \{C_x : \mathbb{Z}_n \to \{\pm 1\} \mid x \in \mathbb{Z}_n\}$ where $C_x(\lambda) = \pi(\lambda \cdot x)$
- An oracle $\Omega$ predicting $\pi(x)$ given $f(x)$

It should be shown that $\mathcal{C}$ meets the following properties

**Accessible** Given $f(x)$, it is possible to get a "noisy" $\tilde{C}_x$ of $C_x$

- They assume $f$ is homomorphic
  i.e., given $\lambda$ and $f(x)$ it is possible to compute $f(\lambda x)$
- Noisy access to $C_x(\lambda)$ is obtained by querying the oracle on $f(\lambda x)$

**Concentrated** Every ... unction

**Recoverable** Given ... algorithm that finds all valu...

> Can be shown when $\pi$ is any individual bit and $\mathcal{C}$ is a multiplication code.

**Fourier-Learnable** It is possible to efficiently learn all the heavy coefficients of $C_x$ given query access to its noisy version.

# More details on Akavia et al. (2003)

- $\mathcal{C} = \{C_x : \mathbb{Z}_n \to \{\pm 1\} \mid x \in \mathbb{Z}_n\}$ where $C_x(\lambda) = \pi(\lambda \cdot x)$
- An oracle $\Omega$ predicting $\pi(x)$ given $f(x)$

It should be shown that $\mathcal{C}$ meets the following properties

Accessible Given $f(x)$, it is possible to get a "noisy" $\tilde{C}_x$ of $C_x$
- They assume $f$ is homomorphic
  i.e., given $\lambda$ and $f(x)$ it is possible to compute $f(\lambda x)$
- Noisy access to $C_x(\lambda)$ is obtained by querying the oracle on $f(\lambda x)$

Concentrated Every[…]unction

Recoverable Given […]algorithm that finds all valu[…]

Can be shown when $\pi$ is any individual bit and $\mathcal{C}$ is a multiplication code.

Fourier-Learnable It is possible to efficiently learn all the heavy coefficients of $C_x$ given query access to its noisy version.

# Elliptic Curves, Short Weierstrass Equations, Isomorphisms

## Short Weierstrass Equations

- An elliptic curve $E$ can be represented by a short Weierstrass equation

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

## Isomorphism Classes

- $W_{a,b}$ is isomorphic to $W'_{a',b'}$ iff $a' = \lambda^{-4}a$, $b' = \lambda^{-6}b$ for $\lambda \in \mathbb{F}_p^\times$
- The isomorphism class of $E$ is given by

$$\mathcal{W}(E) = \{y^2 = x^3 + \lambda^4 ax + \lambda^6 b \mid \lambda \in \mathbb{F}_p^\times\}$$

- The isomorphism $\Phi_\lambda$ is easily computed as

$$\Phi_\lambda((x, y)) = (\lambda^2 x, \lambda^3 y)$$

# Elliptic Curves, Short Weierstrass Equations, Isomorphisms

## Short Weierstrass Equations

- An elliptic curve $E$ can be represented by a short Weierstrass equation

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

## Isomorphism Classes

- $W_{a,b}$ is isomorphic to $W'_{a',b'}$ iff $a' = \lambda^{-4}a,\ b' = \lambda^{-6}b$ for $\lambda \in \mathbb{F}_p^\times$
- The isomorphism class of $E$ is given by

$$\mathcal{W}(E) = \{y^2 = x^3 + \lambda^4 ax + \lambda^6 b \mid \lambda \in \mathbb{F}_p^\times\}$$

- The isomorphism $\Phi_\lambda$ is easily computed as

$$\Phi_\lambda((x, y)) = (\lambda^2 x, \lambda^3 y)$$

# Elliptic Curves, Short Weierstrass Equations, Isomorphisms

## Short Weierstrass Equations

- An elliptic curve $E$ can be represented by a short Weierstrass equation

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

## Isomorphism Classes

- $W_{a,b}$ is isomorphic to $W'_{a',b'}$ iff $a' = \lambda^{-4}a$, $b' = \lambda^{-6}b$ for $\lambda \in \mathbb{F}_p^\times$
- The isomorphism class of $E$ is given by

$$\mathcal{W}(E) = \{y^2 = x^3 + \lambda^4 ax + \lambda^6 b \mid \lambda \in \mathbb{F}_p^\times\}$$

