Obfuscation and Weak Multilinear Maps

Mark Zhandry – Princeton University

Joint work with Saikrishna Badrinarayanan, Sanjam Garg, Eric Miles, Pratyay Mukherjee, Amit Sahai, Akshayaram Srinivasan
Obfuscation [BGIRSVY’01, GGHRSW’13]

Compiler: “scrambles” program, hiding implementation

“Industry accepted” security notion: indist. Obfuscation

$$P_1(x) = P_2(x) \forall (x) \Rightarrow iO(P_1) \approx_c iO(P_2)$$

“Crypto complete”:

[GGHRSW’13, SW’13, BZ’13, BST’13, GGHR’13, BP’14, HJKSWZ’14, CLTV’14, ...]
Multilinear Maps (a.k.a. graded encodings) [BS’03, GGH’13, CLT’13, GGH’15]

Main tool for all constructions of obfuscation

Levels 1,...,k, Field/Ring $F$

secret

$\{a \in F, i \in [k]\}$

$[a]_i + [b]_i \rightarrow [a+b]_i$

$[a]_i \times [b]_j \rightarrow [ab]_{i+j}$

public

$[a]_k \rightarrow \text{IsZero} \rightarrow \text{Yes/No}$
Multilinear Maps (a.k.a. graded encodings) [BS’03, GGH’13, CLT’13, GGH’15]

**k** levels: compute arbitrary degree **k** polynomials

Asymmetric mmaps: additional restrictions
- E.g. multilinear polynomials
Obfuscation From Multilinear Maps

\[ \text{Obfuscate}(P): \]

\[ \text{Eval}(x): \quad X \]

\[ \text{Enc} \]

\[ [p_x]_k \]

\[ \text{IsZero} \]

\[ P(x) = 1 \iff p_x = 0 \]
Applications of Multilinear Maps
“Zeroizing” Attacks on MMaps
“Zeroizing” Attacks on MMaps

(Note: apps still possible using obfuscation)
Central Questions

Q1: Is obfuscation secure?

Q2: If so, how to show it?
This Work: Focus on GGH’13 Mmaps
Background...
High-level description GGH13

Level $i$ encoding of $x$: $\frac{x + g^s}{z^i} \pmod{q}$
High-level description GGH13

Level $i$ encoding of $x$: \[
\frac{x + gs}{{z_i}^{\text{"short"}}} \pmod{q}
\]

• Add within levels

\[
\frac{x_1 + gs_1}{{z_i}^i} + \frac{x_2 + gs_2}{{z_i}^i} = \frac{(x_1 + x_2) + g(s_1 + s_2)}{{z_i}^i}
\]
High-level description GGH13

Level \(i\) encoding of \(x\): \(\frac{x + gs}{z^i} \text{“short” (mod } q)\)

- Add within levels
- Multiplication makes levels add

\[
\frac{x_1 + gs_1}{z^i} \cdot \frac{x_2 + gs_2}{z^j} = \frac{(x_1 x_2) + g(s_1 x_2 + s_2 x_1 + gs_1 s_2)}{z^{i+j}}
\]
High-level description GGH13

Level $i$ encoding of $x$: $\frac{x + gs}{zi} \pmod{q}$

- Add within levels
- Multiplication makes levels add
- Test for zero at “top level” $k$

Public parameter $p_{zt} = \frac{h z^k}{g}$ “not too big”

$$p_{zt} \frac{gs}{z^k} = hs \quad "not\ too\ big"$$

$$p_{zt} \frac{x + gs}{z^k} = \frac{hx}{g} + hs \quad "big"$$
High-level description GGH13

Level \( i \) encoding of \( x \): \( \frac{x + gs}{zi} \) \( \pmod{q} \)

- Add within levels
- Multiplication makes levels add
- Test for zero at “top level” \( k \)

Notes:
- \( z \) must be secret (else can go down levels)
- \( g \) must be secret ([GGH’13] show attack otherwise)
Encodings of Zero – A Necessary Evil

Required for (Most) Applications

“Re-randomization”
• Needed for most (direct) applications
• Needed to use any “simple” assumption on mmaps

