

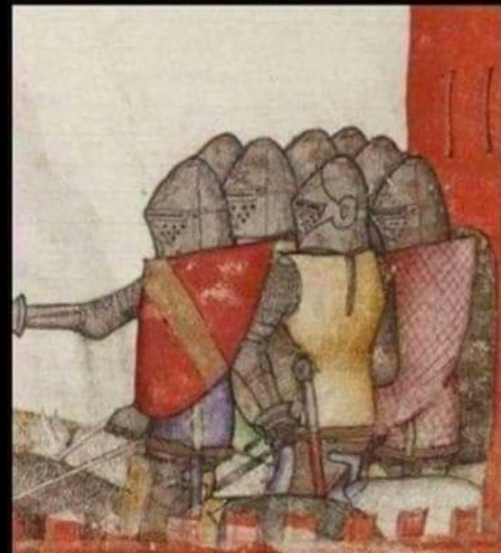
Keep your distance



Wash your hands



Cover your face



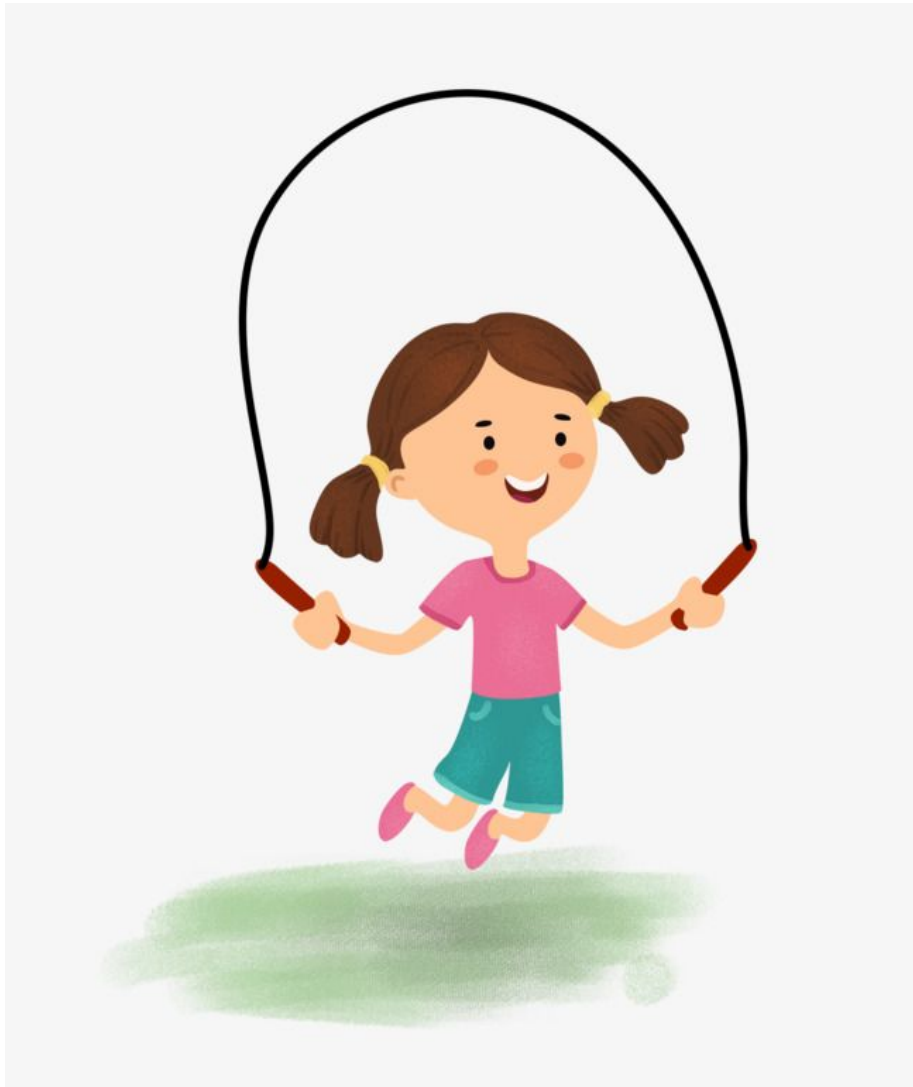
Avoid mass transport



Cryptanalysis of Candidate Program Obfuscators

Yilei Chen
[VISA Research]

2020 Simons Program on Lattices



Cryptanalysis of Candidate Program Obfuscators

Yilei Chen
[VISA Research]

2020 Simons Program on Lattices



2000

Zero-Knowledge and Code Obfuscation

Hada

2001

On the (Im)possibility of Obfuscating Programs

Barak, Goldreich, Impagliazzo, Rudich, Sahai, Vadhan, Yang



2013

Candidate Multilinear Maps from Ideal Lattices

Garg, Gentry, Halevi

Candidate indistinguishability obfuscation and functional encryption for all circuits

Garg, Gentry, Halevi, Raykova, Sahai, Waters

2013

Start the age of discovery in Cryptoland
(assuming the mmaps and iO candidates are secure)

Multilinear maps



Obfuscation

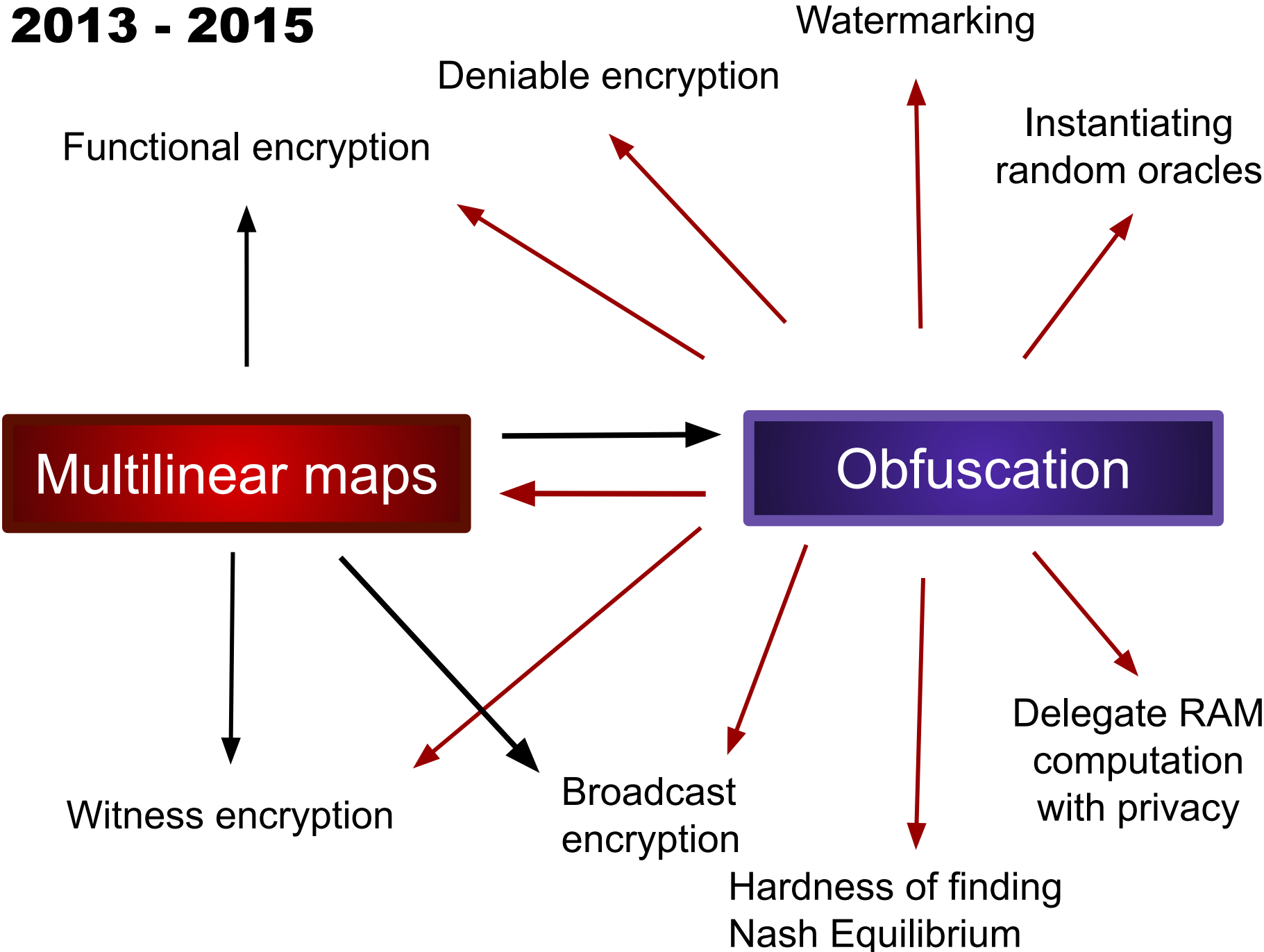
Candidate Multilinear Maps from Ideal Lattices

Garg, Gentry, Halevi

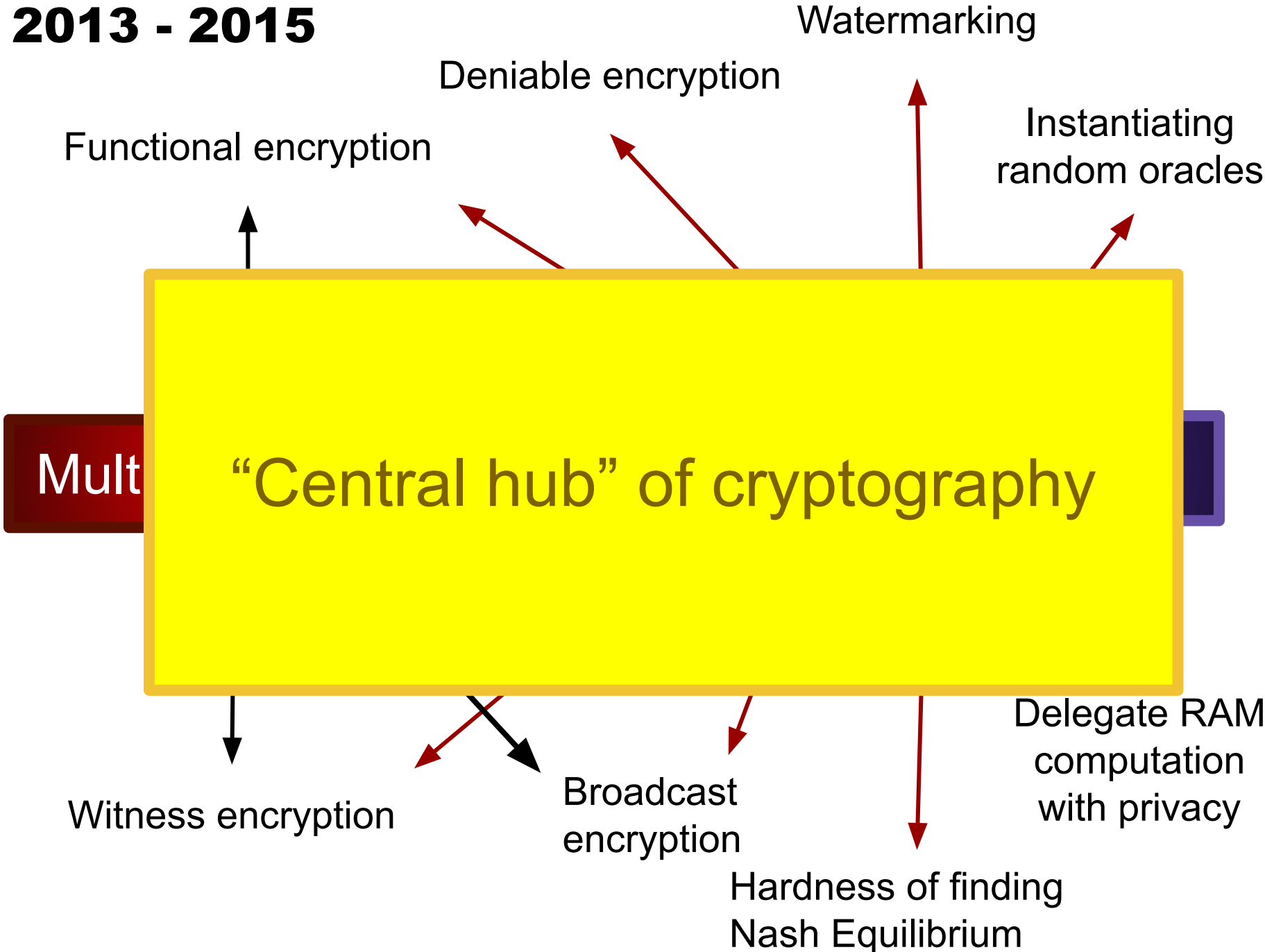
Candidate indistinguishability obfuscation and functional encryption for all circuits

Garg, Gentry, Halevi, Raykova, Sahai, Waters

2013 - 2015



2013 - 2015



Mult

“Central hub” of cryptography

Functional encryption

Deniable encryption

Watermarking

Instantiating random oracles

Witness encryption

Broadcast encryption

Hardness of finding Nash Equilibrium

Delegate RAM computation with privacy

Are the mmaps / obfuscation candidates secure?

Multilinear maps



Obfuscation

2013 - 2015

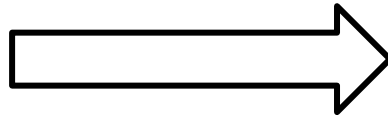
Multilinear maps => Obfuscation

Candidate [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

Idealized multilinear maps => VBB [Barak, Garg, Kalai, Paneth, Sahai 14]

Multilinear Subgroup Elimination => iO [Gentry, Lewko, Sahai, Waters 15]

Multilinear maps



Obfuscation

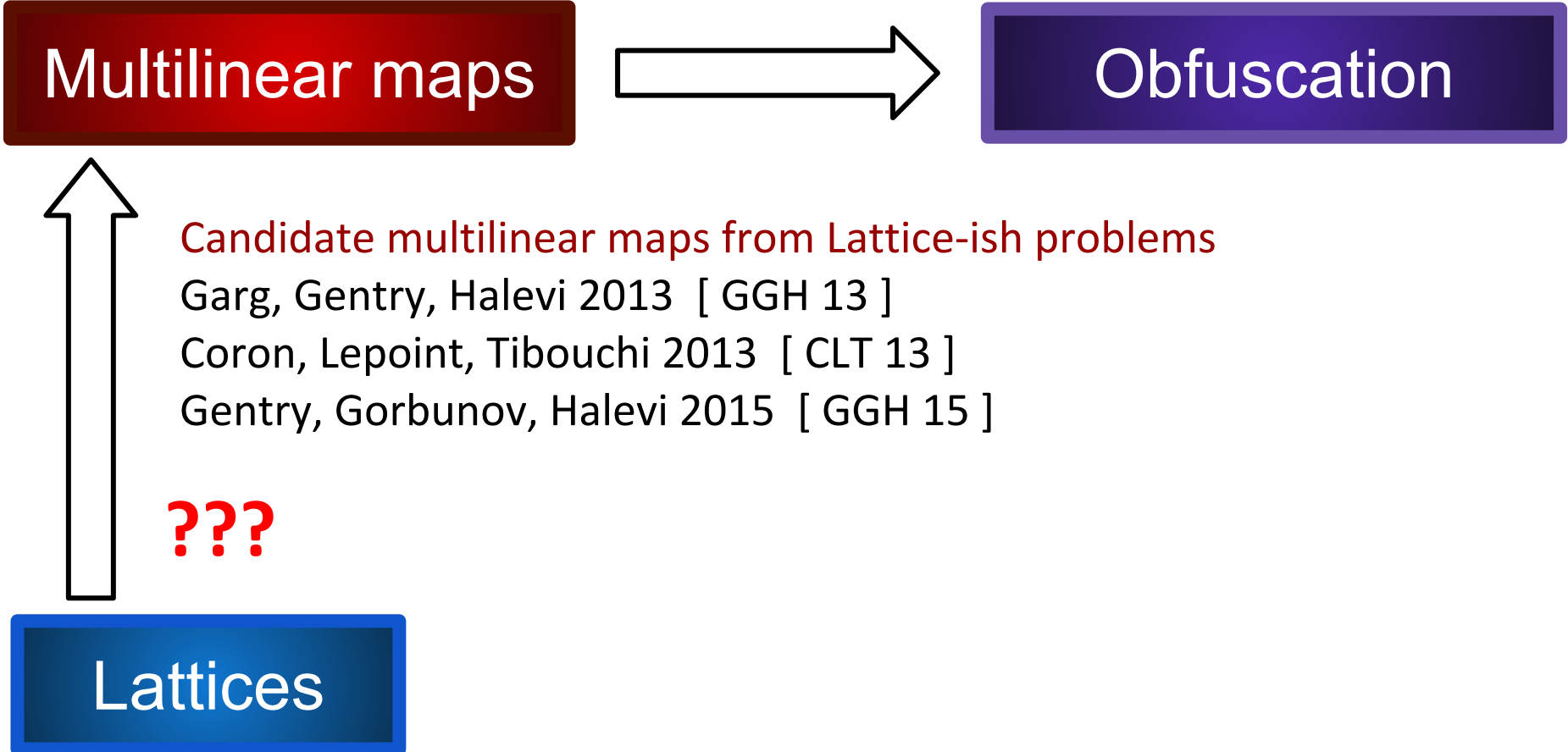
2013 - 2015

Multilinear maps => Obfuscation

Candidate [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

Idealized multilinear maps => VBB [Barak, Garg, Kalai, Paneth, Sahai 14]

Multilinear Subgroup Elimination => iO [Gentry, Lewko, Sahai, Waters 15]



2015

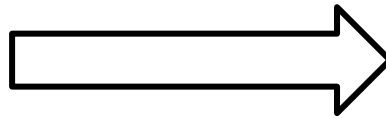
Multilinear maps => Obfuscation

Candidate [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

Idealized multilinear maps => VBB [Barak, Garg, Kalai, Paneth, Sahai 14]

Multilinear Subgroup Elimination => iO [Gentry, Lewko, Sahai, Waters 15]

Multilinear maps



Obfuscation

Candidate multilinear maps from Lattice-ish problems

Garg, Gentry, Halevi 2013 [GGH 13]

Coron, Lepoint, Tibouchi 2013 [CLT 13]

Gentry, Gorbunov, Halevi 2015 [GGH 15]

??????

