Hiding secrets in public random functions

Yilei Chen

Dissertation committee
Adam Smith, Ran Canetti, Leonid Reyzin, Vinod Vaikuntanathan, Mayank Varia

> July 16, 2018, Boston, heavy snow.

> July 16, 2018, Boston, heavy snow. Alice finds a classical polynomial time algorithm for factoring.

> July 16, 2018, Boston, heavy snow. Alice finds a classical polynomial time algorithm for factoring.

> Instead of putting it in her thesis, she thinks it's cool to write a program and post it on Github.

> Wait, let’s have some fun.

> Wait, let’s have some fun.

> obfuscate factoring.hs >> idontknowwhatimdoing.hs

XOpenDisplay(0); z=RootWindow(e,0);
for (XSetForeground(e,k=XCreateGC (e,z,0,0),BlackPixel(e,0));
scanf("%lf%lf%lf",y +n,w+y, y+s)+1; y ++);
XSelectInput(e,z= XCreateSimpleWindow(e,z,0,0,400,400, 0,0,WhitePixel(e,0)),KeyPressMask); **for**(XMapWindow(e,z); ;
T=sin(O)){ **struct** timeval ......

> Maybe more?

> Maybe more?

> watermark idontknowwhatimdoing.hs

XOpenDisplay(0); z=RootWindow(e,0);
for (XSetForeground(e,k=XCreateGC (e,z,0,0),BlackPixel(e,0));
scanf("%lf%lf%lf",y +n,w+y, y+s)+1; y ++);
XSelectInput(e,z= XCreateSimpleWindow(e,z,0,0,400,400, 0,0,WhitePixel(e,0)),KeyPressMask); **for**(XMapWindow(e,z);
; T=sin(O)){ **struct** timeval ......

Alice

… was a nice dream

> Alices~: factoring.hs 21

> Alices~: obfuscate factoring.hs

> Alices~: watermark idontknowwhatimdoing.hs

> Alices~: factoring.hs 21

21 = 1 x 21

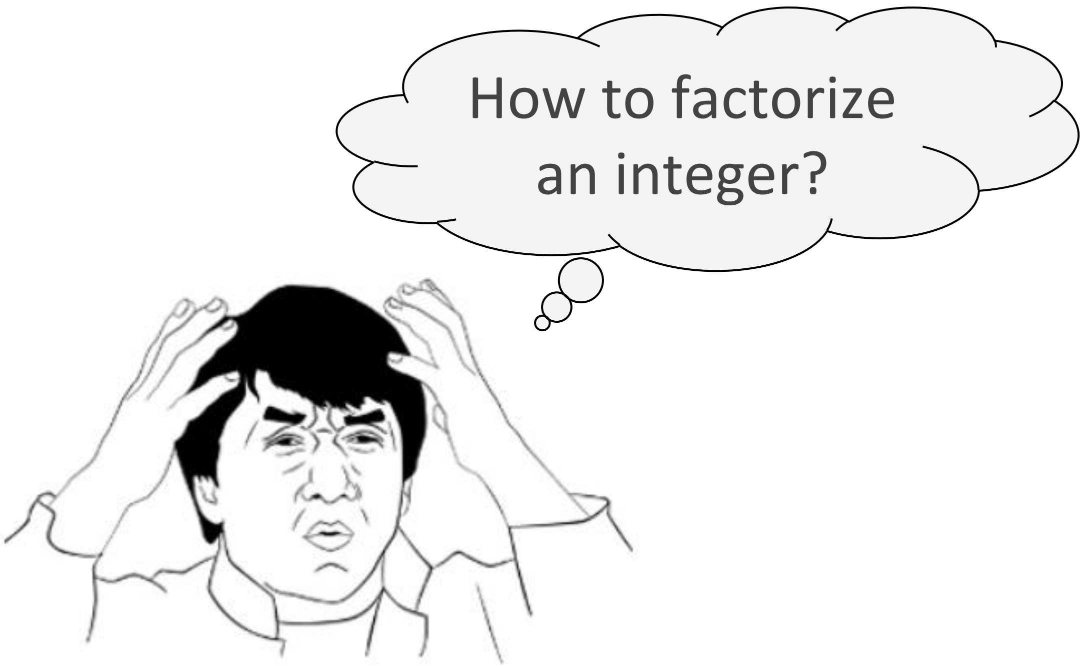
> Alices~: obfuscate factoring.hs

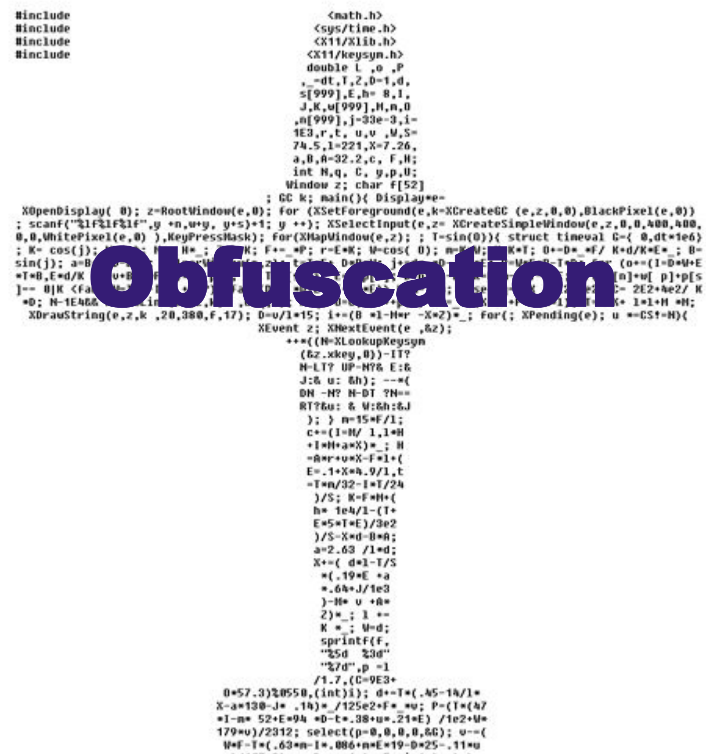
How?

> Alices~: watermark idontknowwhatimdoing.hs

What?

My research:
How to achieve these advanced cryptographic capabilities based on hard mathematical problems (or break them)?





Watermarking

The plan for the talk:

> Overview of our research

> Two specific works related to *hiding secrets in public random functions*

Lattice

Multilinear maps

Obfuscation

buzzwords in cryptography

Lattice

A discrete subgroup of R^n.

Lattice

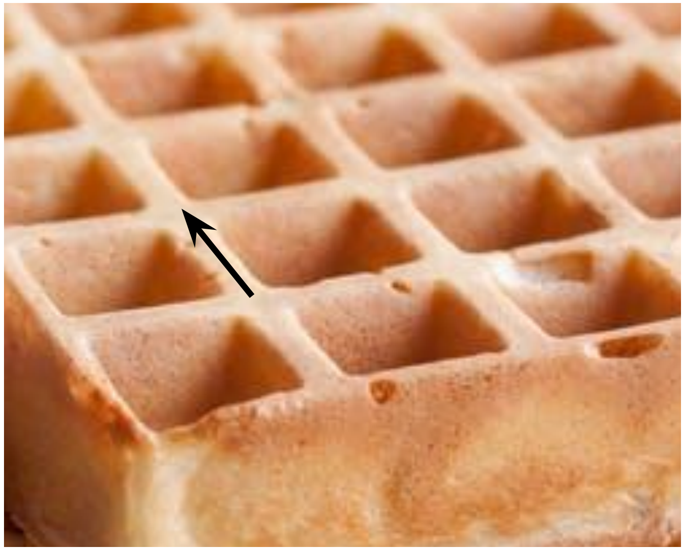
A discrete subgroup of R^n.

Find the shortest non-zero vector in a lattice?

Lattice

A discrete subgroup of R^n.

Find the shortest non-zero vector in a lattice:
Easy for 2 dimensional lattices



Lattice

A discrete subgroup of R^n.

Find the shortest non-zero vector in a lattice:
Easy for 2 dimensional lattices

Find short vectors in n-dimensional lattices

Lattice

A discrete subgroup of R^n.

Find the shortest non-zero vector in a lattice:
Easy for 2 dimensional lattices

Find short vectors in n-dimensional lattices
= > Exponential approximation [Lenstra, Lenstra, Lovasz 82]
…

Lattice

A discrete subgroup of R^n.

Find the shortest non-zero vector in a lattice:

Easy for 2 dimensional lattices

Find short vectors in n-dimensional lattices

= > Exponential approximation [Lenstra, Lenstra, Lovasz 82]

…

= > One-way function [Ajtai 96]

= > Public-key encryption [Ajtai, Dwork 97, Regev 05]

= > Homomorphic enc [Gentry 09, Brakerski, Vaikuntanathan 11]

…

Crypto applications

Lattice

A discrete subgroup of R^n.

Find the shortest non-zero vector in a lattice:

Easy for 2 dimensional lattices

Find short vectors in n-dimensional lattices

= > Exponential approximation [Lenstra, Lenstra, Lovasz 82]

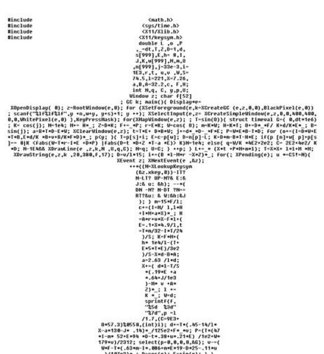
…

= > One-way function [Ajtai 96]

= > Public-key encryption [Ajtai, Dwork 97, Regev 05]

= > Homomorphic enc [Gentry 09, Brakerski, Vaikuntanathan 11]

…

= >  ???

Multilinear maps

A new beast

Multilinear maps

A new beast

Encodings: [a], [b], [c], … so that we can
publicly Add ([a], [b]) -> [a + b]
publicly Mult ([a], [b]) -> [ab]
publicly Test ([a]) -> a = 0 or not

Ideal security: [a], [b], [c], … hide the plaintexts a, b, c, …

Multilinear maps

A new beast

Encodings: [a], [b], [c], … so that we can
publicly Add ([a], [b]) -> [a + b]
publicly Mult ([a], [b]) -> [ab]
publicly Test ([a]) -> a = 0 or not

Ideal security: [a], [b], [c], … hide the plaintexts a, b, c, …

What we know: Bilinear maps from elliptic curves [Miller 1986]
Motivation of n-linear maps [Boneh, Silverberg 2003]

Multilinear maps

A new beast

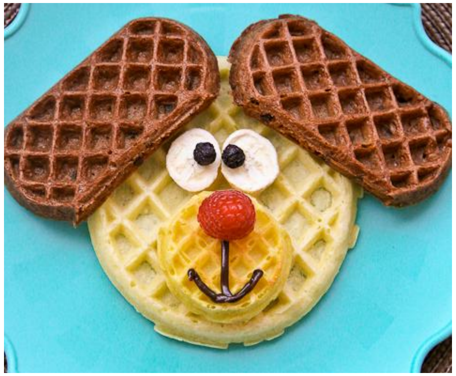
Encodings: [a], [b], [c], … so that we can
publicly Add ([a], [b]) -> [a + b]
publicly Mult ([a], [b]) -> [ab]
publicly Test ([a]) -> a = 0 or not

Ideal security: [a], [b], [c], … hide the plaintexts a, b, c, …

What we know: Bilinear maps from elliptic curves [Miller 1986]
Motivation of n-linear maps [Boneh, Silverberg 2003]

Candidates: Garg, Gentry, Halevi 2013 [GGH 13]
Coron, Lepoint, Tibouchi 2013 [CLT 13]
Gentry, Gorbunov, Halevi 2015 [GGH 15]

All based on nonstandard use of lattices



Obfuscation

Obfuscation:

- A compiler P -> P*
- P* = P in functionality
- P* is “unintelligible”, “hides information” in P

P =

```
factorize()
```

P* =

```
XOpenDisplay( 0); z=RootWindow(e,0); for (XSetForeground(e,k=XCreateGC
(e,z,0,0),BlackPixel(e,0));
scanf("%lf%lf%lf",y +n,w+y, y+s)+1; y ++);
XSelectInput(e,z= XCreateSimpleWindow(e,z,0,0,400,400, 0,0,WhitePixel(e,0)
),KeyPressMask); for(XMapWindow(e,z); ; T=sin(O)){ struct timeval
```

Obfuscation

Obfuscation:

- A compiler P -> P*
- P* = P in functionality
- P* is “unintelligible”, “hides information” in P

Indistinguishability Obfuscation (iO):

- Defined by [Barak et al. 2001]
- Known to be the best-possible [Goldwasser, Rothblum 2007]
- First candidate (for general purpose obfuscation) based on multilinear maps [Garg et al. 2013]

Obfuscation

Obfuscation:

- A compiler P -> P*
- P* = P in functionality
- P* is “unintelligible”, “hides information” in P

Indistinguishability Obfuscation (iO):

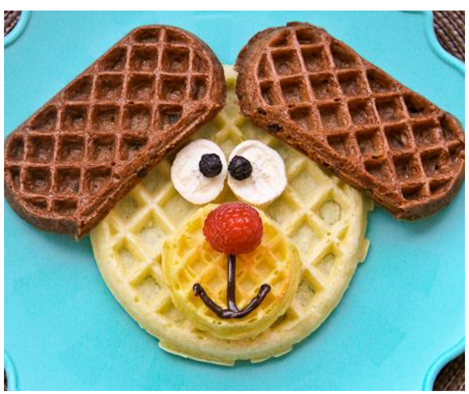
- Defined by [Barak et al. 2001]
- Known to be the best-possible [Goldwasser, Rothblum 2007]
- First candidate (for general purpose obfuscation) based on multilinear maps [Garg et al. 2013]

Oversimplified idea:

- Decompose P into a, b, c, ... (e.g. P(x) = (ax + b)c)
- Use Multilinear maps to encode [a], [b], [c].
- Plus other mechanisms to prevent illegal evaluations.