Add random subset of low-level zeros

Successful zero test $\Rightarrow$ top level zero
Encodings of Zero – A Necessary Evil

Required for (Most) Applications

Two low-level zeros:

\[ e_1 = g \frac{s_1}{z^i}, \quad e_2 = g \frac{s_2}{z^{k-i}} \]

\[ p_{z^i} e_1 e_2 \mod q = hgs_1 s_2 \text{ (over } \mathbb{Z}) \]

Dangerous For Security
Encodings of Zero – A Necessary Evil

Required for (Most) Applications

Zeroizing attacks:
• GGH’13: “Source group” assumptions (e.g. DLin, Subgroup decision) are false
• CGHLMMRST’15: Immunizations don’t work
• HJ’16: MDDH is false, multiparty NIKE broken
• Probably other assumptions broken too (MDHE, etc)

Dangerous For Security
Encodings of Zero – A Necessary Evil

Required for (Most) Applications

Dangerous For Security
What about Obf/WE/SKFE?

**Good News:**
No re-randomization needed in application
no low-level zeros (explicit or implicit)

**Bad News:**
Top level zeros may still be generated during use
Re-rand still needed for “simple” assumptions
Central Questions (Restated)

Q1: Can top-level zeros be used to attack iO?

Q2: How to argue security against zeroizing attacks?
Q1: Affirmative!

**Thm* [MSZ’16]:** The branching program obfuscators in [BGKPS’14, PST’14, AGIS’14, BMSZ’16] over GGH’13 do not satisfy iO

*Small heuristic component*
(Single input) Branching Programs

\[ \text{x} = 11001: \]

\[ \text{IMP}_x( \{ A_{i,b} \} ) = 1 \]

If \( \text{IMP}_x = 0 \), output 1, otherwise output 0
[BMSZ’16] Obfuscator
Building on [GGHRSW’13, BR’14, BGKPS’14, AGIS’14, ...]

“asymmetric” mmap
[BMSZ’16] Over GGH’13

Randomized Branching Program

\[ B_{i,b} = \alpha_{i,b} R_i^{-1} A_{i,b} R_{i+1} \]

Encoding randomness

\[ S_{i,b} \]

Obfuscation encodings

\[ C_{i,b} = B_{i,b} + g S_{i,b} \mod q \]

Evaluation:

\[ T_x = \text{IMP}_x(C_{i,b}) \mod q \]

\( \rightarrow \) test if “not too big”
Attack: Annihilating Polynomials

\[ T_X = p_{zt} \times \text{IMP}_x(C_{i,b}) \mod q \]

\[ = \frac{h}{g} \times \text{IMP}_x(B_{i,b} + gS_{i,b}) \mod q \]

\[ = \frac{h}{g} \times \text{IMP}_x(B_{i,b}) + D_X(\alpha_{i,b}, S_{i,b}, R_i) \]

\[ + g \times E_X(\alpha_{i,b}, S_{i,b}, R_i) \mod q \]
Suppose $P(x) = 1$ \implies $\text{IMP}_x(B_{i,b}) = 0$

\[ T_x = \frac{h}{g} \times \text{IMP}_x(B_{i,b}) + D_x(\alpha_{i,b}, S_{i,b}, R_i) + g \times E_x(\alpha_{i,b}, S_{i,b}, R_i) \mod q \]

“not too big”, so holds over $\mathbb{Z}$
Attack: Annihilating Polynomials

Suppose $P(x) = 1$

$$T_x = D_x(\alpha_{i,b}, S_{i,b}, R_i) + g \times E_x(\alpha_{i,b}, S_{i,b}, R_i)$$

Efficiency: Poly-many free vars

Exp-many inputs: Pick larger poly set of $D_x$

Algebraic dependence: $\exists$ poly $Q$: $Q(D_{x1}, D_{x2}, \ldots ) = 0$
Attack: Annihilating Polynomials