Cryptanalysis of the Multilinear Map over the Integers
Cheon, Han, Lee, Ryu, Stehle [Eurocrypt 2015]

Lattice

2015

Status of multilinear maps and iO

	N party key exchange	iO [GGHRSW 13]
GGH13	Standing	Standing
CLT13	Broken [CHLRS 15]	Broken for simple program [CGHLMRST 15]
GGH15	Standing	Standing

Cryptanalysis of the Multilinear Map over the Integers

Cheon, Han, Lee, Ryu, Stehle

Zeroizing without low-level zeroes: New MMAP attacks and their limitations

Coron, Gentry, Halevi, Lepoint, Maji, Miles, Raykova, Sahai, Tibouchi

2016

Status of multilinear maps and iO

	N party key exchange	iO [GGHRSW 13]
GGH13	Broken [Hu, Jia 16]	Partial attack [Miles, Sahai, Zhandry 16]
CLT13	Broken [CHLRS 15]	Broken for simple program [CGHLMRST 15]
GGH15	Broken [CLLT 16]	Standing

Cryptanalysis of GGH map

Hu, Jia

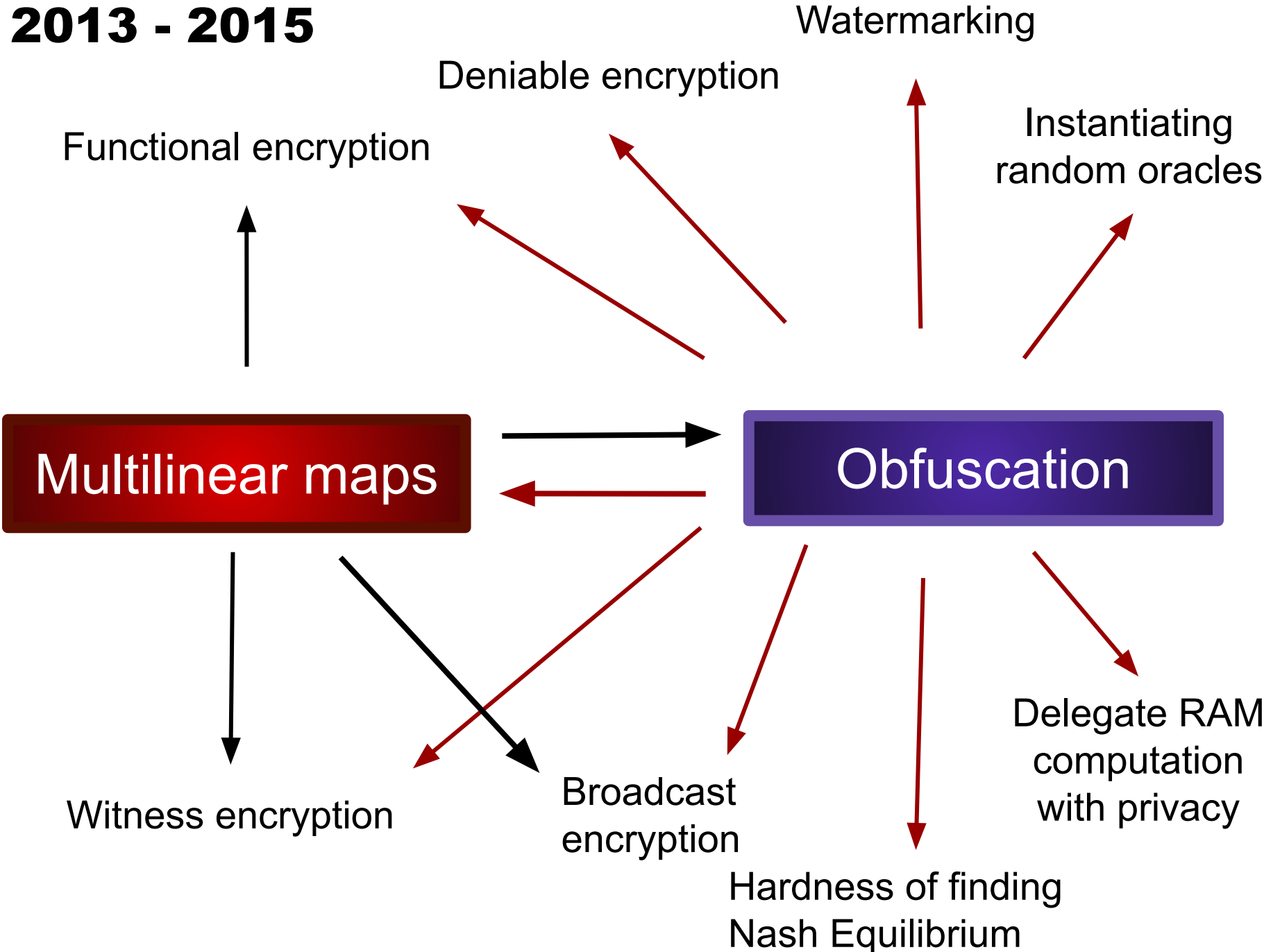
Annihilation attacks: Cryptanalysis of indistinguishability obfuscation over GGH13

Miles, Sahai, Zhandry

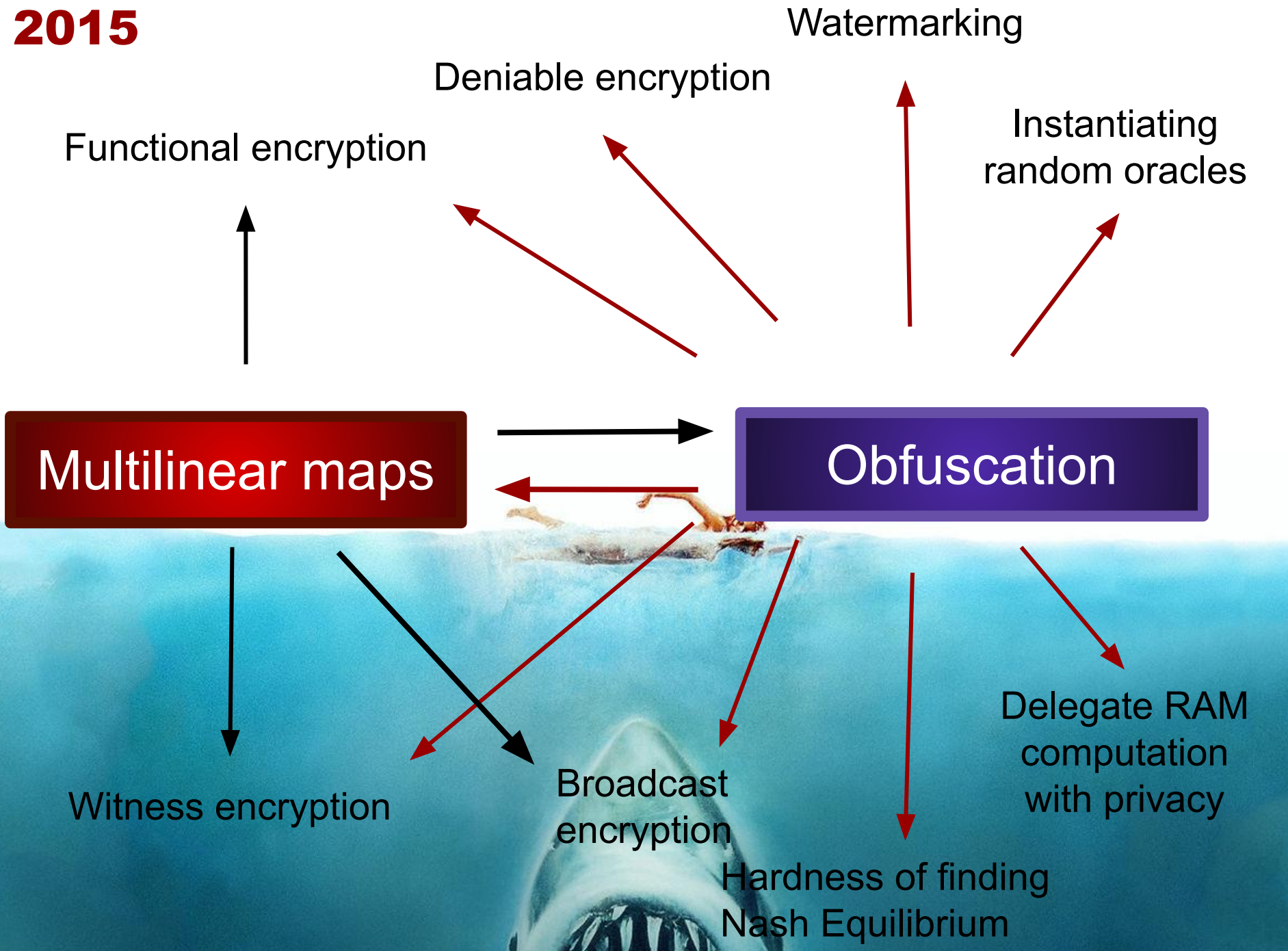
Cryptanalysis of GGH15 multilinear maps

Coron, Lee, Lepoint, Tibouchi

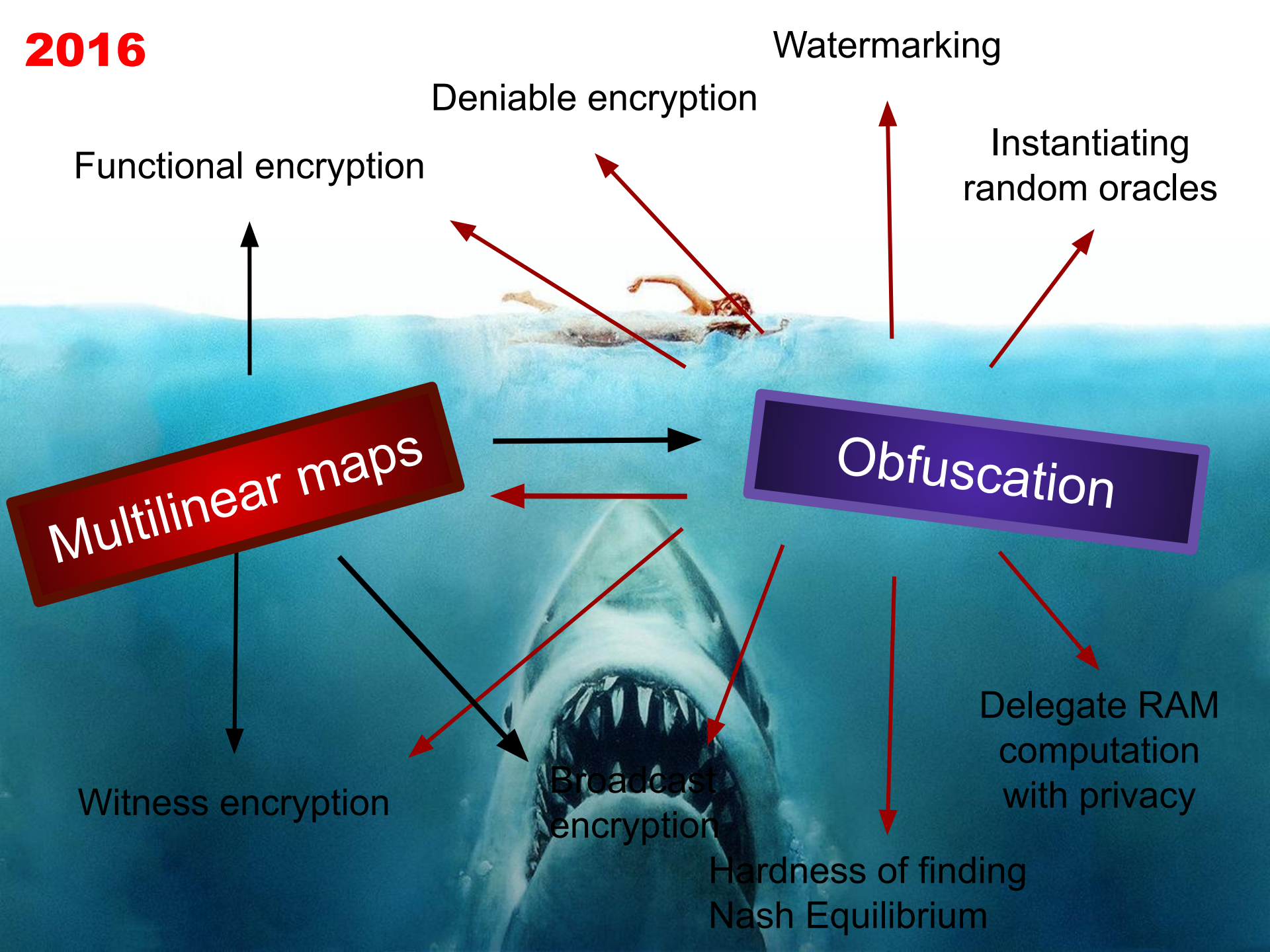
2013 - 2015



2015



2016



Watermarking

Deniable encryption

Instantiating random oracles

Functional encryption

Multilinear maps

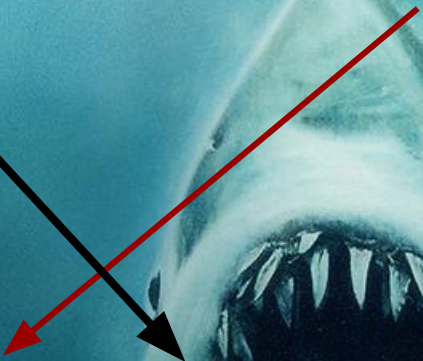
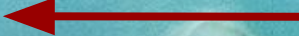
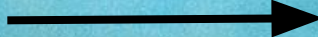
Obfuscation

Delegate RAM computation with privacy

Witness encryption

Broadcast encryption

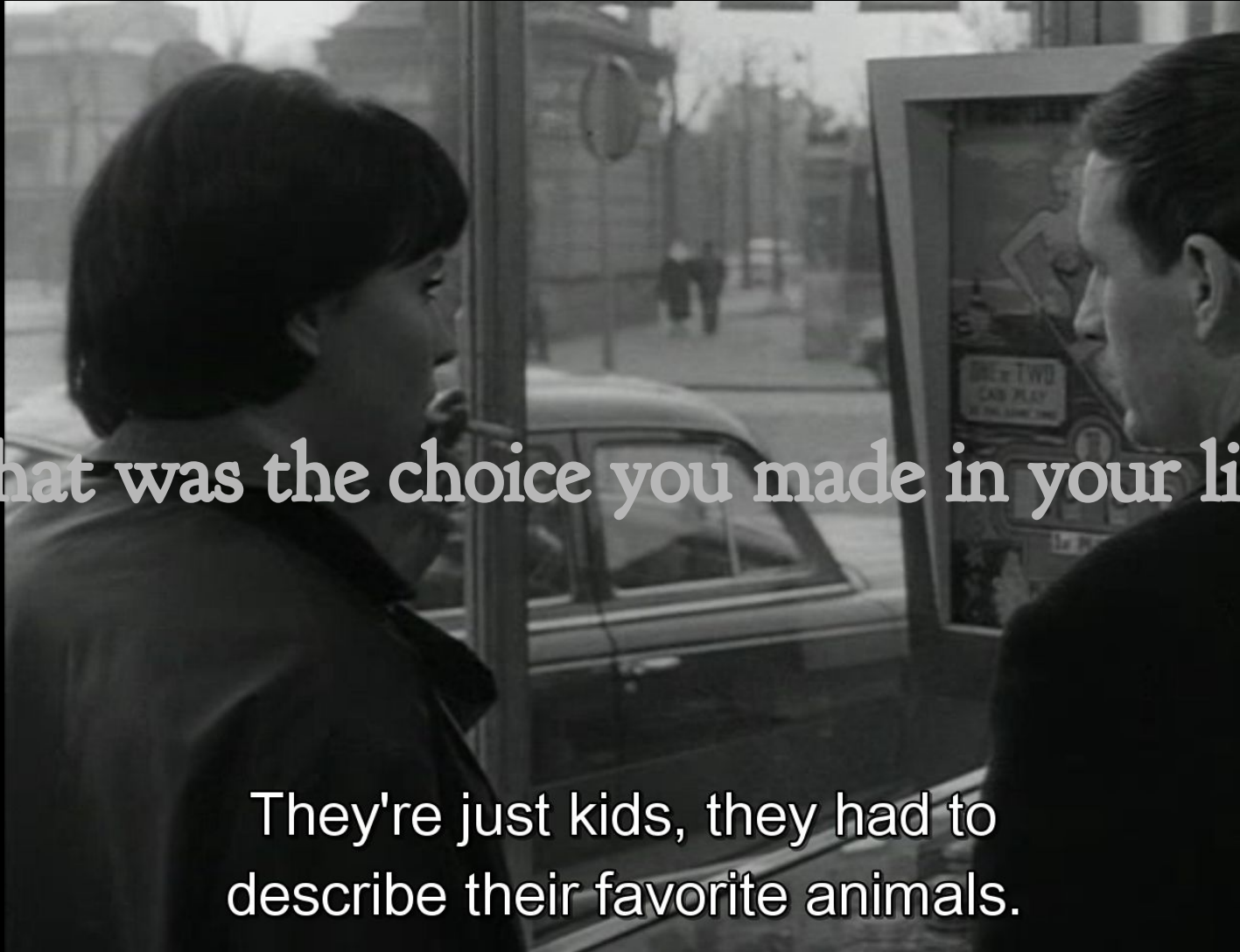
Hardness of finding Nash Equilibrium



2016

What was the choice you made in your life?

2016- 1962



What was the choice you made in your life?

They're just kids, they had to describe their favorite animals.