- The isomorphism $\Phi_\lambda$ is easily computed as

$$\Phi_\lambda((x, y)) = (\lambda^2 x, \lambda^3 y)$$

# Elliptic Curves, Short Weierstrass Equations, Isomorphisms

## Short Weierstrass Equations

- An elliptic curve $E$ can be represented by a short Weierstrass equation

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

## Isomorphism Classes

- $W_{a,b}$ is isomorphic to $W'_{a',b'}$ iff $a' = \lambda^{-4}a,\ b' = \lambda^{-6}b$ for $\lambda \in \mathbb{F}_p^\times$
- The isomorphism class of $E$ is given by

$$\mathcal{W}(E) = \left\{ y^2 = x^3 + \lambda^4 ax + \lambda^6 b \mid \lambda \in \mathbb{F}_p^\times \right\}$$

- The isomorphism $\Phi_\lambda$ is easily computed as

$$\Phi_\lambda((x, y)) = (\lambda^2 x, \lambda^3 y)$$

# Our Result 1: Bit-security of Diffie-Hellman over Elliptic Curves

## Assumption

- DH problem over an EC instance generator $\mathcal{E}$ is hard iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_{a,b}\left[\mathcal{A}(E,P,aP,bP) = abP \mid E \leftarrow \mathcal{E}(1^{\ell})\right] \leq \mathsf{negl}(\ell)$$

## Theorem

- If DH over $\mathcal{E}$ is hard, then

$$\forall \text{ PPT } \Omega \qquad \left|\Pr_{a,b,\lambda}\left[\Omega(\lambda,E,P,aP,bP) = B_k([\Phi_\lambda(abP)]_x)\right] - \beta_k\right| \leq \mathsf{negl}(\ell)$$

# Our Result 1: Bit-security of Diffie-Hellman over Elliptic Curves

## Assumption

- DH problem over an EC instance generator $\mathcal{E}$ is hard iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_{a,b}\left[\mathcal{A}(E,P,aP,bP) = abP \mid E \leftarrow \mathcal{E}(1^{\ell})\right] \leq \mathsf{negl}(\ell)$$

## Theorem

- If DH over $\mathcal{E}$ is hard, then

$$\forall \text{ PPT } \Omega \qquad \left|\Pr_{a,b,\lambda}\left[\Omega(\lambda,E,P,aP,bP) = B_k([\Phi_\lambda(abP)]_x)\right] - \beta_k\right| \leq \mathsf{negl}(\ell)$$

# Our Result 1: Proof Sketch

## What we are given

1. $E$, $P$, $aP$, $bP$
2. $\Omega$ predicting $B_k([\Phi_\lambda(abP)]_x) = B_k(\lambda^2[abP]_x)$ with non-negl adv

## How we do it

1. Define the multiplication code

$$\mathcal{C} = \{C_Q : \mathbb{F}_p^\times \to \{\pm 1\} \mid Q \in \mathbb{F}_p\} \quad \text{where} \quad C_Q(\lambda) = B_k(\lambda \cdot Q_x)$$

2. But $\Phi_\lambda(\cdot)$ squares $\lambda$. So, following BoSh01, define

$$\Omega'(\lambda, E, P, aP, bP) = \begin{cases} \Omega(\sqrt{\lambda}, E, P, aP, bP) & \text{if } \lambda \text{ is a square} \\ B_k\text{-biased coin} & \text{otherwise} \end{cases}$$

3. $\mathcal{C}$ meets three properties required for the framework of Akavia et al.
   - Accessible: $\Omega'$ gives us access to a noisy $\tilde{C}_Q = \Omega'(\lambda, E, P, aP, bP)$
   - Concentrated: Codewords are Fourier concentrated
   - Recoverable: The recovery algorithm of Akavia et al. also works
4. This process yields a poly-size list of candidates: either output one at random or use Shoup's self-corrector

