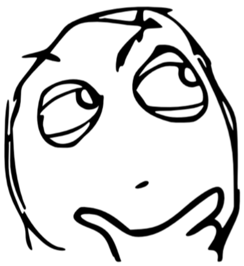
Lattices
Multilinear Maps
Obfuscation

Yilei Chen
Visa Research

Lattices: Algorithms, Complexity, and Cryptography @ Simons Institute

What are Multilinear Maps?



Multilinear maps in cryptography

> Discrete-log problem [Diffie, Hellman 76]

$$\text{Given } g,\ g^{s} \bmod q,\ \text{ finding } s \text{ is hard}$$

> Bilinear maps from Weil pairing over elliptic curve groups

[Miller 86] How to compute Weil pairing

[Sakai, Ohgishi, Kasahara 00] Identity-based key-exchange

[Joux 00] Three-party non-interactive key-exchange

[Boneh, Franklin 02] Identity-base encryption

$$g^{S_1},\ g^{S_2} \rightarrow g_T^{S_1 S_2}$$

> Multilinear maps: motivated in [Boneh, Silverberg 03] with the potential applications of constructing unique signature, broadcast encryption, etc.

$$g^{S_1},\ g^{S_2},\ g^{S_3},\ \ldots \rightarrow g_T^{\prod S}$$

> Discrete-log problem [Diffie, Hellman 76]

Given g, g^{s} mod q, finding s is hard

Turing Award

> Bilinear maps from Weil pairing over elliptic curve groups

[Miller 86] How to compute Weil pairing

[Sakai, Ohgishi, Kasahara 00] Identity-based key-exchange

[Joux 00] Three-party non-interactive key-exchange

[Boneh, Franklin 02] Identity-base encryption

Gödel Prize

$$g^{S_1}, g^{S_2} \rightarrow g_T^{S_1 S_2}$$

> Multilinear maps: motivated in [Boneh, Silverberg 03] with the potential applications of constructing unique signature, broadcast encryption, etc.

$$g^{S_1}, g^{S_2}, g^{S_3}, \ldots \rightarrow g_T^{\prod S}$$

☺

Where to find multilinear maps?

"If an n-multilinear map is computable, it is reasonable to expect it to come from geometry, as is the case for Weil and Tate pairings when $n = 2$."

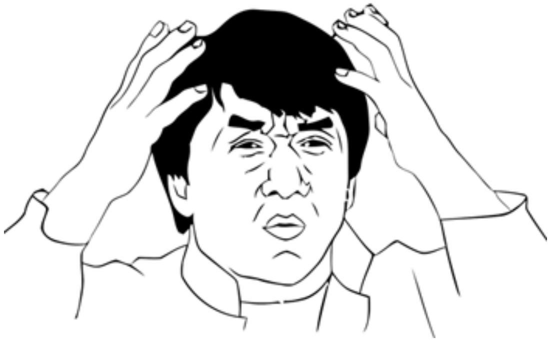
…

"If varieties giving rise to n-multilinear maps cannot be found for $n > 2$, one could at least hope that such maps might arise from *motives*." ☺

– Boneh, Silverberg, 2003

*New: Trilinear maps from abelian varieties [Huang 2019], requires further investigation.

~~What are multilinear maps?~~
Why from lattices?



Multilinear maps since 2013

> Multilinear maps: motivated in [Boneh, Silverberg 2003]

$$g,\ g^{S_1},\ g^{S_2},\ g^{S_3},\ \ldots \rightarrow g_T^{\prod S}$$

Garg, Gentry, Halevi [GGH 13] propose a candidate based on a variant of the NTRU problem
No security reduction is given; cryptanalysis attempts are mentioned.

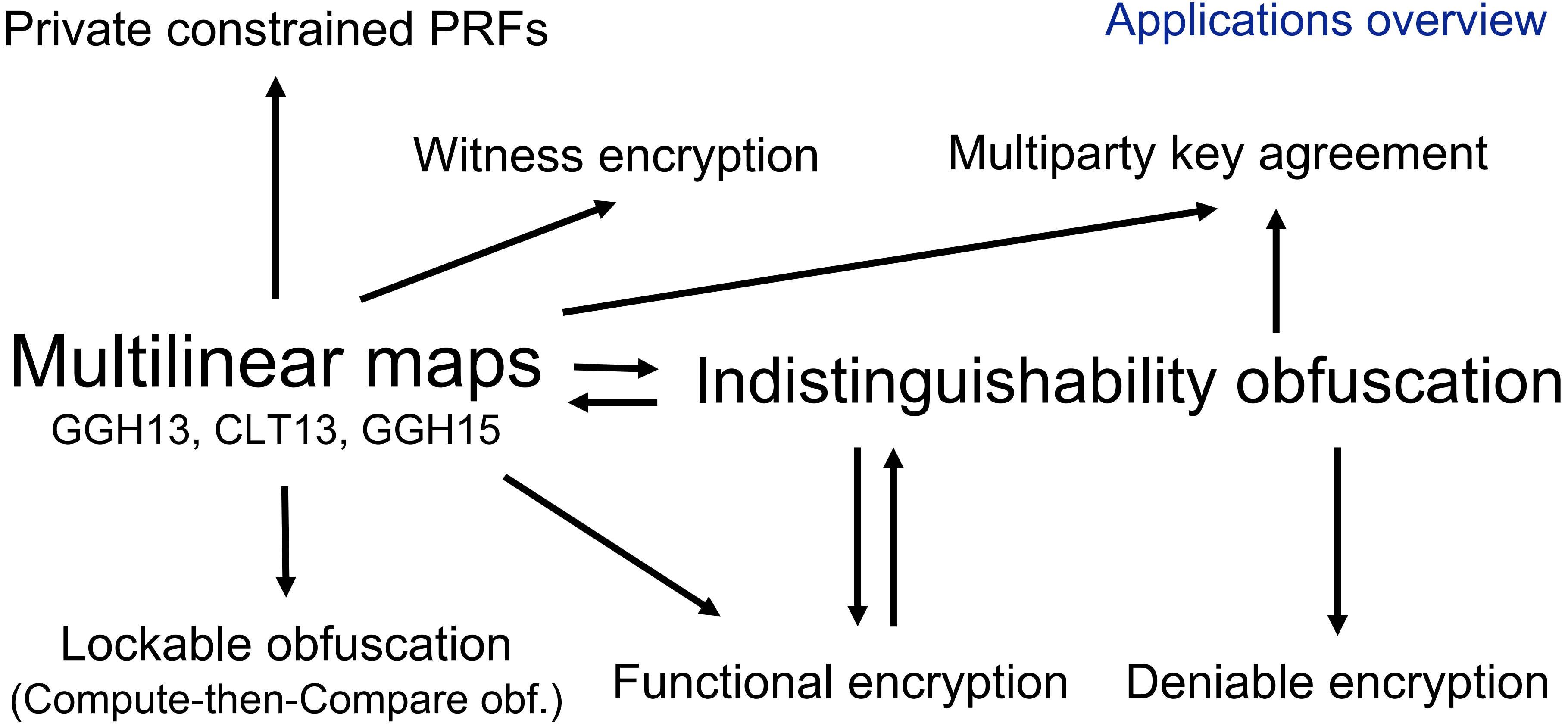
Think of as homomorphic encryption + public zero-test

i.e. everyone can test whether you get g_T^{0} or $g_T^{\text{non-zero}}$

Coron, Lepoint, Tibouchi [CLT 13] propose a candidate based on a variant of approx-gcd

Gentry, Gorbunov, Halevi [GGH 15] propose another candidate inspired by the FHE scheme of [Gentry, Sahai, Waters 13]

Multilinear maps
Applications overview



Multilinear maps ⟶ Indistinguishability obfuscation

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

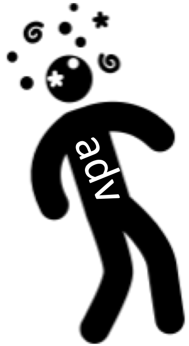
Indistinguishability obfuscation

Defined by [Barak, Goldreich, Impagliazzo, Rudich, Sahai, Vadhan, Yang 01]

Program Obfuscation: P => Obf(P)

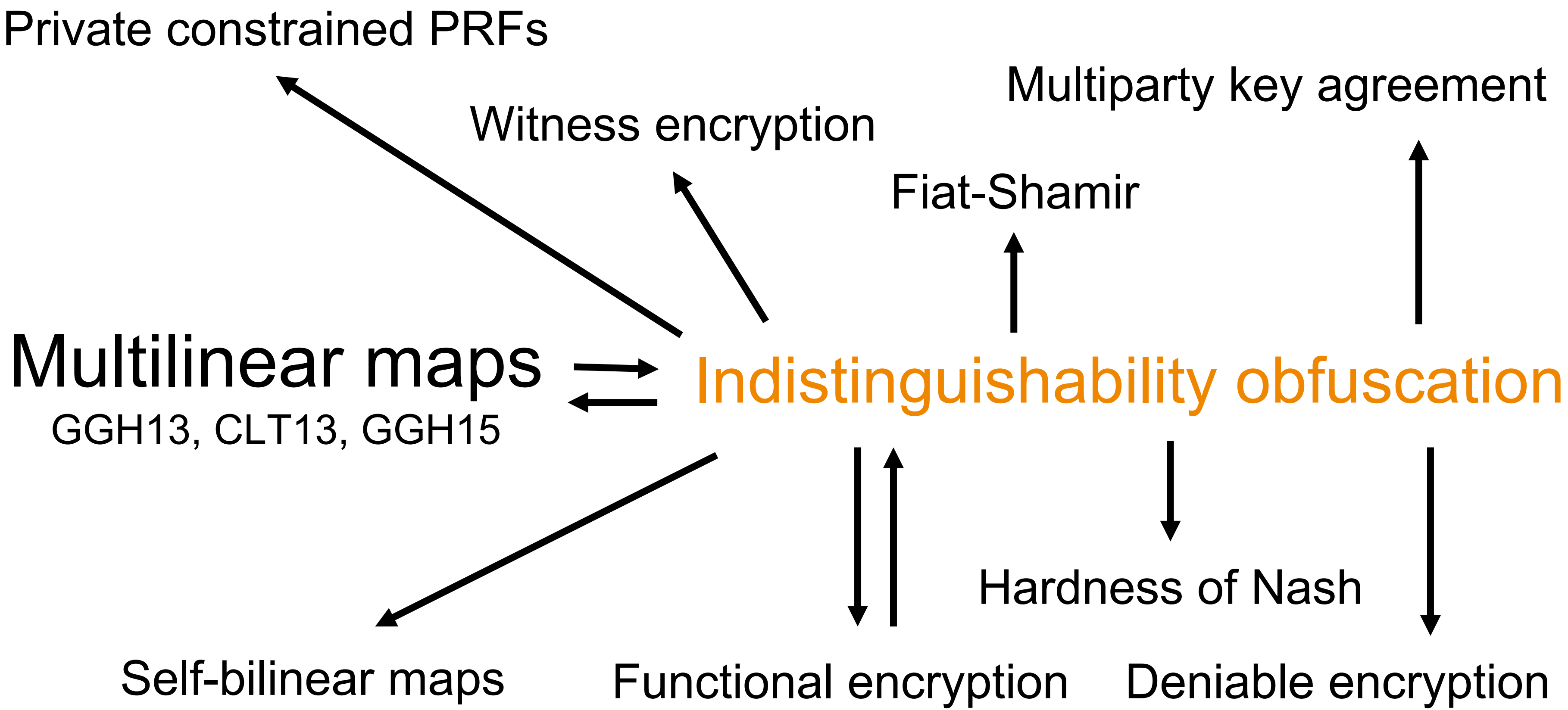
Correctness: Obf(P) preserves the functionality of P

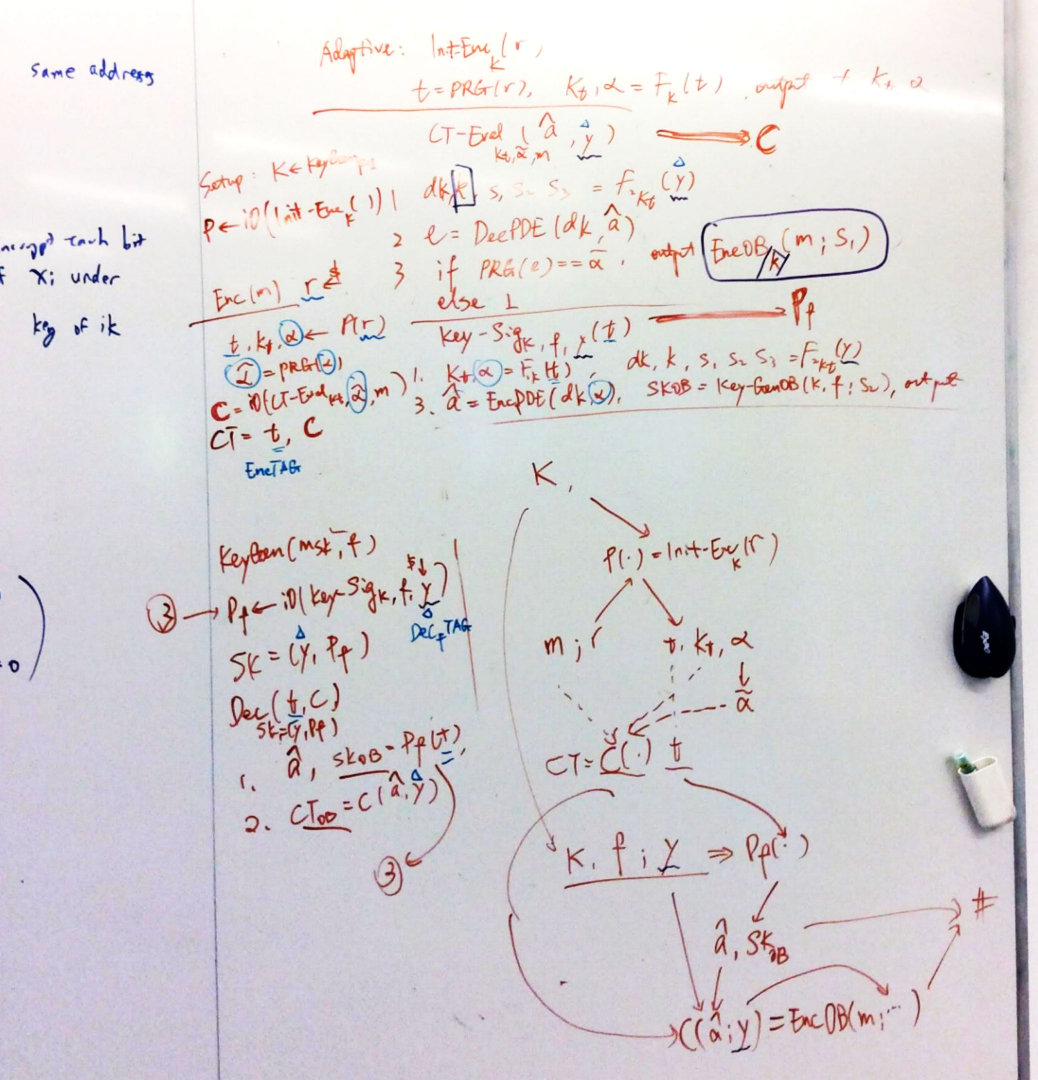
Security: For two programs P_0 and P_1 with identical functionality



$$iO[\ P_0\ ] \approx iO[\ P_1\ ]$$

The big bang in crypto





The big bang in crypto

Indistinguishability obfuscation

↓

Functional encryption
[Waters 14]

← The whiteboard on the 3rd floor of Simons Institute, in a sunny day in Summer 2015.