Vivre sa vie (Jean-Luc Godard)

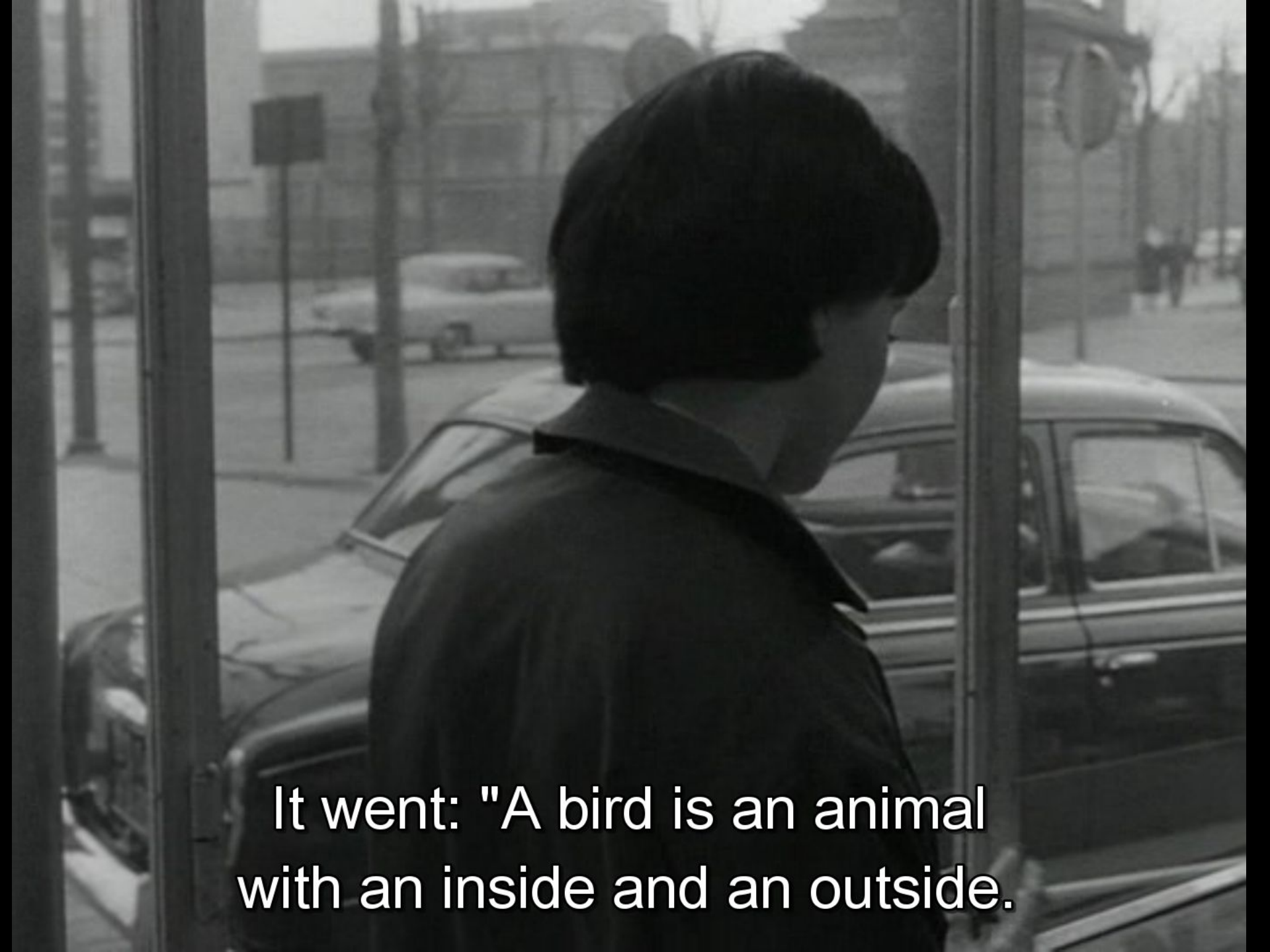
2016- 1962




What was the choice you made in your life?

One little girl of eight
chose a bird.


Vivre sa vie (Jean-Luc Godard)



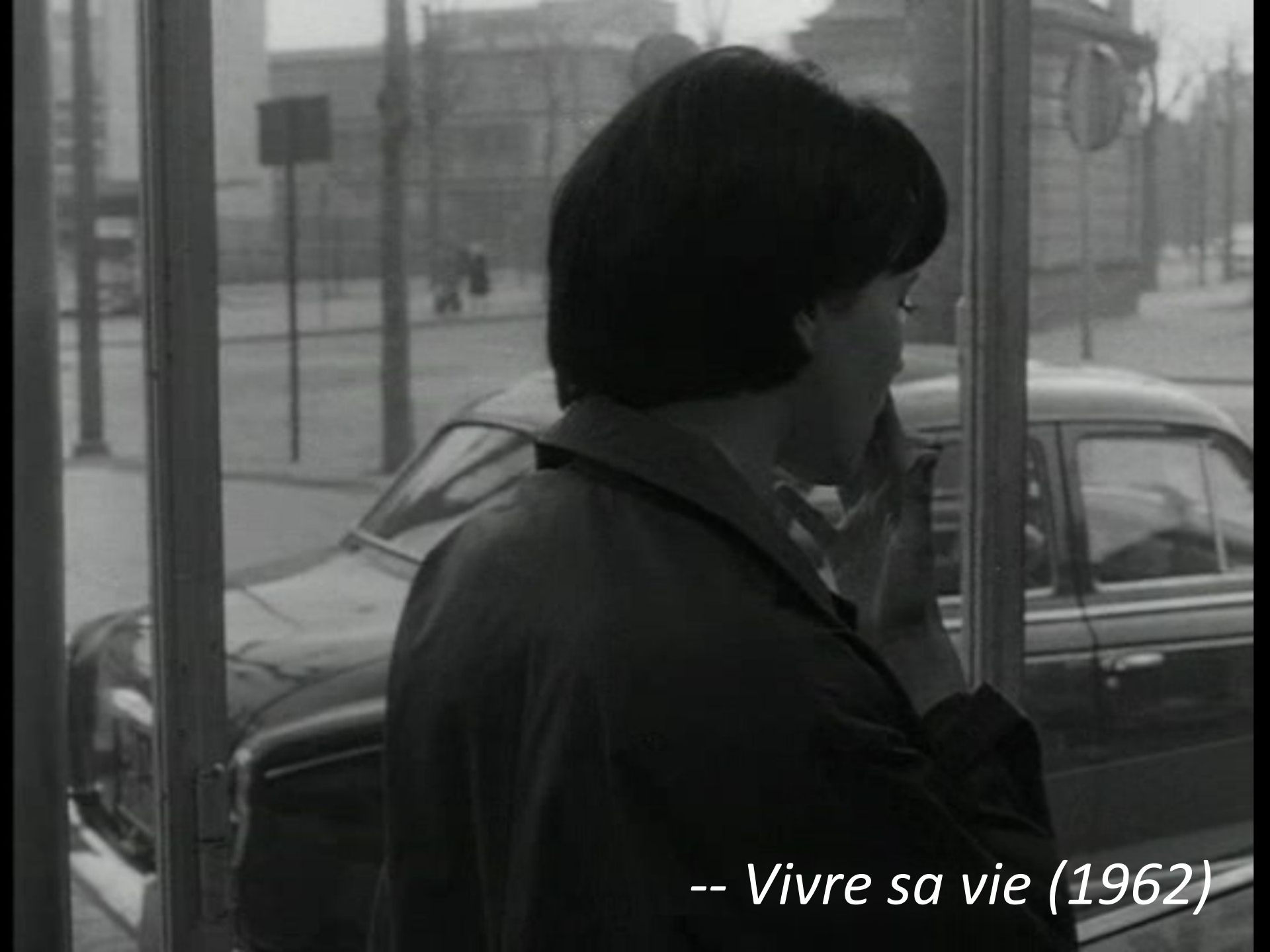
It went: "A bird is an animal
with an inside and an outside."



Remove the outside,
there's the inside.



Remove the inside and you see
the soul."



-- Vivre sa vie (1962)

2016

My choice: See the soul of obfuscation.

Today: the soul of obfuscation

- > Overview of obfuscators and attacks

- > Two attacks on obfuscators based on GGH15

 - Cryptanalyses of candidate branching program obfuscators

 - Chen, Gentry, Halevi [Eurocrypt 2017]

 - GGH15 beyond permutation branching programs

 - Chen, Vaikuntanathan, Wee [Crypto 2018]

- > Explain two interesting open problems

 - “NTRU without mod q ”

 - Better “rank attack” on Obfs & PRFs using quantum

> Overview

>>> How are the iO candidates doing?

>>> What are the general attack strategies?

>>> Why focus on GGH15?

iO => CRYPTO since 2013

Functional Encryption

Deniable encryption

Watermarking

Instantiating random oracles

Multilinear maps

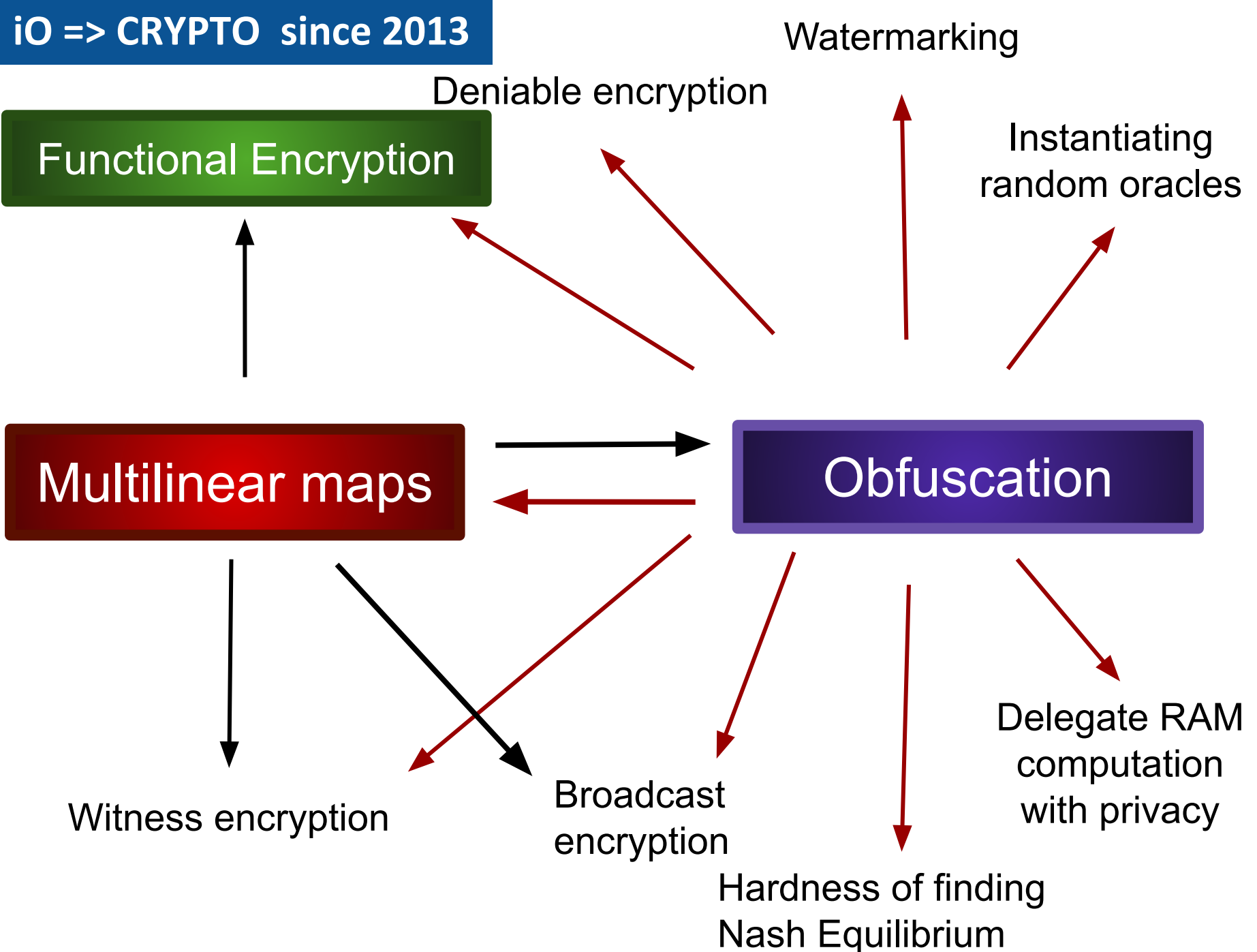
Obfuscation

Witness encryption

Broadcast encryption

Hardness of finding Nash Equilibrium

Delegate RAM computation with privacy



??? => iO

2020: none of them are “well understood”, still exploring.

LWE + Constant degree PRG +
Constant degree (≥ 3) multilinear maps

Algebraic
geometry*

LWE + bilinear maps
+ Low degree “PRG”

Candidate using lattices

Noisy-Linear FE

Candidate using affine
determinant program

Succinct Functional Encryption

Candidate using
tensor product*

Candidates
using lattices

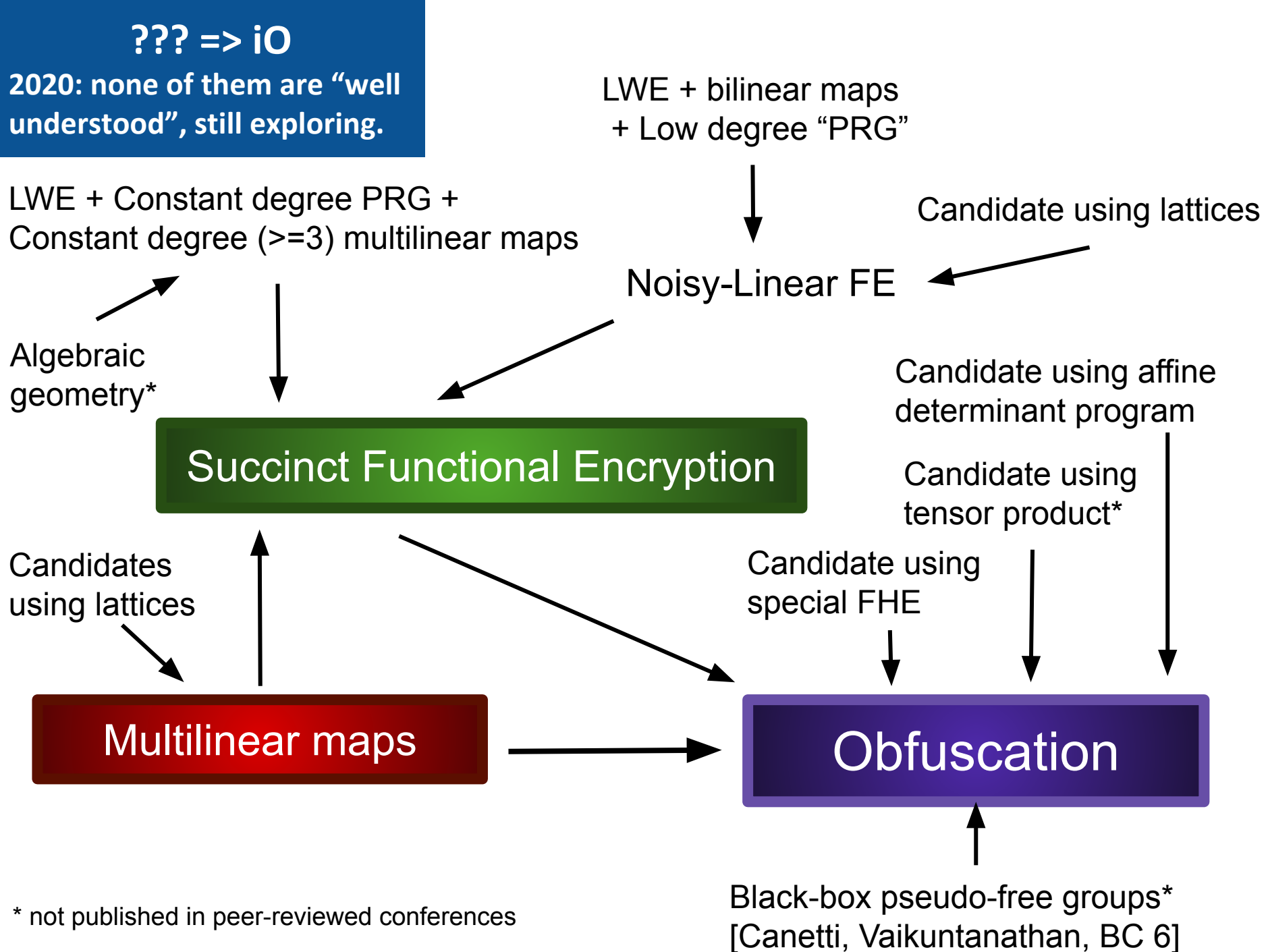
Candidate using
special FHE

Multilinear maps

Obfuscation

* not published in peer-reviewed conferences

Black-box pseudo-free groups*
[Canetti, Vaikuntanathan, BC 6]



General strategies of breaking iO:

In common:

0. Find two programs that are functionally equivalent, then distinguish the obfuscated version of them :)

General strategies of breaking iO:

In common:

0. Find two programs that are functionally equivalent, then distinguish the obfuscated version of them :)
1. Find equations over \mathbb{Z} , \mathbb{Q} , instead of over F_q .
2. Turn high degree equations into linear equations.

Looking ahead, there are many ways of preventing (2), but (1) seems to be hard to prevent in all the “noisy” iO candidates.

General strategies of breaking iO:

In common:

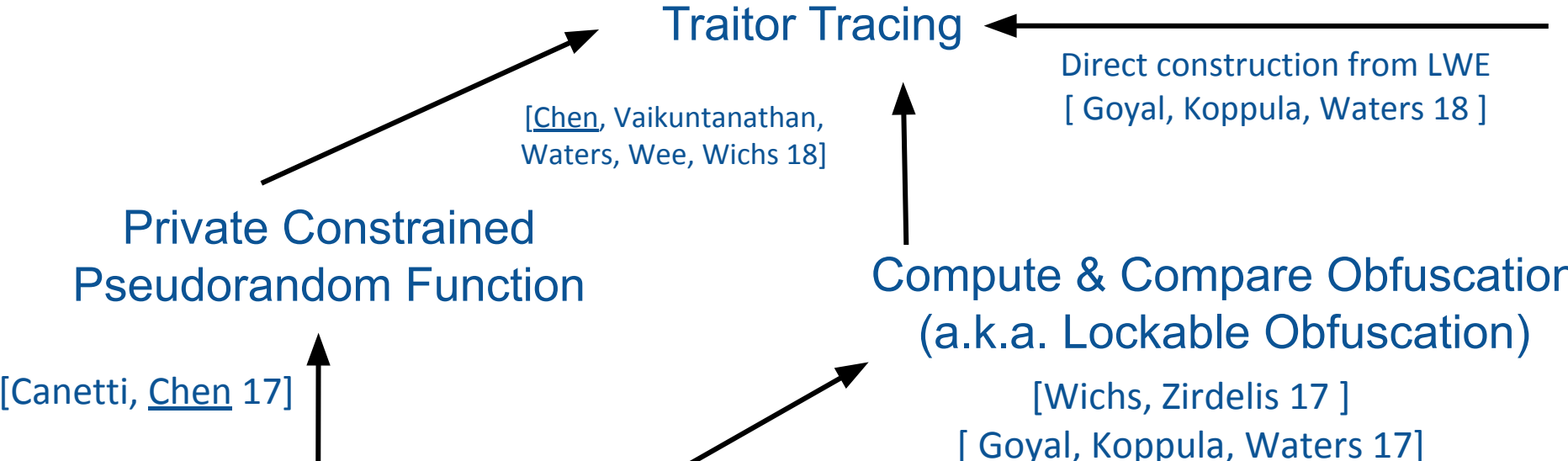
0. Find two programs that are functionally equivalent, then distinguish the obfuscated version of them :)
1. Find equations over \mathbb{Z} , \mathbb{Q} , instead of over F_q .
2. Turn high degree equations into linear equations.