# Our Result 1: Proof Sketch

## What we are given

1. $E$, $P$, $aP$, $bP$
2. $\Omega$ predicting $B_k([\Phi_\lambda(abP)]_x) = B_k(\lambda^2[abP]_x)$ with non-negl adv

## How we do it

1. Define the multiplication code

$$\mathcal{C} = \{C_Q : \mathbb{F}_p^\times \to \{\pm 1\} \mid Q \in \mathbb{F}_p\} \quad \text{where} \quad C_Q(\lambda) = B_k(\lambda \cdot Q_x)$$

2. But $\Phi_\lambda(\cdot)$ squares $\lambda$. So, following BoSh01, define

$$\Omega'(\lambda, E, P, aP, bP) = \begin{cases} \Omega(\sqrt{\lambda}, E, P, aP, bP) & \text{if } \lambda \text{ is a square} \\ \beta_k\text{-biased coin} & \text{otherwise} \end{cases}$$

3. $\mathcal{C}$ meets three properties required for the framework of Akavia et al.
   - Accessible: $\Omega'$ gives us access to a noisy $\tilde{C}_Q = \Omega'(\lambda, E, P, aP, bP)$
   - Concentrated: Codewords are Fourier concentrated
   - Recoverable: The recovery algorithm of Akavia et al. also works
4. This process yields a poly-size list of candidates: either output one at random or use Shoup's self-corrector

# Our Result 1: Proof Sketch

## What we are given

1. $E$, $P$, $aP$, $bP$
2. $\Omega$ predicting $B_k([\Phi_\lambda(abP)]_x) = B_k(\lambda^2[abP]_x)$ with non-negl adv

## How we do it

1. Define the multiplication code

$$\mathcal{C} = \{C_Q : \mathbb{F}_p^\times \to \{\pm 1\} \mid Q \in \mathbb{F}_p\} \quad \text{where} \quad C_Q(\lambda) = B_k(\lambda \cdot Q_x)$$

2. But $\Phi_\lambda(\cdot)$ squares $\lambda$. So, following BoSh01, define

$$\Omega'(\lambda, E, P, aP, bP) = \begin{cases} \Omega(\sqrt{\lambda}, E, P, aP, bP) & \text{if } \lambda \text{ is a square} \\ \beta_k\text{-biased coin} & \text{otherwise} \end{cases}$$

3. $\mathcal{C}$ meets three properties required for the framework of Akavia et al.
   - **Accessible** $\Omega'$ gives us access to a noisy $\tilde{C}_Q = \Omega'(\lambda, E, P, aP, bP)$
   - **Concentrated** Codewords are Fourier concentrated
   - **Recoverable** The recovery algorithm of Akavia et al. also works
4. This process yields a poly-size list of candidates: either output one at random or use Shoup's self-corrector

# Our Result 1: Proof Sketch

## What we are given

1. $E$, $P$, $aP$, $bP$
2. $\Omega$ predicting $B_k([\Phi_\lambda(abP)]_x) = B_k(\lambda^2[abP]_x)$ with non-negl adv

## How we do it

1. Define the multiplication code

$$\mathcal{C} = \{C_Q : \mathbb{F}_p^\times \to \{\pm 1\} \mid Q \in \mathbb{F}_p\} \quad \text{where} \quad C_Q(\lambda) = B_k(\lambda \cdot Q_x)$$

2. But $\Phi_\lambda(\cdot)$ squares $\lambda$. So, following BoSh01, define

$$\Omega'(\lambda, E, P, aP, bP) = \begin{cases} \Omega(\sqrt{\lambda}, E, P, aP, bP) & \text{if } \lambda \text{ is a square} \\ \beta_k\text{-biased coin} & \text{otherwise} \end{cases}$$

3. $\mathcal{C}$ meets three properties required for the framework of Akavia et al.
   - **Accessible** $\Omega'$ gives us access to a noisy $\tilde{C}_Q = \Omega'(\lambda, E, P, aP, bP)$
   - **Concentrated** Codewords are Fourier concentrated
   - **Recoverable** The recovery algorithm of Akavia et al. also works
4. This process yields a poly-size list of candidates: either output one at random or use Shoup's self-corrector