The big bang in crypto

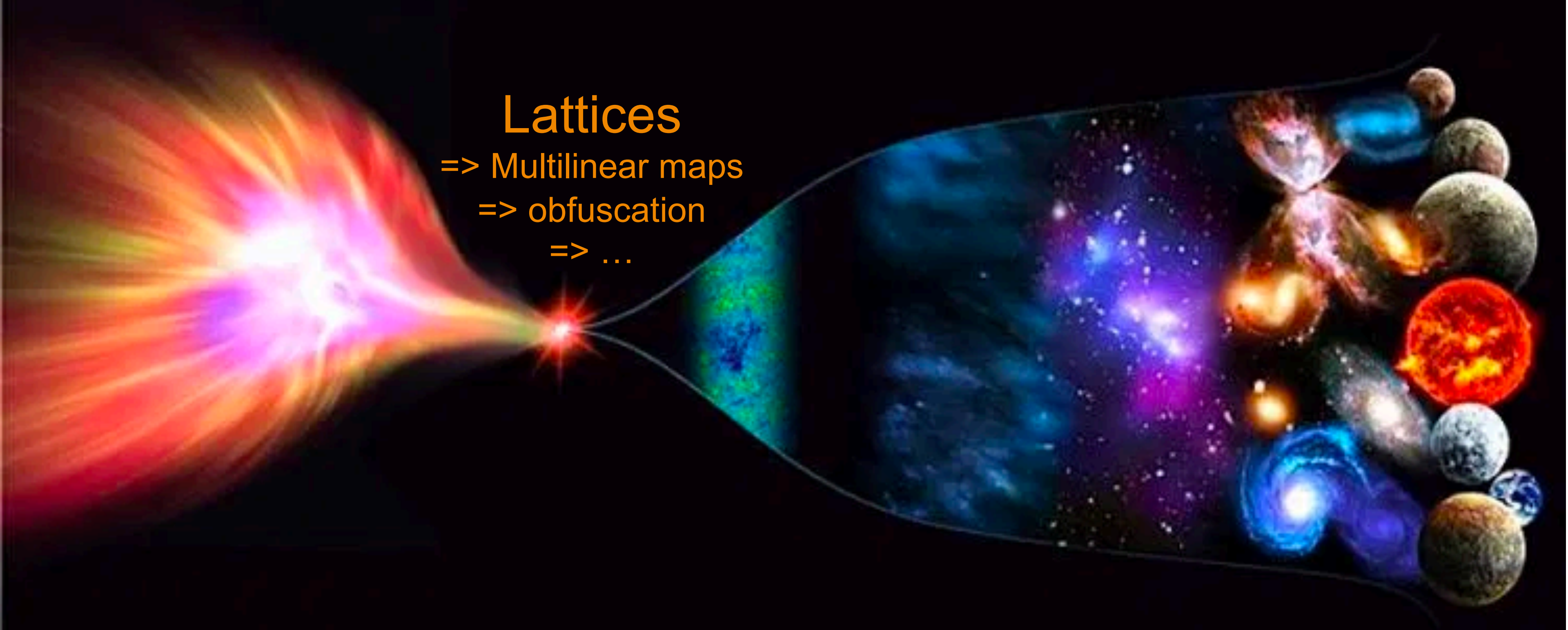
Self-bilinear maps ⟵ Indistinguishability obfuscation

Self-bilinear maps: $g^{S_1}, g^{S_2} \to g^{S_1 S_2}$

[Yamakawa, Yamada, Hanaoka, Kunihiro 14]: When the obfuscation is iO and N is an RSA modulus, the following idea works:

$$\text{Encoding}(S) = \{\ g^S \bmod N,\ \text{Obf}[\ f_S(x) = x^S \bmod N\ ]\ \}$$

The big bang in crypto



Lattices

=> Multilinear maps

=> obfuscation

=> ...

Where are we right now?

Multilinear maps & their friends

security overview

Private constrained PRFs

Witness encryption

Multiparty key agreement

Multilinear maps

GGH13, CLT13, GGH15

Indistinguishability obfuscation

Without multilinear maps

Lockable obfuscation

(Compute-then-Compare obf.)

Functional encryption

Deniable encryption

With a reduction from LWE (via safe use of GGH15); Candidates exists

Current status of multilinear maps and iO

https://malb.io/are-graded-encoding-schemes-broken-yet.html

https://sites.google.com/view/iostate-of-the-art/

Candidate constructions:
[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], …

Cryptanalyses:
[Cheon-Han-Lee-Ryu-Stehle ‘15], [Coron et al ‘15], [Miles-Sahai-Zhandry ‘16], …

48

🡐 Screenshot of my slides at DIMACS workshop in 2016, about delegating RAM computation from iO

SIMONS INSTITUTE for the Theory of Computing

LWE => iO = $100

Open Problem 1

Search

HOME ABOUT PEOPLE PROGRAMS & EVENTS VISITING SUPPORT WATCH / READ CALENDAR

‹ Home

Open Problems, Cryptography, Summer 2015

Below is a list of open problems proposed during the Cryptography program at the Simons Institute for the Theory of Computing, compiled by Ron Rothblum and Alessandra Scafuro. Each problem comes with a symbolic cash prize.

1. One-way permutations from a worst-case lattice assumption ($100 from Vinod Vaikuntanathan).
2. Non-interactive zero-knowledge (NIZK) proofs (or even arguments) for NP from LWE ($100 from Vinod Vaikuntanathan).
3. iO from LWE ($100 from Amit Sahai). This result would also solve problems (1) and (2). For (1) see construction and limitations and for (2) see argument system and proof system.
4. Interactive proofs for languages computable in DTISP(t,s) (time t and space s), where the prover runs in time poly(t) and the verifier runs in time poly(s). The provers in known proofs of IP = PSPACE run in time exponential in $2^{poly(s)}$ or $2^{O(s)}$ ($100 from Yael Kalai).
5. $20 per broken password challenge (from Jeremiah Blocki).
6. (Dis)prove that scrypt requires amortized (space × time) = $\Omega(n^2/polylog(n))$ per evaluation on a parallel machine ($100 from Joël).
7. A 3-linear map with unique encoding (i.e., without noise) for which "discrete log" is "plausibly hard" ($1000 from Dan Boneh).
8. SZK = PZK, or in other words, transform any statistical zero-knowledge proof (SZK) into a perfect zero-knowledge proof (PZK) ($100 from Shafi Goldwasser).

Update: During the talk, Amit raised the award to $1000.

Today: Lattice behind the big bang in crypto

Private constrained PRFs

Multilinear maps

GGH13, CLT13, GGH15

Indistinguishability obfuscation

Gentry, Gorbunov, Halevi (TCC 2015)

"Graph-induced multilinear maps from lattices"

Lockable obfuscation

(Compute-then-Compare obf.)

With a reduction from LWE (via safe use of GGH15); Candidates exists

Today: Lattice behind the big bang in crypto

Private constrained PRFs

Multilinear maps

GGH13, CLT13, GGH15

Indistinguishability obfuscation

- Multilinear maps with security based on LWE
- A new methodology of building lattice applications after "[GSW13]" and "[BGG+14]"

Lockable obfuscation
(Compute-then-Compare obf.)

With a reduction from LWE (via safe use of GGH15); Candidates exists

Plan of today:

~~1. Introduction~~
2. GGH15: functionality and security overview
3. Applications: Obfuscators & Private constrained PRFs

Open problems will be mentioned during the talk

Gentry, Gorbunov, Halevi (TCC 2015)

"Graph-induced multilinear maps from lattices"

Concerto in D minor (BWV 1052)

The arithmetic operations are just matrix operations in $\mathbb{Z}_q^{m\times m}$:

$$\mathsf{neg}(\mathsf{pp},\mathbf{D}) := -\mathbf{D}, \quad \mathsf{add}(\mathsf{pp},\mathbf{D},\mathbf{D}') := \mathbf{D}+\mathbf{D}', \quad \text{and} \quad \mathsf{mult}(\mathsf{pp},\mathbf{D},\mathbf{D}') := \mathbf{D}\cdot\mathbf{D}'.$$

To see that negation and addition maintain the right structure, let $\mathbf{D},\mathbf{D}' \in \mathbb{Z}_q^{m\times m}$ be encodings reltive to the same path $u \rightsquigarrow v$. Namely $\mathbf{D}\cdot\mathbf{A}_u = \mathbf{A}_v\cdot\mathbf{S}+\mathbf{E}$ and $\mathbf{D}'\cdot\mathbf{A}_u = \mathbf{A}_v\cdot\mathbf{S}'$ with the matrices $\mathbf{D},\mathbf{D}',\mathbf{E},\mathbf{E}',\mathbf{S},\mathbf{S}'$ all small. Then we have

$$\begin{aligned} -\mathbf{D}\cdot\mathbf{A}_u &= \mathbf{A}_v\cdot(-\mathbf{S})+(-\mathbf{E}), \\ \text{and } (\mathbf{D}+\mathbf{D}')\cdot\mathbf{A}_u &= (\mathbf{A}_v\cdot\mathbf{S}+\mathbf{E})+(\mathbf{A}_v\cdot\mathbf{S}'+\mathbf{E}') \;=\; \mathbf{A}_v\cdot(\mathbf{S}+\mathbf{S}')+(\mathbf{E}+\mathbf{E}'), \end{aligned}$$

and all the matrices $-\mathbf{D}, -\mathbf{S}, -\mathbf{E},\ \mathbf{D}+\mathbf{D}',\ \mathbf{S}+\mathbf{S}',\ \mathbf{E}+\mathbf{E}'$ are still small. For multiplica consider encodings $\mathbf{D},\mathbf{D}'$ relative to paths $v \rightsquigarrow w$ and $u \rightsquigarrow v$, respectively, then we have

$$\begin{aligned} (\mathbf{D}\cdot\mathbf{D}')\cdot\mathbf{A}_u &= \mathbf{D}\cdot\left(\mathbf{A}_v\cdot\mathbf{S}'+\mathbf{E}'\right) \\ &= \left(\mathbf{A}_w\cdot\mathbf{S}+\mathbf{E}\right)\cdot\mathbf{S}'+\mathbf{D}\cdot\mathbf{E}' \;=\; \mathbf{A}_w\cdot(\mathbf{S}\cdot\mathbf{S}')+\underbrace{(\mathbf{E}\cdot\mathbf{S}'+\mathbf{D}\cdot\mathbf{E}')}_{\mathbf{E}''}, \end{aligned}$$

and the matrices $\mathbf{D}\cdot\mathbf{D}'$, $\mathbf{S}\cdot\mathbf{S}'$, and $\mathbf{E}''$ are still small.

The development of GGH15-like applications: 2015 - 2017

[Gentry, Gorbunov, Halevi 15]: functionality, cryptanalytic attempts, candidate N-party key-exchange and iO.

[Brakerski, Vaikuntanathan, Wee, Wichs 16]: First proof methodology => obfuscating conjunctions

[Coron, Lee, Lepoint, Tibouchi 16]: breaking the candidate N-party key exchange
[Chen, Gentry, Halevi 17]: breaking iO for some parameters

[Canetti, Chen 17]: Private Constrained PRF from LWE
[Goyal, Koppula, Waters 17a]: Circular security counterexample from LWE
[Goyal, Koppula, Waters 17b], [Wichs, Zirdelis 17]: Lockable obfuscation, compute & compare obfuscation from LWE

[GGH15] Via a different view of the FHE scheme of Gentry, Sahai, Waters 13

Different *motives* / views of GGH15

- The arithmetic operations are just matrix operations in $\mathbb{Z}_q^{m\times m}$:

$$\mathsf{neg}(\mathsf{pp},\mathbf{D}) := -\mathbf{D}, \quad \mathsf{add}(\mathsf{pp},\mathbf{D},\mathbf{D}') := \mathbf{D}+\mathbf{D}', \quad \text{and} \quad \mathsf{mult}(\mathsf{pp},\mathbf{D},\mathbf{D}') := \mathbf{D}\cdot\mathbf{D}'.$$

To see that negation and addition maintain the right structure, let $\mathbf{D},\mathbf{D}' \in \mathbb{Z}_q^{m\times m}$ be two encodings reltive to the same path $u \rightsquigarrow v$. Namely $\mathbf{D}\cdot\mathbf{A}_u = \mathbf{A}_v\cdot\mathbf{S}+\mathbf{E}$ and $\mathbf{D}'\cdot\mathbf{A}_u = \mathbf{A}_v\cdot\mathbf{S}'+\mathbf{E}'$, with the matrices $\mathbf{D},\mathbf{D}',\mathbf{E},\mathbf{E}',\mathbf{S},\mathbf{S}'$ all small. Then we have

$$\begin{aligned} -\mathbf{D}\cdot\mathbf{A}_u &= \mathbf{A}_v\cdot(-\mathbf{S})+(-\mathbf{E}), \\ \text{and } (\mathbf{D}+\mathbf{D}')\cdot\mathbf{A}_u &= (\mathbf{A}_v\cdot\mathbf{S}+\mathbf{E})+(\mathbf{A}_v\cdot\mathbf{S}'+\mathbf{E}') = \mathbf{A}_v\cdot(\mathbf{S}+\mathbf{S}')+(\mathbf{E}+\mathbf{E}'), \end{aligned}$$

and all the matrices $-\mathbf{D},-\mathbf{S},-\mathbf{E},\ \mathbf{D}+\mathbf{D}',\ \mathbf{S}+\mathbf{S}',\ \mathbf{E}+\mathbf{E}'$ are still small. For multiplication, consider encodings $\mathbf{D},\mathbf{D}'$ relative to paths $v \rightsquigarrow w$ and $u \rightsquigarrow v$, respectively, then we have

$$\begin{aligned} (\mathbf{D}\cdot\mathbf{D}')\cdot\mathbf{A}_u &= \mathbf{D}\cdot\big(\mathbf{A}_v\cdot\mathbf{S}'+\mathbf{E}'\big) \\ &= \big(\mathbf{A}_w\cdot\mathbf{S}+\mathbf{E}\big)\cdot\mathbf{S}'+\mathbf{D}\cdot\mathbf{E}' = \mathbf{A}_w\cdot(\mathbf{S}\cdot\mathbf{S}')+\underbrace{(\mathbf{E}\cdot\mathbf{S}'+\mathbf{D}\cdot\mathbf{E}')}_{\mathbf{E}''}, \end{aligned}$$

and the matrices $\mathbf{D}\cdot\mathbf{D}'$, $\mathbf{S}\cdot\mathbf{S}'$, and $\mathbf{E}''$ are still small.

Of course, the matrices $\mathbf{D},\mathbf{S},\mathbf{E}$ all grow with arithmetic operations, but our parameter-choice enures that for any encoding relative to any path in the graph $u \rightsquigarrow v$ (of length $\le d$) we have $\mathbf{D}\cdot\mathbf{A}_u = \mathbf{A}_v\cdot\mathbf{S}+\mathbf{E}$ where $\mathbf{E}$ is still small, specifically $\|\mathbf{E}\| < q^{3/4} \le q/2^{t+1}$.

- $\mathsf{ZeroTest}(\mathsf{pp},\mathbf{D})$. Given an encoding $\mathbf{D}$ relative to path $u \rightsquigarrow v$ and the matrix $\mathbf{A}_u$, our zero-test procedure outputs 1 if and only if $\|\mathbf{D}\cdot\mathbf{A}_u\| < q/2^{t+1}$.

[Alamati, Peikert 16],
[Koppula, Waters 16],
[Goyal, Koppula, Waters 17]
"cascaded products" or
"telescoping cancelation",
motivated by showing circular security counterexamples.

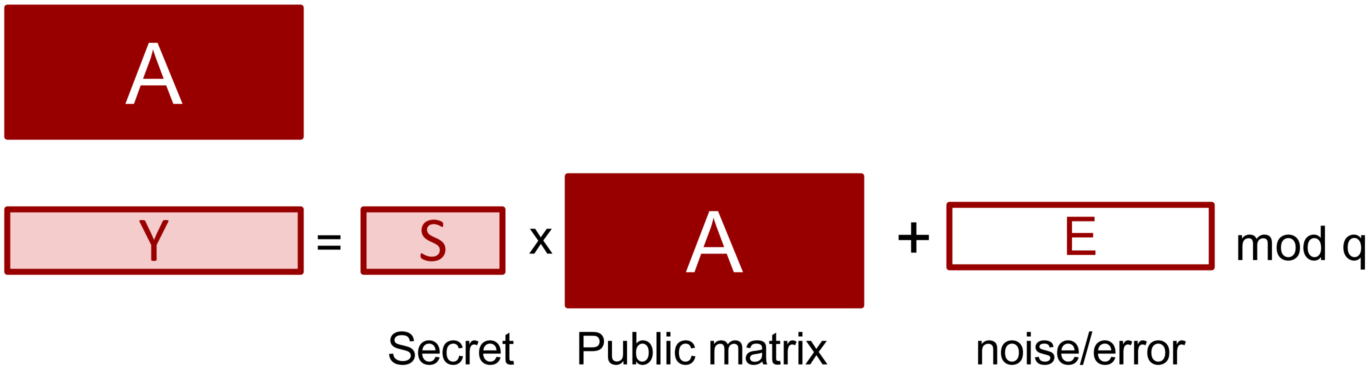
[Canetti, Chen 17]
GGH15 captures two lattice-based PRFs

[Chen, Vaikuntanathan, Wee 18]
A generalization of Kilian randomization

Today: chaining LWE samples

Recall Learning with Errors
[Regev 05]

Uniform Small Unspecified



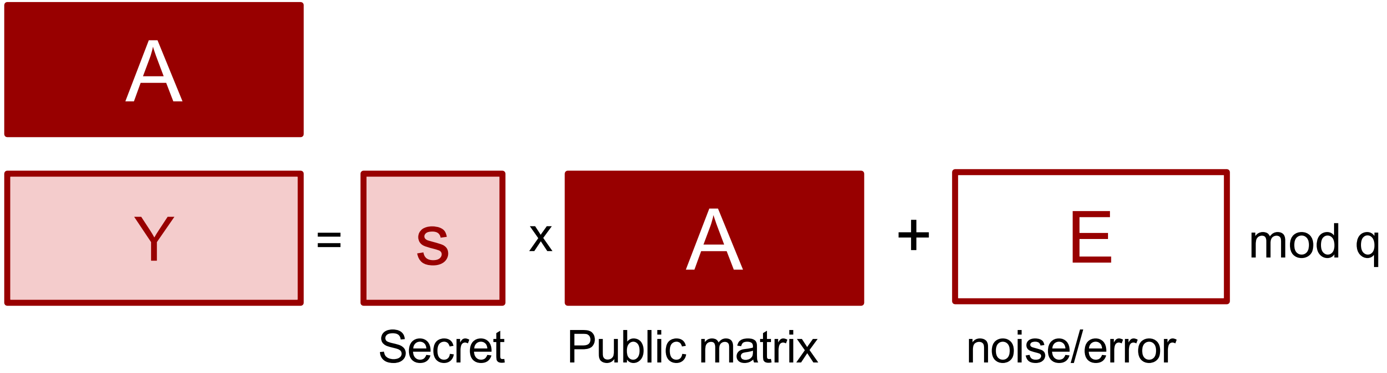
$A \in Z_q^{n \times m}$ (m > n log q)

Search LWE: Given $A, Y = SA + E$, find S.