Looking ahead, there are many ways of preventing (2), but (1) seems to be hard to prevent in all the “noisy” iO candidates.

Interesting attacks on specific components of obfuscation:

1. Lattice attacks:
 - Trace attack on the NTRU variant used in GGH13 [Cheon, Jeong, Lee 16]
 - Subfield attacks on overstretched NTRU [Albrecht, Bai, Ducas 16]
2. SOS attacks on low-degree “PRGs”: [Lombardi, Vaikuntanathan 17], [Barak, Brakerski, Komargodski, Kothari 18], [Barak, Hopkins, Jain, Kothari, Sahai 19]

Why focus on GGH15 (Gentry, Gorbunov, Halevi)?

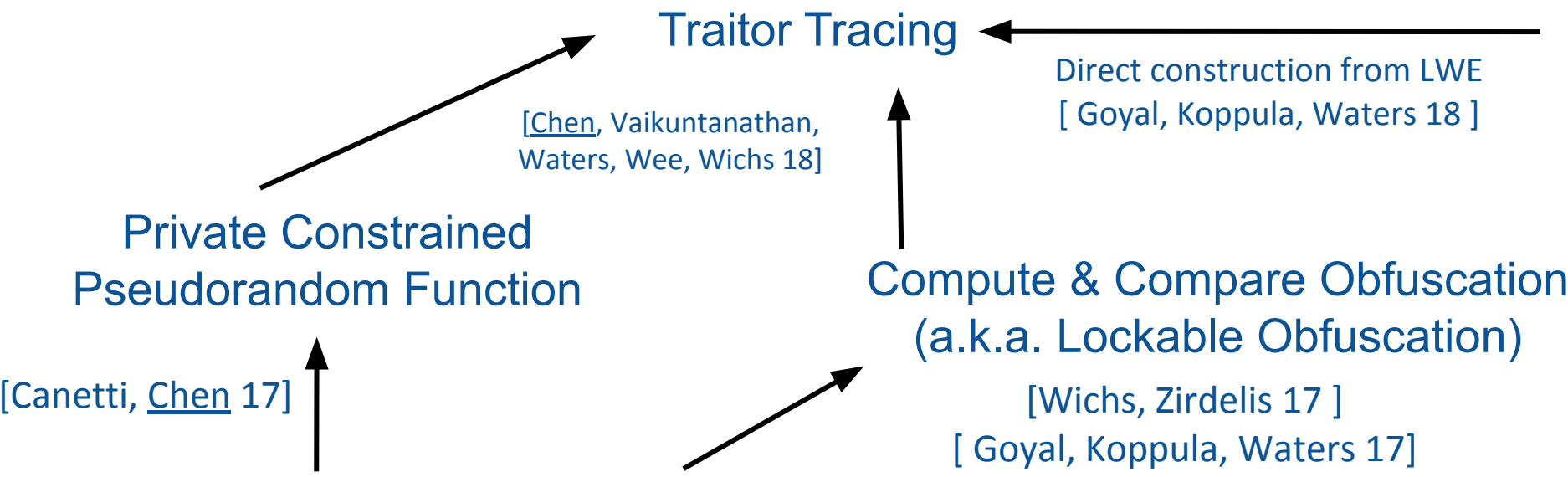


Safe modes of GGH15 multilinear maps

Lattices



Why focus on GGH15 (Gentry, Gorbunov, Halevi)?



Safe modes of GGH15



Lattices, Multilinear Maps and Program Obfuscation
Simons Institute • 343 views • Streamed 2 months ago
Yilei Chen (Visa Research) <https://simons.berkeley.edu/talks/advanced-lattices-etc-0> Lattices: ...



Constraint hiding constrained PRFs for NC1 from LWE
TheIACR • 277 views • 2 years ago
Paper by Ran Canetti and Yilei Chen presented at Eurocrypt 2017. See <https://iacr.org/cryptodb/data/paper.php?pubke>



Upcoming:

9:45 Venkata Koppula
> Lockable obfuscation

10:50 Rishab Goyal
> Traitor Tracing



Remove the inside and you see
the soul."

Start now: the soul of obfuscation

à la [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13] + GGH15 multilinear maps

Candidate iO from [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

(0) Representation of plaintext program.

(1) Safeguard 1

(2) Safeguard 2

(3) Safeguard 3

(4) Wrap (0-3) by multilinear maps (GGH13, CLT13, or GGH15)

Safeguards aim at randomizing the plaintext program, preventing illegal operations; mmaps is the source of “computational hardness”

$$g, g^{s_1}, \dots, g^{s_k} \rightarrow g^{\prod s}$$

Multilinear maps in the
group representation

Candidate iO from [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

(0) Representation of plaintext program: Oblivious branching program

(1) Safeguard 1

(2) Safeguard 2

(3) Safeguard 3

(4) Wrap (0-3) by multilinear maps (GGH13, CLT13, or GGH15)

Candidate iO from [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

(0) Representation of plaintext program: Oblivious branching program

(1) **Safeguard 1: Kilian randomization [Kilian 88]**

(2) Safeguard 2

(3) Safeguard 3

(4) Wrap (0-3) by **multilinear maps (GGH13, CLT13, or GGH15)**



1	$B_{1,1} K_1$	$K_1^{-1} B_{2,1} K_2$	$K_2^{-1} B_{3,1} K_3$	$K_3^{-1} B_{4,1}$
0	$B_{1,0} K_1$	$K_1^{-1} B_{2,0} K_2$	$K_2^{-1} B_{3,0} K_3$	$K_3^{-1} B_{4,0}$
i	1	2	1	2

Random
matrix K, K'

1	$B'_{1,1} K'_1$	$K'^{-1}_{1} B'_{2,1} K'_2$	$K'^{-1}_{2} B'_{3,1} K'_3$	$K'^{-1}_{3} B'_{4,1}$
0	$B'_{1,0} K'_1$	$K'^{-1}_{1} B'_{2,0} K'_2$	$K'^{-1}_{2} B'_{3,0} K'_3$	$K'^{-1}_{3} B'_{4,0}$
i	1	2	1	2

Candidate iO from [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

(0) Representation of plaintext program: Oblivious branching program

(1) Safeguard 1: Kilian randomization

(2) Safeguard 2: Bundling scalars (against mix-input attack)

(3) Safeguard 3

(4) Wrap (0-3) by multilinear maps (GGH13, CLT13, or GGH15)



1	$a_{1,1} B_{1,1} K_1$	$a_{2,1} K_1^{-1} B_{2,1} K_2$	$a_{3,1} K_2^{-1} B_{3,1} K_3$	$a_{4,1} K_3^{-1} B_{4,1}$	$a_{1,1} a_{3,1} = a'_{1,1} a'_{3,1}$ $a_{1,0} a_{3,0} = a'_{1,0} a'_{3,0}$ $a_{2,1} a_{4,1} = a'_{2,1} a'_{4,1}$ $a_{2,0} a_{4,0} = a'_{2,0} a'_{4,0}$
0	$a_{1,0} B_{1,0} K_1$	$a_{2,0} K_1^{-1} B_{2,0} K_2$	$a_{3,0} K_2^{-1} B_{3,0} K_3$	$a_{4,0} K_3^{-1} B_{4,0}$	
i	1	2	1	2	

1	$a'_{1,1} B'_{1,1} K'_1$	$a'_{2,1} K'^{-1}_{1,1} B'_{2,1} K'_2$	$a'_{3,1} K'^{-1}_{2,1} B'_{3,1} K'_3$	$a'_{4,1} K'^{-1}_{3,1} B'_{4,1}$
0	$a'_{1,0} B'_{1,0} K'_1$	$a'_{2,0} K'^{-1}_{1,0} B'_{2,0} K'_2$	$a'_{3,0} K'^{-1}_{2,0} B'_{3,0} K'_3$	$a'_{4,0} K'^{-1}_{3,0} B'_{4,0}$
i	1	2	1	2

Candidate iO from [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

(0) Representation of plaintext program: Oblivious branching program

(1) Safeguard 1: Kilian randomization

(2) Safeguard 2: Bundling scalars (against mix-input attack)

(3) Safeguard 3: random diagonal entries and bookends

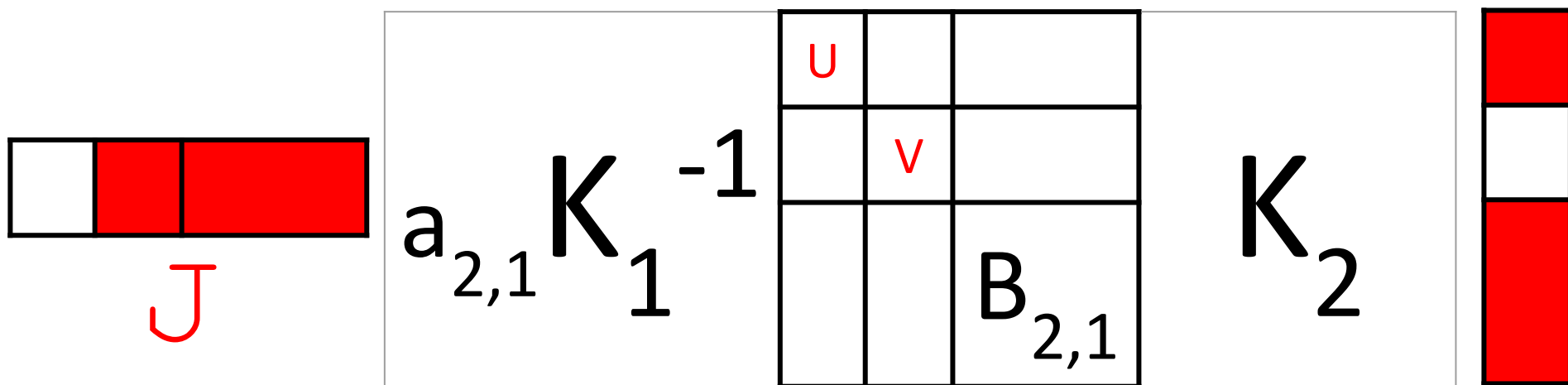
(4) Wrap (0-3) by **multilinear maps (GGH13, CLT13, or GGH15)**



1	$a_{1,1} J B_{1,1} K_1$	$a_{2,1} K_1^{-1} B_{2,1} K_2$	$a_{3,1} K_2^{-1} B_{3,1} K_3$	$a_{4,1} K_3^{-1} B_{4,1} L$
0	$a_{1,0} J B_{1,0} K_1$	$a_{2,0} K_1^{-1} B_{2,0} K_2$	$a_{3,0} K_2^{-1} B_{3,0} K_3$	$a_{4,0} K_3^{-1} B_{4,0} L$
i	1	2	1	2

1	$a'_{1,1} J' B'_{1,1} K'_1$	$a'_{2,1} K'_{1,1}^{-1} B'_{2,1} K'_2$	$a'_{3,1} K'_{2,2}^{-1} B'_{3,1} K'_3$	$a'_{4,1} K'_{3,3}^{-1} B'_{4,1} L'$
0	$a'_{1,0} J' B'_{1,0} K'_1$	$a'_{2,0} K'_{1,1}^{-1} B'_{2,0} K'_2$	$a'_{3,0} K'_{2,2}^{-1} B'_{3,0} K'_3$	$a'_{4,0} K'_{3,3}^{-1} B'_{4,0} L'$
i	1	2	1	2

Zoom in: random diagonal entries and bookends



$$S_{1,1} = a_{1,1} J \begin{bmatrix} * \\ B_{1,1} \end{bmatrix} K_1$$

$$S_{2,1} = a_{2,1} K_1^{-1} \begin{bmatrix} * \\ B_{2,1} \end{bmatrix} K_2$$

$$S_{h,1} = a_{h,1} K_{h-1}^{-1} \begin{bmatrix} * \\ B_{h,1} \end{bmatrix} L$$

1	$S_{1,1}$	$S_{2,1}$...	$S_{h,1}$
0	$S_{1,0}$	$S_{2,0}$...	$S_{h,0}$
i	i_1	i_2	...	i_h

Candidate iO from [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

(0) Representation of plaintext program: Oblivious branching program

(1) Safeguard 1: Kilian randomization

(2) Safeguard 2: Bundling scalars

(3) Safeguard 3: random diagonal entries and bookends

(4) Wrap (0-3) by **multilinear maps (GGH13, CLT13, or GGH15)**



1	$a_{1,1} J B_{1,1} K_1$	$a_{2,1} K_1^{-1} B_{2,1} K_2$	$a_{3,1} K_2^{-1} B_{3,1} K_3$	$a_{4,1} K_3^{-1} B_{4,1} L$
0	$a_{1,0} J B_{1,0} K_1$	$a_{2,0} K_1^{-1} B_{2,0} K_2$	$a_{3,0} K_2^{-1} B_{3,0} K_3$	$a_{4,0} K_3^{-1} B_{4,0} L$
i	1	2	1	2

1	$a'_{1,1} J' B'_{1,1} K'_1$	$a'_{2,1} K'^{-1}_{1} B'_{2,1} K'_2$	$a'_{3,1} K'^{-1}_{2} B'_{3,1} K'_3$	$a'_{4,1} K'^{-1}_{3} B'_{4,1} L'$
0	$a'_{1,0} J' B'_{1,0} K'_1$	$a'_{2,0} K'^{-1}_{1} B'_{2,0} K'_2$	$a'_{3,0} K'^{-1}_{2} B'_{3,0} K'_3$	$a'_{4,0} K'^{-1}_{3} B'_{4,0} L'$
i	1	2	1	2

Candidate iO from [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

(0) Representation of plaintext program: Oblivious branching program

(1) Safeguard 1: Kilian randomization

(2) Safeguard 2: Bundling scalars

(3) Safeguard 3: random diagonal entries and bookends

(4) Wrap (0-3) by multilinear maps (GGH13, CLT13, or GGH15)

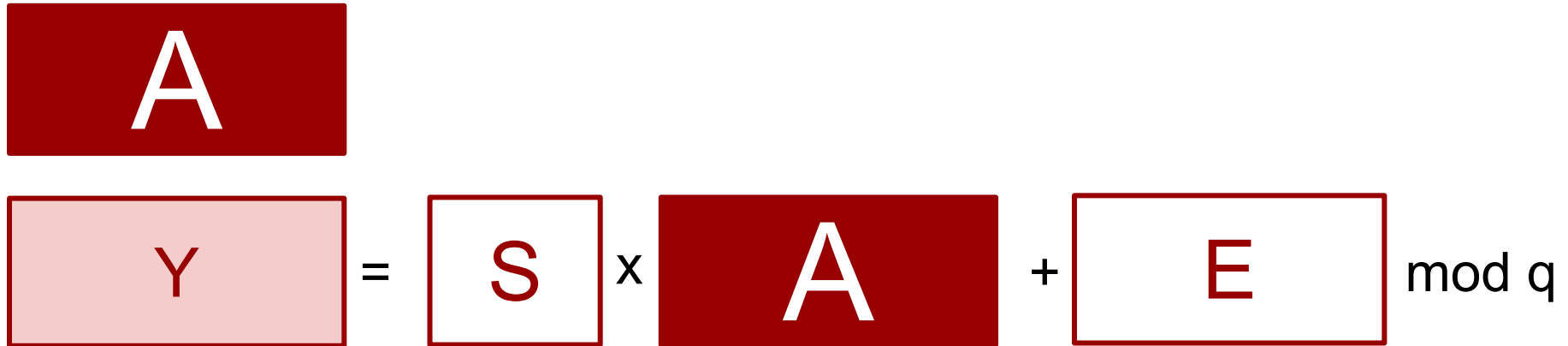
1	$a_{1,1} J B_{1,1} K_1$	$a_{2,1} K_1^{-1} B_{2,1} K_2$	$a_{3,1} K_2^{-1} B_{3,1} K_3$	$a_{4,1} K_3^{-1} B_{4,1} L$
0	$a_{1,0} J B_{1,0} K_1$	$a_{2,0} K_1^{-1} B_{2,0} K_2$	$a_{3,0} K_2^{-1} B_{3,0} K_3$	$a_{4,0} K_3^{-1} B_{4,0} L$
i	1	2	1	2



Already lost?

*Then remember "Barrington"
and "bundling scalars"*

**Recap of GGH15 multilinear maps
[Gentry, Gorbunov, Halevi 15]**

$$A = SA + E \pmod{q}$$


Learning with errors [Regev 2005]

Search LWE: Given A , $Y = SA + E \pmod{q}$, find S ,

Decisional LWE: Distinguish A , Y from random.

What is GGH15 trying to do?

$$g, g^{s_1}, \dots, g^{s_k} \rightarrow g^{\prod s}$$

Multilinear maps in the group representation

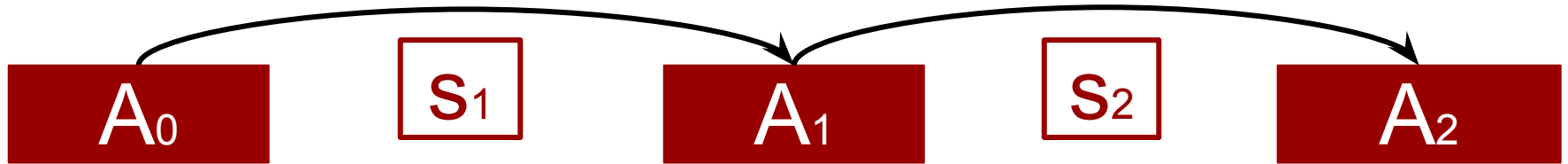
$$A, s_1 A + E_1, \dots, s_k A + E_k \rightarrow \prod s A + E$$

GGH15: (Ring)LWE analogy

The difficulty is to compute the map without revealing the secrets.