# Our Result 1: Proof Sketch

## What we are given

1. $E$, $P$, $aP$, $bP$
2. $\Omega$ predicting $B_k([\Phi_\lambda(abP)]_x) = B_k(\lambda^2[abP]_x)$ with non-negl adv

## How we do it

1. Define the multiplication code

$$\mathcal{C} = \{C_Q : \mathbb{F}_p^\times \to \{\pm 1\} \mid Q \in \mathbb{F}_p\} \quad \text{where} \quad C_Q(\lambda) = B_k(\lambda \cdot Q_x)$$

2. But $\Phi_\lambda(\cdot)$ squares $\lambda$. So, following BoSh01, define

$$\Omega'(\lambda, E, P, aP, bP) = \begin{cases} \Omega(\sqrt{\lambda}, E, P, aP, bP) & \text{if } \lambda \text{ is a square} \\ \beta_k\text{-biased coin} & \text{otherwise} \end{cases}$$

3. $\mathcal{C}$ meets three properties required for the framework of Akavia et al.
   - **Accessible** $\Omega'$ gives us access to a noisy $\tilde{C}_Q = \Omega'(\lambda, E, P, aP, bP)$
   - **Concentrated** Codewords are Fourier concentrated
   - **Recoverable** The recovery algorithm of Akavia et al. also works
4. This process yields a poly-size list of candidates: either output one at random or use Shoup's self-corrector

# Other Candidate Settings?

## The Finite Field $\mathbb{F}_{p^2}$

- For a given prime $p$, there are around $p^2/2$ fields of the form $\mathbb{F}_{p^2}$, all isomorphic to each other
- Each such field can be represented by a monic irreducible polynomial $h(x) = x^2 + h_1 x + h_0$ so that the field is isomorphic to $\mathbb{F}_p[x]/(h)$
- Then, $g \in \mathbb{F}_{p^2}$ is a linear polynomial $g = g_0 + g_1 x$. Let $[g]_i$ denote $g_i$.
- Also for $h, \hat{h}$ there exists an easily computable isomorphism $\phi_{h,\hat{h}}$, computed by right multiplication of the coefficients by a matrix $\left[\begin{smallmatrix} 1 & 0 \\ \mu & \lambda \end{smallmatrix}\right]$.
- For example,

$$\begin{aligned} \phi_{h,\hat{h}}(g) &= \phi_{h,\hat{h}}(\begin{bmatrix} g_0 & g_1 \end{bmatrix}) \\ &= \begin{bmatrix} g_0 & g_1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ \mu & \lambda \end{bmatrix} \\ &= \begin{bmatrix} g_0 + \mu g_1 & \lambda g_1 \end{bmatrix}. \end{aligned}$$

# Other Candidate Settings?

## The Finite Field $\mathbb{F}_{p^2}$

- For a given prime $p$, there are around $p^2/2$ fields of the form $\mathbb{F}_{p^2}$, all isomorphic to each other
- Each such field can be represented by a monic irreducible polynomial $h(x) = x^2 + h_1 x + h_0$ so that the field is isomorphic to $\mathbb{F}_p[x]/(h)$
- Then, $g \in \mathbb{F}_{p^2}$ is a linear polynomial $g = g_0 + g_1 x$. Let $[g]_i$ denote $g_i$.
- Also for $h, \hat{h}$ there exists an easily computable isomorphism $\phi_{h,\hat{h}}$, computed by right multiplication of the coefficients by a matrix $\left[\begin{smallmatrix} 1 & 0 \\ \mu & \lambda \end{smallmatrix}\right]$.
- For example,

$$\begin{aligned}
\phi_{h,\hat{h}}(g) &= \phi_{h,\hat{h}}(\begin{bmatrix} g_0 & g_1 \end{bmatrix}) \\
&= \begin{bmatrix} g_0 & g_1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ \mu & \lambda \end{bmatrix} \\
&= \begin{bmatrix} g_0 + \mu g_1 & \lambda g_1 \end{bmatrix}.
\end{aligned}$$