Decisional LWE: Given A, distinguish Y from random.

Recall Learning with Errors [Regev 05]

Uniform Small Unspecified



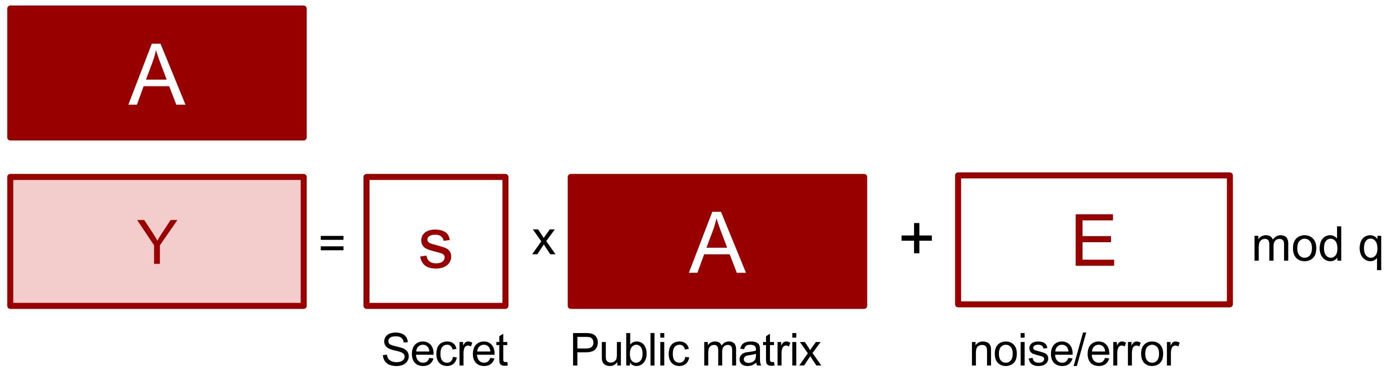
$A \in Z_q^{n \times m}$ (m > n log q)

Search LWE: Given $A, Y = SA + E$, find S.

Decisional LWE: Given A, distinguish Y from random.

Recall Learning with Errors
[Regev 05]

Uniform Small Unspecified



Entries of S from the error distribution
As hard as normal LWE [Applebaum, Cash, Peikert, Sahai 09]

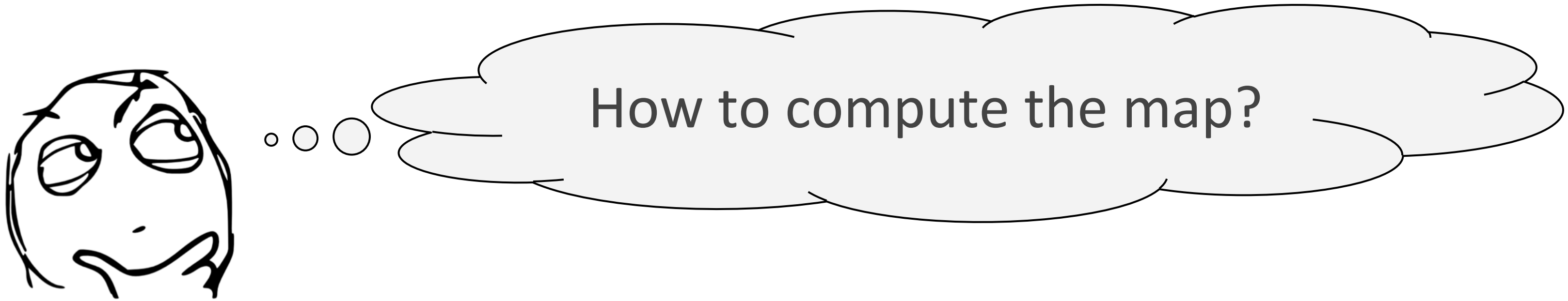
GGH15 in a nutshell

> Multilinear maps: motivated in [Boneh, Silverberg 2003]

$$g,\ g^{S_1},\ g^{S_2},\ g^{S_3},\ \ldots \rightarrow g_T^{\prod S}$$

> (Ring)LWE analogy:

$$A,\ S_1A+E_1, \ldots,\ S_kA+E_k \rightarrow \prod S A+E \quad \text{mod } q$$



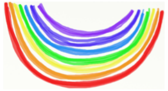
GGH15 in a nutshell

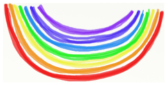
> Multilinear maps: motivated in [Boneh, Silverberg 2003]

$$g,\ g^{S_1},\ g^{S_2},\ g^{S_3},\ \ldots \rightarrow g_T^{\prod S}$$

> (Ring)LWE analogy:

$$A,\ S_1A+E_1, \ldots, S_kA+E_k \rightarrow \prod S A+E \mod q$$





Idea: using lattice trapdoor sampling to chain them together

Recall lattice trapdoor

[Ajtai 99], [Alwen, Peikert 09], [Micciancio, Peikert 12]

The trapdoor for A can be used to solve SIS and LWE.

Given an image Y , find a short vector D s.t.

$$A \times D = Y \bmod q$$

Lattice trapdoor
[Ajtai 99]

T is short and full rank in Z

A x T = 0 mod q

GGH15 in a nutshell

> (Ring)LWE analogy:

$$A,\ S_1A+E_1, \ldots, S_kA+E_k \rightarrow \prod SA+E \quad \text{mod } q$$

> GGH15:

A_0 $\quad S_1A_1+E_1,\quad A_1 \quad S_2A_2+E_2$

GGH15 in a nutshell

> (Ring)LWE analogy:

$$A,\ S_1A+E_1, \ldots, S_kA+E_k \rightarrow \prod SA+E \quad \text{mod } q$$

> GGH15:

$$A_0\, D_1 = S_1A_1+E_1, \quad A_1\, D_2 = S_2A_2+E_2 \quad \text{mod } q$$

D_i is sampled using the trapdoor of A_{i-1}

GGH15 in a nutshell

> (Ring)LWE analogy:

$$A,\ S_1A+E_1, \ldots,\ S_kA+E_k \rightarrow \prod SA+E \quad \bmod q$$

> GGH15:

$$A_0\, D_1 = S_1A_1+E_1, \quad A_1\, D_2 = S_2A_2+E_2 \quad \bmod q$$

D_i is sampled using the trapdoor of A_{i-1}

A_0 D_1 = $S_1A_1+E_1$ mod q

A_1 D_2 = $S_2A_2+E_2$ mod q

$$A_0 D_1 = S_1 A_1 + E_1, \quad A_1 D_2 = S_2 A_2 + E_2 \mod q$$

$$A_{i-1} \; D_i = s_i \times A_i + E_i \mod q$$

D_i is sampled using the trapdoor of A_{i-1}

GGH15
in a nutshell

> (Ring)LWE analogy:

$$A,\ S_1A+E_1,\ldots,\ S_kA+E_k \rightarrow \prod S A+E \mod q$$

> GGH15:

$$A_0\,D_1 = S_1A_1+E_1,\quad A_1\,D_2 = S_2A_2+E_2 \mod q$$

Publish A_0 , D_1 , D_2 as the encodings of S_1 , S_2

GGH15 in a nutshell

> (Ring)LWE analogy:

$$A,\ S_1A+E_1, \ldots, S_kA+E_k \rightarrow \prod SA+E \mod q$$

> GGH15:

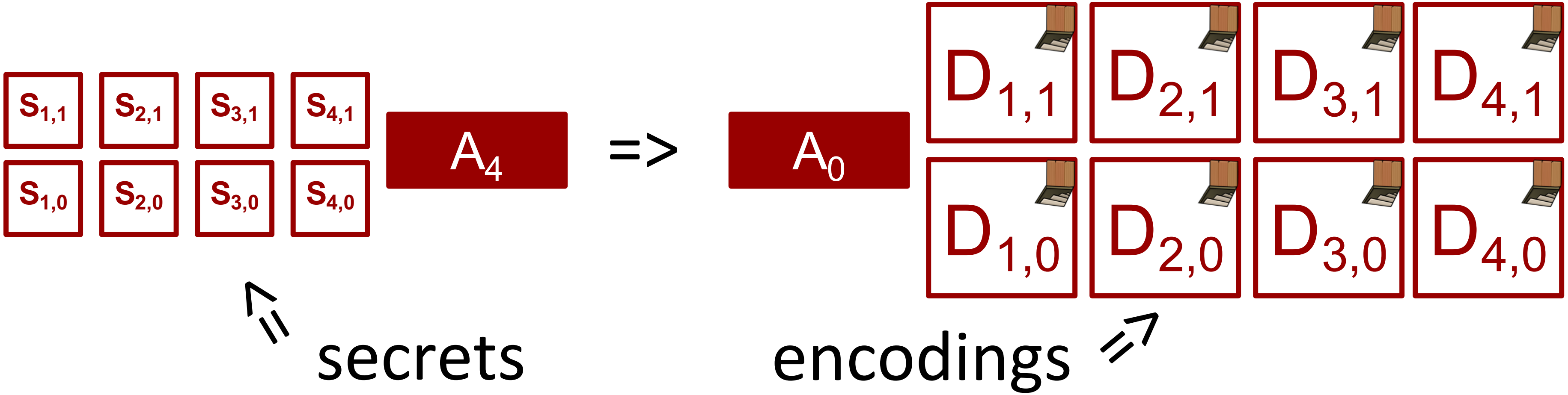
$$A_0\, D_1 = S_1A_1+E_1, \quad A_1\, D_2 = S_2A_2+E_2 \quad \mod q$$

Publish A_0 , D_1 , D_2 as the encodings of S_1 , S_2

$$A_0D_1D_2 = (S_1A_1+E_1)D_2 = S_1A_1D_2+E_1D_2$$

$$= S_1(S_2A_2+E_2)+E_1D_2 = \underbrace{S_1S_2A_2}_{\text{functionality}} + \underbrace{S_1E_2 + E_1D_2}_{\text{small error}}$$

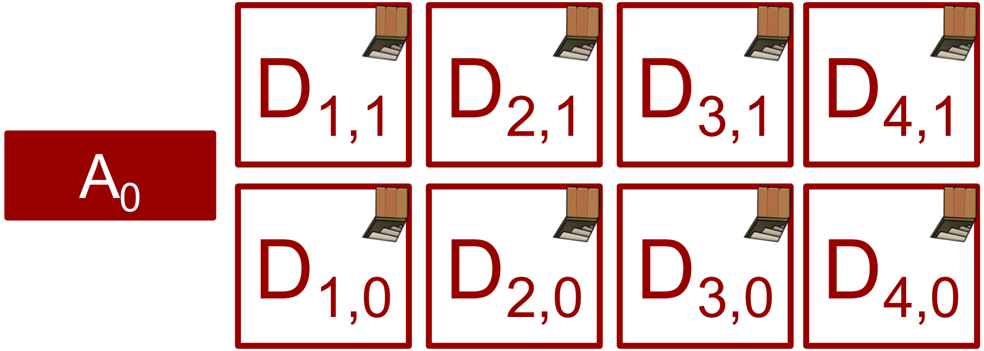
A typical evaluation pattern for GGH15: subset product



via

$$A_{i-1} \times D_{i,b} = S_{i,b} \times A_i + E_{i,b} \mod q$$

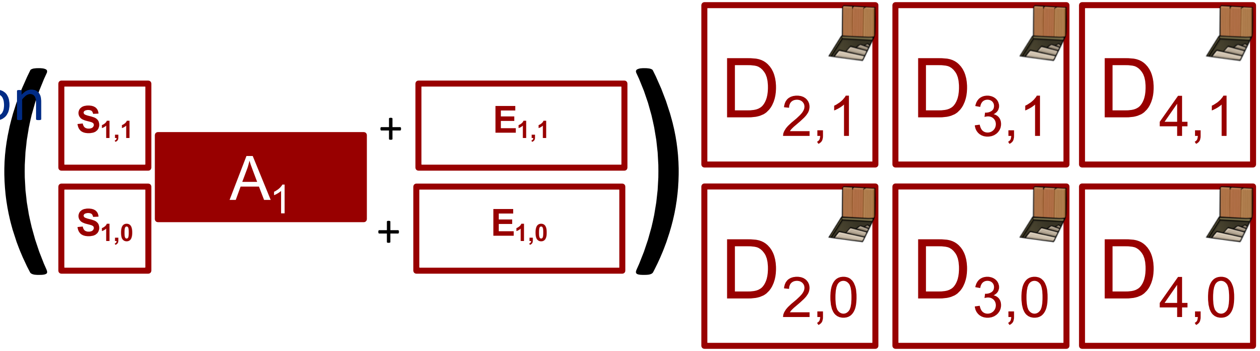
Subset product evaluation



$$\text{Eval}(0110) = A_0 D_{1,0} D_{2,1} D_{3,1} D_{4,0}$$

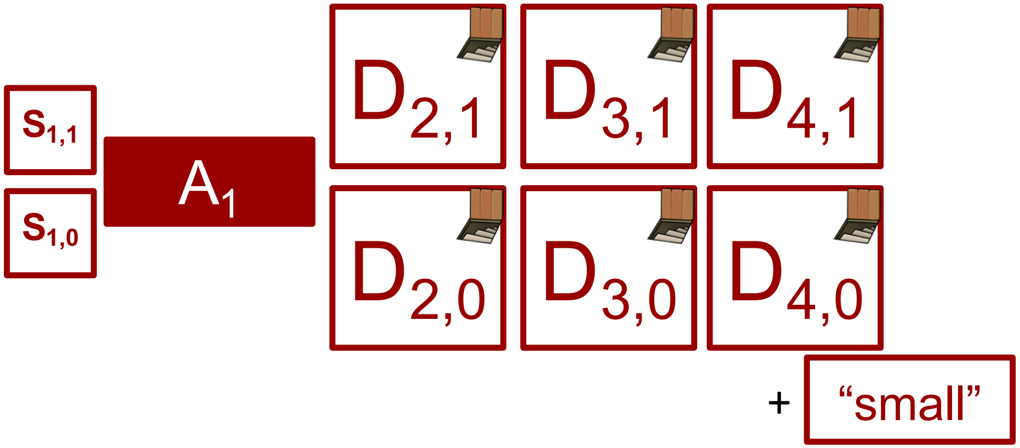
<= The input is a bit string that selects which $D_{i,b}$ to multiply

Subset product evaluation



$$\mathrm{Eval}(0110)$$
$$= A_0 D_{1,0} D_{2,1} D_{3,1} D_{4,0}$$
$$= (s_{1,0} A_1 + E_{1,0}) D_{2,1} D_{3,1} D_{4,0}$$

Subset product evaluation



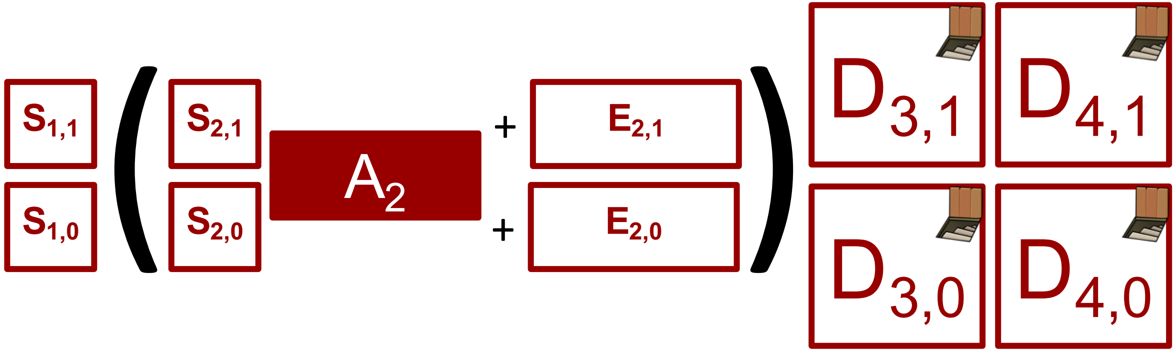
Eval(0110)