Obfuscation

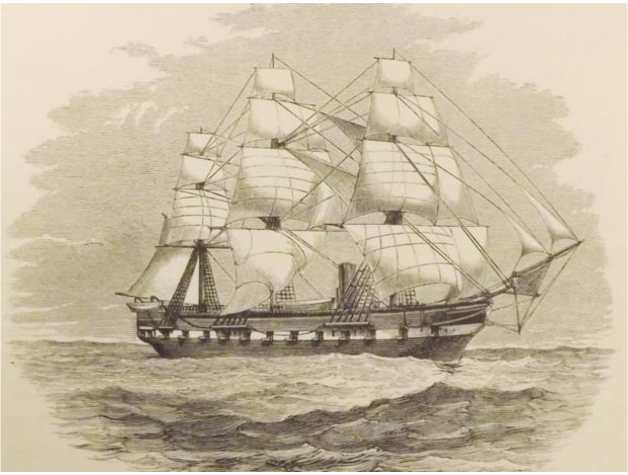
Multilinear maps



From nonstandard use of lattices

2013 - 2016

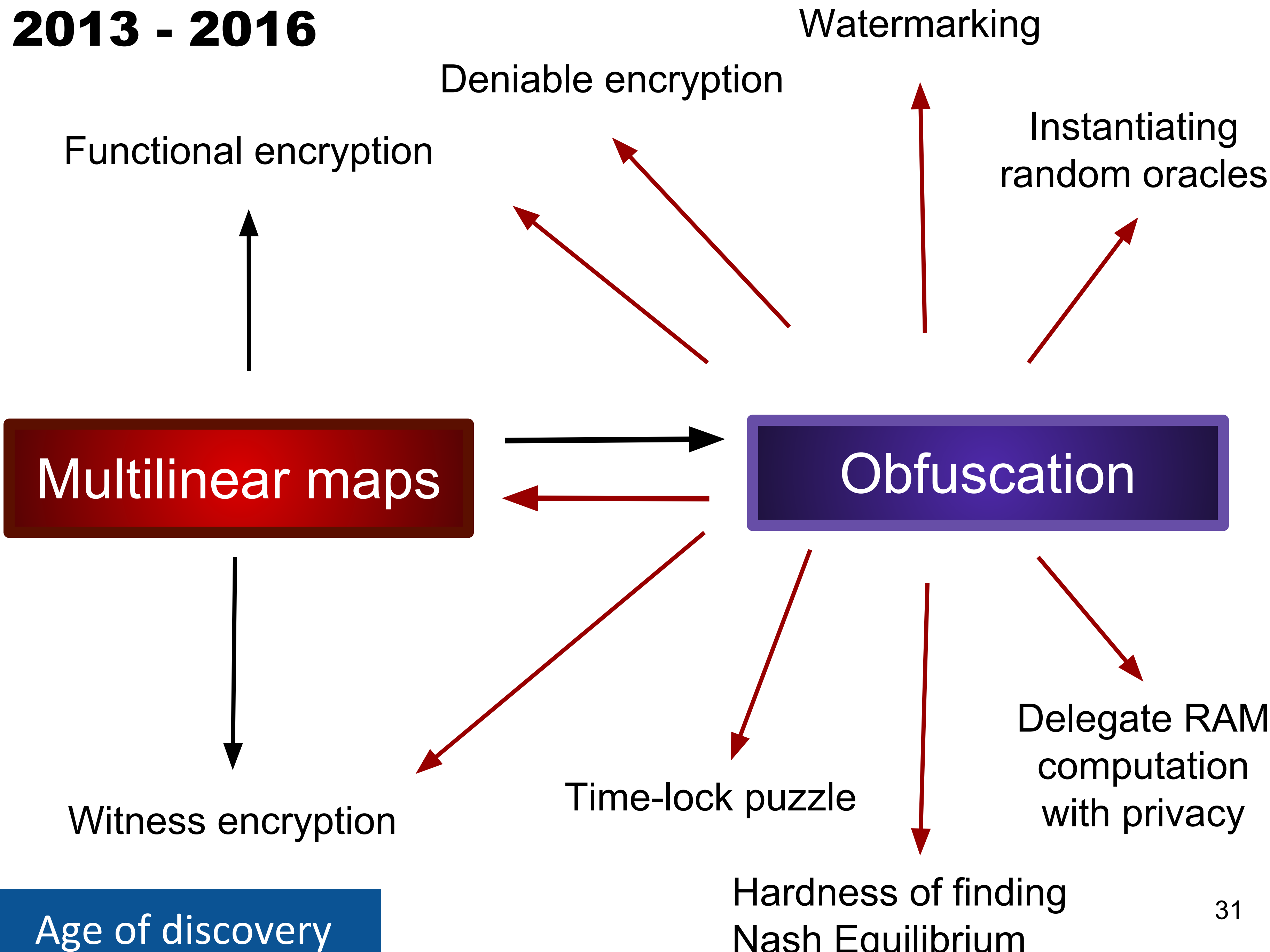
Start the age of discovery in Cryptoland



Multilinear maps

Obfuscation

2013 - 2016



Age of discovery

On the correlation intractability of obfuscated pseudorandom functions
Ran Canetti, Yilei Chen, Leonid Reyzin
Theory of Cryptography Conference 2016-A

Instantiating random oracles

Obfuscation

Delegate RAM computation with privacy

Adaptive succinct garbled RAM, or How to delegate your database
Ran Canetti, Yilei Chen, Justin Holmgren, Mariana Raykova
Theory of Cryptography Conference 2016-B

On the correlation intractability of obfuscated pseudorandom functions
Ran Canetti, Yilei Chen, Leonid Reyzin
Theory of Cryptography Conference 2016-A

Instantiating random oracles

[Full story of Alice]

Obfuscation

Adaptive succinct garbled RAM, or How to delegate your database
Ran Canetti, Yilei Chen, Justin Holmgren, Mariana Raykova
Theory of Cryptography Conference 2016-B

Delegate RAM computation with privacy

Age of discovery

Multilinear maps

Obfuscation

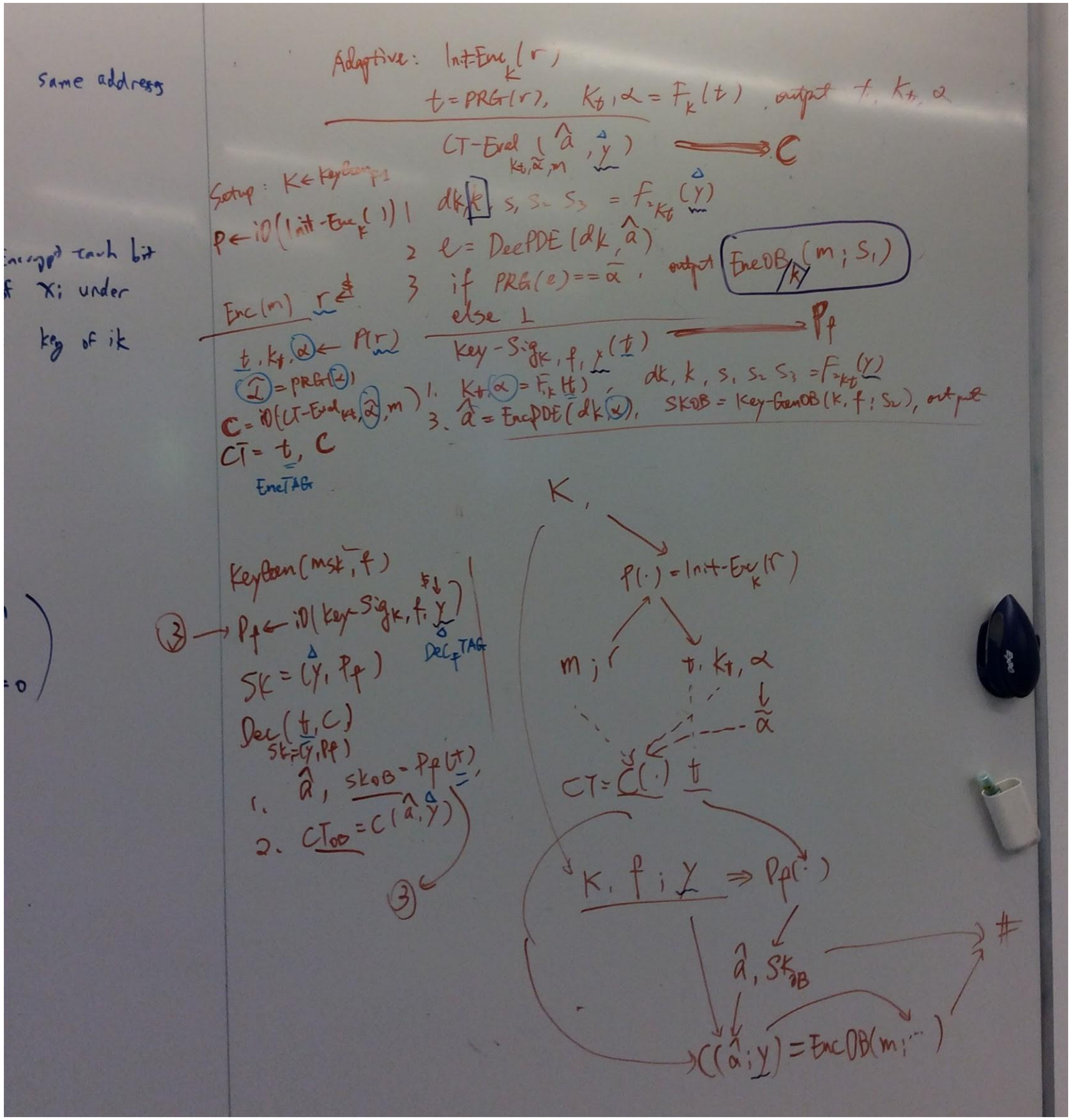
Witness encryption

Time-lock puzzle

Hardness of finding Nash Equilibrium

Delegate RAM computation with privacy

Age of discovery



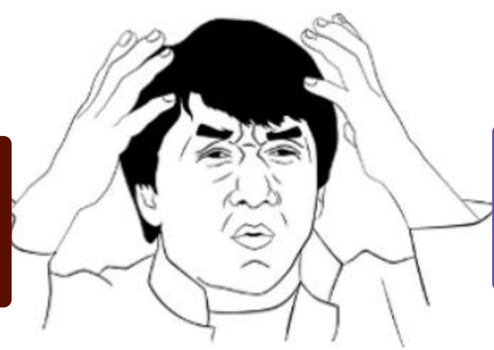
Functional encryption
[Waters 14]

Obfuscation

Memory from the glorious old days

How about the security of mmaps / iO candidates themselves?

Multilinear maps



Obfuscation

Candidate program obfuscators:

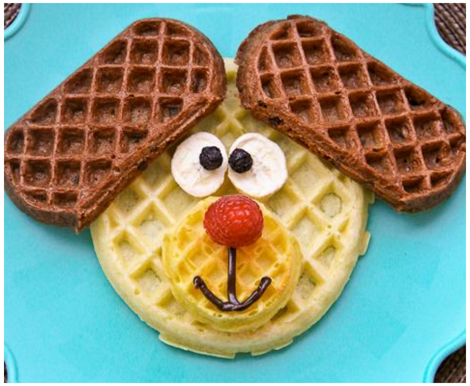
Since [Garg, Gentry, Halevi, Raykova, Sahai, Waters '13], around 20 variants
[Barak, Garg, Kalai, Paneth, Sahai '14], [Brakerski, Rothblum '14], [Pass, Seth, Telang '14], [Zimmerman '15], [Applebaum, Brakerski '15], [Ananth, Jain '15], [Bitansky, Vaikuntanathan '15], [Gentry, Gorbunov, Halevi '15], [Lin '16], [Lin, Vaikuntanathan '16], [Garg, Miles, Mukherjee, Sahai, Srinivasan, Zhandry '16] ...

So far, all based on n-linear maps (n>2).

Candidate multilinear maps for n>2:

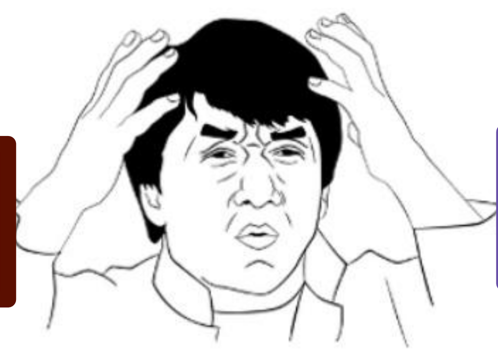
GGH13, CLT13, GGH15

All make non-standard uses of lattices.



You never know ...

Multilinear maps



Obfuscation

GGH13, CLT13, GGH15

[Garg, Gentry, Halevi, Raykova, Sahai, Waters '13], [Barak, Garg, Kalai, Paneth, Sahai '14], [Brakerski, Rothblum '14], [Pass, Seth, Telang '14], [Zimmerman '15], [Applebaum, Brakerski '15], [Ananth, Jain '15], [Bitansky, Vaikuntanathan '15], [Gentry, Gorbunov, Halevi '15], [Lin '16], [Lin, Vaikuntanathan '16], [Garg, Miles, Mukherjee, Sahai, Srinivasan, Zhandry '16] ...

2016

Status of indistinguishability obfuscators

under the framework of [Garg et al. 2013]

Type of program (branching programs)	Simple (read-once BP)	Complex (read-many BP)	Very Complex (dual-input BP)
GGH13	Standing	Standing	Standing
CLT13	[Cheon et al. 15] [Coron et al. 15]	Standing	Standing
GGH15	Standing	Standing	Standing

2017

Status of indistinguishability obfuscators

under the framework of [Garg et al. 2013]

Type of program (branching programs)	Simple (read-once BP)	Complex (read-many BP)	Very Complex (dual-input BP)
GGH13	[CGH 17]	Standing	Standing
CLT13	[Cheon et al. 15] [Coron et al. 15]	[Coron et al. 17]	Standing
GGH15	[CGH 17]*	Standing	Standing

Cryptanalysis of candidate branching program obfuscators
Yilei Chen, Craig Gentry, Shai Halevi
Eurocrypt 2017