GGH15 for 1 multiplication

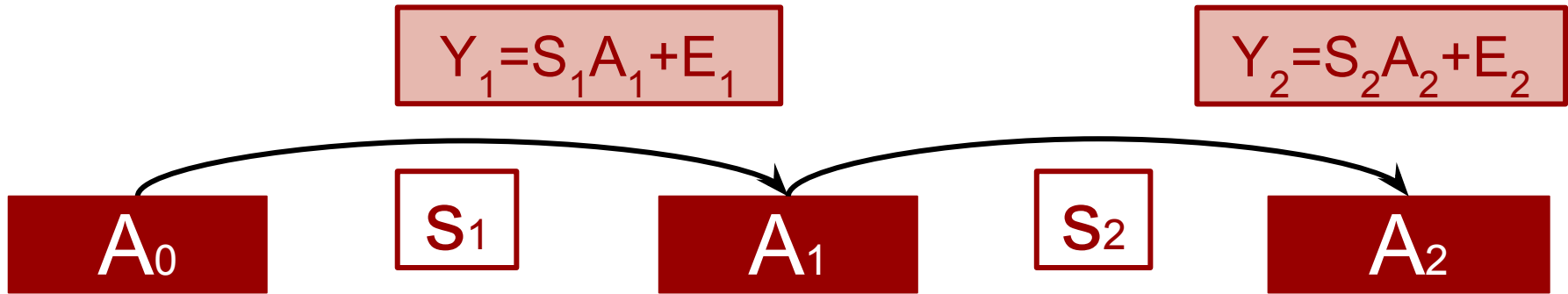
Uniform Small Unclear



Encoding(S_i):

GGH15 for 1 multiplication

Uniform Small Unclear

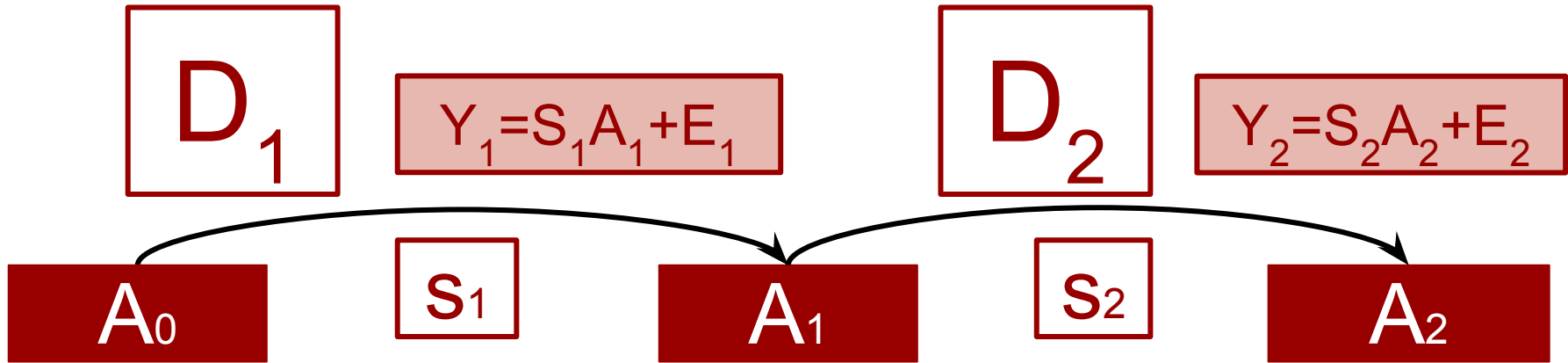


Encoding(S_i): 2 steps

1. Compute $Y_i = S_i A_i + E_i$

GGH15 for 1 multiplication

Uniform Small Unclear



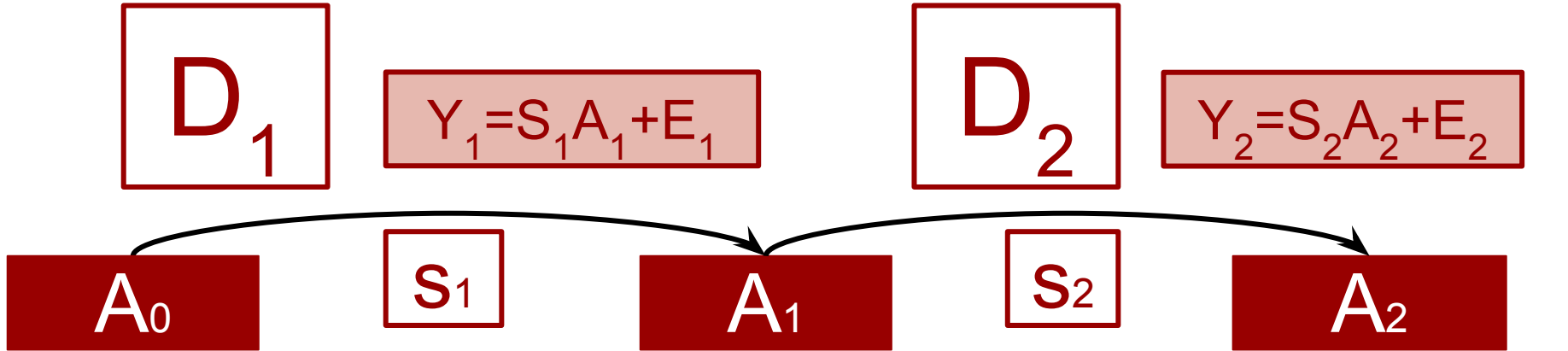
Encoding(S_i): 2 steps

1. Compute $Y_i = S_i A_i + E_i$
2. Sample (by the trapdoor of A_{i-1}) small D_i such that $A_{i-1} D_i = Y_i$

$$D_i = \text{Encoding}(S_i)$$

GGH15 for 1 multiplication

Uniform Small Unclear



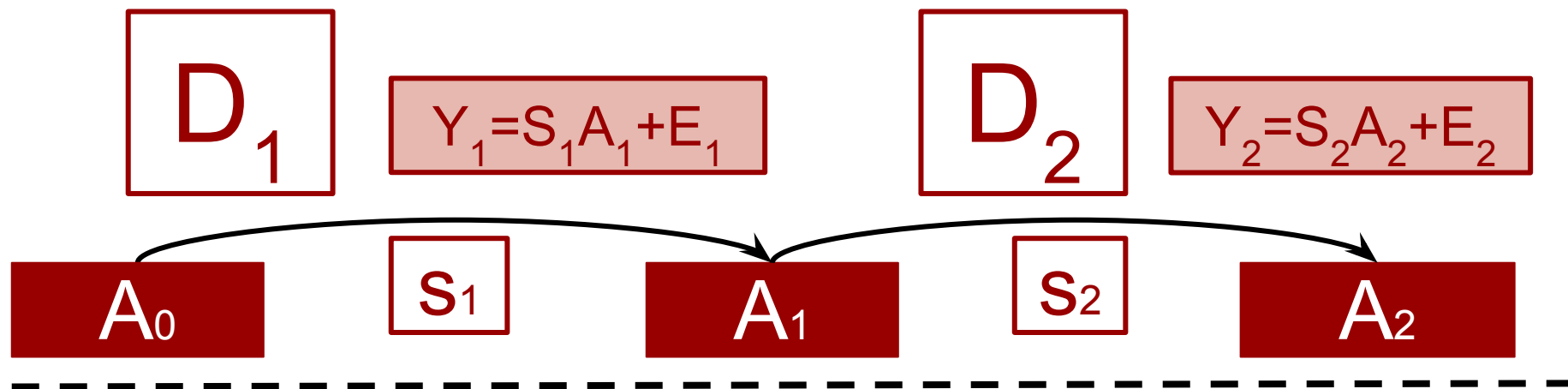
Functionality

$$A_0 \times D_1 \times D_2$$

Recall the goal: $S_1 A + E_1, \dots, S_k A + E_k \rightarrow \prod S A + E$

GGH15 for 1 multiplication

Uniform Small Unclear

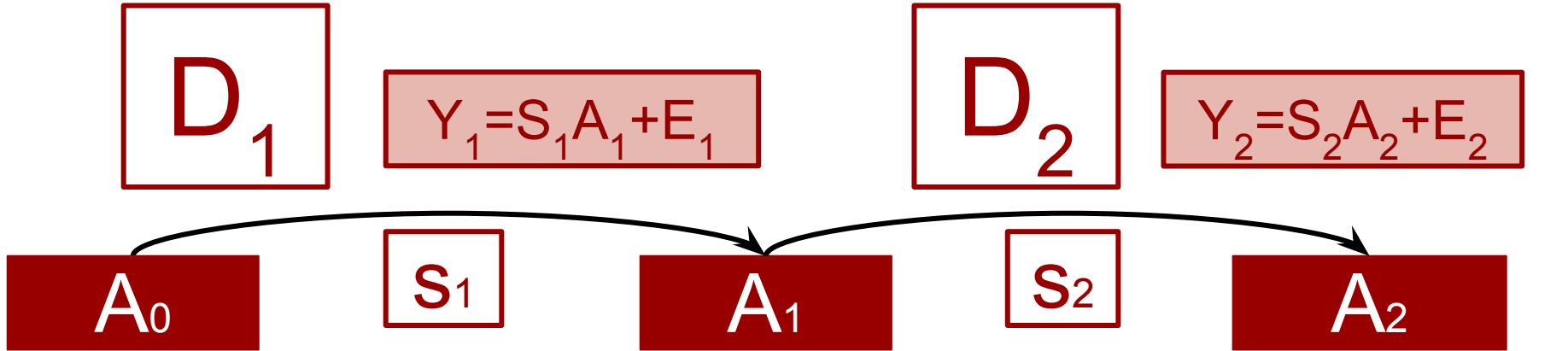


Functionality

$$A_0 \times D_1 \times D_2 = S_1 A_1 D_2 + E_1 D_2$$

GGH15 for 1 multiplication

Uniform Small Unclear

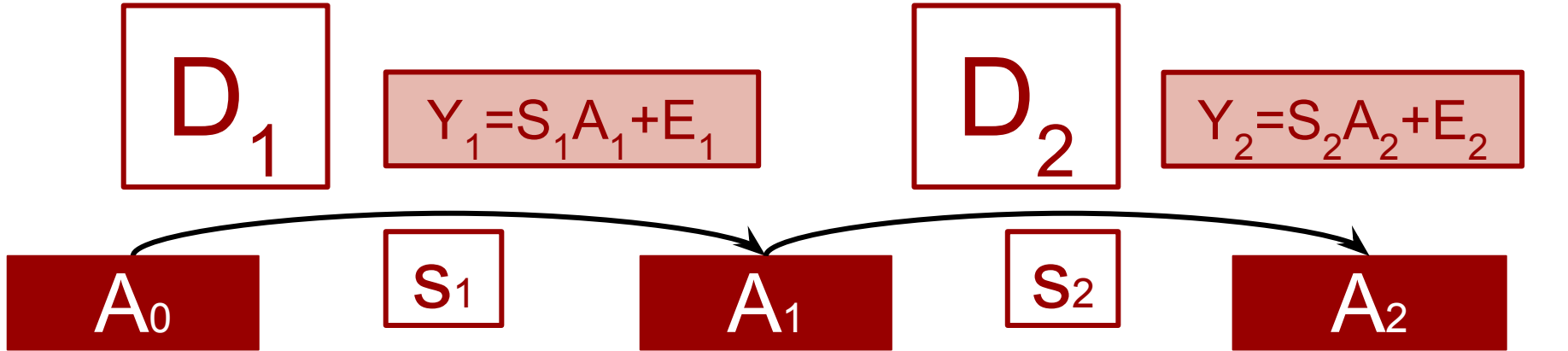


Functionality

$$\begin{aligned}
 & A_0 \times D_1 \times D_2 \\
 = & S_1 A_1 D_2 + E_1 D_2 \\
 = & S_1 S_2 A_2 + S_1 E_2 + E_1 D_2
 \end{aligned}$$