# Other Candidate Settings?

## The Finite Field $\mathbb{F}_{p^2}$

- For a given prime $p$, there are around $p^2/2$ fields of the form $\mathbb{F}_{p^2}$, all isomorphic to each other
- Each such field can be represented by a monic irreducible polynomial $h(x) = x^2 + h_1 x + h_0$ so that the field is isomorphic to $\mathbb{F}_p[x]/(h)$
- Then, $g \in \mathbb{F}_{p^2}$ is a linear polynomial $g = g_0 + g_1 x$. Let $[g]_i$ denote $g_i$.
- Also for $h, \hat{h}$ there exists an easily computable isomorphism $\phi_{h,\hat{h}}$, computed by right multiplication of the coefficients by a matrix $\left[\begin{smallmatrix} 1 & 0 \\ \mu & \lambda \end{smallmatrix}\right]$.
- For example,

$$\begin{aligned}\phi_{h,\hat{h}}(g) &= \phi_{h,\hat{h}}(\begin{bmatrix} g_0 & g_1 \end{bmatrix}) \\ &= \begin{bmatrix} g_0 & g_1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ \mu & \lambda \end{bmatrix} \\ &= \begin{bmatrix} g_0 + \mu g_1 & \lambda g_1 \end{bmatrix}.\end{aligned}$$

# Other Candidate Settings?

## The Finite Field $\mathbb{F}_{p^2}$

- For a given prime $p$, there are around $p^2/2$ fields of the form $\mathbb{F}_{p^2}$, all isomorphic to each other
- Each such field can be represented by a monic irreducible polynomial $h(x) = x^2 + h_1 x + h_0$ so that the field is isomorphic to $\mathbb{F}_p[x]/(h)$
- Then, $g \in \mathbb{F}_{p^2}$ is a linear polynomial $g = g_0 + g_1 x$. Let $[g]_i$ denote $g_i$.
- Also for $h, \hat{h}$ there exists an easily computable isomorphism $\phi_{h,\hat{h}}$, computed by right multiplication of the coefficients by a matrix $\left[\begin{smallmatrix} 1 & 0 \\ \mu & \lambda \end{smallmatrix}\right]$.
- For example,

$$\begin{aligned}
\phi_{h,\hat{h}}(g) &= \phi_{h,\hat{h}}(\begin{bmatrix} g_0 & g_1 \end{bmatrix}) \\
&= \begin{bmatrix} g_0 & g_1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ \mu & \lambda \end{bmatrix} \\
&= \begin{bmatrix} g_0 + \mu g_1 & \lambda g_1 \end{bmatrix}.
\end{aligned}$$

# Other Candidate Settings?

## The Finite Field $\mathbb{F}_{p^2}$

- For a given prime $p$, there are around $p^2/2$ fields of the form $\mathbb{F}_{p^2}$, all isomorphic to each other
- Each such field can be represented by a monic irreducible polynomial $h(x) = x^2 + h_1 x + h_0$ so that the field is isomorphic to $\mathbb{F}_p[x]/(h)$
- Then, $g \in \mathbb{F}_{p^2}$ is a linear polynomial $g = g_0 + g_1 x$. Let $[g]_i$ denote $g_i$.
- Also for $h, \hat{h}$ there exists an easily computable isomorphism $\phi_{h,\hat{h}}$, computed by right multiplication of the coefficients by a matrix $\left[\begin{smallmatrix} 1 & 0 \\ \mu & \lambda \end{smallmatrix}\right]$.
- For example,

$$\begin{aligned}
\phi_{h,\hat{h}}(g) &= \phi_{h,\hat{h}}(\begin{bmatrix} g_0 & g_1 \end{bmatrix}) \\
&= \begin{bmatrix} g_0 & g_1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ \mu & \lambda \end{bmatrix} \\
&= \begin{bmatrix} g_0 + \mu g_1 & \lambda g_1 \end{bmatrix}.
\end{aligned}$$