2018

Status of indistinguishability obfuscators

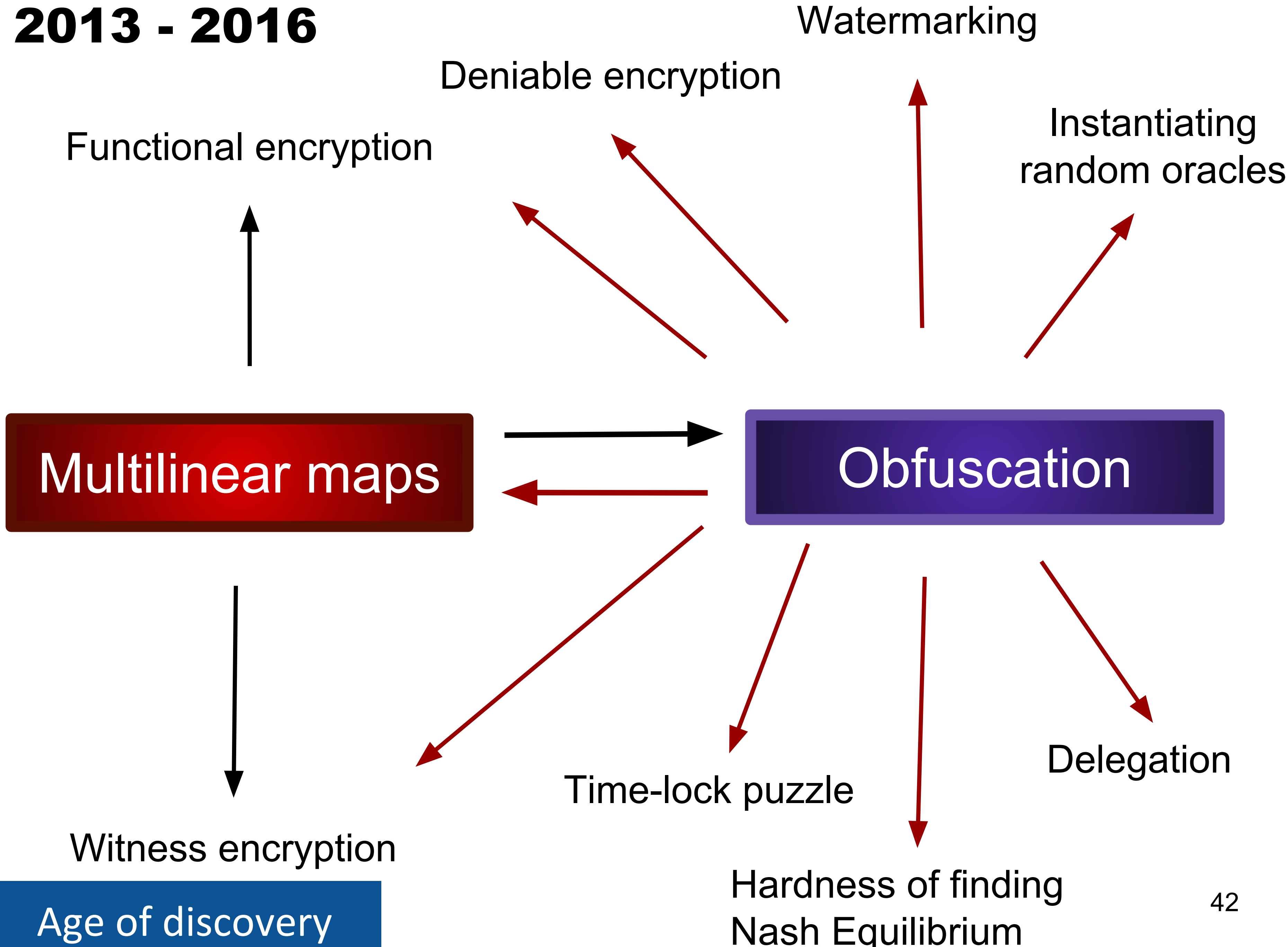
under the framework of [Garg et al. 2013]

Type of program (branching programs)	Simple (read-once BP)	Complex (read-many BP)	Very Complex (dual-input BP)
GGH13	[CGH 17]	[CVW 18]	Standing
CLT13	[Cheon et al. 15] [Coron et al. 15]	[Coron et al. 17]	Standing
GGH15	[CGH 17]* [CVW 18]	[CVW 18]	Standing

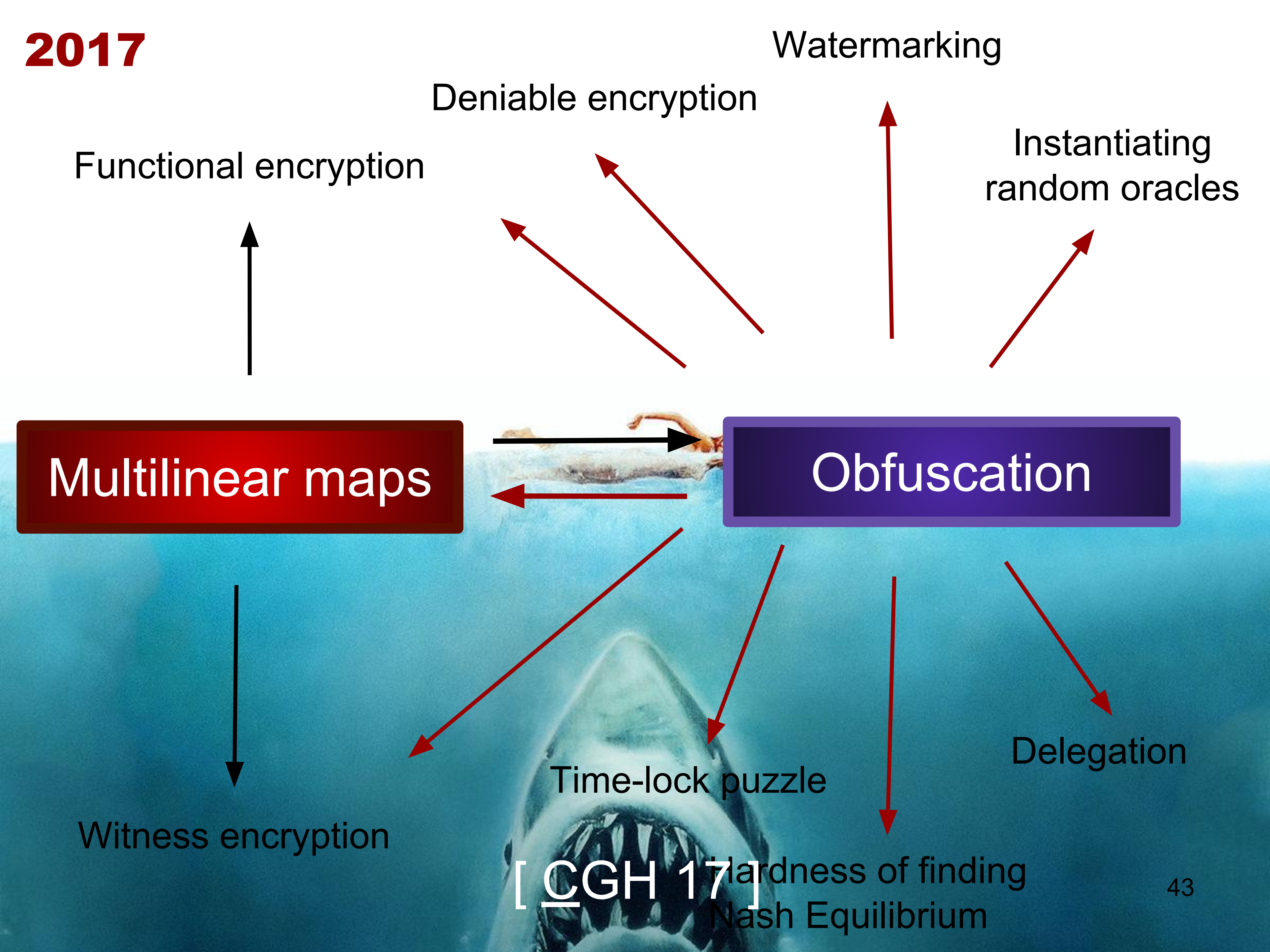
Cryptanalysis of candidate branching program obfuscators
Yilei Chen, Craig Gentry, Shai Halevi
Eurocrypt 2017

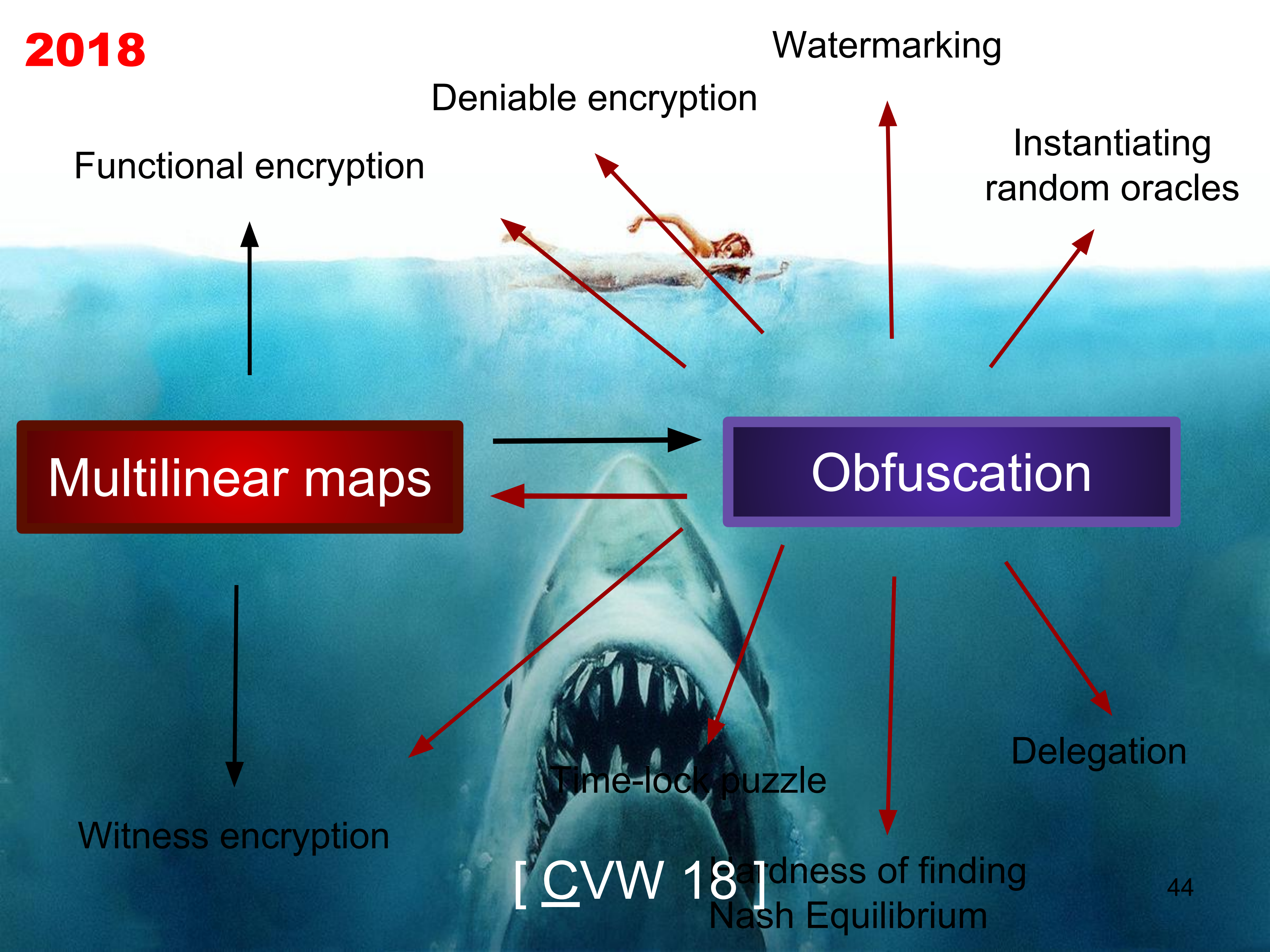
GGH15 beyond permutation branching programs
Yilei Chen, Vinod Vaikuntanathan, Hoeteck Wee
In submission 2018

2013 - 2016



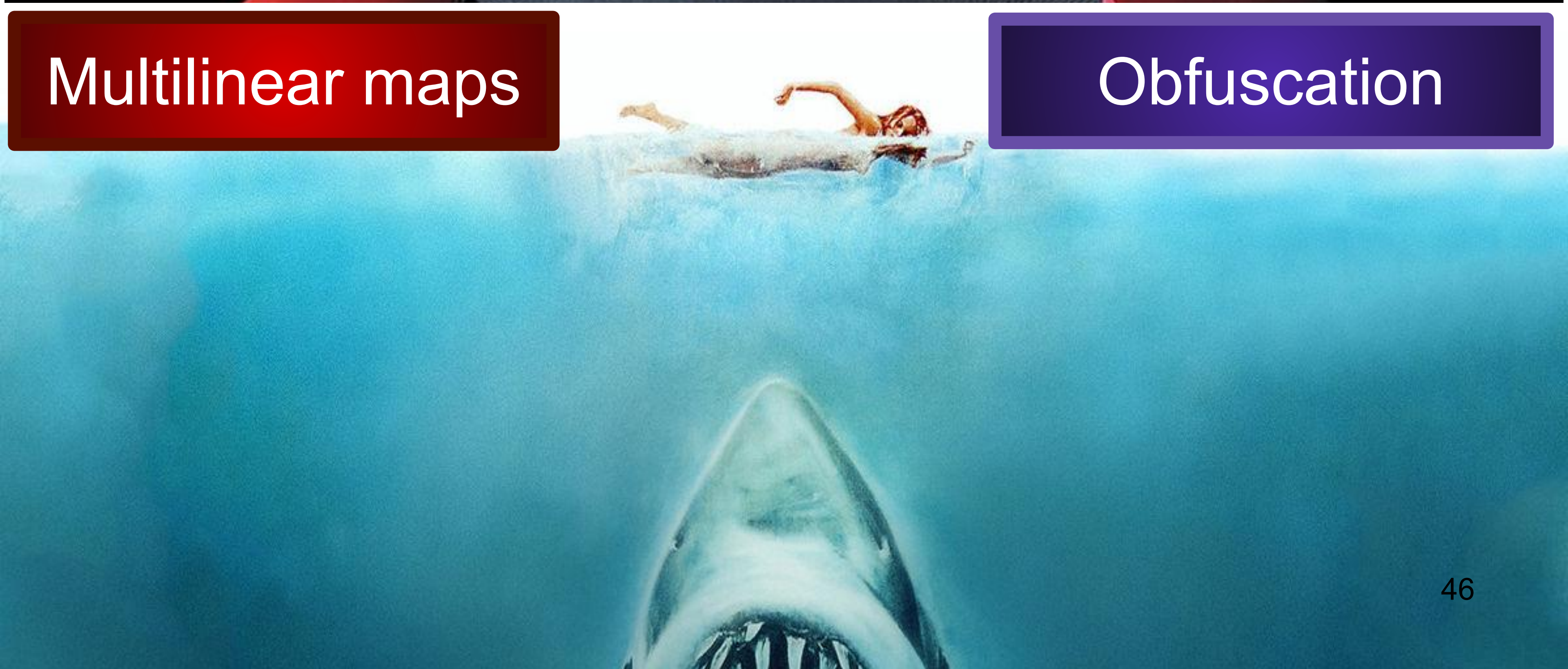
Age of discovery


2017
Watermarking
Deniable encryption
Instantiating
random oracles
Functional encryption
Multilinear maps
Obfuscation
Witness encryption
Time-lock puzzle
Delegation
Hardness of finding
Nash Equilibrium
[CGH 17]


2018
Watermarking
Deniable encryption
Functional encryption
Instantiating random oracles
Multilinear maps
Obfuscation
Time-lock puzzle
Delegation
Witness encryption
Hardness of finding Nash Equilibrium
[CVW 18]

Where do we stand today?