$= A_0 D_{1,0} D_{2,1} D_{3,1} D_{4,0}$

$= (s_{1,0} A_1 + E_{1,0}) D_{2,1} D_{3,1} D_{4,0}$

$= s_{1,0} A_1 D_{2,1} D_{3,1} D_{4,0}$ + "small"

Subset product evaluation



Eval(0110)

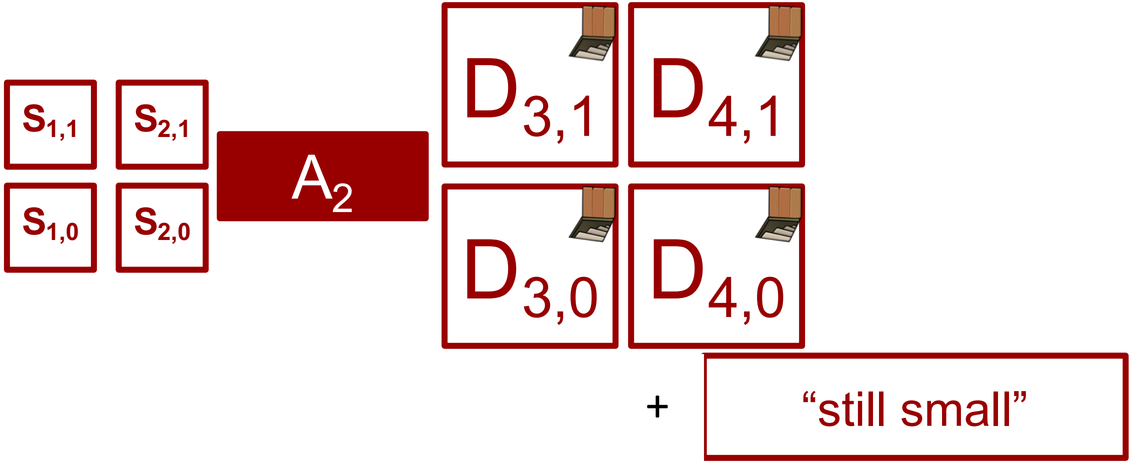
$= A_0 D_{1,0} D_{2,1} D_{3,1} D_{4,0}$

$= (s_{1,0} A_1 + E_{1,0}) D_{2,1} D_{3,1} D_{4,0}$

$= s_{1,0} A_1 D_{2,1} D_{3,1} D_{4,0}$ + “small”

$= s_{1,0} (s_{2,1} A_2 + E_{2,1}) D_{3,1} D_{4,0}$ + “small”

Subset product evaluation



Eval(0110)

$= A_0 D_{1,0} D_{2,1} D_{3,1} D_{4,0}$

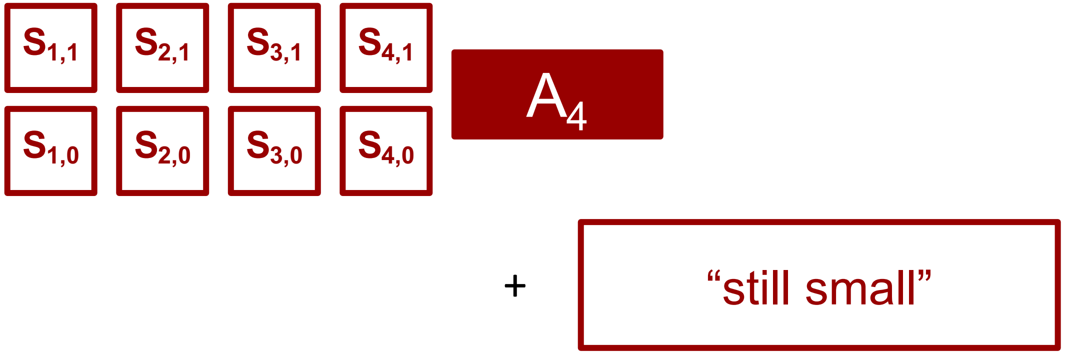
$= (s_{1,0} A_1 + E_{1,0}) D_{2,1} D_{3,1} D_{4,0}$

$= s_{1,0} A_1 D_{2,1} D_{3,1} D_{4,0}$ + “small”

$= s_{1,0}(s_{2,1} A_2 + E_{2,1}) D_{3,1} D_{4,0}$ + “small”

$= s_{1,0} s_{2,1} A_2 D_{3,1} D_{4,0}$ + “still small”

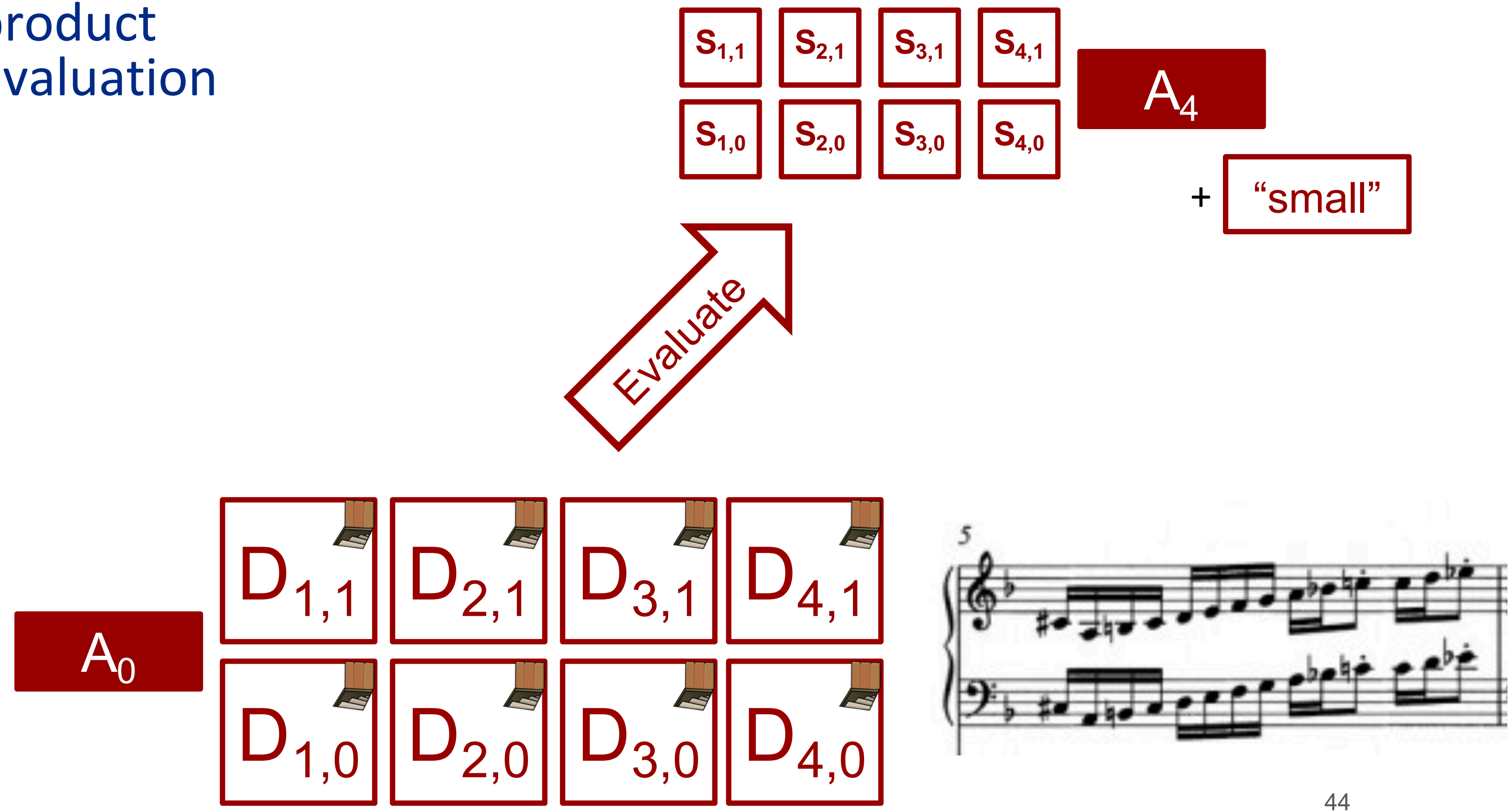
Subset product evaluation



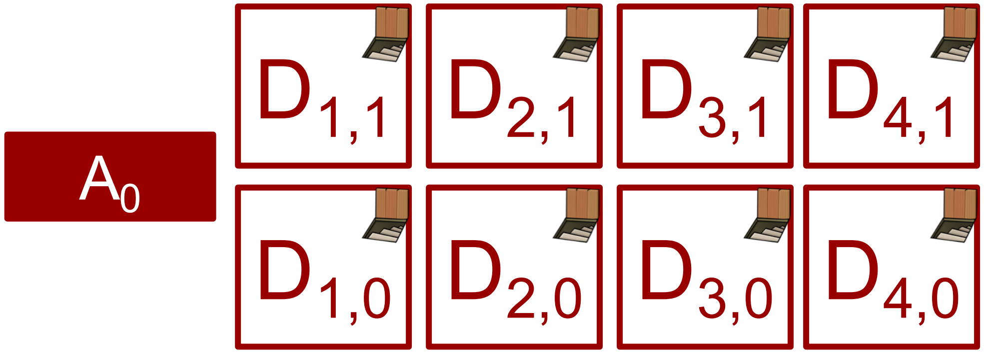
$$
\begin{aligned}
&\text{Eval}(0110) \\
&= A_0 D_{1,0} D_{2,1} D_{3,1} D_{4,0} \\
&= (s_{1,0}A_1 + E_{1,0}) D_{2,1} D_{3,1} D_{4,0} \\
&= s_{1,0} A_1 D_{2,1} D_{3,1} D_{4,0} + \text{“small”} \\
&= s_{1,0}(s_{2,1}A_2 + E_{2,1}) D_{3,1} D_{4,0} + \text{“small”} \\
&= s_{1,0} s_{2,1} A_2 D_{3,1} D_{4,0} + \text{“still small”} \\
&= s_{1,0} s_{2,1} s_{3,1} A_3 D_{4,0} + \text{“still smallish”} \\
&= s_{1,0} s_{2,1} s_{3,1} s_{4,0} A_4 + \text{“small”}
\end{aligned}
$$

The "small" noise grows exponentially with #levels, becomes a problem in the efficiency.

Subset product evaluation



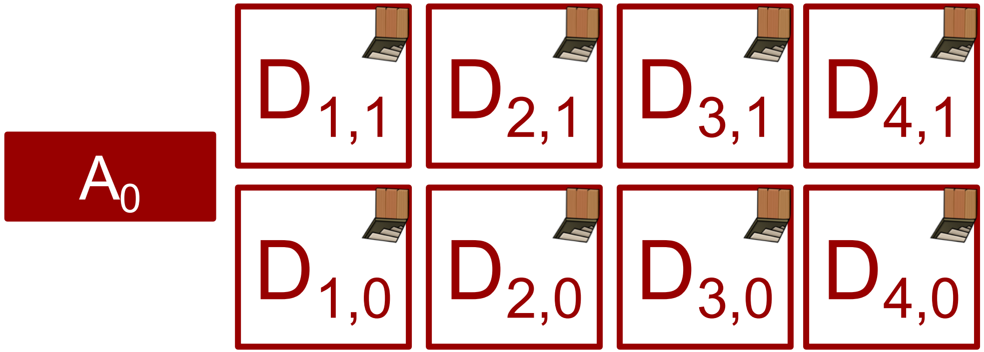
Functionality



$A_0, S_1A_1+E_1, \ldots, S_kA_k+E_k \rightarrow \prod S A_k + E \mod q$

Functionality: evaluate and test whether $\prod S$ is zero or not.

(Designing GGH15 applications: put structures in $S_{i,b}$)



Functionality and Security

$A_0, S_1A_1+E_1, \ldots, S_kA_k+E_k \rightarrow \prod S A_k+E \mod q$

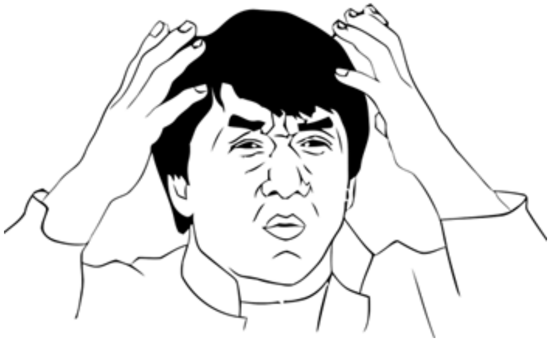
Functionality: evaluate and test whether $\prod S$ is zero or not. (Designing GGH15 applications: put structures in $S_{i,b}$)

Security (goal): hides $S_{i,b}$ for all i, b. But the reality is …

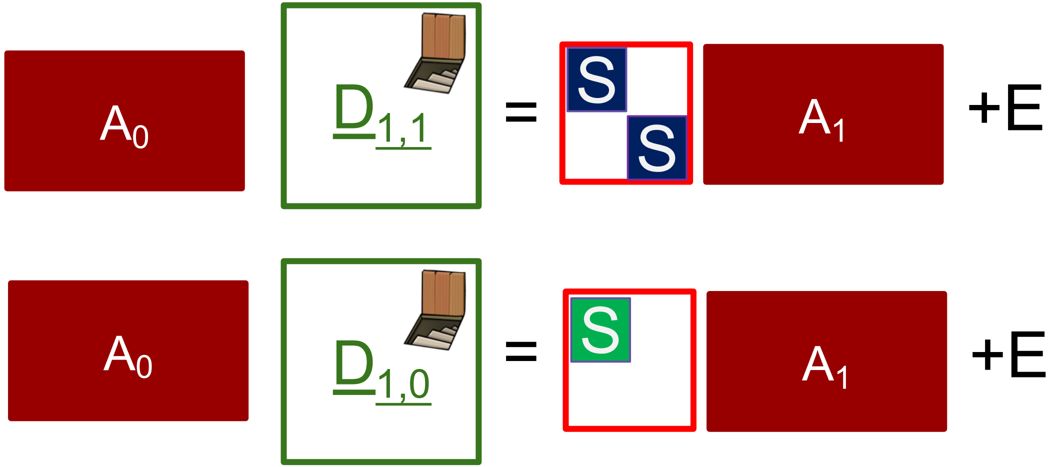
complicated, depends on the structure inside $S_{i,b}$

Security (goal): hides $S_{i,b}$ for all i, b. But the reality is …

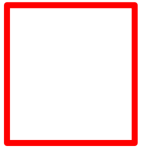
What does “structure” in $S_{i,b}$ look like?