Algebraic dependence:
\[ \exists \text{poly } Q: Q(D_{x_1}, D_{x_2}, \ldots ) = 0 \]

\[ Q(T_{x_1}, T_{x_2}, \ldots ) = Q(D_{x_1}+gE_{x_1}, D_{x_2}+gE_{x_2}, \ldots ) \]
\[ = Q(D_{x_1}, D_{x_2}, \ldots ) + gQ' + g^2Q'' + \ldots \]
\[ = gQ' + g^2Q'' + \ldots \]
Goal: find $Q$ that annihilates $P_1$, but not $P_2$

Distinguishing Attack*

Extends to any “purely algebraic” obfuscator

Problem: in general, annihilation is hard

Thm ([Kay’09]): Unless PH collapses, there are dependent polys for which an annihilating polynomial requires super-polynomial sized circuits

Question: Can annihilating polys be found for particular obfuscators/programs?

* Additional work needed to test for multiple of $g$
Attack: Annihilating Polynomials

Consider “single-input” setting (used to prove iO)
Suppose “trivial” branching program: \( A_{i,0} = A_{i,1} = A_i \)

Explicit annihilating polynomial for [BMSZ’16]:

\[
q = (D_{000}D_{111})^2 + (D_{001}D_{110})^2 + (D_{010}D_{101})^2 + (D_{100}D_{011})^2 \\
- 2D_{000}D_{111}D_{001}D_{110} - 2D_{000}D_{111}D_{010}D_{101} - 2D_{000}D_{111}D_{100}D_{011} \\
- 2D_{001}D_{110}D_{010}D_{101} - 2D_{001}D_{110}D_{100}D_{011} - 2D_{010}D_{101}D_{100}D_{011} \\
+ 4D_{000}D_{011}D_{101}D_{111} + 4D_{111}D_{001}D_{010}D_{100}
\]

Computed by reducing problem to finite size, then brute-force search
Attack: Annihilating Polynomials

For dual input:
• First, reduce problem to finite size
• Brute-force annihilating poly in constant time
• Haven’t found it yet, but still gives poly-time attack

Other obfuscators:
• [BR’14, BGKPS’14, PST’14, AGIS’14]: similar analysis

Also attack ORE (SKFE) [BLRSZZ’15] over GGH’13
Now What?

Goal: Argue security of other schemes

Problem: Cannot use “simple” assumptions

Solution: Argue security in abstract attack models
Restricted Black Box Fields

\( \mathcal{F} = \text{Field}, \mathcal{P} = \text{class of polynomials on } n \text{ variables} \)

Generic Groups*:
\[ \mathcal{P} = \{ \text{Linear functions} \} \]

Black Box Fields*:
\[ \mathcal{P} = \{ \text{Polys with small algebraic circuits} \} \]

Symmetrix multilinear maps*:
\[ \mathcal{P} = \{ \text{Degree } k \text{ polynomials} \} \]

Asymmetric multilinear maps*:
\[ \mathcal{P} = \{ \text{More complicated restrictions} \} \]

* Often need greater functionality requirements for protocols. This model suffices for our discussion.
Obfuscation in Restricted BBFs

( model used by [BR’14,BGKPS’14,AGIS’14,Z’15,AB’15,BMSZ’16] )

Obfuscate(C):

\[ \text{Eval}(x) : \text{IsZero}( p(a_1, a_2, \ldots, a_n) ) \]

- If \text{IsZero} gives “True”, output 1
- If \text{IsZero} gives “False”, output 0

Our Attack: Model is false for GGH’13
A Conservative Model [BMSZ’16]

\[
\begin{align*}
\text{BBF with restricted polynomial class } P \\
\text{If } \text{IsZero}( p( a_1, a_2, ..., a_n ) ), \\
\Rightarrow \\
\text{ADVERSARY WINS}
\end{align*}
\]
Obfuscation for evasive functions [BMSZ’16]

Honest executions always give non-zero

**Thm([BMSZ’16]):** Only way for “level respecting” adversary to get zero is through honest program executions

Impossible to find zeros anywhere for evasive funcs

Compare to prior “abstract model” theorems:

**Thm([BR’14, BGKPS’14, ...]):** For “level respecting” adversary, can guess output of `IsZero` just by knowing `P(x)`

Doesn’t say if/when finding a zero is possible
A Conservative Model [BMSZ’16]