Today



Multilinear maps

Obfuscation

Today

Multilinear maps

Obfuscation

Construct (or fix the existing) mmaps and iO: open problems

Award prices see
https://simons.berkeley.edu/crypto2015/open-problems

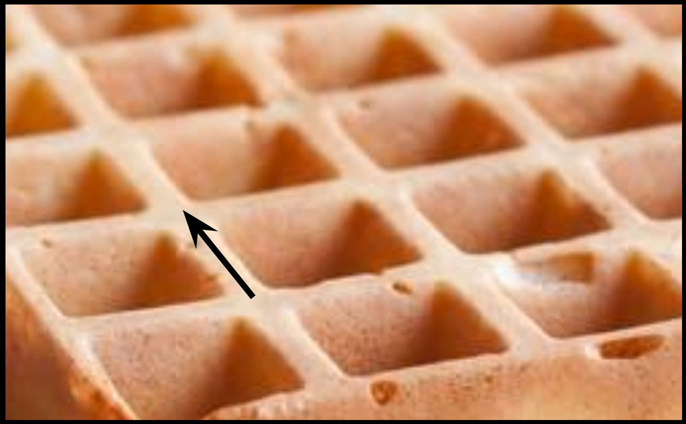
Multilinear maps

Obfuscation

Lattices

The plan for the rest of the talk:

> *Hiding secrets in public random functions*
>
> *Based on short vector problems on lattices*

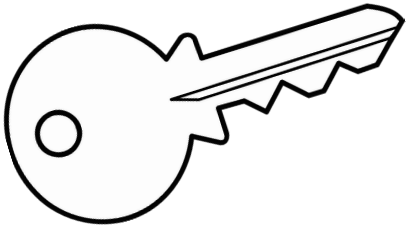


Construct private constrained PRFs from lattices

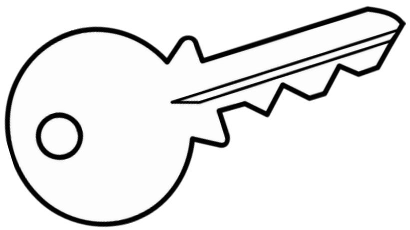
Constraint-hiding PRFs for NC1 circuits from LWE
Ran Canetti, Yilei Chen
Eurocrypt 2017

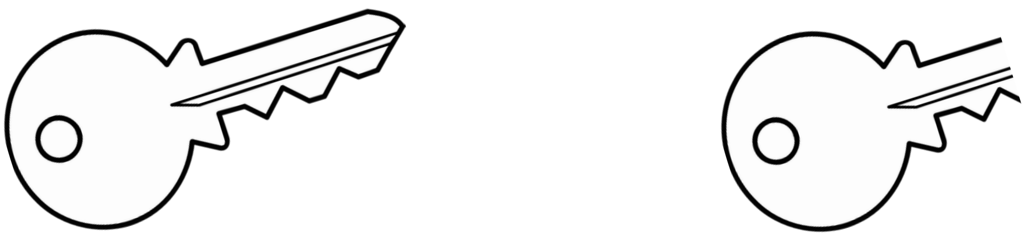
GGH15 beyond permutation branching programs
Yilei Chen, Vinod Vaikuntanathan, Hoeteck Wee
In submission 2018

Private Constrained PRF in 3 mins



PRF = Pseudorandom function [Goldwasser, Goldreich, Micali 84]



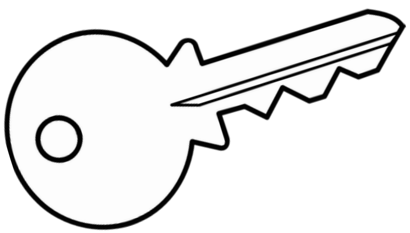


original key

constrained key (a modified key)

Constrained PRF

Private Constrained PRF in a nutshell



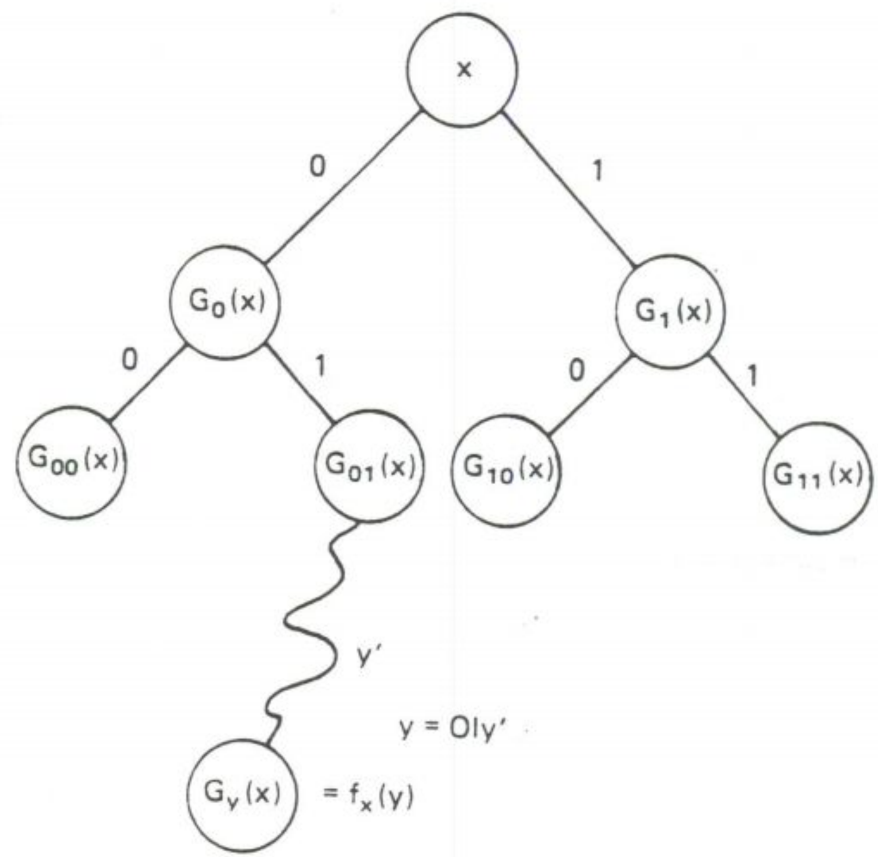
original key

constrained key (a modified key)

Constraint-hiding: hide where it is modified

Private Constrained PRF in a nutshell

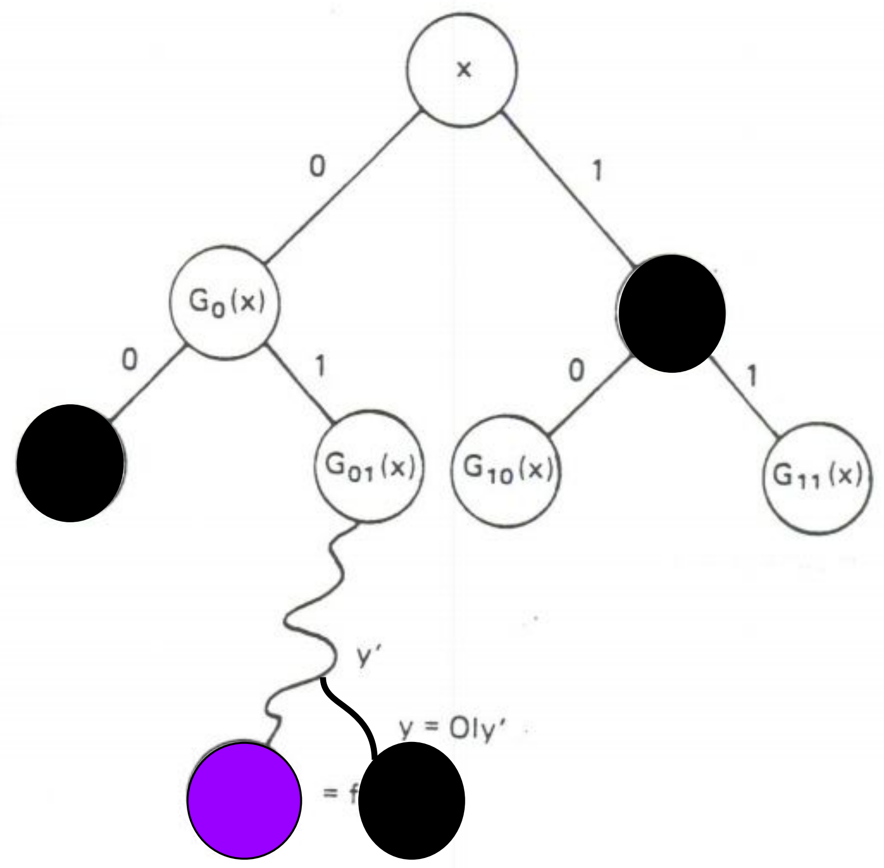
Constrained PRF (not hiding)
Puncturable PRF from [GGM 84]



Constrained PRF (not hiding)
Puncturable PRF from [GGM 84]

original

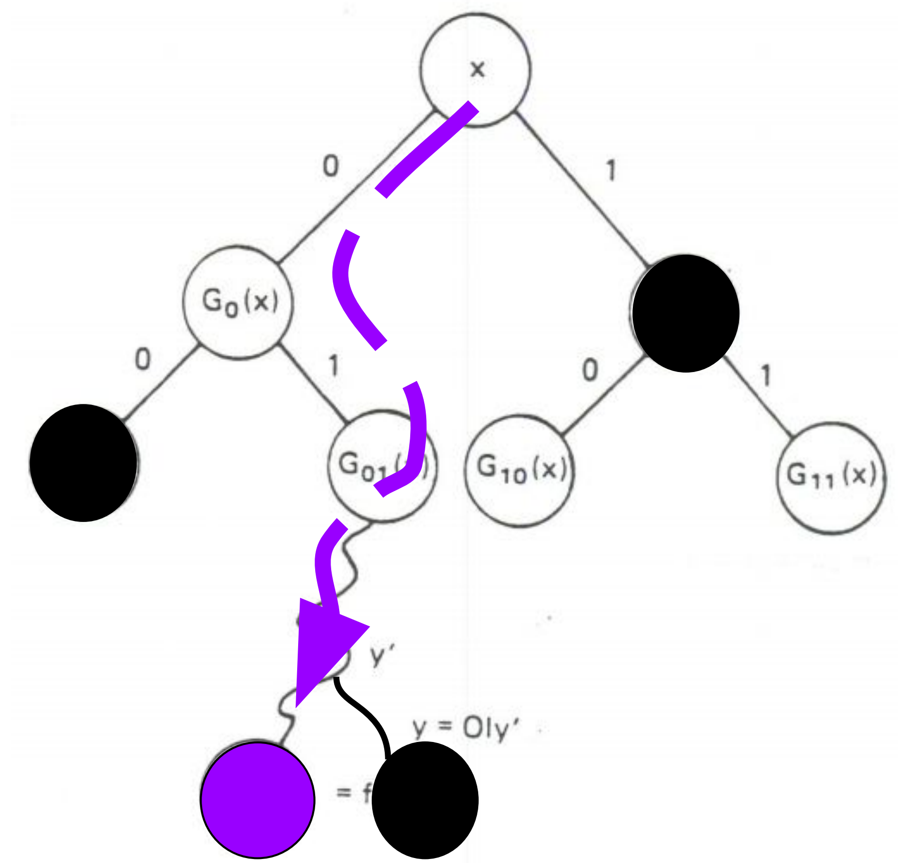
fresh random



Constrained PRF (not hiding)
Puncturable PRF from [GGM 84]

original

fresh random



The punctured key reveals the point x*

Constrained PRF

Private Constrained PRF

hide

Private Constrained PRF [Boneh, Lewi, Wu 17]

Private Constrained PRF [Boneh, Lewi, Wu 17]

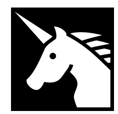
? = embed a secret message
  => watermarking [BLW17]

$$CK\_\{x^*\} = \begin{cases} \text{“Alice”}, & \text{if } x = x^* \\ F(x), & \text{else} \end{cases}$$

Watermarked key

Private Constrained PRF [Boneh, Lewi, Wu 17]

? = embed a secret message
=> watermarking [BLW17]

? = modify the key according to a function F
=> functional encryption [CC17] 

[The functional decryption key]

CK_{F,sk} = F[Dec_sk(x)]

Private Constrained PRF [Boneh, Lewi, Wu 17]

? = embed a secret message
  => watermarking [BLW17]

? = modify the key according to a function F
  => functional encryption [CC17]

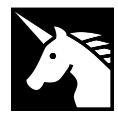
? = if you can achieve 2-key security
  => Obfuscation [CC17]

CK[C]

CK[I]

Private Constrained PRF [Boneh, Lewi, Wu 17]

? = embed a secret message
=> watermarking [BLW17]

? = modify the key according to a function F
=> functional encryption [CC17] 

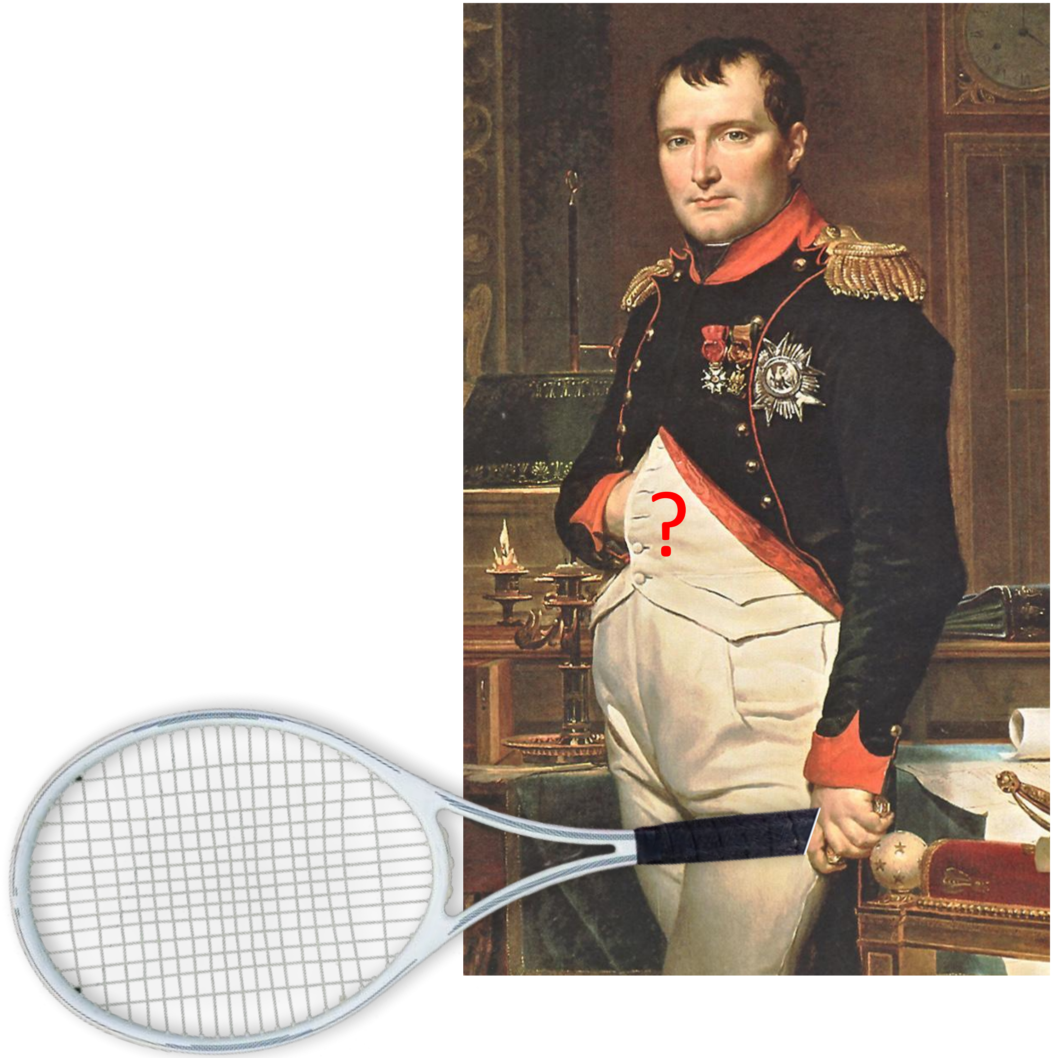
? = if you can achieve 2-key security
=> Obfuscation [CC17] 