Toy example 1



Each $S_{i,\,b} = M_{i,\,b} \otimes s_{i,\,b}$



= 0, else = 1

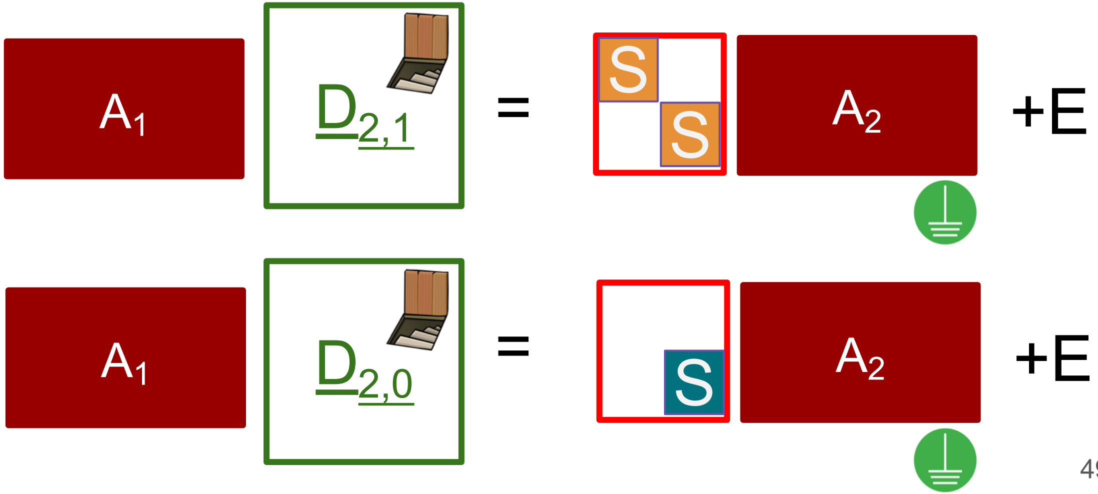
$\prod S A_2 + E$

F(00) = 0

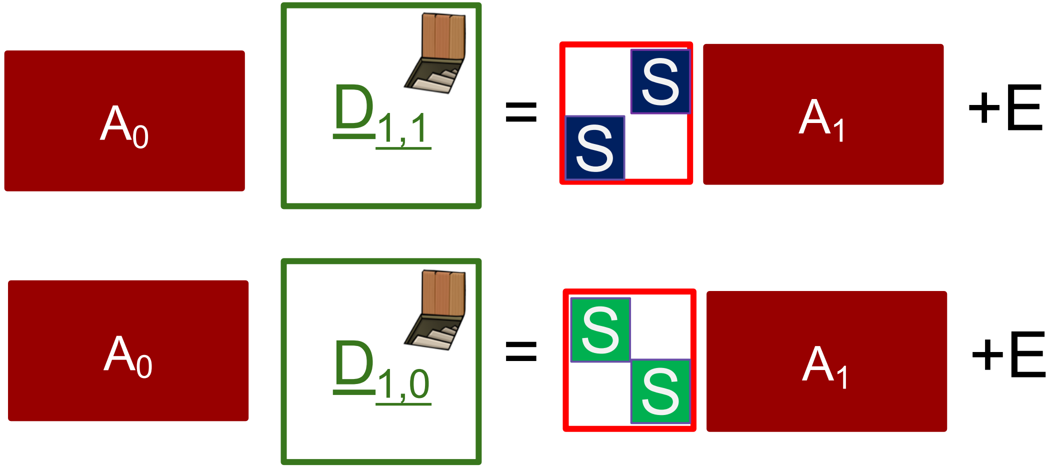
F(01) = 1

F(10) = 1

F(11) = 1



Toy example 2



Claim: this construction hides all the structures in the S matrices.

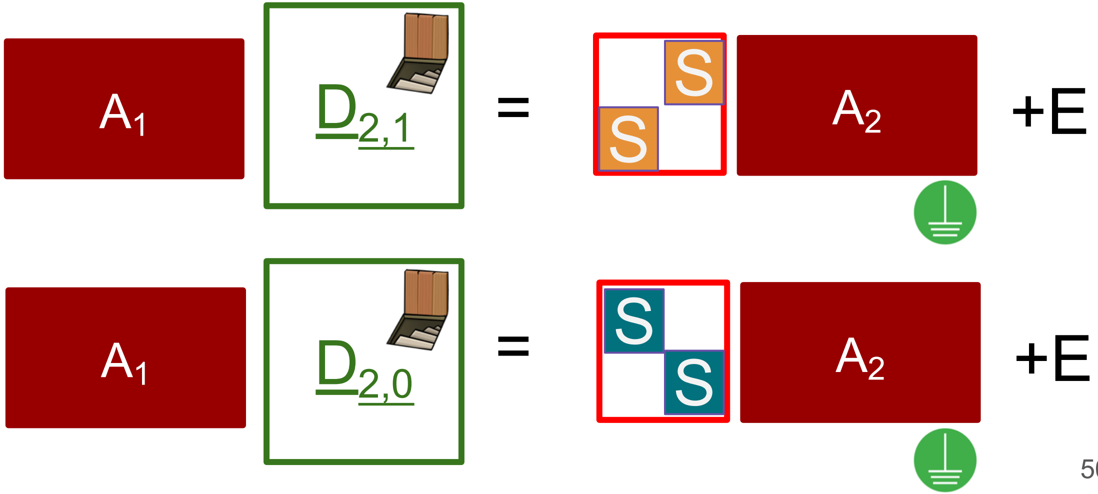
$\prod S A_2 + E$

F(00) = 1

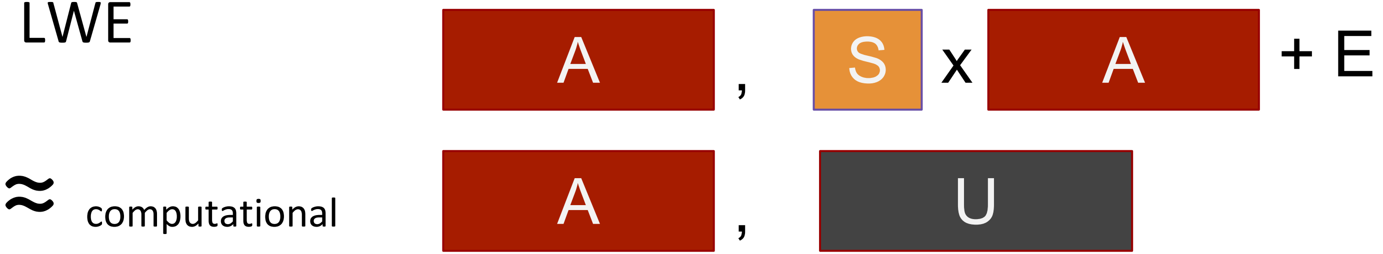
F(01) = 1

F(10) = 1

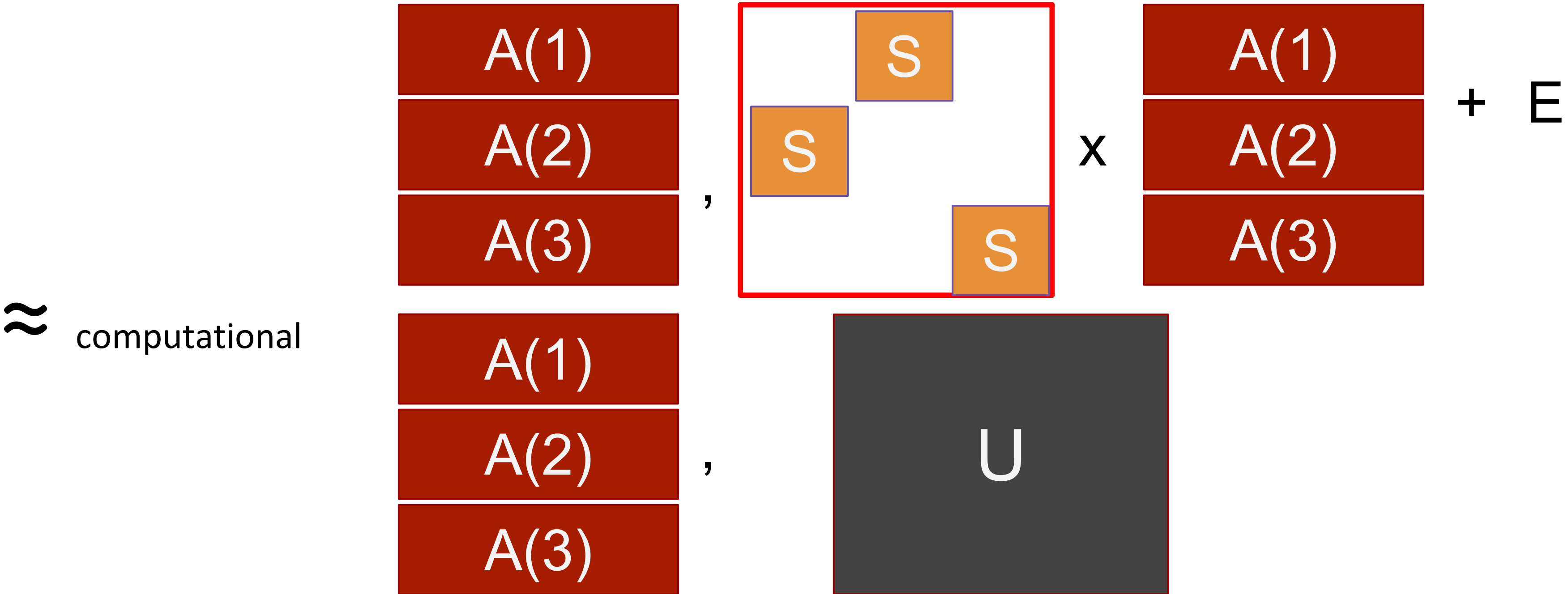
F(11) = 1



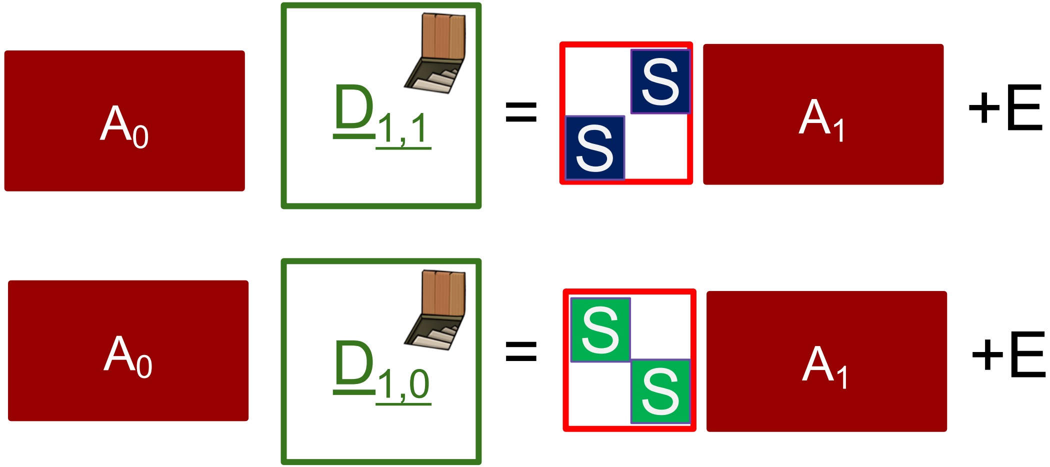
Recall decisional LWE



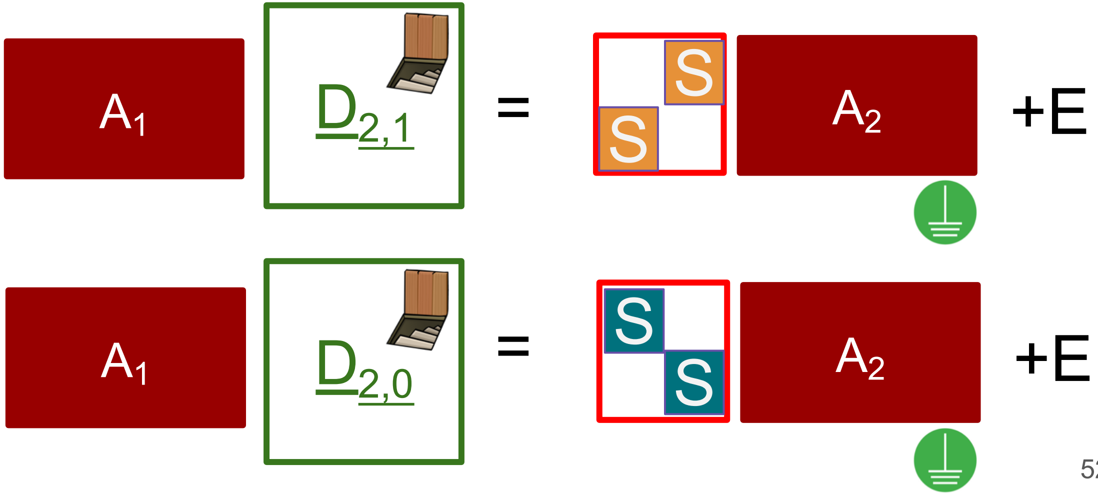
Permutation - LWE:



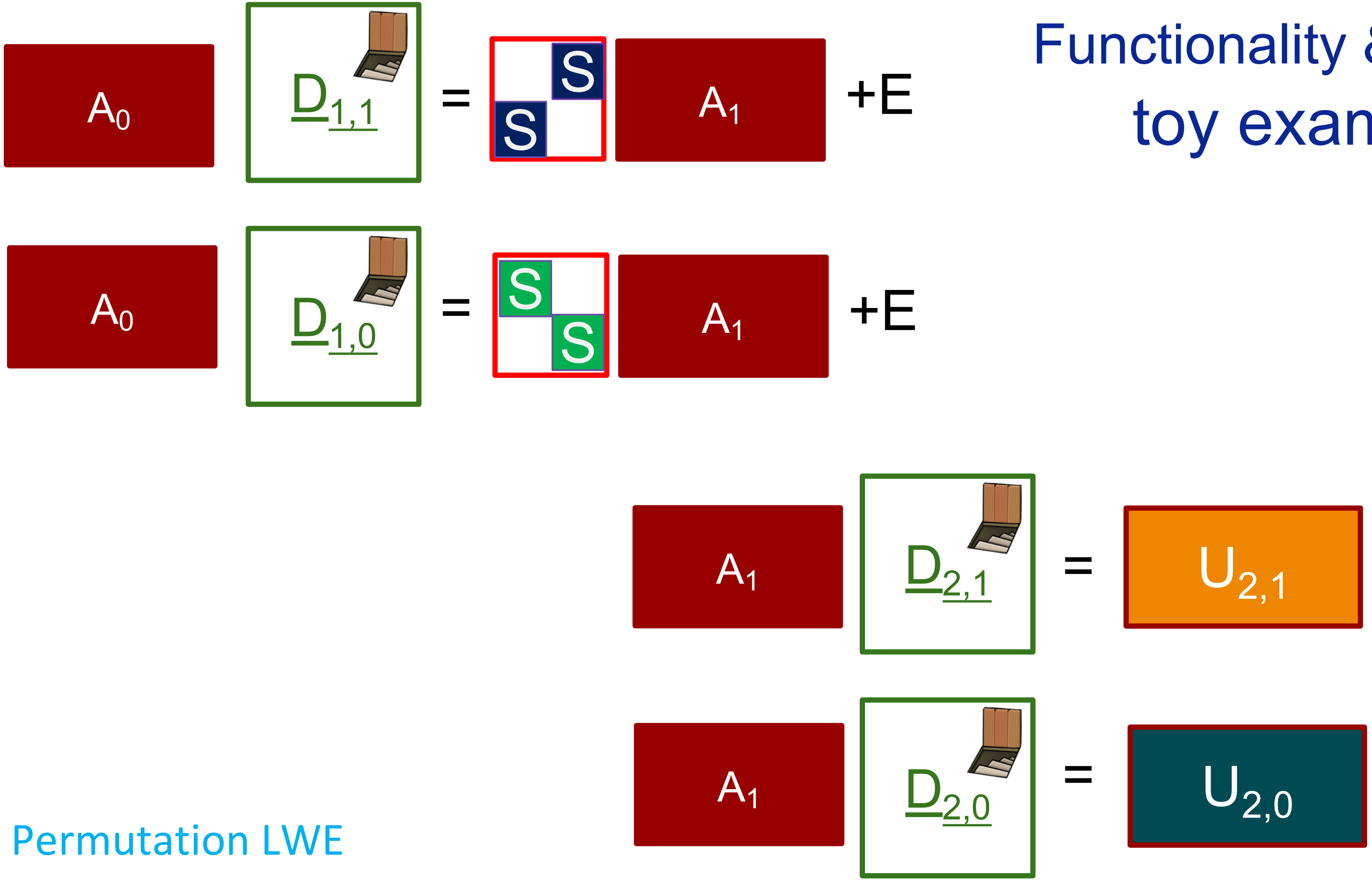
Functionality & Security toy examples



Claim: this construction hides all the structures in the S matrices.