Model useless in “non-evasive” settings, e.g. iO, SKFE

Need model that allows for zeros to occur
Characterizing Attacks

**All Known Classical Attacks**

Compute polynomials obeying level restrictions

Several top level zero encodings

Attack
Characterizing Attacks

**All Known Classical Attacks**

- Compute polynomials obeying level restrictions
- Several top level zero encodings
- Polynomial in the zeros
- Multiple of $g$
- Attack
Refined Abstract Model for Mmap attacks

\[ a_1, a_2, \ldots, a_n \in F \]
\[ s_1, s_2, \ldots, s_n \leftarrow \$ F \]

Write \( p(a_1+gs_1, \ldots, a_n+gs_n) \)
\[ = c + dg + \ldots \]

If \( c \neq 0 \), output “False”
If \( c = 0 \), output “True”, \( d \)

Efficient polys*

\[ q \in Q \]

Unrestricted BBF

\[ d_1 \ d_2 \ d_3 \ldots \]

If \( q(d_1, d_2, \ldots) = 0 \), adversary wins

* Also need to assume degree \( \ll |F| \)
Refined Abstract Model for Mmap attacks

• Seems to capture intuition behind attacks

Proof in refined model → Heuristic evidence of security against current attacks

But keep in mind that:

Attack in refined model → Attack on actual protocol
Not trivially
Blocking Attacks [GMMSSZ’16]

Notably absent from attacked schemes: [GGHRSW’13]

Random diagonal converts even “trivial” branching programs into non-trivial ones

\[ A'_{i,b}: \]

\[ A_{i,b}, D_{i,b} \]

size > multilinearity
Blocking Attacks [GMMSSSZ’16]

Our fix: append random block matrix

\[ A'_{i,b} : \]

\[ \begin{array}{c|c}
  A_{i,b} & E_{i,b} \\
  \hline
  \end{array} \]

Potentially as small as 2×2
Blocking Attacks [GMMSSZ’16]

Let $\mathbf{BP}_E$ be branching program defined by $E$ matrices
Let $E_x$ be evaluation of $\mathbf{BP}_E$ on input $x$

**Thm:** If polynomial $Q$ annihilates $\{D_x\}^*$, then it annihilates $\{E_x\}$ as well

Let $\mathbf{BP}_F$ be any $\mathbf{BP}$, $F_x$ evaluation of $\mathbf{BP}_F$

**Thm:** If polynomial $Q$ annihilates $\{E_x\}^*$, then it annihilates $\{F_x\}$ as well

$^* = \text{with noticeable probability}$
Example Proof Sketch

**Thm:** If polynomial $Q$ annihilates $\{E_x\}^*$, then it annihilates $\{F_x\}$ as well

Let $E_{i,b} = F_{i,b} + r E'_{i,b}$

⇒ $E_x = F_x + r F'_x + r^2 F''_x + \ldots$

⇒ $Q(\{E_x\}) = Q(\{F_x\}) + r Q' + r^2 Q'' + \ldots$

By Schwartz-Zippel, if $\Pr[Q = 0] = \text{non-negl}$, Then $Q$ must be identically $0$

⇒ $Q(\{F_x\}) = 0$
Branching Program Unannihilateability

**Assumption:** For any efficient polynomial \( Q^* \), there is a branching program not annihilated by \( Q \).

**“Easy” fact:** PRFs in \( \text{NC}^1 \) give unannihilateable branching programs.

**Corollary:** Assuming BPUA (or \( \text{NC}^1 \) PRFs), our obfuscator is secure in the weak mmap model for [GGH’13].

* of not too high degree
Future Directions

• Substantiate BPUA ($P \neq NP$, general OWF, etc)

• Attack GGH’13 without annihilating polys

• Extend to obfuscation for circuits
  Mostly solved: [DGGMM’16] assuming $NC^1$ PRFs

• Extend attacks to CLT’13, GGH’15
  Partial progress: [CLLT’16] for single-input iO over CLT’13

• Useful abstract attack model for CLT’13, GGH’15

Thanks!