# Our Result 2: Bit-security of (Partial) DH over $\mathbb{F}_{p^2}$

## Assumption

- DH problem over a FF instance generator $\mathcal{F}$ is hard iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_{a,b}\left[\mathcal{A}(F, g, g^a, g^b) = \quad g^{ab} \quad \mid F \leftarrow \mathcal{F}(1^\ell)\right] \leq \mathsf{negl}(\ell)$$

## Theorem

- If (Partial) DH over $\mathcal{F}$ is hard, then

$$\forall \text{ PPT } \Omega \quad \left| \Pr_{a,b,h,\hat{h}}\left[\Omega(h, \hat{h}, F, g, g^a, g^b) = B_k\left(\left[\phi_{h,\hat{h}}(g^{ab})\right]_1\right)\right] - \beta_k \right| \leq \mathsf{negl}(\ell)$$

## Proof Idea

- Apply the framework of Akavia et al.

# Our Result 2: Bit-security of (Partial) DH over $\mathbb{F}_{p^2}$

## Assumption

- DH problem over a FF instance generator $\mathcal{F}$ is hard iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_{a,b}\left[\mathcal{A}(F, g, g^a, g^b) = \boxed{g^{ab}} \mid F \leftarrow \mathcal{F}(1^\ell)\right] \leq \mathsf{negl}(\ell)$$

a linear polynomial

## Theorem

- If (Partial) DH over $\mathcal{F}$ is hard, then

$$\forall \text{ PPT } \Omega \quad \left| \Pr_{a,b,h,\hat{h}}\left[\Omega(h, \hat{h}, F, g, g^a, g^b) = B_k\left(\left[\phi_{h,\hat{h}}(g^{ab})\right]_1\right)\right] - \beta_k \right| \leq \mathsf{negl}(\ell)$$

## Proof Idea

- Apply the framework of Akavia et al.

# Our Result 2: Bit-security of (Partial) DH over $\mathbb{F}_{p^2}$

## New Assumption

- (Partial) DH problem over a FF instance generator $\mathcal{F}$ is hard iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_{a,b}\left[\mathcal{A}(F, g, g^a, g^b) = \boxed{\left[g^{ab}\right]_1} \mid F \leftarrow \mathcal{F}(1^\ell)\right] \leq \mathsf{negl}(\ell)$$

the degree-1 coefficient

## Theorem

- If (Partial) DH over $\mathcal{F}$ is hard, then

$$\forall \text{ PPT } \Omega \quad \left| \Pr_{a,b,h,\hat{h}}\left[\Omega(h, \hat{h}, F, g, g^a, g^b) = B_k\left(\left[\phi_{h,\hat{h}}(g^{ab})\right]_1\right)\right] - \beta_k \right| \leq \mathsf{negl}(\ell)$$

## Proof Idea

- Apply the framework of Akavia et al.

# Our Result 2: Bit-security of (Partial) DH over $\mathbb{F}_{p^2}$

## New Assumption

- (Partial) DH problem over a FF instance generator $\mathcal{F}$ is hard iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_{a,b}\left[\mathcal{A}(F, g, g^a, g^b) = \boxed{\left[g^{ab}\right]_1} \mid F \leftarrow \mathcal{F}(1^\ell)\right] \leq \mathsf{negl}(\ell)$$

the degree-1 coefficient

## Theorem

- If (Partial) DH over $\mathcal{F}$ is hard, then

$$\forall \text{ PPT } \Omega \quad \left| \Pr_{a,b,h,\hat{h}}\left[\Omega(h, \hat{h}, F, g, g^a, g^b) = B_k\left(\left[\phi_{h,\hat{h}}(g^{ab})\right]_1\right)\right] - \beta_k \right| \leq \mathsf{negl}(\ell)$$

## Proof Idea

- Apply the framework of Akavia et al.