Functionality & Security toy examples



Permutation LWE

Preimage sampling

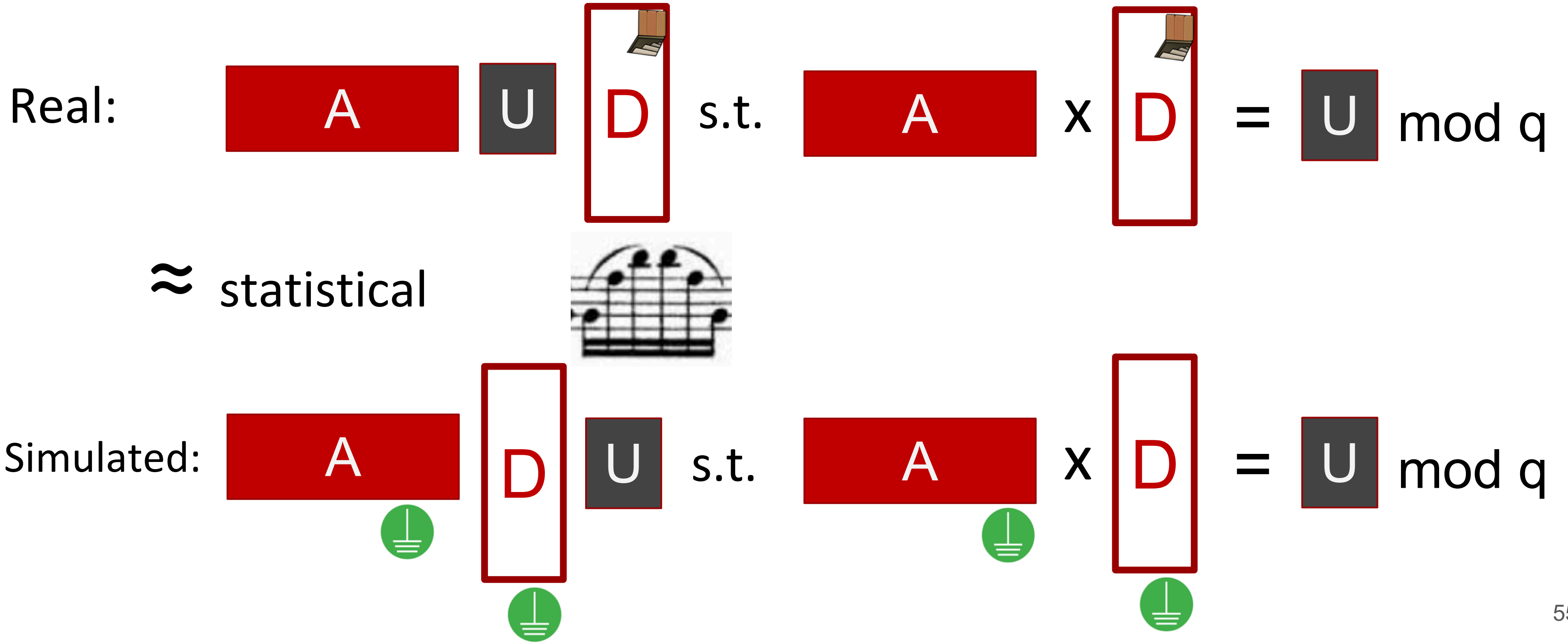
[Gentry, Peikert, Vaikuntanathan 08]

For random images, there is a way to sample the preimage without revealing the trapdoor.

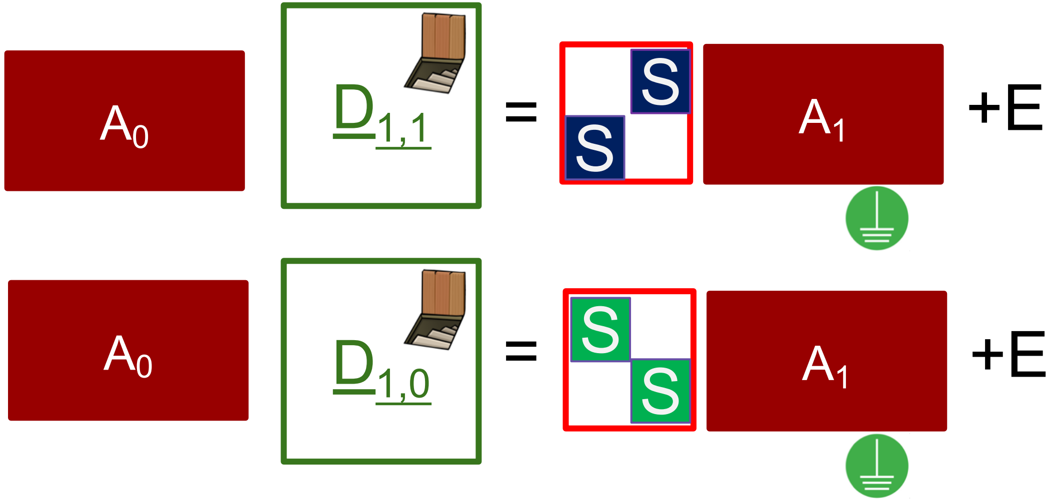
Preimage sampling

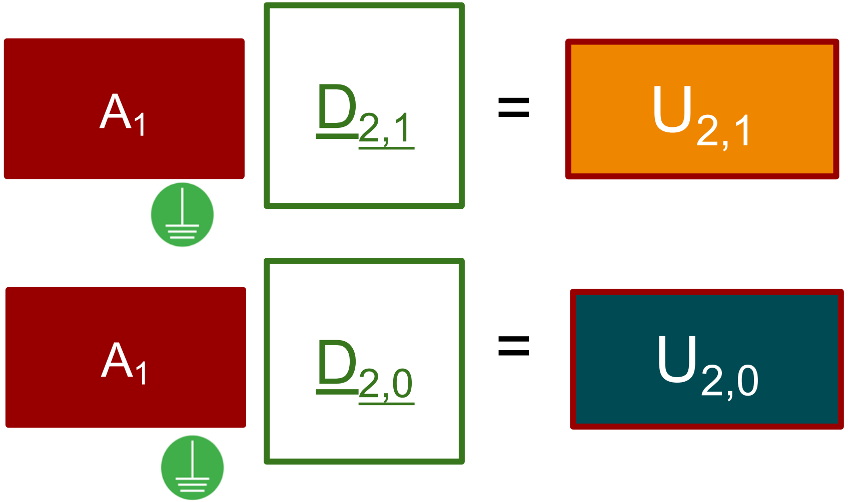
[Gentry, Peikert, Vaikuntanathan 08]

For random images, there is a way to sample the preimage without revealing the trapdoor.



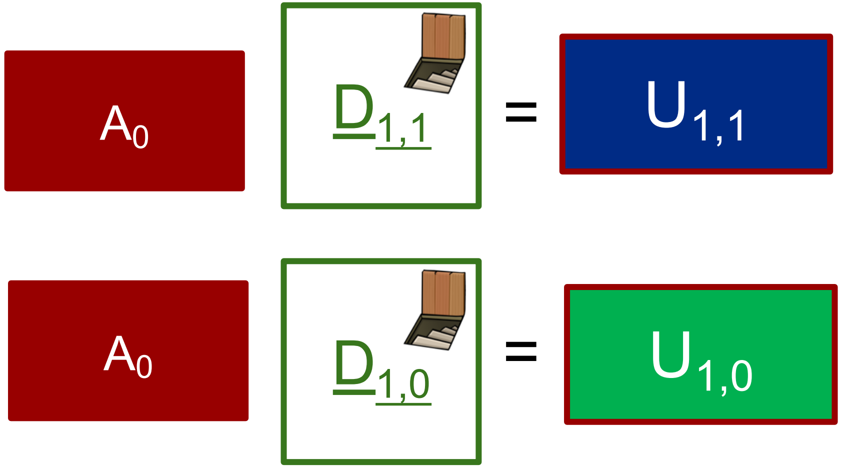
Functionality & Security toy examples

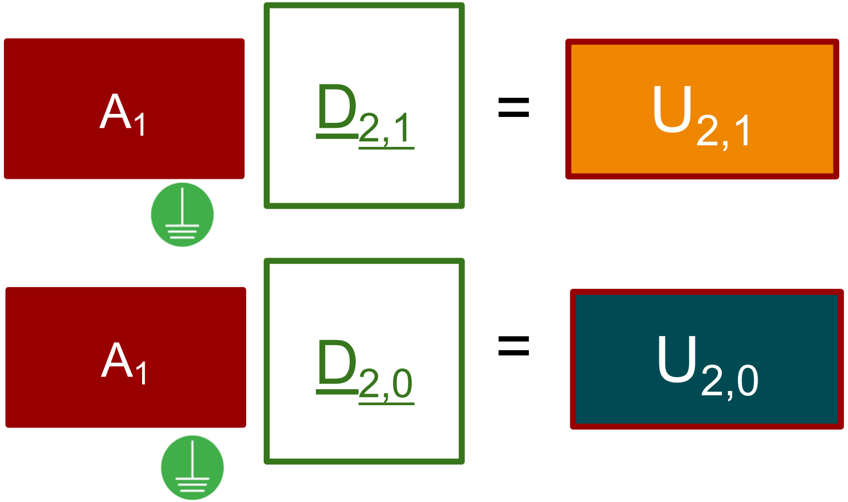




Turn off the trapdoor using GPV

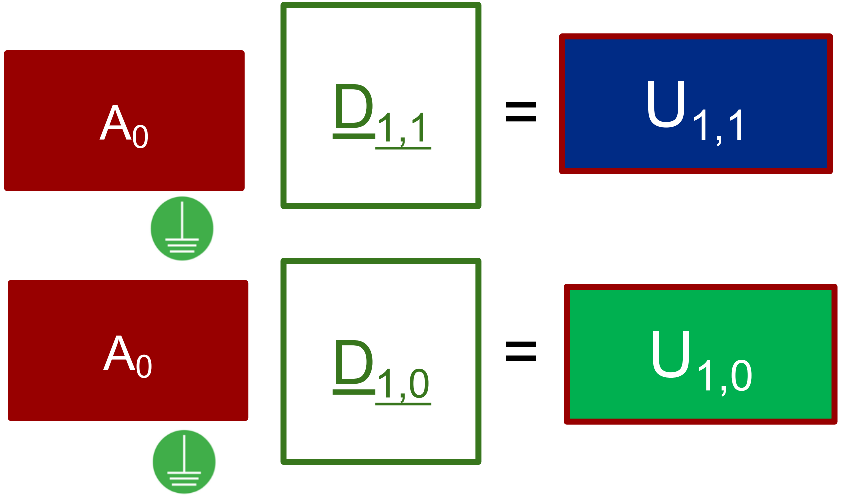
Functionality & Security toy examples

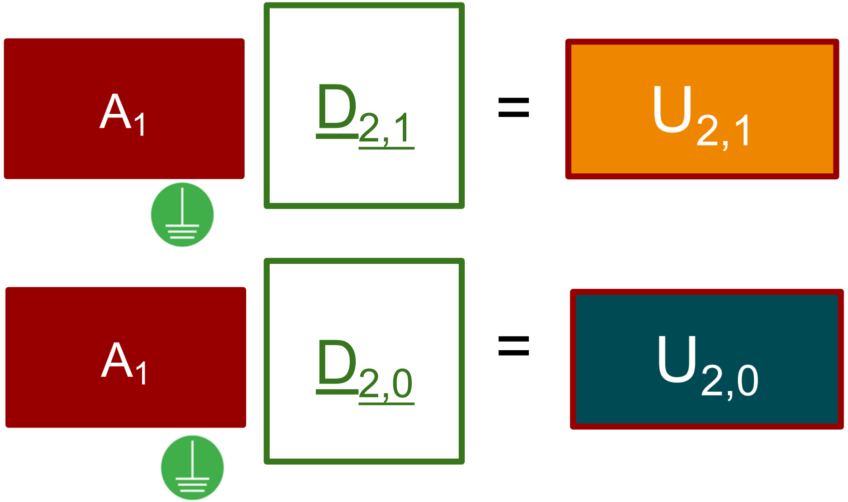




Permutation LWE

Functionality & Security toy examples

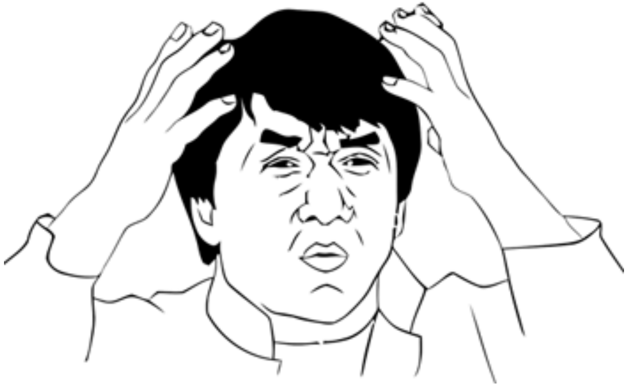




Turn off the trapdoor using GPV

Looks simple to achieve security based on LWE!

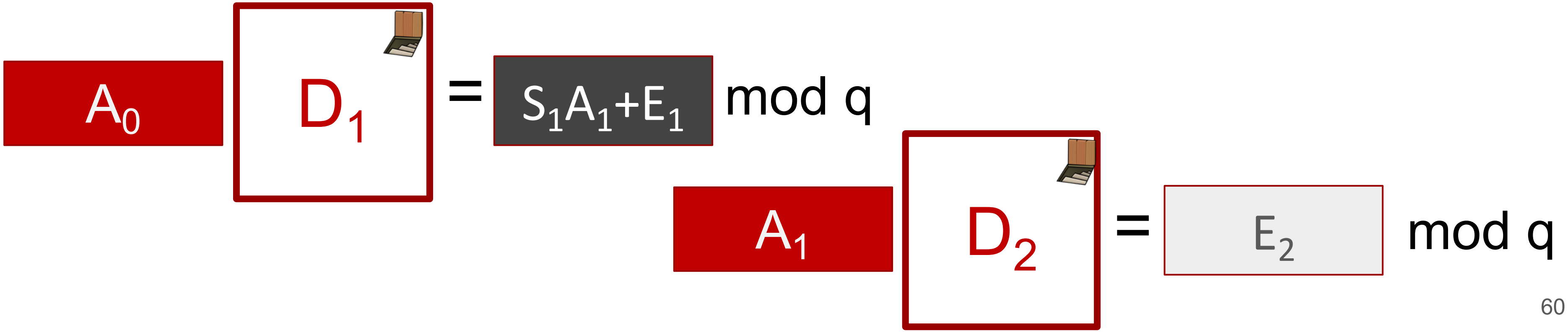
How do the insecure examples look like?



For example, let $S_2 = 0$ in

Insecure example

$A_0 D_1 = S_1 A_1 + E_1, \quad A_1 D_2 = S_2 A_2 + E_2 \mod q$

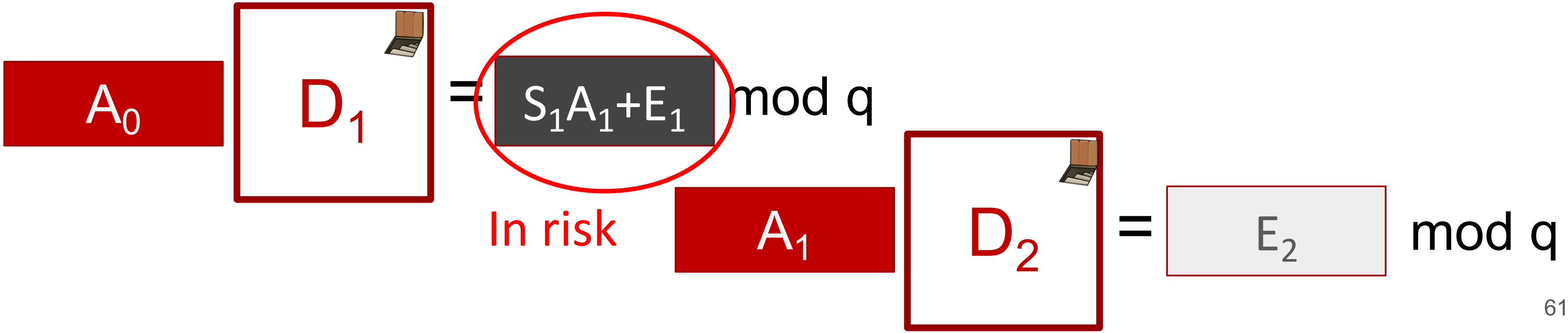


For example, let $S_2 = 0$ in

$A_0 D_1 = S_1 A_1 + E_1, \quad A_1 D_2 = S_2 A_2 + E_2 \mod q$

Insecure example

D_2 becomes a "weak trapdoor" of A_1, then S_1 is in danger



Insecure example

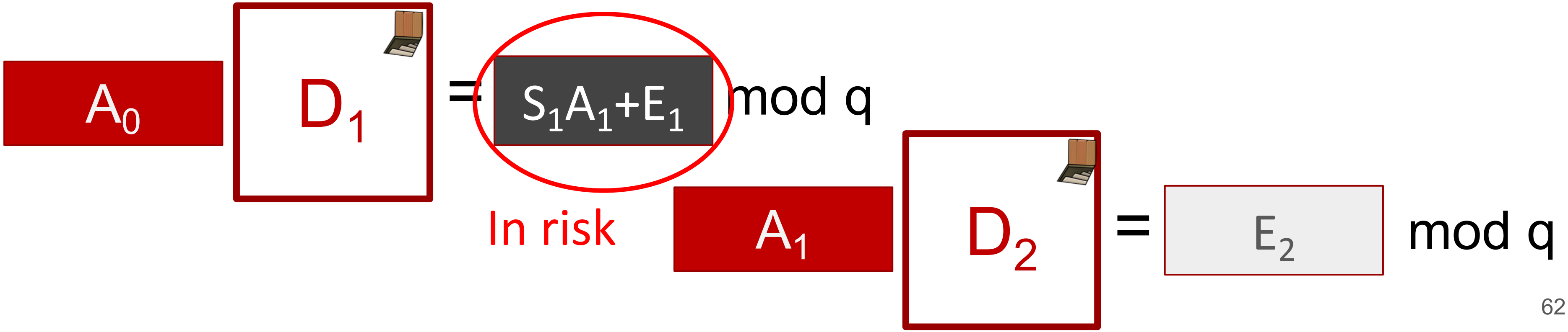
For example, let $S_2 = 0$ in

$A_0 D_1 = S_1 A_1 + E_1, \quad A_1 D_2 = S_2 A_2 + E_2 \quad \bmod q$

D_2 becomes a "weak trapdoor" of A_1, then S_1 is in danger

$\text{Eval} = A_0 D_1 D_2 = (S_1 A_1 + E_1) D_2 = S_1 E_2 + E_1 D_2 \bmod q$

Recover $S_1 E_2 + E_1 D_2$ over integers, can do many things.