? = 

Watermarking

Functional encryption

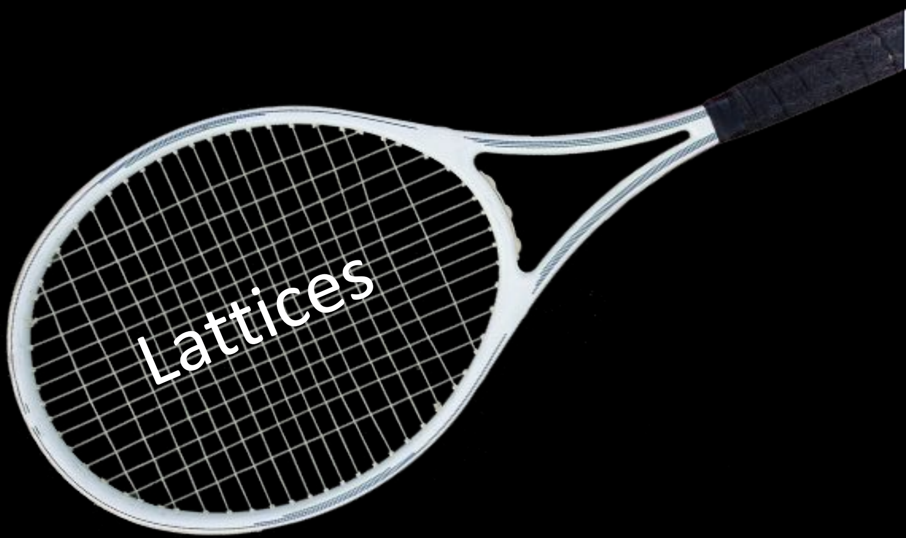
Indistinguishability obfuscation?
(if PCPRF is two-key secure)



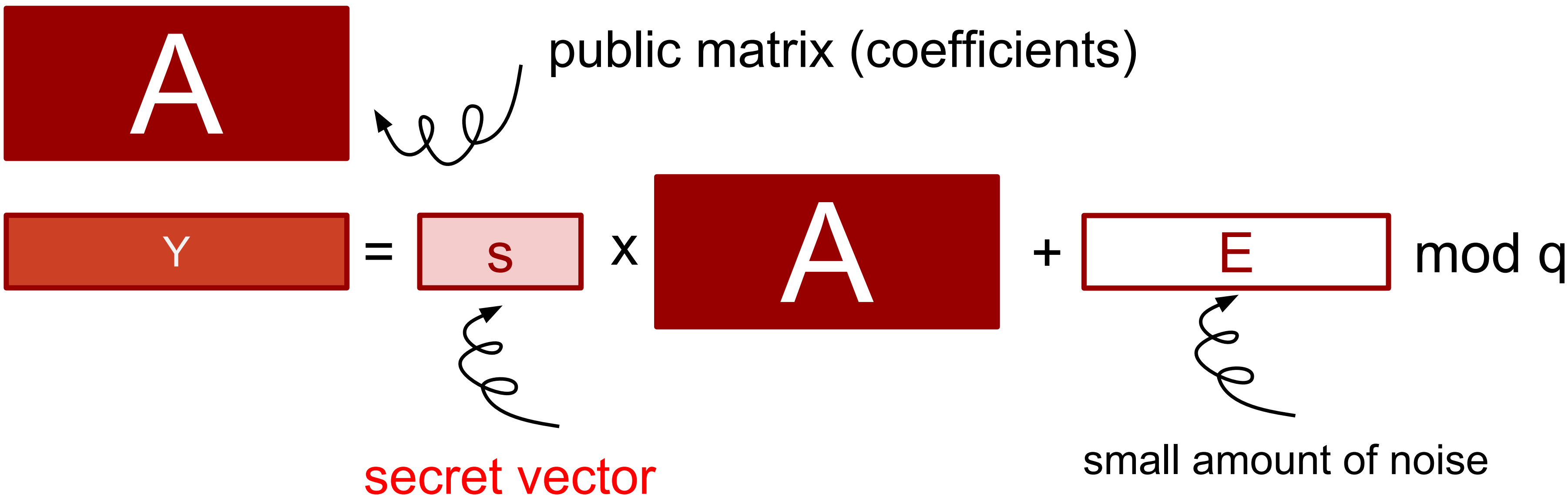
How to build PCPRF?

Main construction [CC17]

Private constrained PRF for NC1 with 1-key security from Learning With Errors.

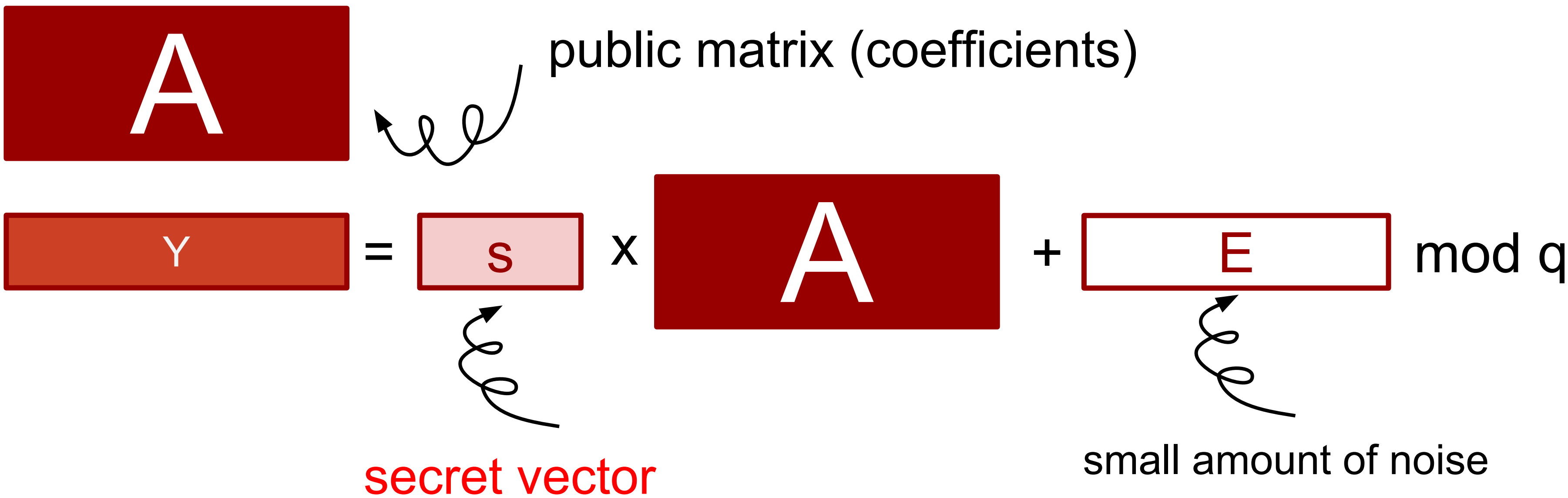


Learning with errors [Regev 05]



LWE: given A, Y, find s.

Learning with errors [Regev 05]



LWE: given A, Y, find s.

Proved to be as hard as short-vector problems in lattices [Regev 05].
Conjectured to be hard even for quantum computers.

Starting point: a plain PRF by [Banerjee, Peikert, Rosen 12]

Key:

$s_{1,1}$	$s_{2,1}$	…	$s_{n,1}$
$s_{1,0}$	$s_{2,0}$	…	$s_{n,0}$

secret matrices

A mod q

public matrix

Eval: $F(x) = \{ \prod s_{i,x_i} A \}_2$

Starting point: a plain PRF by [Banerjee, Peikert, Rosen 12]

Key: $\begin{matrix} \boxed{s_{1,1}} & \boxed{s_{2,1}} & \cdots & \boxed{s_{n,1}} \\ \boxed{s_{1,0}} & \boxed{s_{2,0}} & \cdots & \boxed{s_{n,0}} \end{matrix} \quad \boxed{A} \quad \text{mod } q$

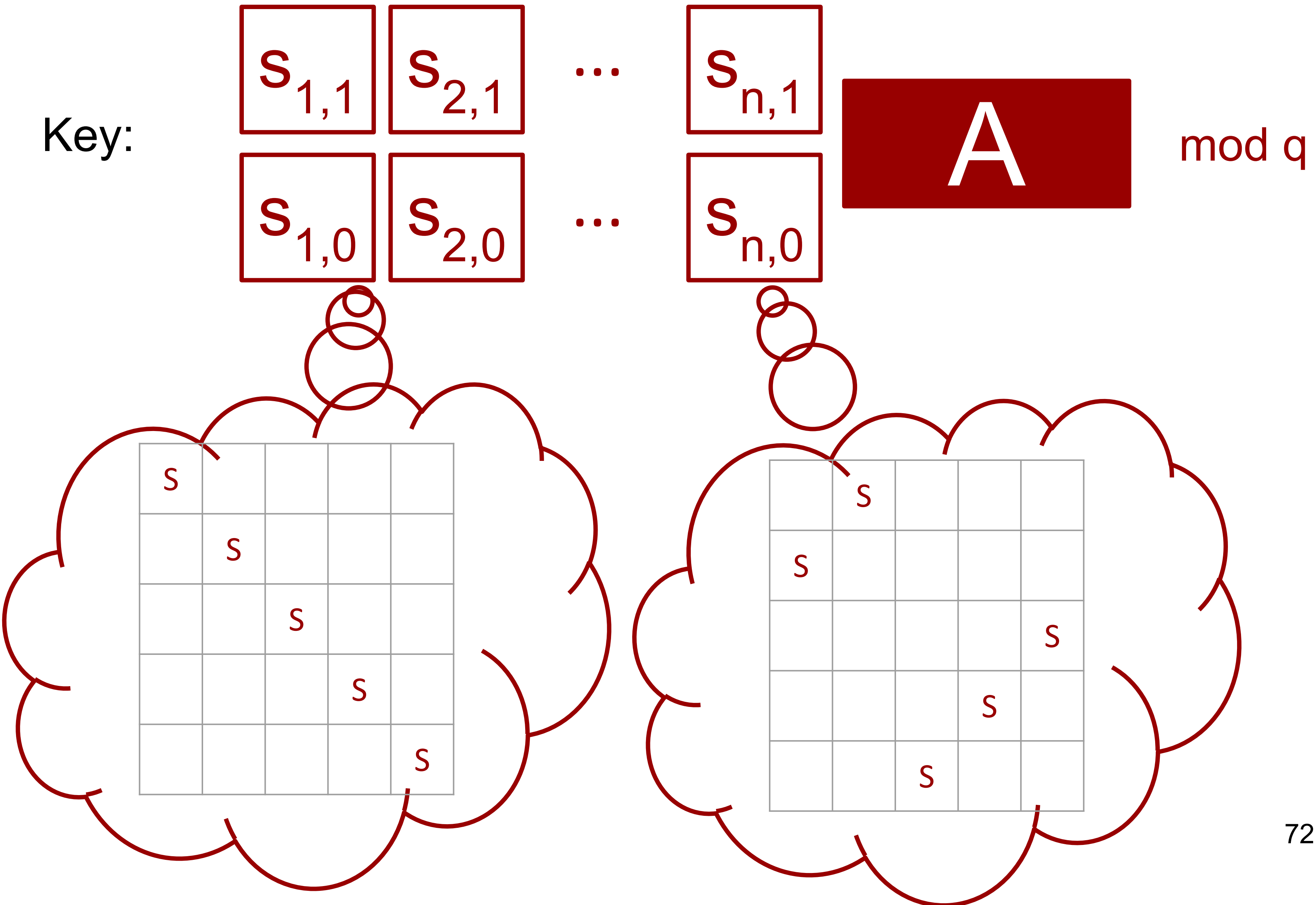
Eval: $F(x) = \{ \prod s_{i,xi} \; A \}_2$

<u>What we need in addition to build a PCPRF</u>:

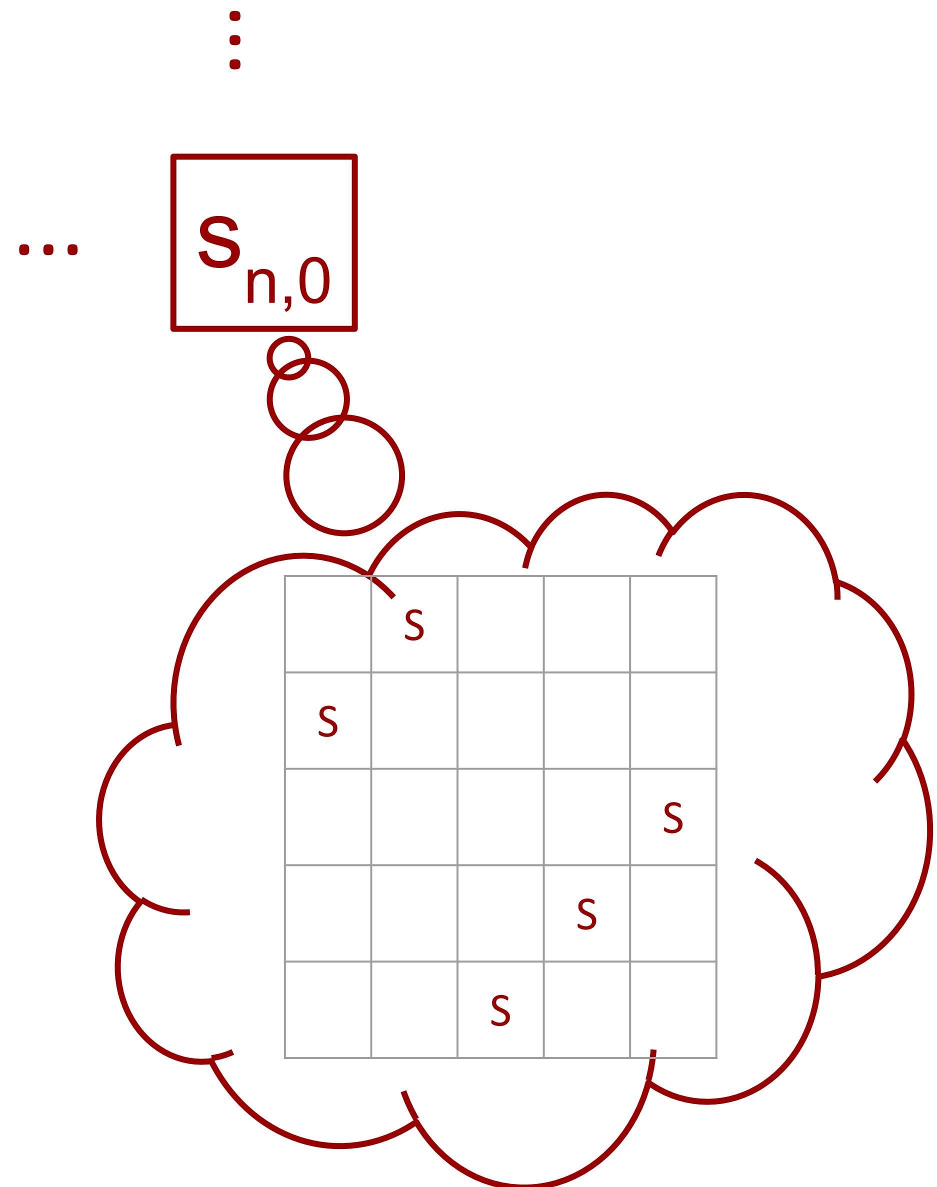
+ Embed structures in the secret terms to perform functionality
+ A proper public mode of the function