# Our Result 2: Bit-security of (Partial) DH over $\mathbb{F}_{p^2}$

## New Assumption

- (Partial) DH problem over a FF instance generator $\mathcal{F}$ is hard iff

$$\forall \text{ PPT } \mathcal{A} \qquad \Pr_{a,b}\left[\mathcal{A}(F, g, g^a, g^b) = \boxed{\left[g^{ab}\right]_1} \mid F \leftarrow \mathcal{F}(1^\ell)\right] \leq \mathsf{negl}(\ell)$$

the degree-1 coefficient

## Theorem

- If (Partial) DH over $\mathcal{F}$ is hard, then

$$\forall \text{ PPT } \Omega \quad \left| \Pr_{a,b,h,\hat{h}}\left[\Omega(h, \hat{h}, F, g, g^a, g^b) = B_k\left(\left[\phi_{h,\hat{h}}(g^{ab})\right]_1\right)\right] - \beta_k \right| \leq \mathsf{negl}(\ell)$$

## Proof Idea

- Apply the framework of Akavia et al.

# Our Result 2: Proof Sketch

## What we are given

1. $F$, $g$, $g^a$, $g^b$
2. $\Omega$ predicting $B_k([\phi_{h,\hat{h}}(g^{ab})]_1) = B_k(\lambda[g^{ab}]_1)$ with non-negl adv

## How we do it

1. Define the multiplication code

$$\mathcal{C} = \{C_\alpha : \mathbb{F}_p^\times \to \{\pm 1\} \mid \alpha \in \mathbb{F}_{p^2}\} \quad \text{where} \quad C_\alpha(\lambda) = B_k(\lambda \cdot [\alpha]_1)$$

2. $\mathcal{C}$ meets three properties required for the framework of Akavia et al.
   - Accessible $\Omega$ gives us access to a noisy $\tilde{C}_\alpha = \Omega(\lambda, g, g^a, g^b)$
   - Concentrated Codewords are Fourier concentrated
   - Recoverable The recovery algorithm of Akavia et al. also works
3. But, we only get a poly-list of degree-1 coefficients. So, we pick one coefficient at random.

# Our Result 2: Proof Sketch

## What we are given

1. $F$, $g$, $g^a$, $g^b$
2. $\Omega$ predicting $B_k([\phi_{h,\hat{h}}(g^{ab})]_1) = B_k(\lambda[g^{ab}]_1)$ with non-negl adv

## How we do it

1. Define the multiplication code

$$\mathcal{C} = \left\{ C_\alpha : \mathbb{F}_p^\times \to \{\pm 1\} \mid \alpha \in \mathbb{F}_{p^2} \right\} \quad \text{where} \quad C_\alpha(\lambda) = B_k(\lambda \cdot [\alpha]_1)$$

2. $\mathcal{C}$ meets three properties required for the framework of Akavia et al.
   - **Accessible** $\Omega$ gives us access to a noisy $\tilde{C}_\alpha = \Omega(\lambda, g, g^a, g^b)$
   - **Concentrated** Codewords are Fourier concentrated
   - **Recoverable** The recovery algorithm of Akavia et al. also works
3. But, we only get a poly-list of degree-1 coefficients. So, we pick one coefficient at random.

# Our Result 2: Proof Sketch

## What we are given

1. $F$, $g$, $g^a$, $g^b$
2. $\Omega$ predicting $B_k([\phi_{h,\hat{h}}(g^{ab})]_1) = B_k(\lambda[g^{ab}]_1)$ with non-negl adv

## How we do it

1. Define the multiplication code

$$\mathcal{C} = \left\{ C_\alpha : \mathbb{F}_p^\times \to \{\pm 1\} \mid \alpha \in \mathbb{F}_{p^2} \right\} \quad \text{where} \quad C_\alpha(\lambda) = B_k(\lambda \cdot [\alpha]_1)$$

2. $\mathcal{C}$ meets three properties required for the framework of Akavia et al.
   - **Accessible** $\Omega$ gives us access to a noisy $\tilde{C}_\alpha = \Omega(\lambda, g, g^a, g^b)$
   - **Concentrated** Codewords are Fourier concentrated
   - **Recoverable** The recovery algorithm of Akavia et al. also works
3. But, we only get a poly-list of degree-1 coefficients. So, we pick one coefficient at random.