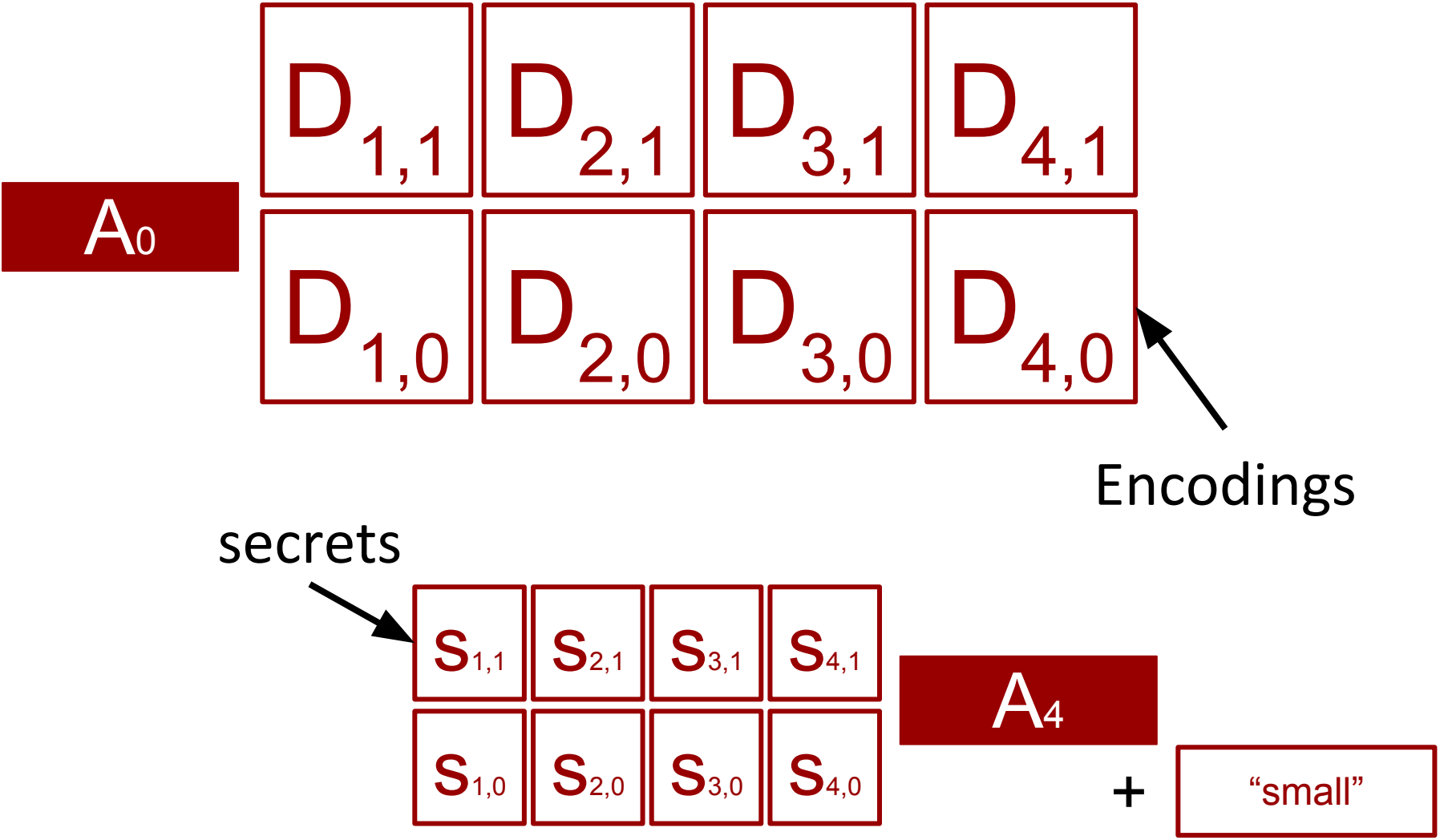
GGH15 for 1 multiplication

Uniform Small Unclear

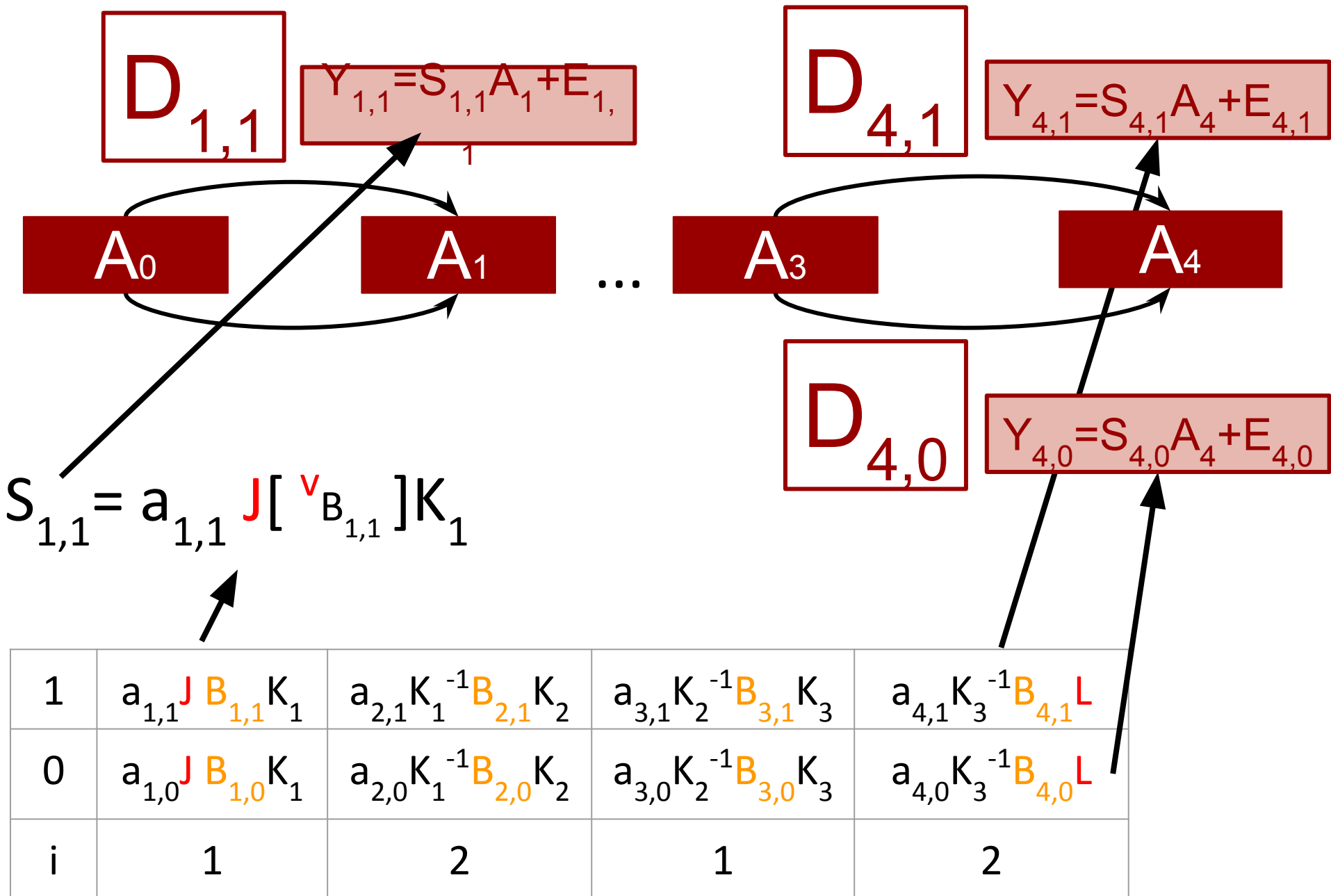


Functionality

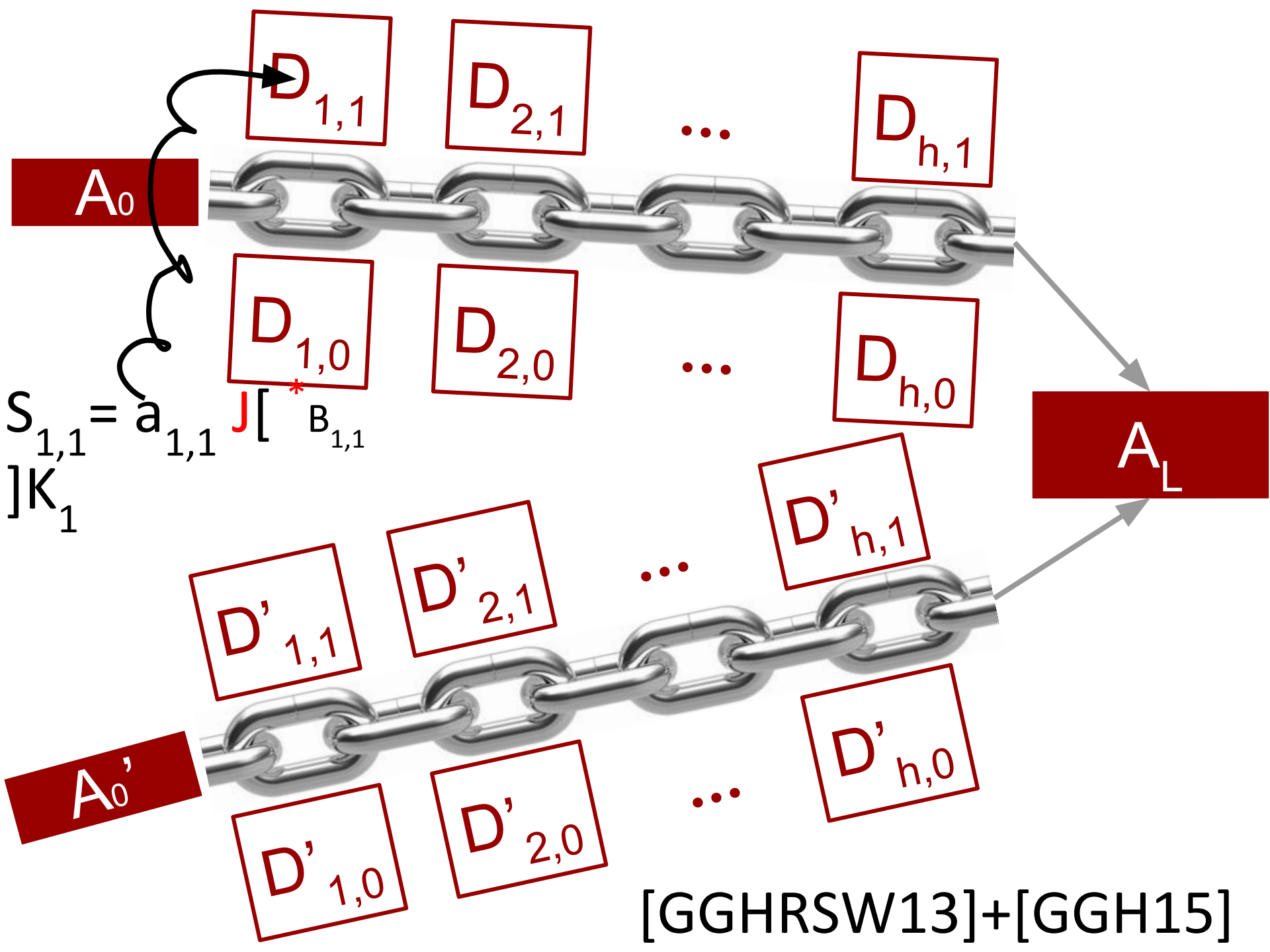
$$\begin{aligned}
 & A_0 \times D_1 \times D_2 \\
 = & S_1 A_1 D_2 + E_1 D_2 \\
 = & S_1 S_2 A_2 + S_1 E_2 + E_1 D_2 \\
 = & S_1 S_2 A_2 + \text{"small"}
 \end{aligned}$$



A typical evaluation pattern for GGH15: subset product



[GGHRSW13]+[GGH15]





Remove the inside and you see
the soul."

Where is the soul?

à la GGHRSW13 Obfuscation + GGH15 multilinear maps

Candidate iO from [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

(0) Representation of plaintext program: Oblivious branching program

(1) Safeguard 1: Kilian randomization

(2) Safeguard 2: Bundling scalars

(3) Safeguard 3: random diagonal entries and bookends

(4) Wrap (0-3) by GGH15

1	$a_{1,1} J B_{1,1} K_1$	$a_{2,1} K_1^{-1} B_{2,1} K_2$	$a_{3,1} K_2^{-1} B_{3,1} K_3$	$a_{4,1} K_3^{-1} B_{4,1} L$
0	$a_{1,0} J B_{1,0} K_1$	$a_{2,0} K_1^{-1} B_{2,0} K_2$	$a_{3,0} K_2^{-1} B_{3,0} K_3$	$a_{4,0} K_3^{-1} B_{4,0} L$
i	1	2	1	2

Attack: weakness of GGH15 + the weakness in (0)-(3)

Key idea of the attack (from [Coron, Lee, Lepoint, Tibouchi 16])

$$\begin{aligned} & \mathbf{A}_0 \times \mathbf{D}_1 \times \mathbf{D}_2 \pmod q \\ = & \mathbf{S}_1 \mathbf{S}_2 \mathbf{A}_2 + \mathbf{S}_1 \mathbf{E}_2 + \mathbf{E}_1 \mathbf{D}_2 \end{aligned}$$

Key idea of the attack (from [Coron, Lee, Lepoint, Tibouchi 16])

$$\begin{aligned} & \mathbf{A}_0 \times \mathbf{D}_1 \times \mathbf{D}_2 \pmod{q} \\ = & \mathbf{S}_1 \mathbf{S}_2 \mathbf{A}_2 + \mathbf{S}_1 \mathbf{E}_2 + \mathbf{E}_1 \mathbf{D}_2 \end{aligned}$$

If $\mathbf{S}_1 \mathbf{S}_2 = 0$, then

$$\begin{aligned} & \mathbf{A}_0 \times \mathbf{D}_1 \times \mathbf{D}_2 \pmod{q} \\ = & \mathbf{S}_1 \mathbf{E}_2 + \mathbf{E}_1 \mathbf{D}_2, \text{ holds over } \mathbb{Z} \end{aligned}$$

Key idea of the attack (from [Coron, Lee, Lepoint, Tibouchi 16])

If $\boxed{s_1} \boxed{s_2} = 0$, then

$$\boxed{A_0} \times \boxed{D_1} \times \boxed{D_2} \pmod q$$

$$= \boxed{s_1} \boxed{E_2} + \boxed{E_1} \boxed{D_2}$$

Key idea of the attack (from [Coron, Lee, Lepoint, Tibouchi 16])

If $\boxed{s_1} \boxed{s_2} = 0$, then

$$\boxed{A_0} \times \boxed{D_1} \times \boxed{D_2} \pmod q$$

$$= \boxed{s_1} \boxed{E_2} + \boxed{E_1} \boxed{D_2}$$

$$= \boxed{s_1 \mid E_1} \times \begin{array}{|c} E_2 \\ \hline D_2 \end{array}$$

Key idea of the attack (from [Coron, Lee, Lepoint, Tibouchi 16])

$$\begin{aligned} \text{Let } \boxed{W_{xz}} &= \boxed{A_0} \times \boxed{D_x} \times \boxed{D_z} \pmod{q} \\ &= \boxed{S_x \mid E_x} \times \begin{array}{|c|} \hline E_z \\ \hline D_z \\ \hline \end{array} \end{aligned}$$

Key idea of the attack (from [Coron, Lee, Lepoint, Tibouchi 16])

$$\text{Let } \boxed{W_{xz}} = \boxed{A_0} \times \boxed{D_x} \times \boxed{D_z} \pmod q$$
$$= \boxed{S_x \mid E_x} \times \begin{array}{|c|} \hline E_z \\ \hline D_z \\ \hline \end{array}$$

If we have more inputs at x and z, say $x_1, x_2, x_3, z_1, z_2, \dots$ then

$$\begin{array}{|c|c|} \hline W_{x_1 z_1} & W_{x_1 z_2} \\ \hline W_{x_2 z_1} & W_{x_2 z_2} \\ \hline W_{x_3 z_1} & W_{x_3 z_2} \\ \hline \end{array} = \begin{array}{|c|c|} \hline S_{x_1} \mid E_{x_1} \\ \hline S_{x_2} \mid E_{x_2} \\ \hline S_{x_3} \mid E_{x_3} \\ \hline \end{array} \times \begin{array}{|c|c|} \hline E_{z_1} & E_{z_2} \\ \hline D_{z_1} & D_{z_2} \\ \hline \end{array}$$

Key idea of the attack (from [Coron, Lee, Lepoint, Tibouchi 16])

By evaluating on the combinations of $x_1, x_2, x_3, z_1, z_2, \dots$, we got

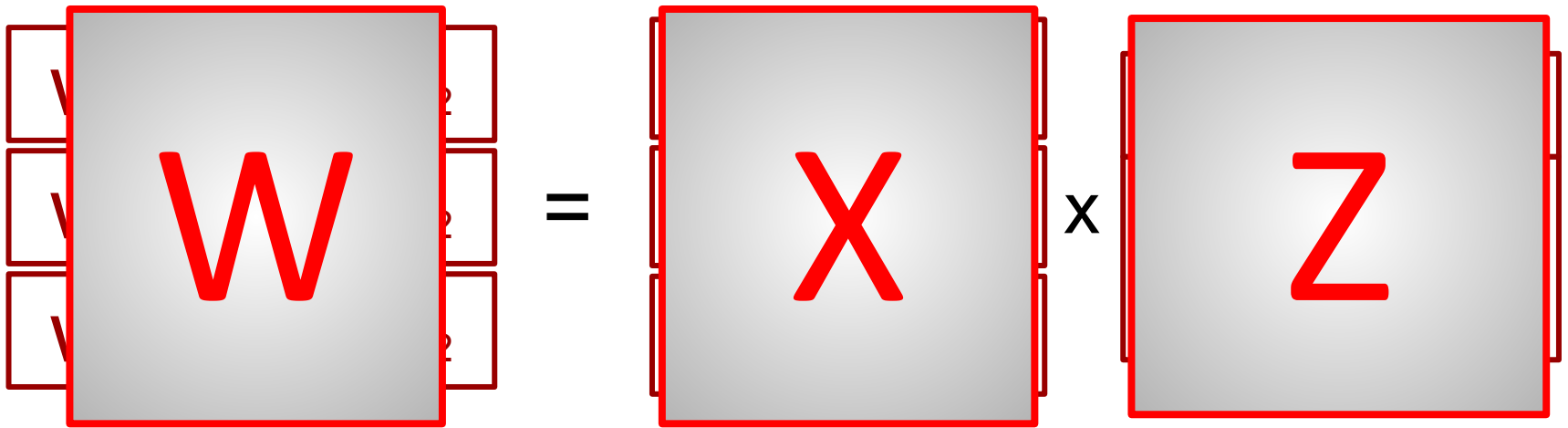
$$\begin{array}{|c|c|} \hline W_{x_1 z_1} & W_{x_1 z_2} \\ \hline W_{x_2 z_1} & W_{x_2 z_2} \\ \hline W_{x_3 z_1} & W_{x_3 z_2} \\ \hline \end{array} = \begin{array}{|c|c|} \hline S_{x_1} & E_{x_1} \\ \hline S_{x_2} & E_{x_2} \\ \hline S_{x_3} & E_{x_3} \\ \hline \end{array} \times \begin{array}{|c|c|} \hline E_{z_1} & E_{z_2} \\ \hline D_{z_1} & D_{z_2} \\ \hline \end{array}$$

In obfuscation, S_{xi} contains useful information,
Assuming the rest are random (and small)

1	$a_{1,1} J B_{1,1} K_1$	$a_{2,1} K_1^{-1} B_{2,1} K_2$	$a_{3,1} K_2^{-1} B_{3,1} K_3$	$a_{4,1} K_3^{-1} B_{4,1} L$
0	$a_{1,0} J B_{1,0} K_1$	$a_{2,0} K_1^{-1} B_{2,0} K_2$	$a_{3,0} K_2^{-1} B_{3,0} K_3$	$a_{4,0} K_3^{-1} B_{4,0} L$
i	1	2	1	2

Key idea of the attack (from [Coron, Lee, Lepoint, Tibouchi 16])

By evaluating on the combinations of $x_1, x_2, x_3, z_1, z_2, \dots$, we got



Denote the whole matrix as $W = X * Z$.

Key idea of the attack (from [Coron, Lee, Lepoint, Tibouchi 16])

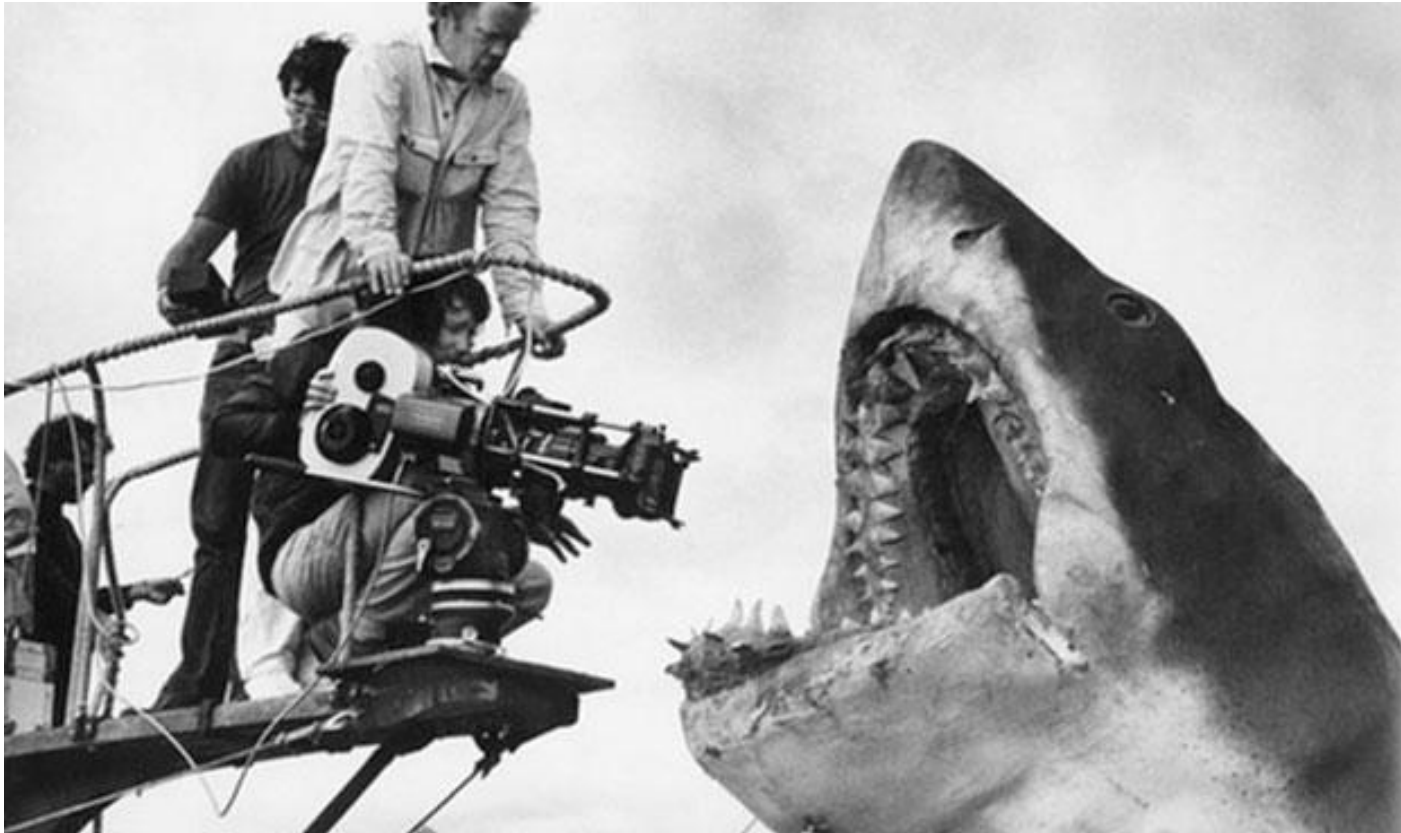
By evaluating on the combinations of $x_1, x_2, x_3, z_1, z_2, \dots$, we got

$$W = X * Z$$

Denote the whole matrix as $W = X * Z$.

In [[Chen, Gentry, Halevi 17](#)], collect W s.t. X is tall, Z is square, compute **the left kernel of W** , then extract **the bundling scalars**.

In [[Chen, Vaikuntanathan, Wee 18](#)], collect W s.t. X and Z are square, compute **the rank of W** , reveal information about **the matrix BP** directly.



More details about [CGH 17]

Target: Distinguish these two programs

Program 1

1	I	I	I	I	I	I	I	I	I
0	I	I	I	I	I	I	I	I	I
i	1	2	1	2	3	4	3	4	4

Goal of the attack:
extract the bundling
scalars in the X zone, run
the mixed-input attack.