Compared to other lattice application frameworks

"Regev-like schemes" [Regev 05]

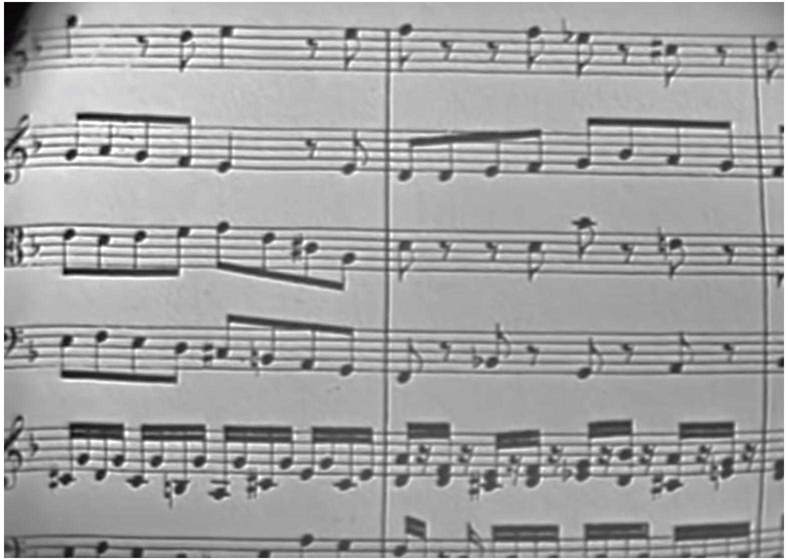
Public key: A, SA+E; secret key: S; message: (SA+E)*R + m*(q/2)

"Dual-Regev-like schemes" [Gentry, Peikert, Vaikuntanathan 08]

Public key: A_0, A_1, …, A_d, (master) secret key: the trapdoor of A_0

"GGH15-like" $A_0, S_1A_1+E_1, \ldots, S_kA_k+E_k \rightarrow \prod S A_k+E$

Both the message/function to be hidden are in the LWE secret terms



Plan of today:

~~1. Introduction~~
~~2. GGH15: functionality and security overview~~
3. Applications: Obfuscators & Private constrained PRFs

Open problems will be mentioned during the talk

Multilinear maps

GGH13, CLT13, GGH15

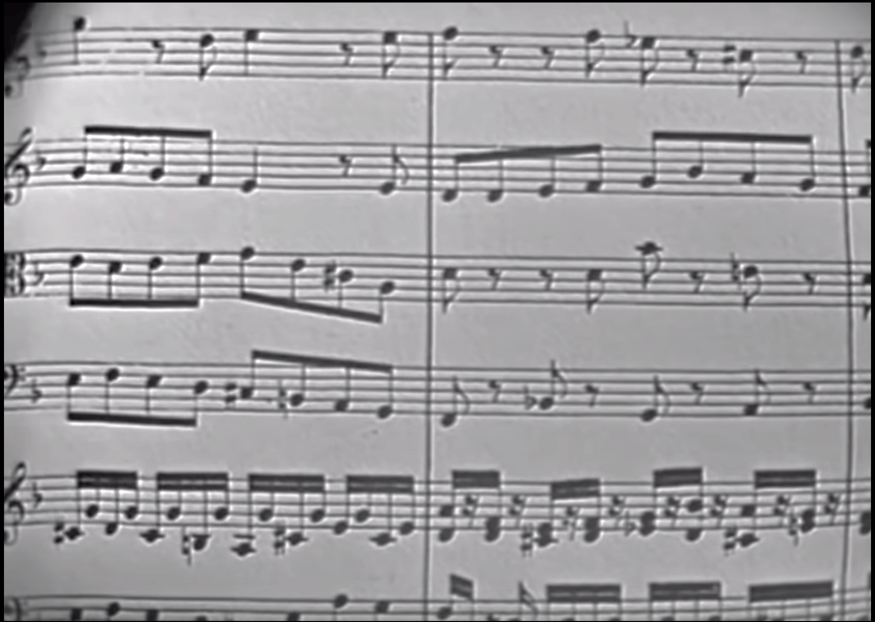
1. Private Constrained PRFs

[Canetti, Chen 17]

2. General-purpose obfuscation

[Gentry, Gorbunov, Halevi 15], …

With a reduction from LWE (via safe use of GGH15); Candidates exists



Private Constrained PRFs

Private Constrained Pseudorandom Function in 3 slides

Private Constrained Pseudorandom Function in 3 slides

[Goldreich, Goldwasser, Micali 86]

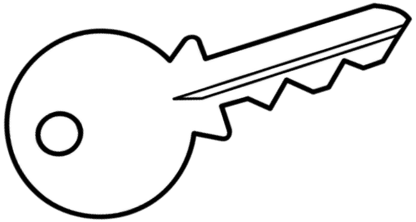
PRF

A truly random function

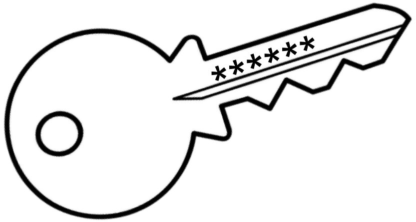
With oracle access
to either left or right

Private Constrained Pseudorandom Function in 3 slides

[Boneh, Waters 13], [Kiayias, Papadopoulos, Triandopoulos, Zacharias 13], [Boyle, Goldwasser, Ivan 14]



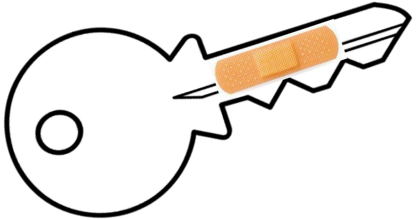
original key



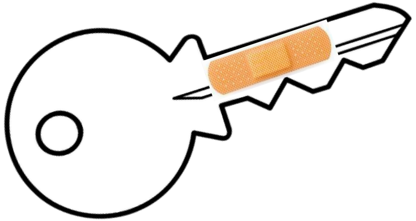
modified key

Private Constrained Pseudorandom Function in 3 slides

[Boneh, Lewi, Wu 17]

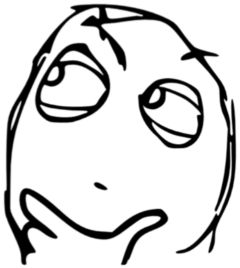


original key



privately modified key

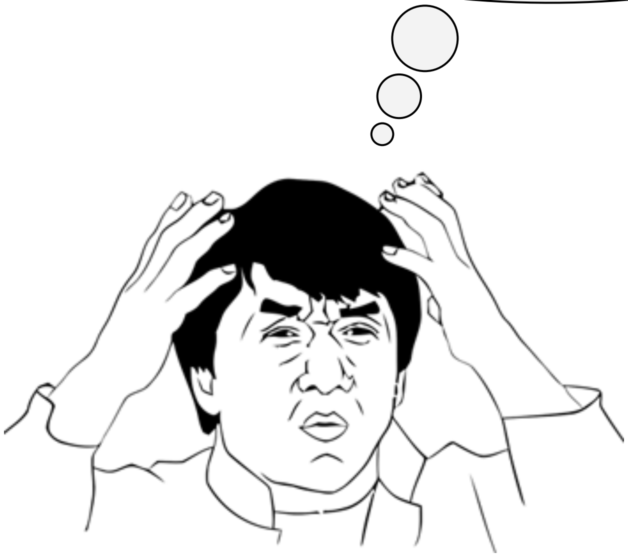
either the original key
or the modified one



Private key owner

~~What are private constrained PRFs?~~

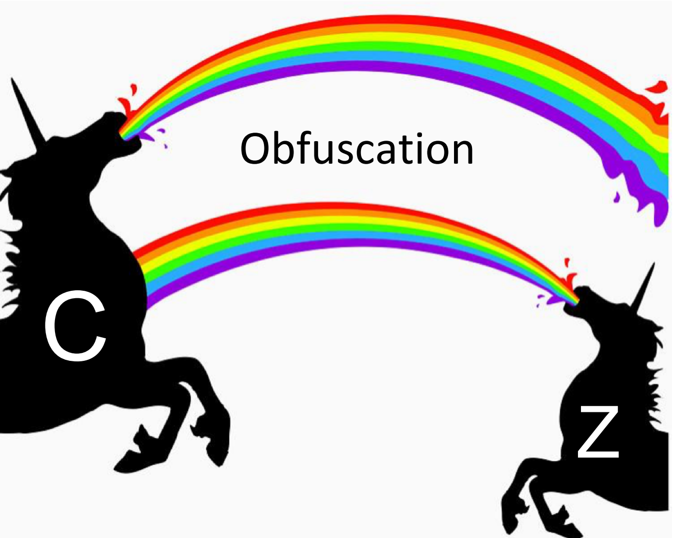
What is the *motive*?



[Canetti Chen 17]: Two-key secure PCPRF (for a circuit class C) implies obfuscation (for C)

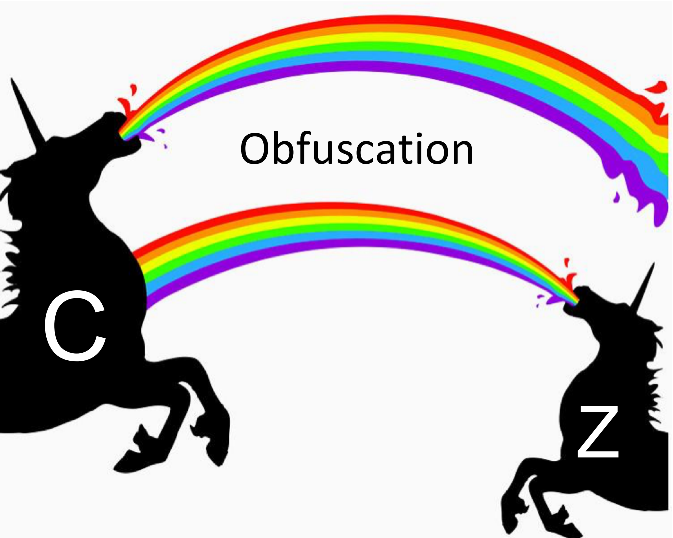
Obf = { K[C], K[original] }

Eval(Obf, x): Compare K[C](x) and K[original](x)



[Canetti Chen 17]: Two-key secure PCPRF (for a circuit class C) implies obfuscation (for C)

Obf = { K[C], K[original] }
Eval(Obf, x): Compare K[C](x) and K[original](x)



But if two constrained keys are published, then we don’t know how to prove constraint-hiding based on LWE.

[Canetti, Chen 17] 1-key PCPRF implies 1-key private-key functional encryption (a.k.a. reusable garbled circuits).

Construction:

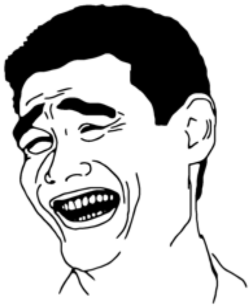
Enc(m;r): ct = $Enc_{Sym.K}$(m;r); tag = PRF.K[original](ct)

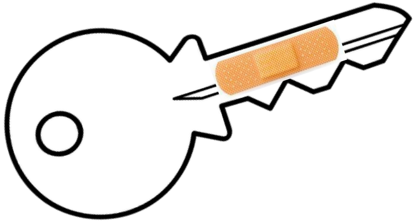
Functional_SK[Sym.K, PRF.K, C]:

A private constrained key for the "decryption and eval" functionality

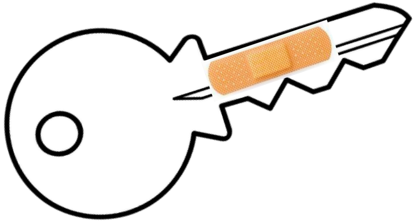
PRF.K[C($Dec_{Sym.K}$(.))]

Eval: compute PRF.K[C($Dec_{Sym.K}$(.))](ct), and compare with tag





original key



privately modified key

Applications of Private Constrained PRFs:

Obfuscation (if it is 2-key secure)*

Reusable garbled circuits

Privately-detectable watermarking

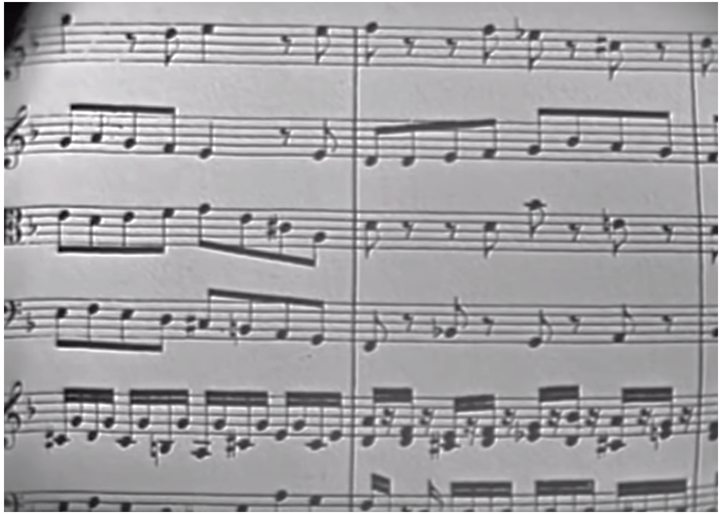
With key homomorphism => traitor tracing

Maybe more …

~~What are private constrained PRFs?~~

~~What is the *motive*?~~

How to construct from *lattices*?



Private Constrained PRFs from Lattices?

Step 1: Start from a lattice PRF.
[Banerjee, Peikert, Rosen 12]

Step 2: Embed a constraint.
[Barrington 86]

Step 3: Do Step 2 privately.
[GGH15]

[Banerjee, Peikert, Rosen 12]

Key:

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

mod q

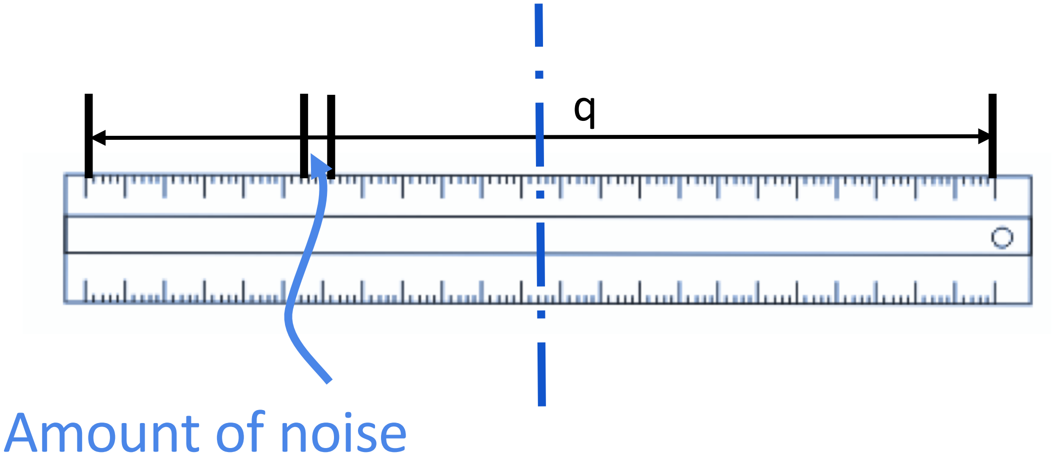
Eval: $F(x) = \{ \prod s_{i,x_i} A \}_2$

$s_{i,b}$ are LWE secrets from low-norm distributions

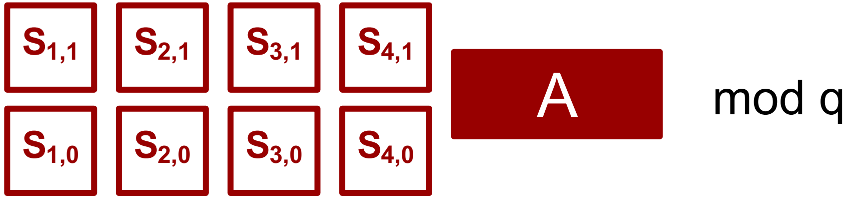
Rounding: $\{t\}_p$: $Z_q \to Z_p$

Compute t*p/q, then round to the nearest integer

In this talk, p=2, q/p>exp(L), q/p ~ super-polynomial



[Banerjee, Peikert, Rosen 12]



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

Main observation: After rounding, can inject noises without changing the functionality with high probability.