Step 1: Embed structures: permutation branching programs

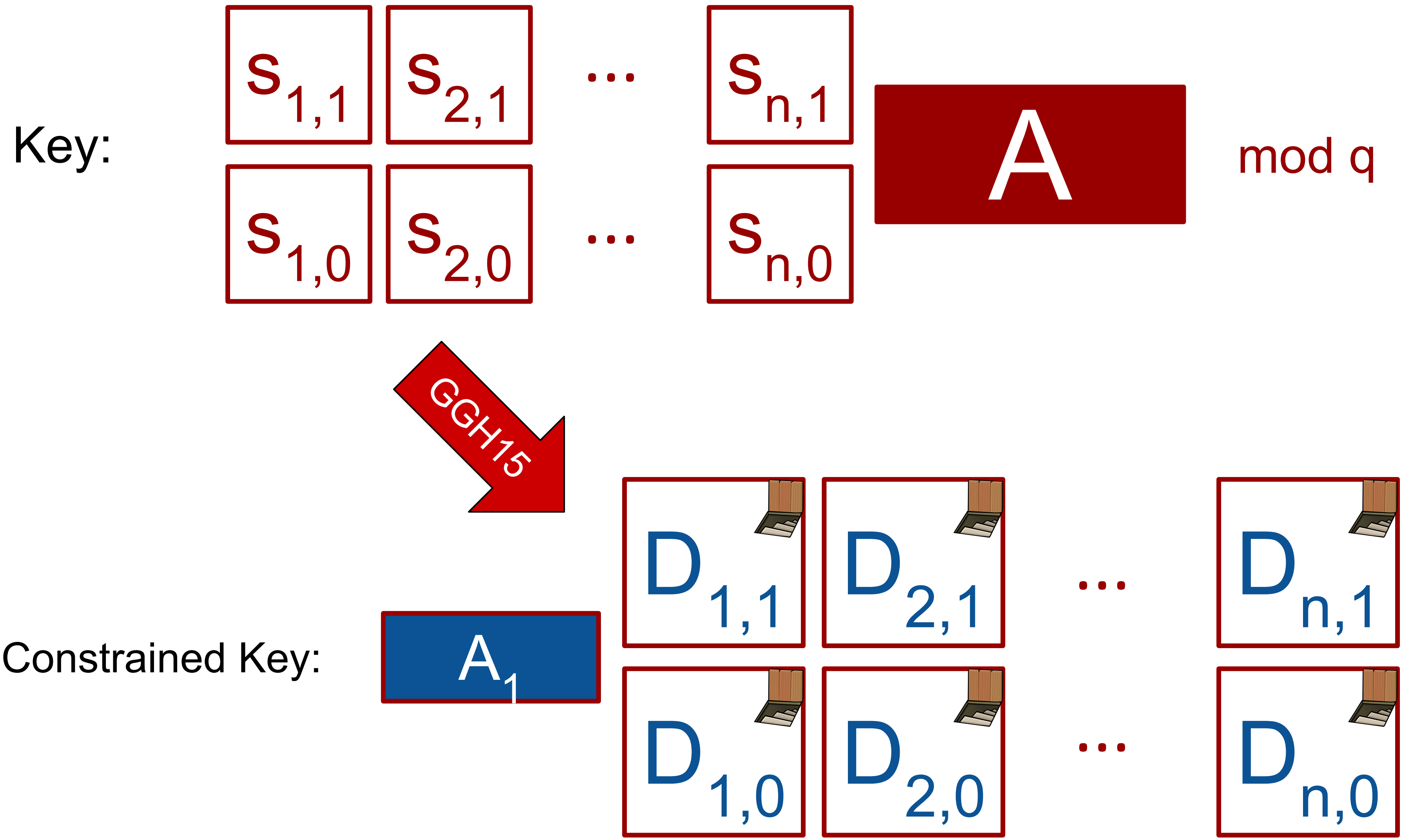
Step 1: Embed structures: permutation branching programs



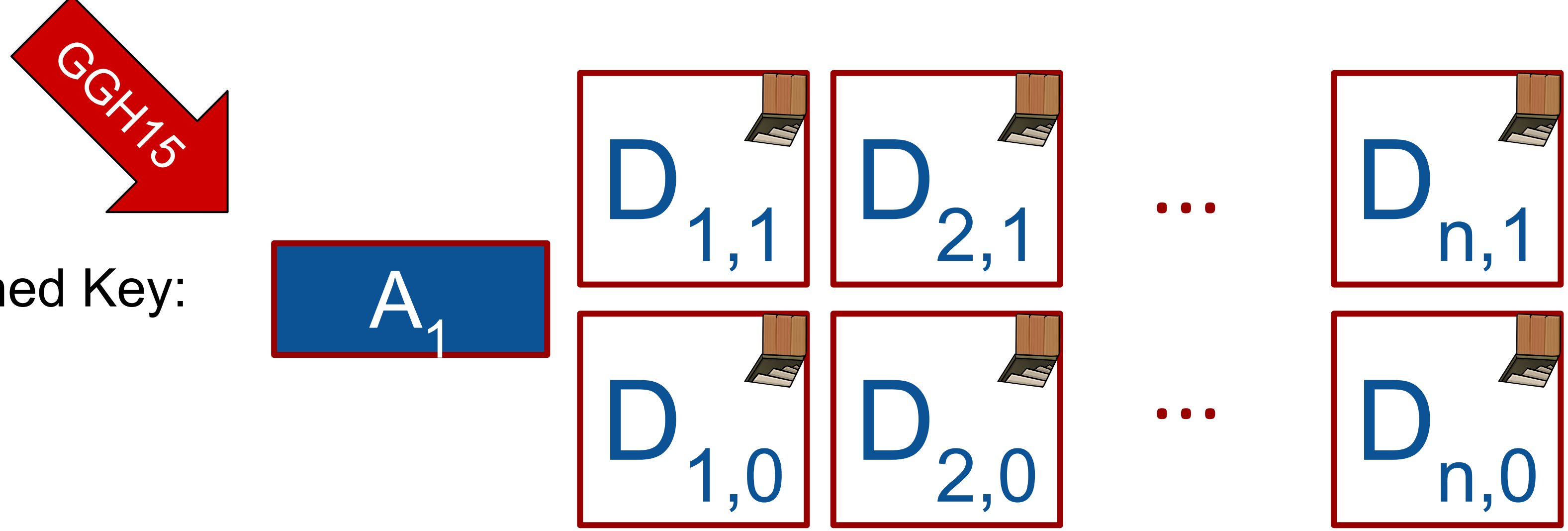
Step 2: Encode the structured key?



Step 2: Encode the structured key?

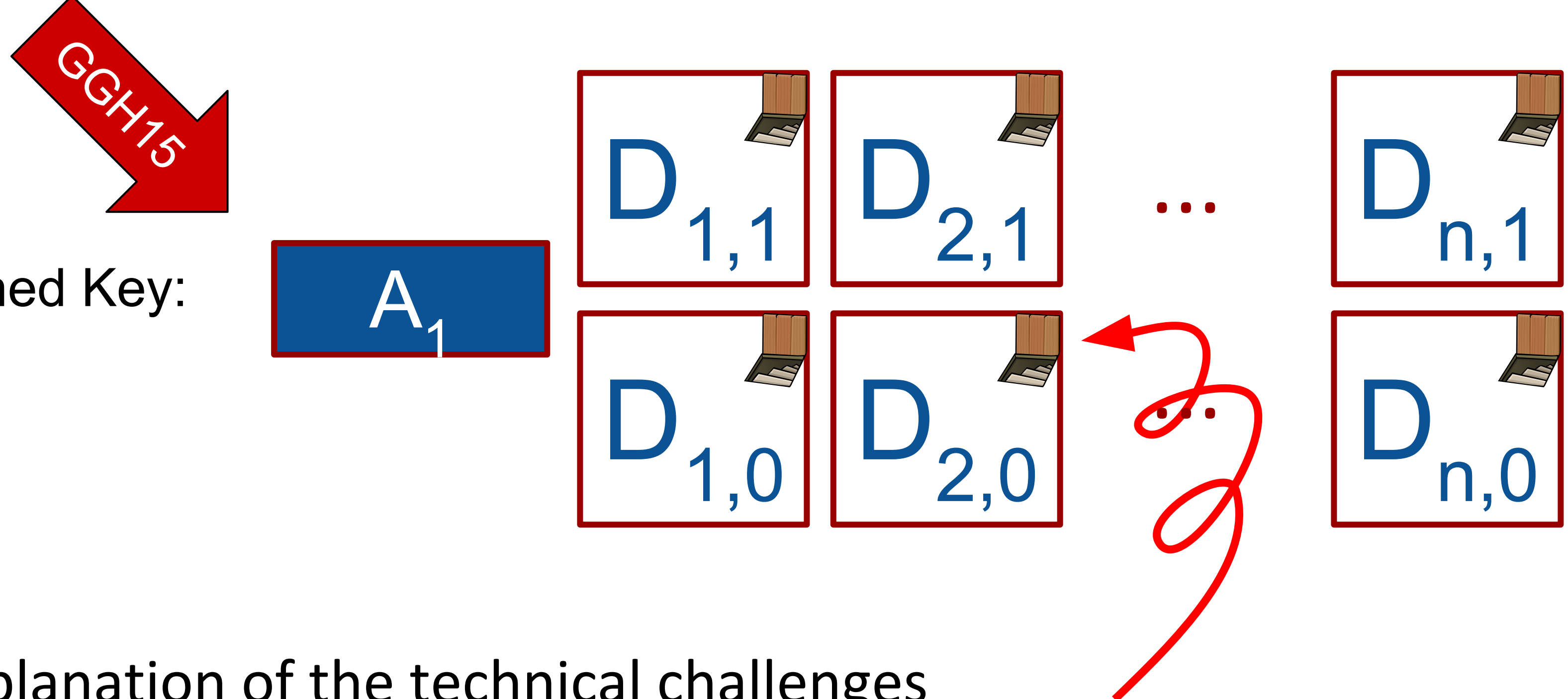


Constrained Key:



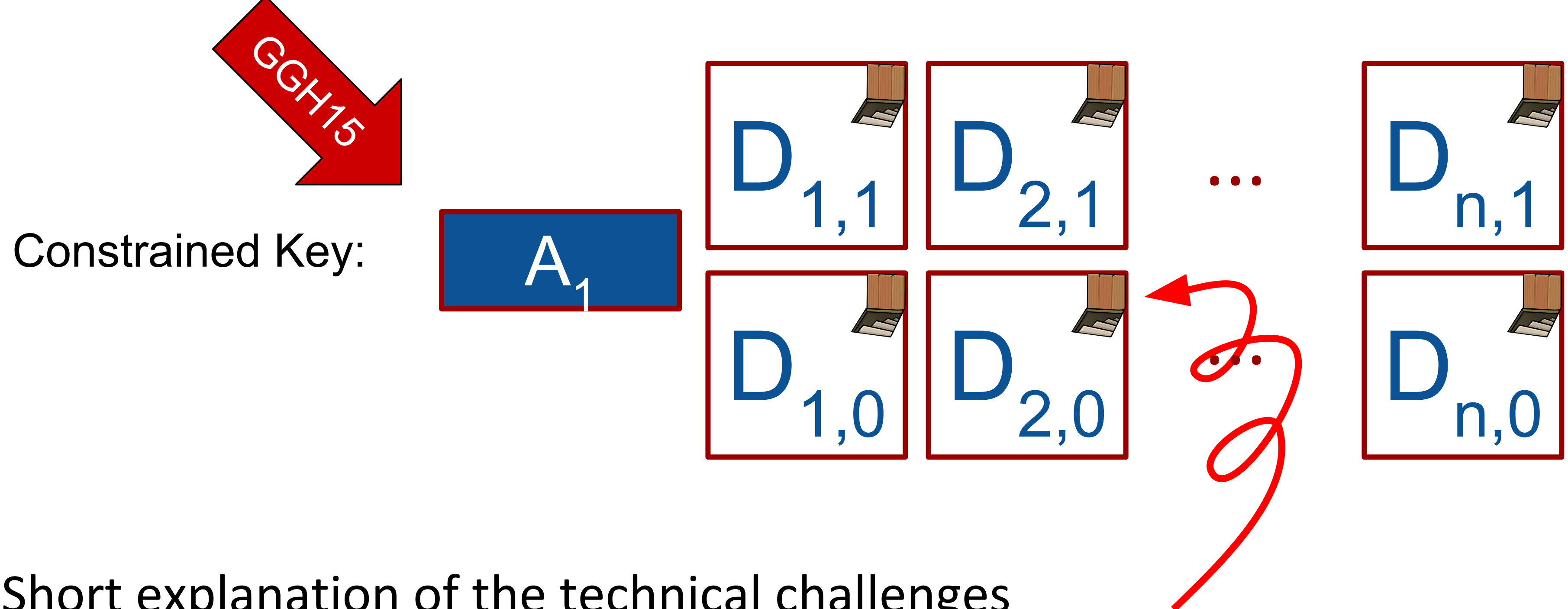
Short explanation of the technical challenges

Constrained Key:



Short explanation of the technical challenges

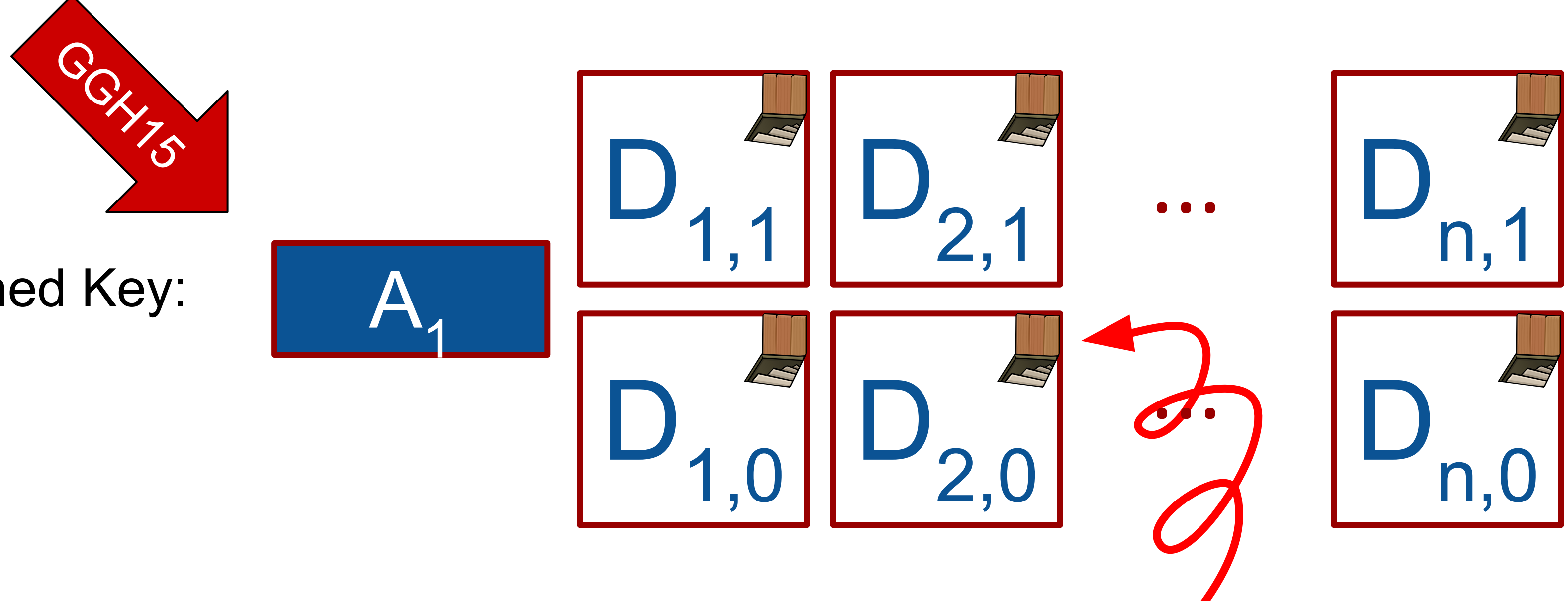
- GGH15 encoding uses LWE with lattice trapdoors, trapdoor sampling of arbitrary plaintext can be dangerous (as revealed by cryptanalytic attacks)



Short explanation of the technical challenges

- GGH15 encoding uses LWE with lattice trapdoors, trapdoor sampling of arbitrary plaintext can be dangerous (as revealed by cryptanalytic attacks)
- [CC 17] discovers a "safe mode" related to permutation matrices.

Constrained Key:

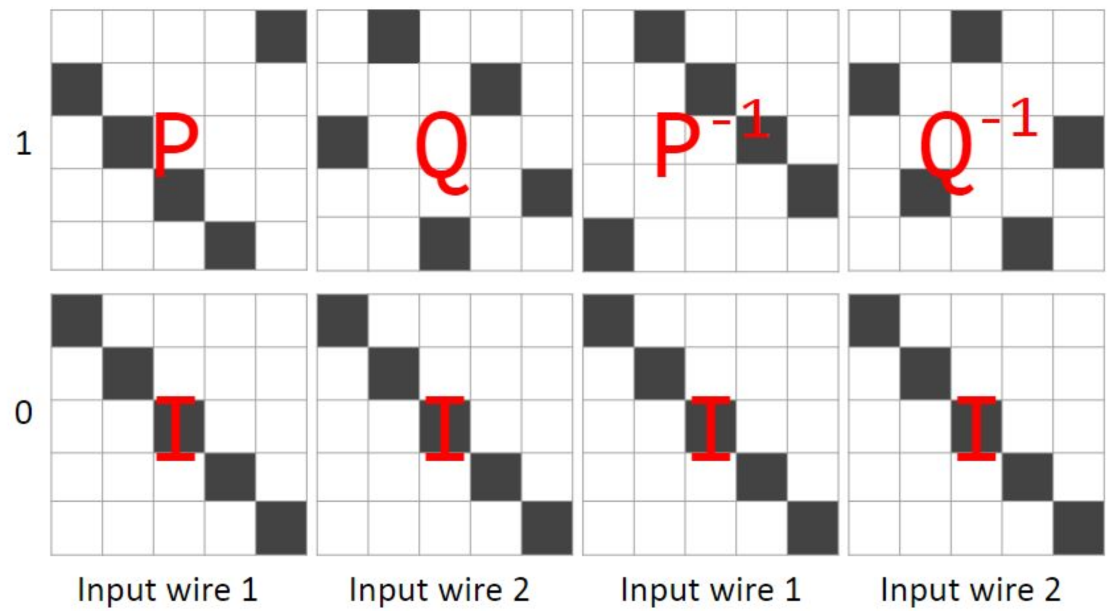


Short explanation of the technical challenges

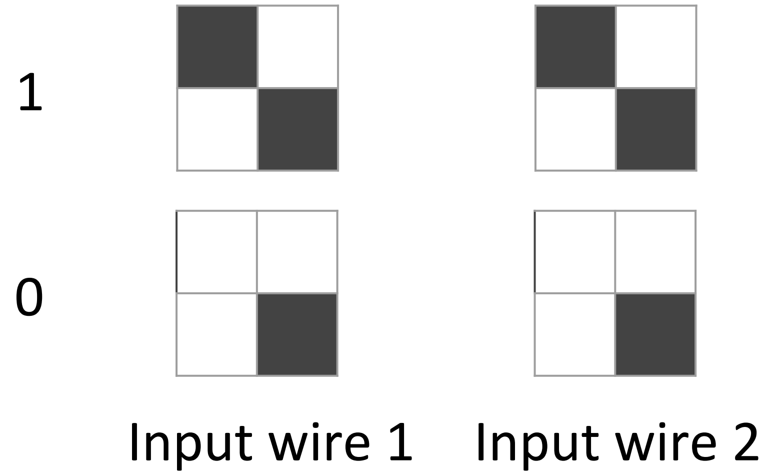
- GGH15 encoding uses LWE with lattice trapdoors, trapdoor sampling of arbitrary plaintext can be dangerous (as revealed by cryptanalytic attacks)
- [CC 17] discovers a "safe mode" related to permutation matrices.
- [CVW 18] shows "more general safe modes". As a result, it improves the efficiency of e.g. private puncturable PRFs.

Demo: 2-bit PRF that is punctured on x*=11:

[CC 17] uses permutation matrices.



[CVW 18] uses diagonal matrices (low-rank).



Performance?

Implementations of GGH15-based schemes:

1. BPobfus
2. PALISADE

Current status: (for 80-bit security)

Multilinearity	generate the encodings	storage	time per evaluation
4	1 min	1 GB	< 1 sec
8	8 mins	8 GB	< 4 sec
16	120 mins	300 GB	< 100 sec

Performance?

Implementations of GGH15-based schemes:
1. BPobfus
2. PALISADE

Current status: (for 80-bit security)

Multilinearity	generate the encodings	storage	time per evaluation
4	1 min	1 GB	< 1 sec
8	8 mins	8 GB	< 4 sec
16	120 mins	300 GB	< 100 sec

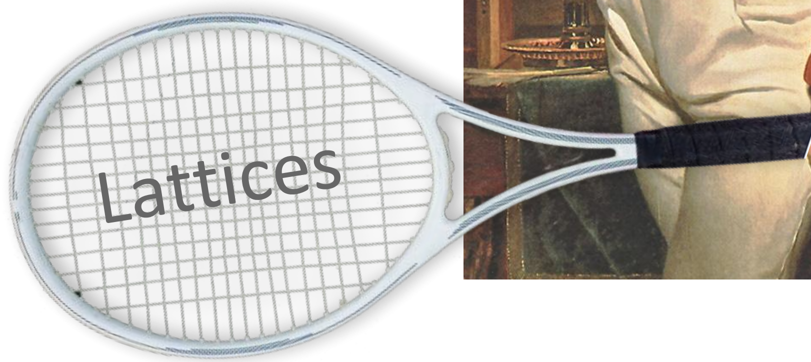
Estimation of 16 bit private puncturable PRF using [CVW 18]:
use multilinearity = 4, wordsize of $2^4 = 16$,

16 mins, 16 GB storage, 1 sec per eval.

Watermarking

Functional encryption

Indistinguishability obfuscation?
(if PCPRF is two-key secure)



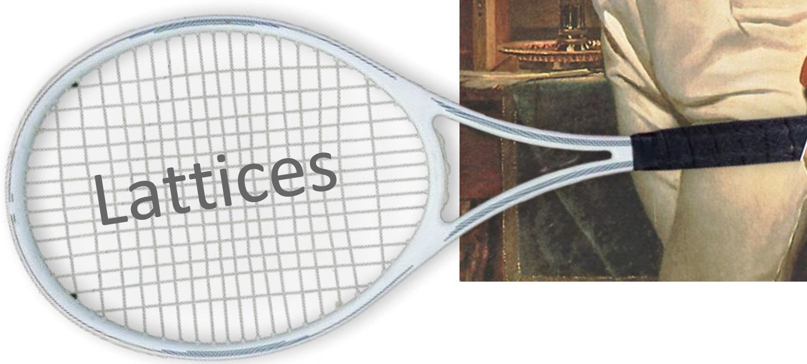
What’s more?

Watermarking

Functional encryption

Indistinguishability obfuscation?
(if PCPRF is two-key secure)

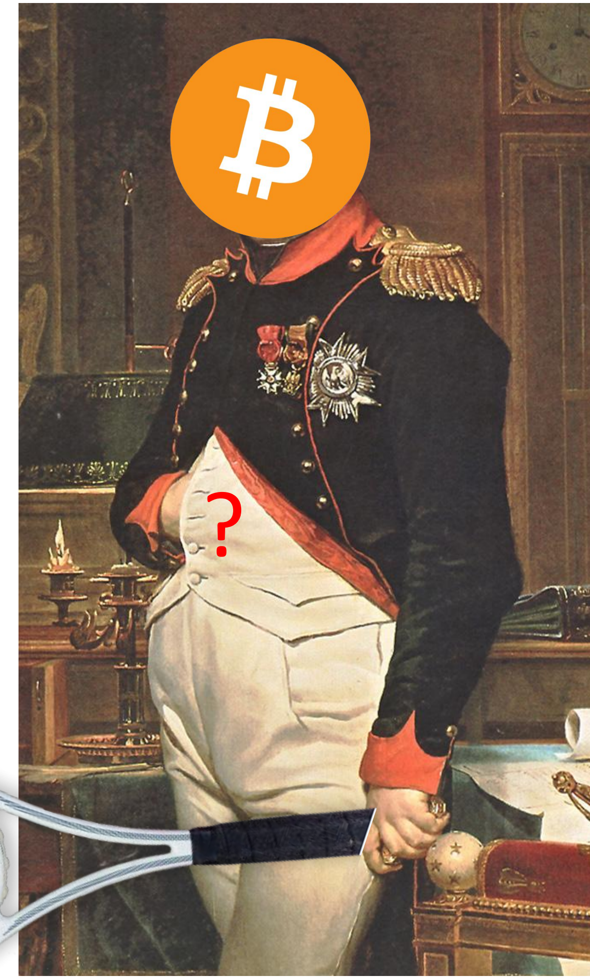
Correlation Intractable functions

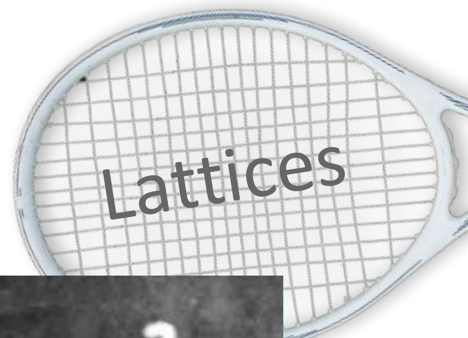


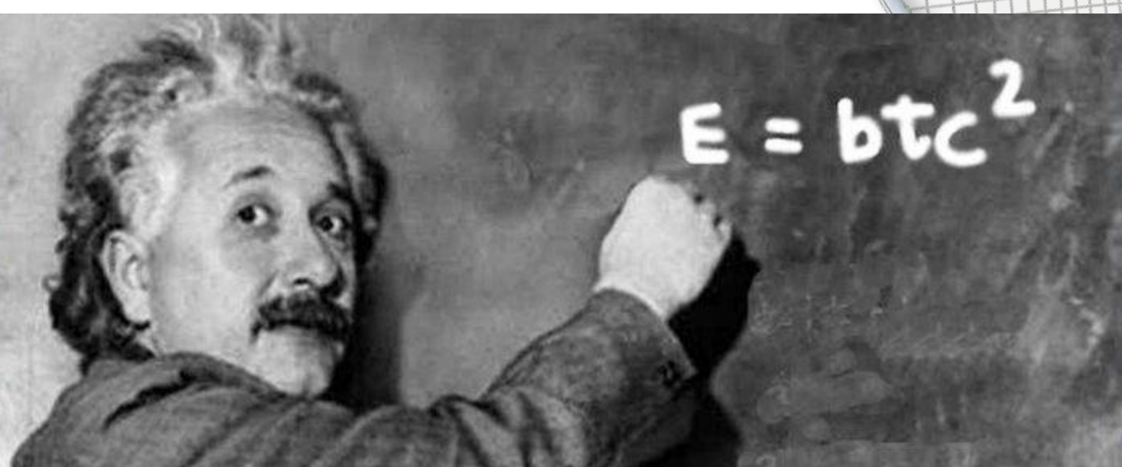
Watermarking

Functional encryption

Indistinguishability obfuscation?
(if PCPRF is two-key secure)







Constructing cryptographic hash functions:

- On the correlation intractability of obfuscated pseudorandom functions
  Ran Canetti, Yilei Chen, Leonid Reyzin
  Theory of Cryptography Conference 2016-A

- Fiat-Shamir from strong KDM encryption schemes
  Ran Canetti, Yilei Chen, Leonid Reyzin, Ron Rothblum
  Eurocrypt 2018

A <u>public</u> function

$h\colon \{0,1\}^l \rightarrow \{0,1\}^m$

behaves like a random function?