Program 2

1	I	I	I	I	I	I	I	I	I
0	P	I	P ⁻¹	I	I	I	I	I	I
i	1	2	1	2	3	4	3	4	4

here $P \neq I$

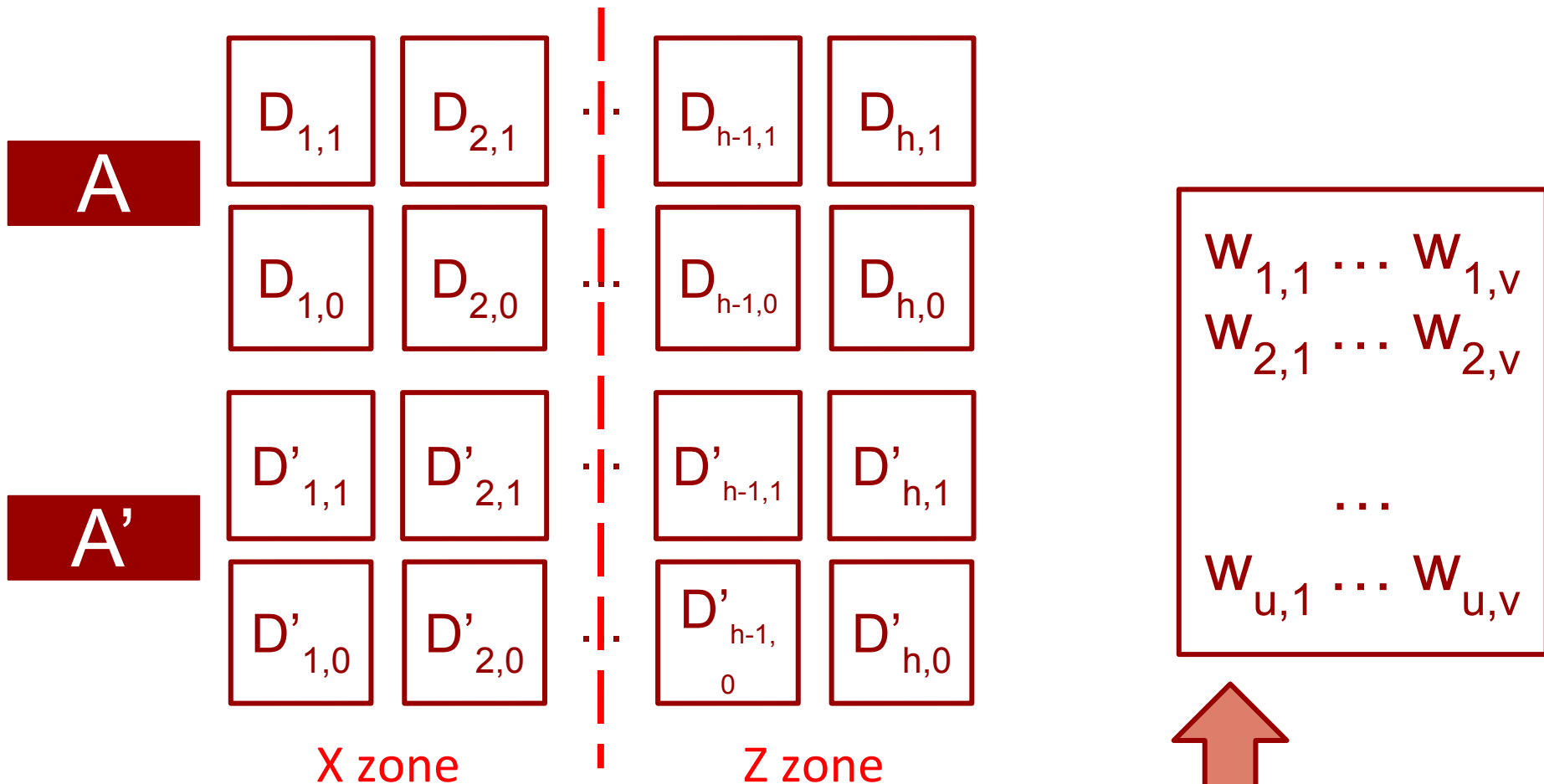
X zone

Z zone

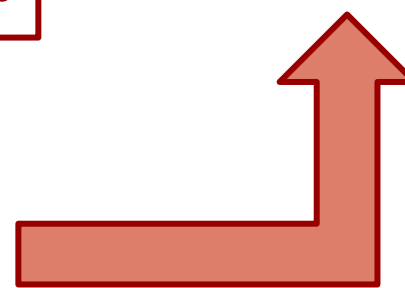
Attack GGHRSW13+GGH15

Step 1: Accumulate a matrix W via many evaluations that yield zero.

I.e., evaluate on $000|000, 001|000, 010|000, \dots, 000|001, 001|001, \dots$



$$w_{i,j} = A D_{x_i} D_{z_j} - A' D'_{x_i} D'_{z_j} \text{ mod } q$$

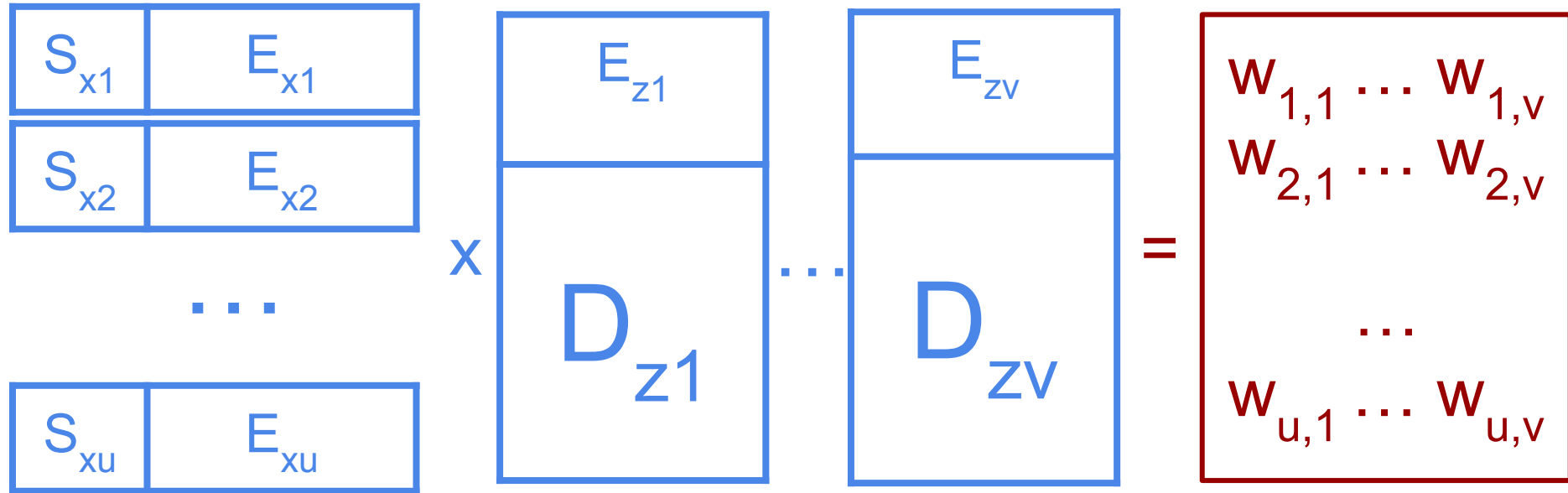


Attack GGHRSW13+GGH15

Step 1: Accumulate a matrix W via many evaluations that yield zero.

Step 2: Compute **the left-kernel of W**

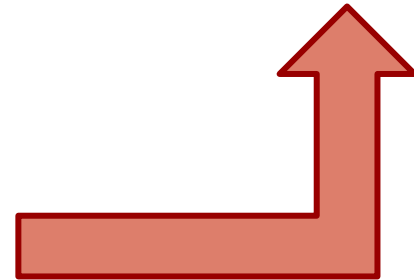
(In the rest of the analysis in this talk, I will ignore the dummy branch.)



X zone

Z zone

$$w_{i,j} = A D_{xi} D_{zj} - A' D'_{xi} D'_{zj} \text{ mod } q$$

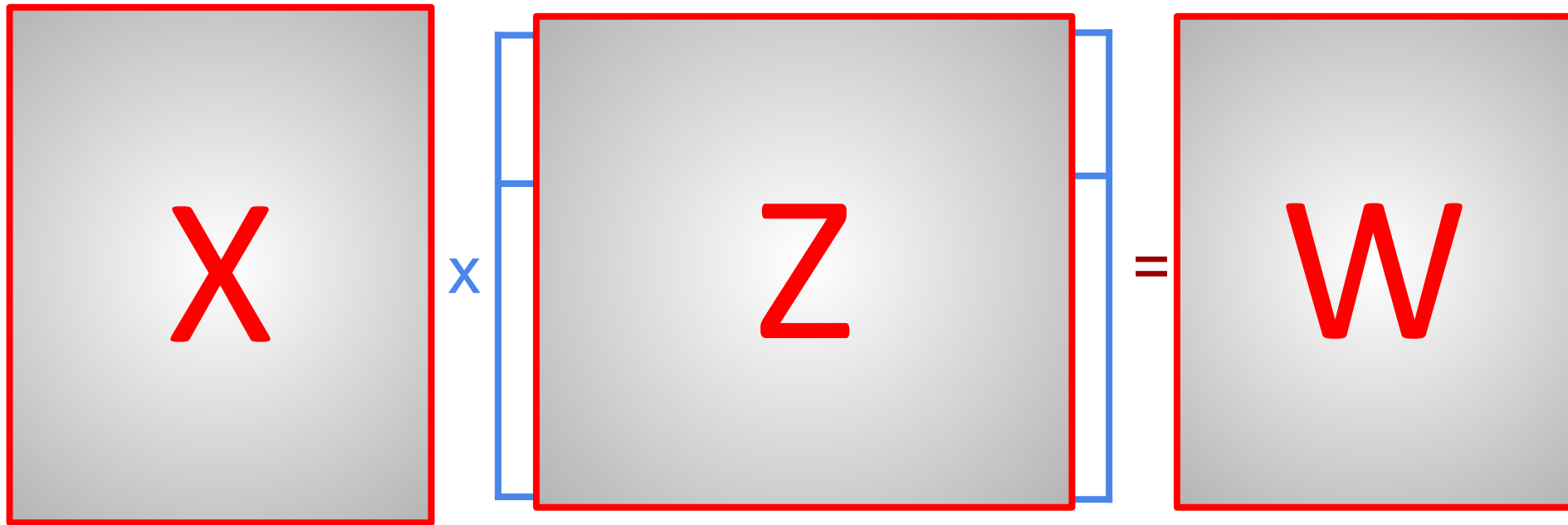


Attack GGHSW13+GGH15

Step 1: Accumulate a matrix W via many evaluations that yield zero.

Step 2: Compute **the left-kernel of W**

(In the rest of the analysis in this talk, I will ignore the dummy branch.)



$$F \cdot W = F \cdot X \cdot Z = 0 \Rightarrow F \cdot X = 0$$

(Z is square and full-rank whp)

Attack GGHRSW13+GGH15

Step 1: Accumulate a matrix W via many evaluations that yield zero.

Step 2: Compute the left-kernel of W : $FW = FXZ = 0 \Rightarrow FX=0$

Step 3: From F , learn something about **scalars**

$$\begin{array}{|c|} \hline f_{1,1}, f_{1,2}, \dots, f_{1,u} \\ \hline \dots \\ \hline f_{d,1}, f_{d,2}, \dots, f_{d,u} \\ \hline \end{array} \times \begin{array}{|c|c|} \hline S_{x1} & E_{x1} \\ \hline S_{x2} & E_{x2} \\ \hline \dots & \dots \\ \hline S_{xu} & E_{xu} \\ \hline \end{array} = 0$$

The useful equations:

$\forall k \text{ in } [1,d]$

$$\sum_{i=1}^u f_{k,i} \cdot a_{xi} = 0$$

\nearrow coefficients \nwarrow unknowns

Use the homogeneous feature, possible

to get $a_{1,1} a_{3,1} / a_{1,0} a_{3,0}, a_{1,1} a_{3,1} / a_{2,1} a_{4,1}$

\nearrow

$$S_{xi} = a_{xi} J [B_{xi}^*] K_x$$

where $a_{xi} = \prod_j a_{j,xij}$

Attack GGHRSW13+GGH15

Step 1: Accumulate a matrix W via many evaluations that yield zero.

Step 2: Compute the left-kernel of W : $FW = FXZ = 0 \Rightarrow FX=0$

Step 3: From F , learn something about scalars

What we can get:

$$a_{1,1} a_{3,1} / a_{1,0} a_{3,0}$$

$$a_{1,1} a_{3,1} / a_{2,1} a_{4,1} \text{ in } \mathbb{Q}[x]/\phi(x)$$

What we want:

The small multiples of each of them

$$a_{1,1}, a_{3,1}, a_{1,0}, a_{3,0}, \dots$$

1	I	I	I	I	I	I	I	I	I
0	I	I	I	I	I	I	I	I	I
i	1	2	1	2	3	4	3	4	
	1	2	3	4					
1	I	I	I	I	I	I	I	I	I
0	P	I	P ⁻¹	I	I	I	I	I	I
i	1	2	1	2	3	4	3	4	

Attack GGHRSW13+GGH15

Step 1: Accumulate a matrix W via many evaluations that yield zero.

Step 2: Compute the left-kernel of W : $FW = FXZ = 0 \Rightarrow FX=0$

Step 3: From F , learn something about scalars

What we can get:

$$a_{1,1}a_{3,1}/a_{1,0}a_{3,0}$$

$$a_{1,1}a_{3,1}/a_{2,1}a_{4,1} \text{ in } \mathbb{Q}[x]/\phi(x)$$

1	I	I	I	I	I	I	I	I	I
0	I	I	I	I	I	I	I	I	I
i	1	2	1	2	3	4	3	4	4

What we want:

The small multiples of each of them

$$a_{1,1}, a_{3,1}, a_{1,0}, a_{3,0}, \dots$$

1	2	3	4
---	---	---	---

For certain parameters, can use averaging attack.

But in general, need quantum poly or classical subexp!

1	I	I	I	I	I	I	I	I	I
0	P	I	P ⁻¹	I	I	I	I	I	I
i	1	2	1	2	3	4	3	4	4

NTRU problem:

Given $h = f/g$ in $Z_q[x]/\phi(x)$, where f, g in $Z[x]/\phi(x)$ with small coefficients, find small multiples of f and g .

Open problem 1:

“NTRU problem without mod q ”

Given $h = f/g$ in $Q[x]/\phi(x)$, where f, g in $Z[x]/\phi(x)$ with small coefficients, find small multiples of f and g in $Z[x]/\phi(x)$.

Open problem 1:

“NTRU problem without mod q ”

Given $h = f/g$ in $\mathbb{Q}[x]/\phi(x)$, where f, g in $\mathbb{Z}[x]/\phi(x)$ with small coefficients, find small multiples of f and g in $\mathbb{Z}[x]/\phi(x)$.

Example: (Sage)

```
n = 8
P.<x> = ZZ["x"]
phi = x^n + 1
f = P.random_element(x=-100, y=101, degree=n-1)
g = P.random_element(x=-100, y=101, degree=n-1)
h = (f * g.change_ring(QQ).inverse_mod(phi)) % phi
```

```
print(f, g, h)
```

```
-5*x^7 + 11*x^6 - 38*x^5 + 13*x^4 + 72*x^3 - 7*x^2 - 88*x + 23
-24*x^7 + x^6 - 4*x^5 - 40*x^4 + 99*x^3 + 30*x^2 - 33*x + 96
-5967273975693765/23276530089158977*x^7 + 21013131762418457/23276530089158977*x^6 +
16763779410495968/23276530089158977*x^5 + 4594423085422955/23276530089158977*x^4 -
19985854142124613/23276530089158977*x^3 - 19258587086998750/23276530089158977*x^2 +
5782611576011603/23276530089158977*x + 28754308083338628/23276530089158977
```

Open problem 1:

“NTRU problem without mod q ”

Given $h = f/g$ in $\mathbb{Q}[x]/\phi(x)$, where f, g in $\mathbb{Z}[x]/\phi(x)$ with small coefficients, find small multiples of f and g in $\mathbb{Z}[x]/\phi(x)$.

If given “many” $f_1/g, f_2/g, \dots, f_n/g$ with the same g , then the problem can be solved using the averaging attack.

Summary of the [CGH 17] attack on GGHRW13+GGH15

Step 1: Accumulate a matrix W via many evaluations that yield zero.

Step 2: Compute the left-kernel of W : $FW = FXZ = 0 \Rightarrow FX=0$

Step 3: From F , learn the ratios of the scalars.

Step 4: (Quantum poly. or classically sub.exp.) from the ratios, find small scalars.

Two limitations:

1. Need quantum poly or classical subexp in the last step.
2. Only applies to BPs where the indices can be partitioned in X and Z zones.

1	I	I	I	I	I	I	I	I	I
0	I	I	I	I	I	I	I	I	I
i	1	2	1	2	3	4	3	4	

1	I	I	I	I	I	I	I	I	I
0	P	I	P ⁻¹	I	I	I	I	I	I
i	1	2	1	2	3	4	3	4	

Summary of the [CGH 17] attack on GGHRW13+GGH15

Step 1: Accumulate a matrix W via many evaluations that yield zero.

Step 2: Compute the left-kernel of W : $FW = FXZ = 0 \Rightarrow FX=0$

Step 3: From F , learn the ratios of the scalars.

Step 4: (Quantum poly. or classically sub.exp.) from the ratios, find small scalars.

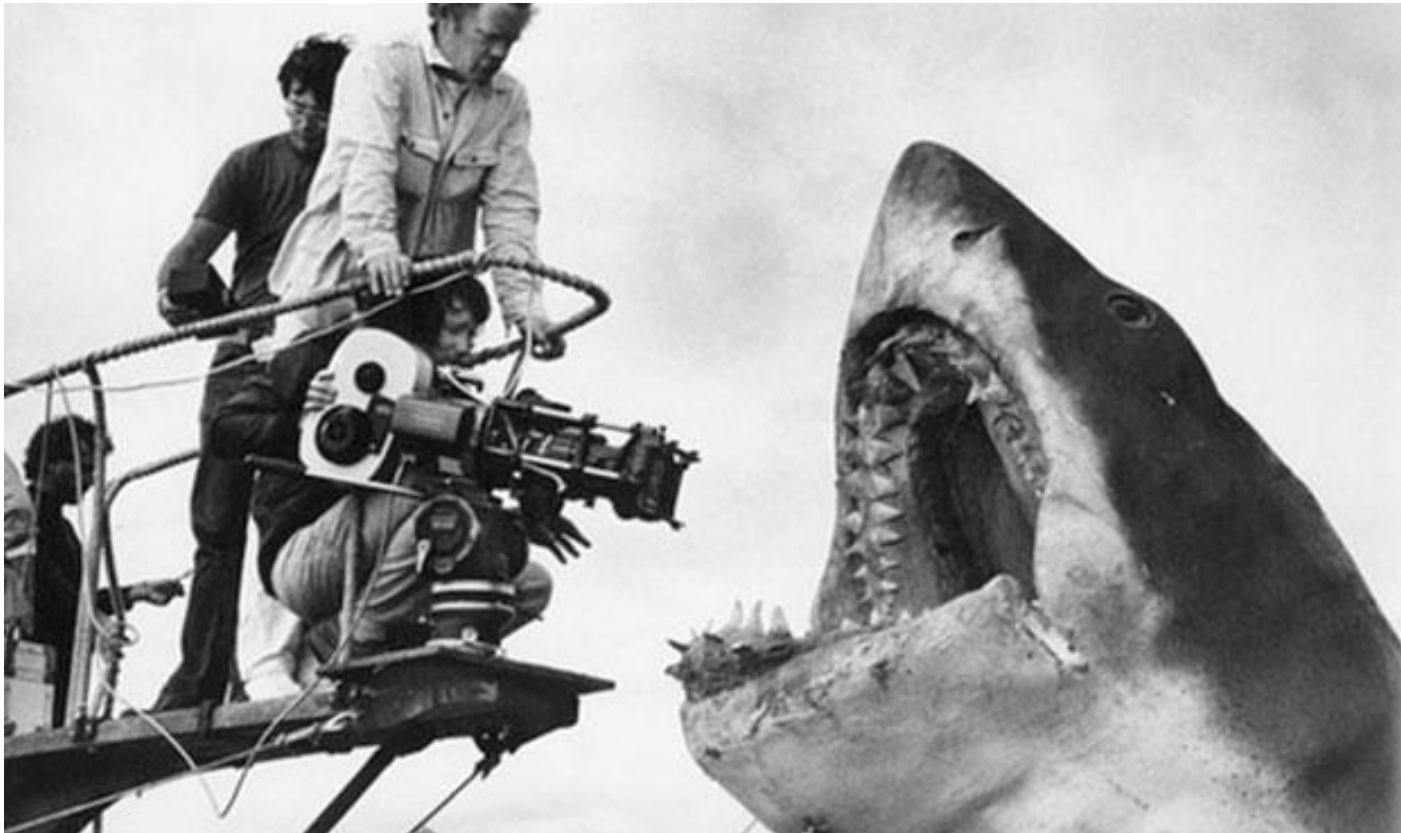
Two limitations:

1. Need quantum poly or classical subexp in the last step.
2. Only applies to BPs where the indices can be partitioned in X and Z zones.