# Our Result 2: Proof Sketch

## What we are given

1. $F$, $g$, $g^a$, $g^b$
2. $\Omega$ predicting $B_k([\phi_{h,\hat{h}}(g^{ab})]_1) = B_k(\lambda[g^{ab}]_1)$ with non-negl adv

## How we do it

1. Define the multiplication code

$$\mathcal{C} = \left\{C_\alpha : \mathbb{F}_p^\times \to \{\pm 1\} \mid \alpha \in \mathbb{F}_{p^2}\right\} \quad \text{where} \quad C_\alpha(\lambda) = B_k(\lambda \cdot [\alpha]_1)$$

2. $\mathcal{C}$ meets three properties required for the framework of Akavia et al.
   - **Accessible** $\Omega$ gives us access to a noisy $\tilde{C}_\alpha = \Omega(\lambda, g, g^a, g^b)$
   - **Concentrated** Codewords are Fourier concentrated
   - **Recoverable** The recovery algorithm of Akavia et al. also works
3. But, we only get a poly-list of degree-1 coefficients. So, we pick one coefficient at random.

# Our Result 3: Bit-security of FFB-POWFs

- Proof of second result also applies to finite field-based partial OWF
- A function $f$ is a FFB-POWF iff
  1. $f$ is easy to compute given $\alpha$
  2. It is hard to compute $[\alpha]_1$ from $f(\alpha)$
  3. $f$ does not depend on a particular isomorphism class of $\mathbb{F}_{p^2}$
- Duc and Jetchev (2012) proved this for ECB-OWF

## Our Result 3: Bit-security of FFB-POWFs

- Proof of second result also applies to finite field-based partial OWF
- A function $f$ is a FFB-POWF iff
  1. $f$ is easy to compute given $\alpha$
  2. It is hard to compute $[\alpha]_1$ from $f(\alpha)$
  3. $f$ does not depend on a particular isomorphism class of $\mathbb{F}_{p^2}$
- Duc and Jetchev (2012) proved this for ECB-OWF

# Our Result 3: Bit-security of FFB-POWFs

- Proof of second result also applies to finite field-based partial OWF
- A function $f$ is a FFB-POWF iff
  1. $f$ is easy to compute given $\alpha$
  2. It is hard to compute $[\alpha]_1$ from $f(\alpha)$
  3. $f$ does not depend on a particular isomorphism class of $\mathbb{F}_{p^2}$
- Duc and Jetchev (2012) proved this for ECB-OWF

# Summary & Open Problems

## Summary

1. Every bit of the EC DH secret value is hard-core
2. Above result also applies to (partial) DH problem over finite fields $\mathbb{F}_{p^2}$
3. The second result also applies to FFB-POWFs over $\mathbb{F}_{p^2}$
4. Our approach "augments" the input to the computationally hard problem with a random description of the underlying group

## Open Problems

1. Extend our results to $\mathbb{F}_{p^t}$ for $t > 2$
2. Show that DH problem over $\mathbb{F}_{p^2}$ $\rightarrow$ (Partial) DH problem over $\mathbb{F}_{p^2}$
3. Show that DH problem over $\mathbb{F}_{p}$ $\rightarrow$ (Partial) DH problem over $\mathbb{F}_{p^2}$
4. **Find hard-core predicates for DH over $\mathbb{F}_{p}$**

# Summary & Open Problems

## Summary

1. Every bit of the EC DH secret value is hard-core
2. Above result also applies to (partial) DH problem over finite fields $\mathbb{F}_{p^2}$
3. The second result also applies to FFB-POWFs over $\mathbb{F}_{p^2}$
4. Our approach "augments" the input to the computationally hard problem with a random description of the underlying group

## Open Problems

1. Extend our results to $\mathbb{F}_{p^t}$ for $t > 2$
2. Show that DH problem over $\mathbb{F}_{p^2} \rightarrow$ (Partial) DH problem over $\mathbb{F}_{p^2}$
3. Show that DH problem over $\mathbb{F}_{p} \rightarrow$ (Partial) DH problem over $\mathbb{F}_{p^2}$
4. **Find hard-core predicates for DH over $\mathbb{F}_{p}$**