One of the desirable property of a public random function is

Correlation Intractability

“infeasibility of finding ‘rare’ input-output relations”

Sparse Relations

"For each input (x),
the fraction of outputs (y) in the relation is negligible"

Sparse Relations

“For each input (x),
the fraction of outputs (y) in the relation is negligible”

Correlation intractability [Canetti, Goldreich, Halevi ‘98]

For all sparse relations R:

Adversary Challenger

h

x, (as a result, y=h(x))

Adversary wins if R(x, y)=1

H(???...?)=000000….XYZ3d83h

$H(???...?)=\underbrace{000000}_{d}....XYZ3d83h$

=> the best algorithm shall run in time 2^d

Correlation intractability
for moderately sparse relations [CCRR 18]

How to construct?

What we know about CI:

[Canetti, Goldreich, Halevi 98] some parameters are impossible.

[Goldwasser, Kalai 03] more parameters are impossible.

[Bitanski et al. 13] Impossible to prove based on black-box reduction to efficiently falsifiable assumptions. (difficult to prove)

Before 2015: sometime cited as unconditionally impossible

What we know about CI:

…

SHA256 (and others) are heuristical candidates.

We want to base CI on clear mathematical problems.

Intuitive difficulty: how to handle "weird relations" like

R(x, y) = 1 iff the "2nd ~ 2n-th bits of (3^x +x^2)" = y

Canetti, Chen, Reyzin (TCC 2016-A)

Bounded correlation intractability from iO(Puncturable.PRF)

Canetti, Chen, Reyzin (TCC 2016-A)

Bounded correlation intractability from iO(Puncturable.PRF)

Canetti, Chen, Reyzin, Rothblum (Eurocrypt 2018)

CI for every sparse relation from exponentially KDM secure encryption schemes.
(with candidates from LWE and discrete-log like problems + KDM assumption)

Canetti, Chen, Reyzin (TCC 2016-A)

Bounded correlation intractability from iO(Puncturable.PRF)

Canetti, Chen, Reyzin, Rothblum (Eurocrypt 2018)

CI for every sparse relation from exponentially KDM secure encryption schemes.
(with candidates from LWE and discrete-log like problems + KDM assumption)

Main idea in the analysis:

1. Starting from a relation R and a random function F
2. Moving to F_R that is indistinguishable from F
3. Finally, argue the correlation intractability of F_R

Canetti, Chen, Reyzin (TCC 2016-A)

Bounded correlation intractability from iO(Puncturable.PRF)

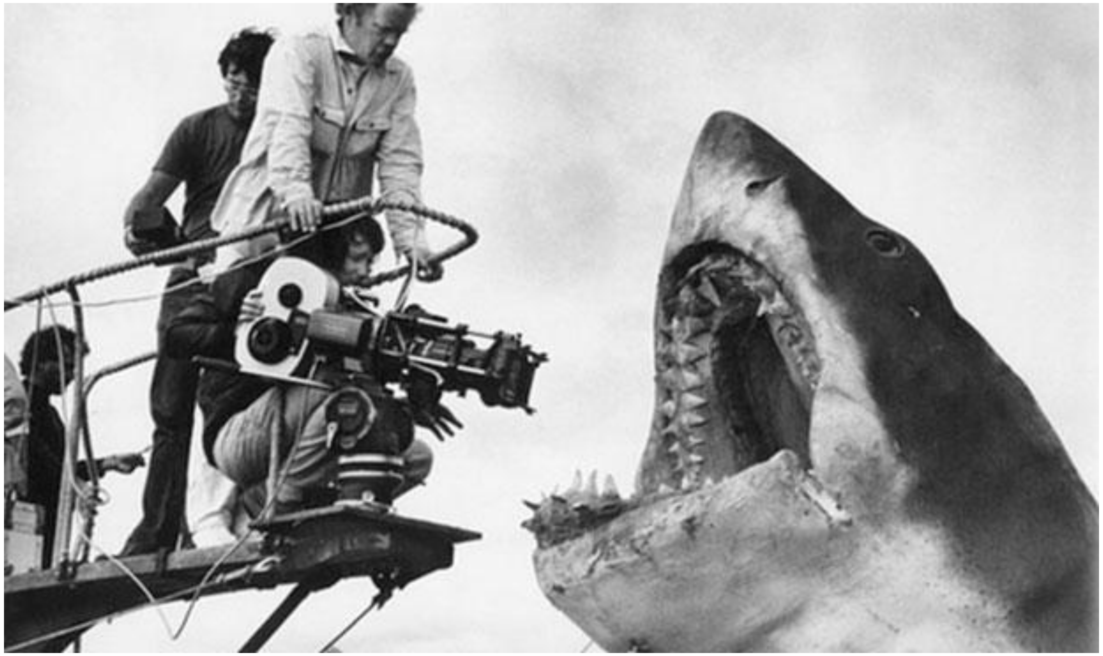
Canetti, Chen, Reyzin, Rothblum (Eurocrypt 2018)

CI for every sparse relation from exponentially KDM secure encryption schemes.
(with candidates from LWE and discrete-log like problems + KDM assumption)

Main idea in the analysis:

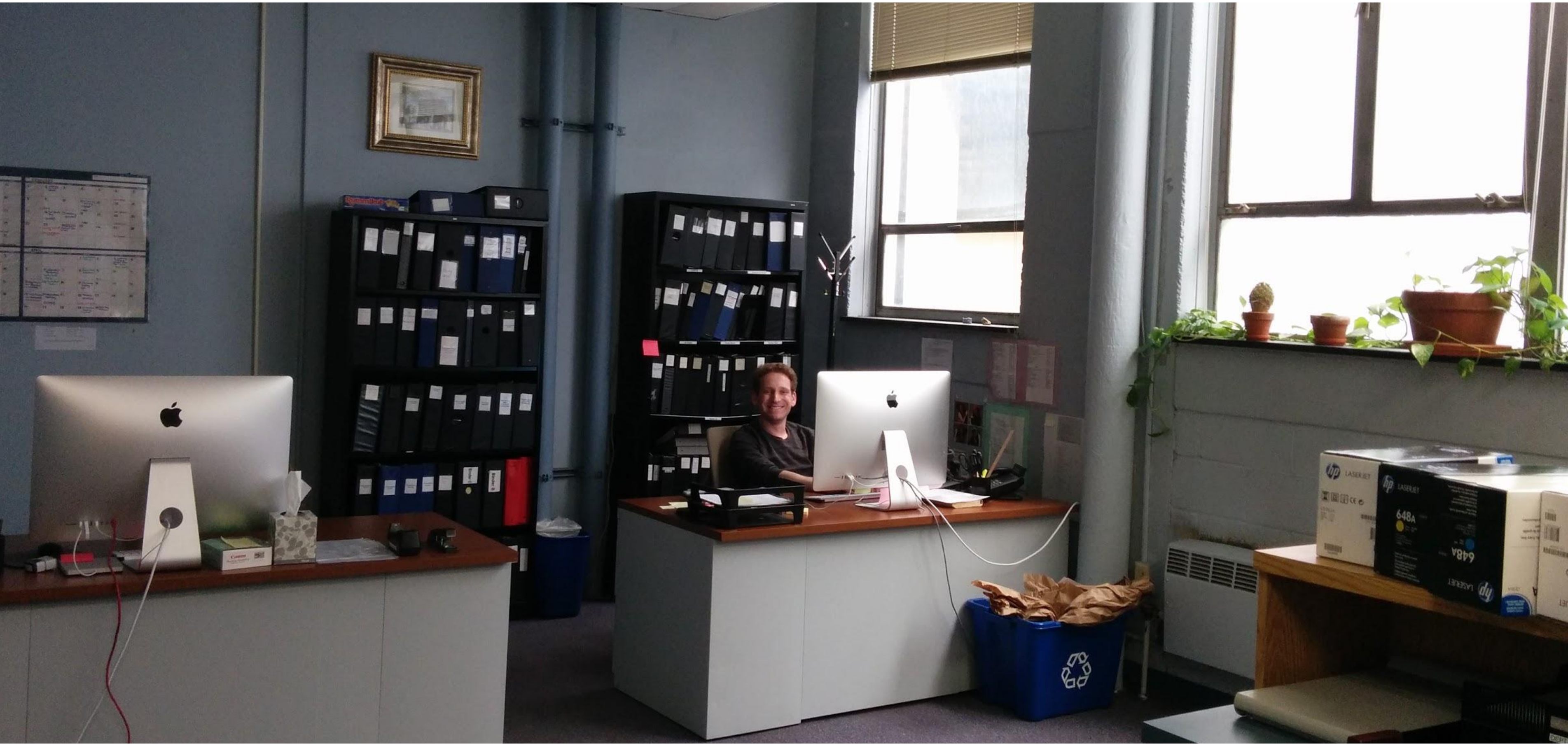
1. Starting from a relation R and a random function F
2. Moving to F_R that is indistinguishable from F
3. Finally, argue the correlation intractability of F_R

Reasoning about moderate hardness is still an open problem.



3 more mins … Acknowledgements

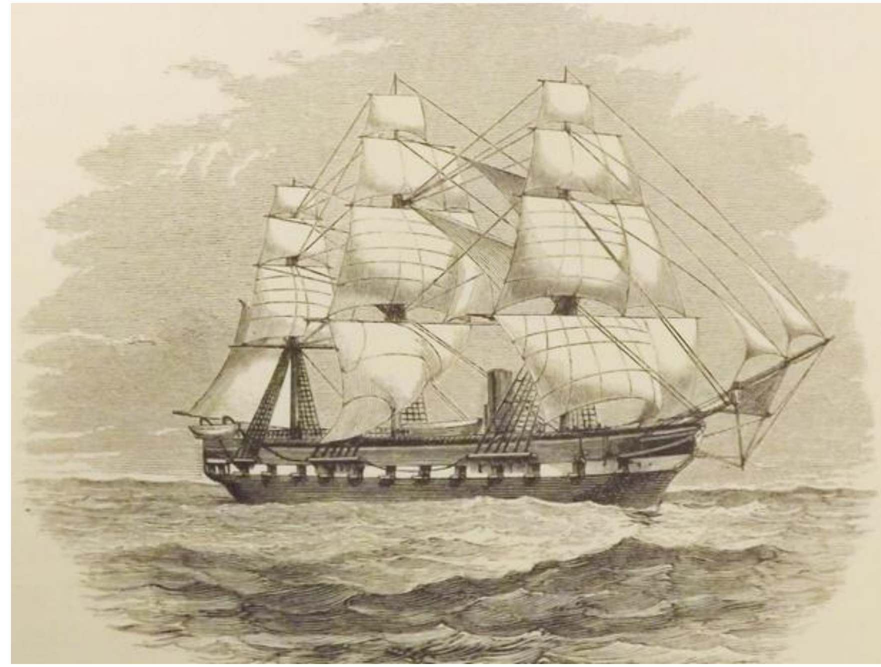
Ran watching thrillers in a restaurant (Jan 2015)



Rare footage of Leo running the main office (Mar 2015)

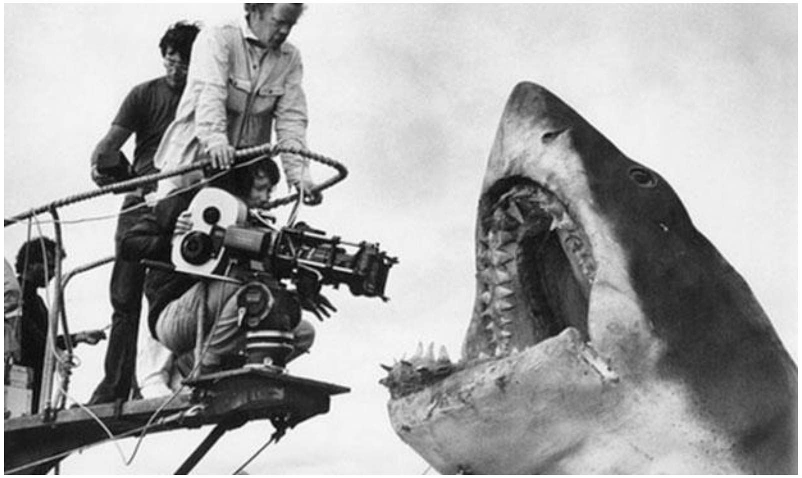
Vinod looking for the most beautiful short non-zero vector in the desert. (Jan 2016)

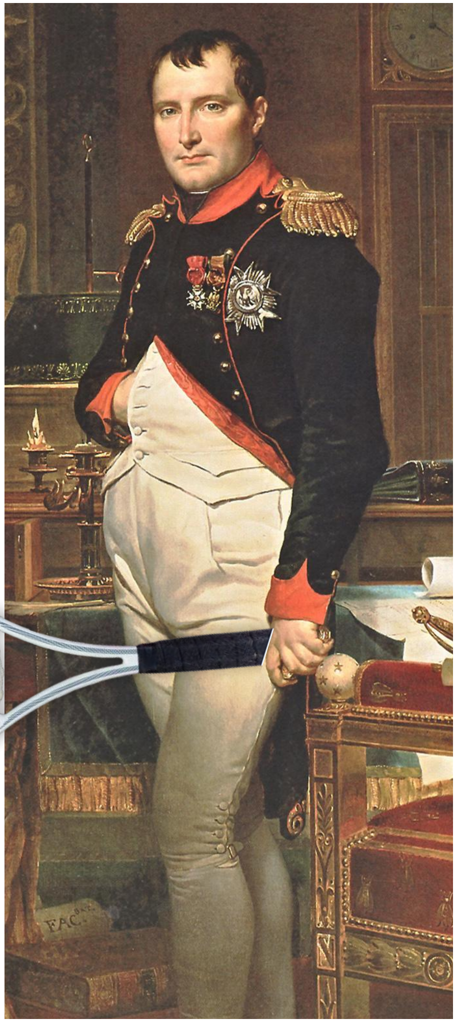
Boston University
Security Group

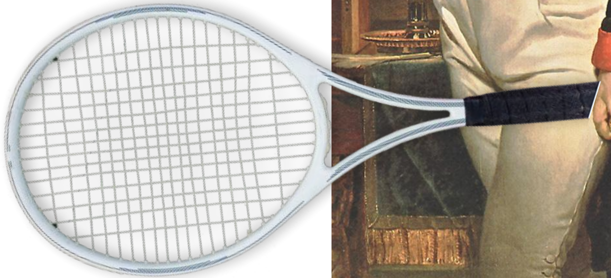


Coauthors:

Craig Gentry,
Shai Halevi,
Justin Holmgren,
Mariana Raykova,
Ron Rothblum,
Hoeteck Wee.







Thanks for your time!