1	I	I	I	I	I	I	I	I	I
0	I	I	I	I	I	I	I	I	I
i	1	2	1	2	3	4	3	4	

1	I	I	I	I	I	I	I	I	I
0	P	I	P ⁻¹	I	I	I	I	I	I
i	1	2	1	2	3	4	3	4	

Overcomed in [CVW 18]



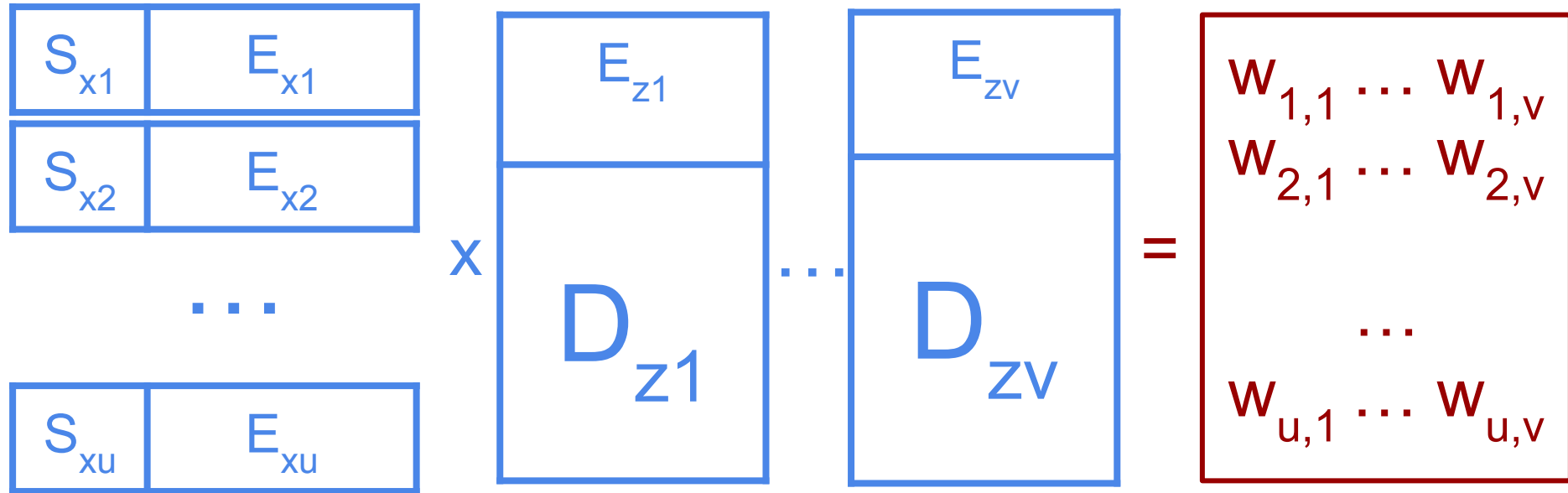
[CVW 18]: “rank attack”

Attack GGHRSW13+GGH15

Step 1: Accumulate a matrix W via many evaluations that yield zero.

~~Step 2: Compute the left kernel of W [CGH 17]~~

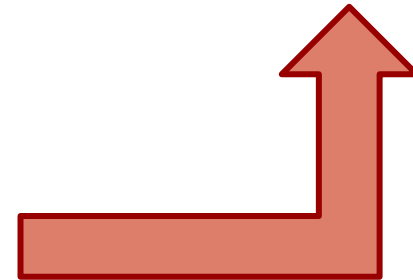
Step 2: Compute the rank of W :) [CVW 18]



X zone

Z zone

$$w_{i,j} = A D_{xi} D_{zj} - A' D'_{xi} D'_{zj} \text{ mod } q$$



Attack GGHRSW13+GGH15

Step 1: Accumulate a matrix W via many evaluations that yield zero.

~~Step 2: Compute the left kernel of W [CGH17]~~

Step 2: Compute the rank of W :) The rank of W reveals the rank of "S".

$$S_{x_i} = a_{x_i} J [\begin{matrix} * \\ B_{x_i} \end{matrix}] K_x \quad S'_{x_i} = a'_{x_i} J' [\begin{matrix} * \\ l_{x_i} \end{matrix}] K'_x$$



$$w_{i,j} = A D_{x_i} D_{z_j} - A' D'_{x_i} D'_{z_j} \text{ mod } q$$

How to find two programs where this attack applies?

(Caveat: these two programs are not functionally equivalent.)

Program 1

1	I	I	I	I	I	I	I	I
0	I	I	I	I	I	I	I	I
i	1	2	3	4	5	6	7	8

Program 2

1	I	I	I	I	I	I	I	I
0	P	I	I	I	I	I	I	I
i	1	2	3	4	5	6	7	8

here $P \neq I$

X zone

Z zone

How to find two programs where this attack applies?

(Caveat: these two programs are not functionally equivalent.)

In the analysis, you will see an intermediate step

where for program 1, rank(W) depends on

I	I
I	I

whereas for program 2, rank(W) depends on

I	I
P	I

Program 1

1	I	I	I	I	I	I	I	I
0	I	I	I	I	I	I	I	I
i	1	2	3	4	5	6	7	8

Program 2

1	I	I	I	I	I	I	I	I
0	P	I	I	I	I	I	I	I
i	1	2	3	4	5	6	7	8

here $P \neq I$

X zone

Z zone

How to find two programs where inputs are not partitioned?

Program 1

1^*	I	I	I	I
0^*	I	I	I	I
i	X1	Z1	X2	Z2

Program 2

1^*	I	I	I	I
0^*	P	I	P^{-1}	I
i	X1	Z1	X2	Z2

here $P \neq I$

How to find two programs where inputs are not partitioned?



The attack algorithm is the same, except that $\dim(W)$ is bigger, and the analysis is harder.

Program 1

1^*	I	I	I	I
0^*	I	I	I	I
i	X1	Z1	X2	Z2

Program 2

1^*	I	I	I	I
0^*	P	I	P^{-1}	I
i	X1	Z1	X2	Z2

here $P \neq I$

How to find two programs where inputs are not partitioned?

The analysis uses a matrix-product switching trick used in [Apon, Döttling, Garg, Mukherjee 17] & [Coron, Lee, Lepoint, Tibouchi 17]

Given four matrices A, B, C, D where B, C are square and of dimension n, A, D are vectors of dimension n.

Then

$$w = A * B * C * D = u(A, C) * v(B, D),$$

where u and v are vectors of dimension n^3 .

I	I
I	I

vs

I	I
M(P)	I

Program 1

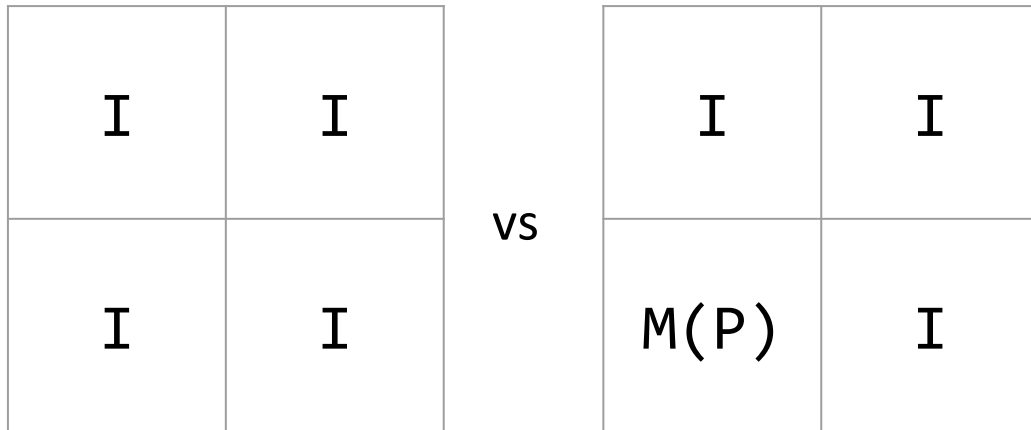
1*	I	I	I	I
0*	I	I	I	I
i	X1	Z1	X2	Z2

Program 2

1*	I	I	I	I
0*	P	I	P ⁻¹	I
i	X1	Z1	X2	Z2

here $P \neq I$

How to find two programs where inputs are not partitioned?



Program 1

1*	I	I	I	I
0*	I	I	I	I
i	X1	Z1	X2	Z2

Program 2

1*	I	I	I	I
0*	P	I	P ⁻¹	I
i	X1	Z1	X2	Z2

Implication: can switch the index from 1212 to 1122 by blowing up the dimension.

In general, if 123123...123 repeat c times, the dimension of the matrix W is 2^{2c-1} .

here $P \neq I$

*Beyond iO: Rank attack can be used to break “PRFs” from matrix products.

1	$B_{1,1}$	$B_{2,1}$	$B_{3,1}$	$B_{4,1}$	$B_{5,1}$	$B_{6,1}$
0	$B_{1,0}$	$B_{2,0}$	$B_{3,0}$	$B_{4,0}$	$B_{5,0}$	$B_{6,0}$
i	1	2	3	1	2	3

Here is a candidate PRF: F

Key: sample $2*c*L$ matrices $\{ B_{i,b} \}$ of dimension w . (In the example, $c = 2, L=3$)

On an L -bit input x , repeat x for c times, then do the subset-product evaluation.

*Beyond iO: Rank attack can be used to break “PRFs” from matrix products.

1	$B_{1,1}$	$B_{2,1}$	$B_{3,1}$	$B_{4,1}$	$B_{5,1}$	$B_{6,1}$
0	$B_{1,0}$	$B_{2,0}$	$B_{3,0}$	$B_{4,0}$	$B_{5,0}$	$B_{6,0}$
i	1	2	3	1	2	3

Here is a candidate PRF: F

Key: sample $2 \cdot c \cdot L$ matrices $\{ B_{i,b} \}$ of dimension w . (In the example, $c = 2$, $L=3$)

On an L -bit input x , repeat x for c times, then do the subset-product evaluation.

[[Chen](#), Hhan, Vaikuntanathan, Wee 19]:

The rank attack runs in time $w^{O(c)}$. So when c is a constant, F is not a PRF.

The attack collects a matrix of dimension w^{2c} , then computes its rank. If the rank is lower than some threshold, then it is not a random function.

We also simplify the proof of the matrix switching lemma.

Open problem 2:

Better rank attack for matrix PRFs or iO using quantum?

Or: Quantum advantage in the rank problem => breaking LWE?

Background:

> Quantum algorithms for deciding whether the rank is super high, or super low:
[Harrow, Hassidim, Lloyd 09], [Ambainis 12], [Belovs 11]
The precise runtime depends on the singular values of the matrix and others.

> Solving the rank problem of an exponential dimensional matrix
=> ? => break all the PRFs computable in NC1?

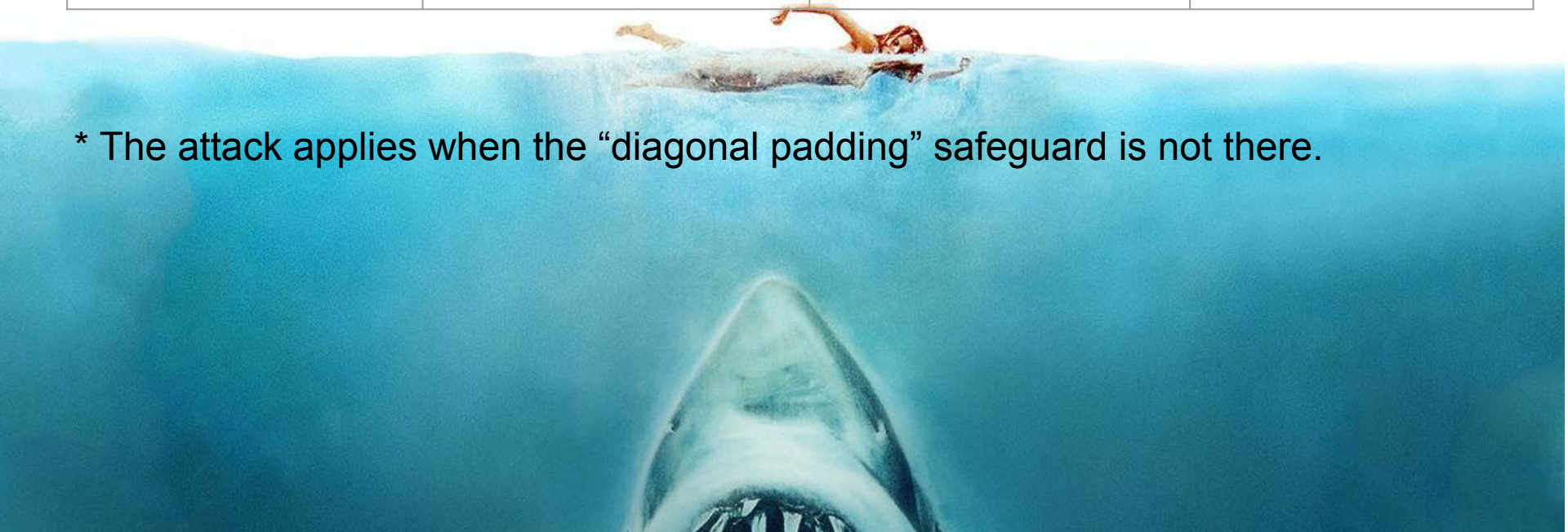
This is based on two steps. First, use Barrington theorem to convert any NC1 circuit into a matrix branching program (of polynomially many input repetitions). Second, use the algorithm in Section 3 of [Chen, Hhan, Vaikuntanathan, Wee 19] to break all the PRFs based on matrix branching program in $2^{\{ O(r) \}}$ times, where r is the time that the input repeats.

> [Zhandry 13] shows that the classical PRF computable in NC1 from [Banerjee, Peikert, Rosen 12] is a PRF in the quantum-query model under LWE assumption.

Status of iOs under the framework of [GGHRSW 13]

Type of program (branching programs)	Simple (read-once BP)	Complex (read-const-many BP)	Very Complex (read-n or dual-input BP)
GGH13	[MSZ 16]* [<u>CGH</u> 17]	[ADGM 17]* [<u>CVW</u> 18]	Standing
CLT13	[CHLRS 15] [Coron et al. 15]	[CLLT 17]	Standing
GGH15	[<u>CGH</u> 17]	[<u>CVW</u> 18]	Standing

* The attack applies when the “diagonal padding” safeguard is not there.



Candidate iO from [Garg, Gentry, Halevi, Raykova, Sahai, Waters 13]

(0) Representation of plaintext program: Oblivious branching program

(1) Safeguard 1: Kilian randomization

(2) Safeguard 2: Bundling scalars

(3) Safeguard 3: random diagonal entries and bookends

(4) Wrap (0-3) by [GGH15](#)

In [[Chen](#), Vaikuntanathan, Wee 18], we propose a variant of GGHRSW13.

The main differences are

(0) The oblivious branching program has to **repeat the indices for n times**.

(2) The “bundling factors” are **non-commutative**.

Attacked by [Cheon, Cho, Hhan, Kim, Lee 19] using statistical method,
fixed in [[Chen](#), Hhan, Vaikuntanathan, Wee 19].

[Bartusek, Guan, Ma, Zhandry 18] provides a different variant that is also standing right now.

2020

Open problem 1:

“NTRU problem without mod q ”

Given $h = f/g$ in $\mathbb{Q}[x]/\phi(x)$, where f, g in $\mathbb{Z}[x]/\phi(x)$ with small coefficients, find small multiples of f and g in $\mathbb{Z}[x]/\phi(x)$.

Open problem 2:

Better rank attack for matrix PRFs or iO using **quantum**?

Or: **Quantum** advantage in the rank problem \Rightarrow breaking **LWE**?

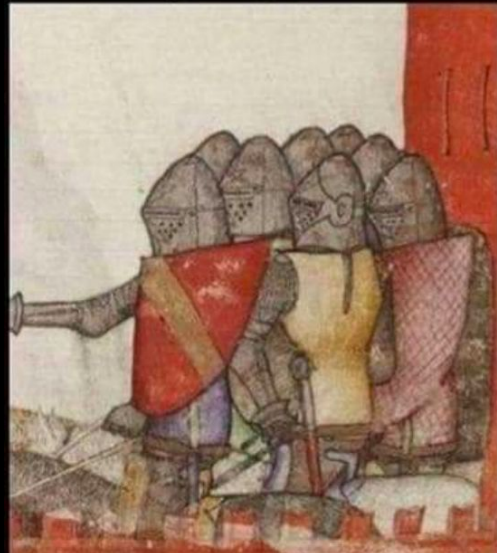
Keep your distance



Wash your hands



Cover your face



Avoid mass transport



Thanks for
your time!

Yilei Chen
[VISA Reseach]

2020 Simons Lattices