[Banerjee, Peikert, Rosen 12]

$$F(0110) = \{ s_{1,0}s_{2,1}s_{3,1}s_{4,0} \, A \}_2$$

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

[Banerjee, Peikert, Rosen 12]

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

$F(0110)$

$= \{ s_{1,0} s_{2,1} s_{3,1} s_{4,0} A \}_2$

$\approx_s \{ s_{1,0} s_{2,1} s_{3,1} (s_{4,0} A + E_{4,0}) \}_2$

[Banerjee, Peikert, Rosen 12]

mod q

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

F(0110)

$= \{ s_{1,0} s_{2,1} s_{3,1} s_{4,0} A \}_2$

$\approx_s \{ s_{1,0} s_{2,1} s_{3,1} (s_{4,0} A + E_{4,0}) \}_2$

$\approx_c \{ s_{1,0} s_{2,1} s_{3,1} Y_{***0} \}_2$

[Banerjee, Peikert, Rosen 12]

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

F(0110)

$= \{ s_{1,0}s_{2,1}s_{3,1}s_{4,0} \, A \}_2$

$\approx_s \{ s_{1,0}s_{2,1}s_{3,1}(s_{4,0} \, A + E_{4,0}) \}_2$

$\approx_c \{ s_{1,0}s_{2,1}s_{3,1} Y_{***0} \}_2$

$\approx_s \{ s_{1,0}s_{2,1}(s_{3,1} Y_{***0} + E_{3,1}) \}_2$

[Banerjee, Peikert, Rosen 12]

$s_{1,1}$	$s_{2,1}$	$s_{3,1}$	$s_{4,1}$	A	mod q
$s_{1,0}$	$s_{2,0}$	$s_{3,0}$	$s_{4,0}$		

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

$$\begin{aligned} &F(0110) \\ &= \{ s_{1,0}s_{2,1}s_{3,1}s_{4,0} A \}_2 \\ &\approx_s \{ s_{1,0}s_{2,1}s_{3,1}(s_{4,0} A + E_{4,0}) \}_2 \\ &\approx_c \{ s_{1,0}s_{2,1}s_{3,1}Y_{***0} \}_2 \\ &\approx_s \{ s_{1,0}s_{2,1}(s_{3,1}Y_{***0} + E_{3,1}) \}_2 \\ &\approx_c \{ s_{1,0}s_{2,1}Y_{**10} \}_2 \\ &\approx \ldots \approx \{ Y_{0110} \}_2 \end{aligned}$$

Key:

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

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

Exercise: show that taking matrix subset-product without rounding does not give a PRF.

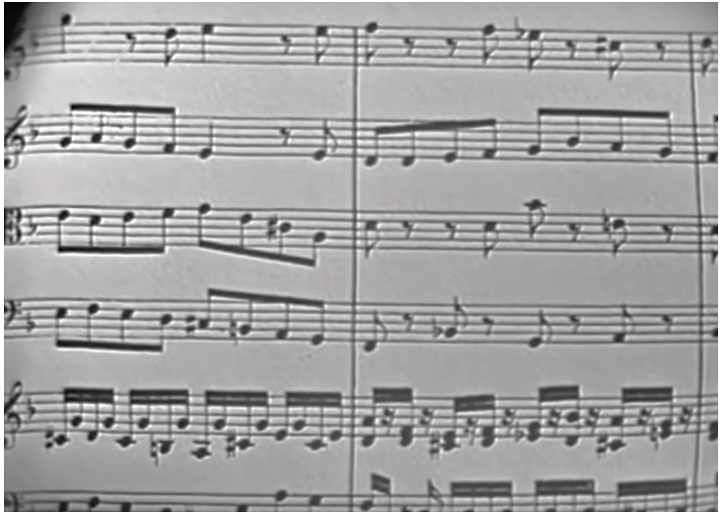
Open Problem 2

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

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

Open problem: prove or disprove that when q is a polynomial, the construction is a PRF.

The distribution of the S matrices can be uniformly from Z_q



Private Constrained PRFs from Lattices?

Step 1: Start from a lattice PRF.

[Banerjee, Peikert, Rosen 12]

Step 2: Embed a constraint.

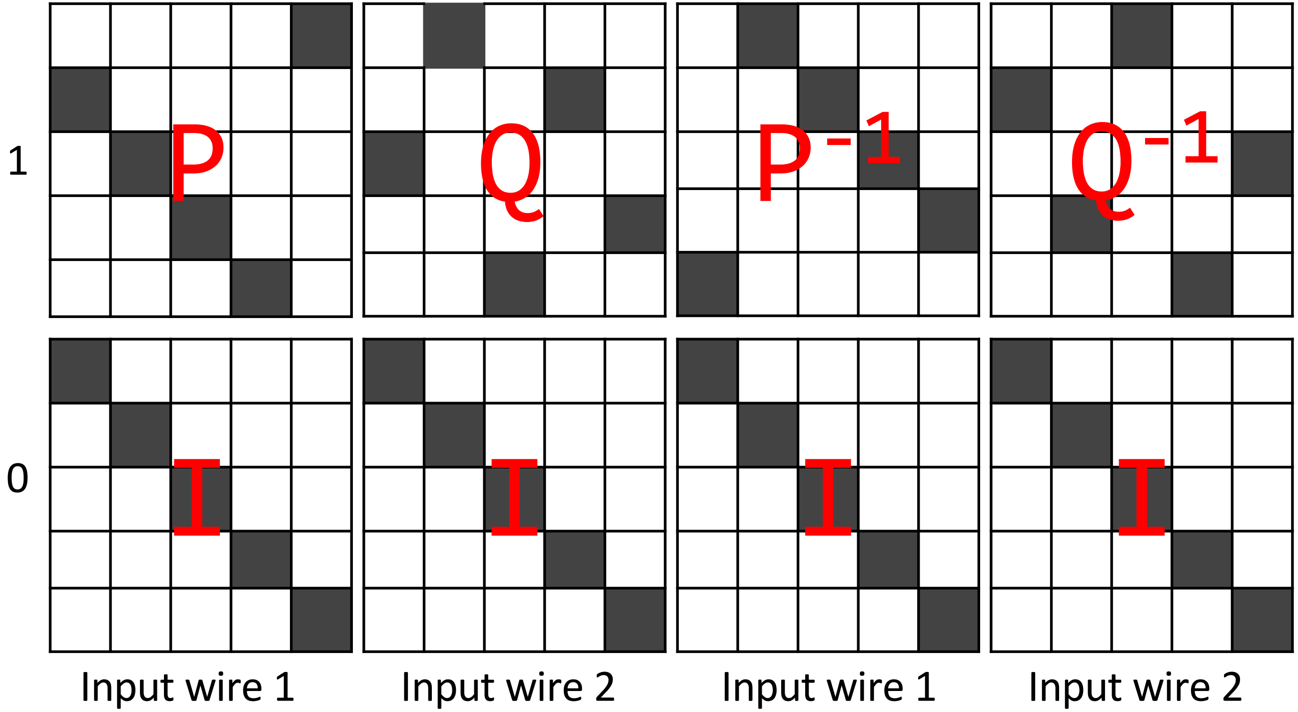
[Barrington 86]

Step 3: Do Step 2 privately.

[GGH15]

Barrington 1986: log-depth circuit => matrix branching program

Example: how to represent an AND gate

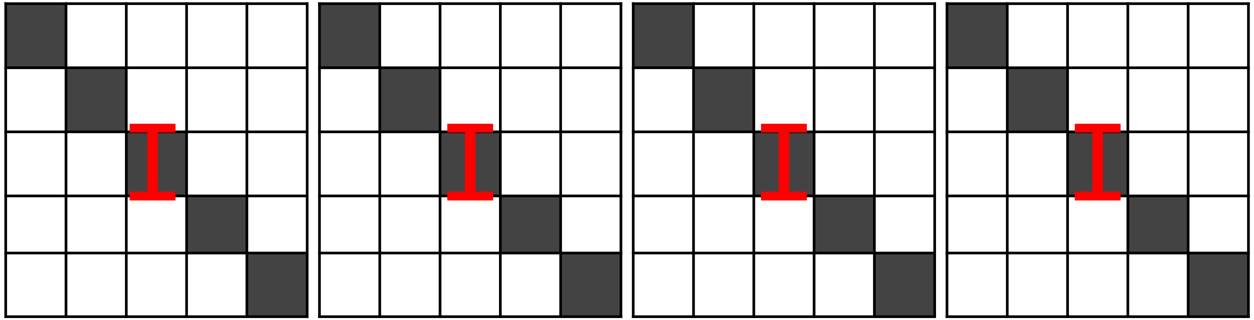


Barrington 1986: log-depth circuit => matrix branching program

Example: how to represent an AND gate 0 and 0

1

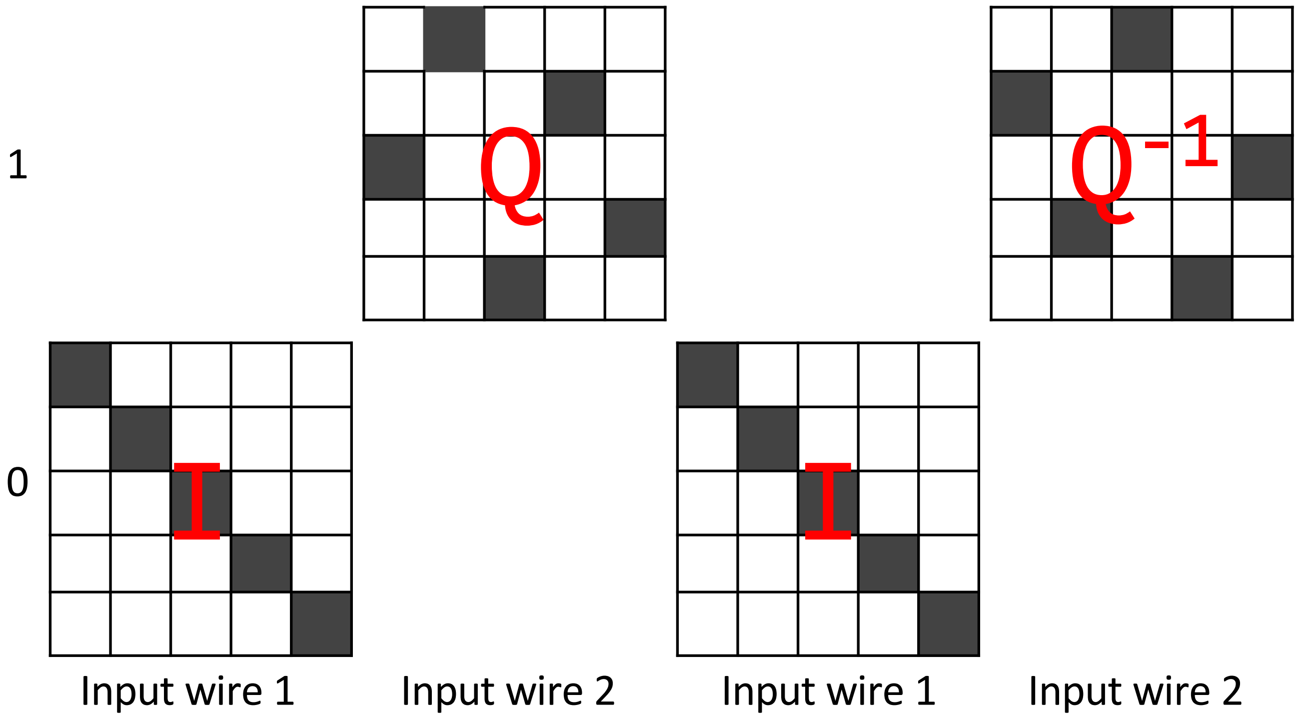
0



Input wire 1 Input wire 2 Input wire 1 Input wire 2

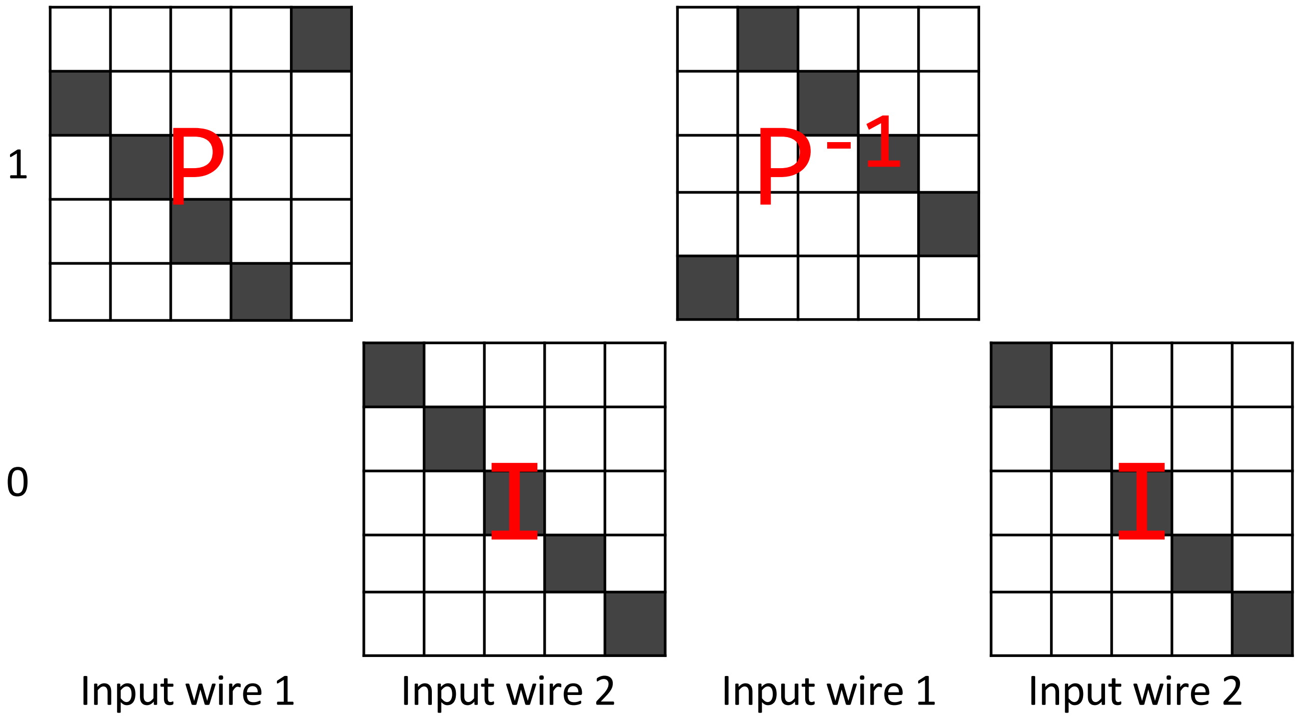
Barrington 1986: log-depth circuit => matrix branching program

Example: how to represent an AND gate 0 and 1



Barrington 1986: log-depth circuit => matrix branching program

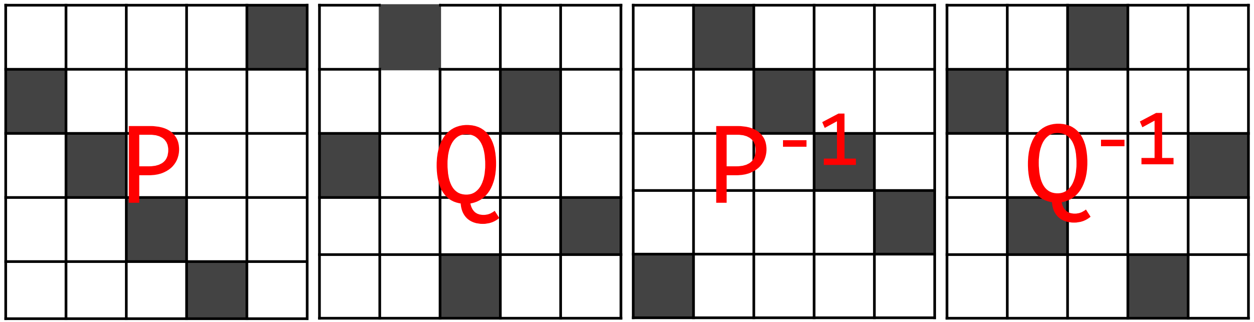
Example: how to represent an AND gate 1 and 0



Barrington 1986: log-depth circuit => matrix branching program

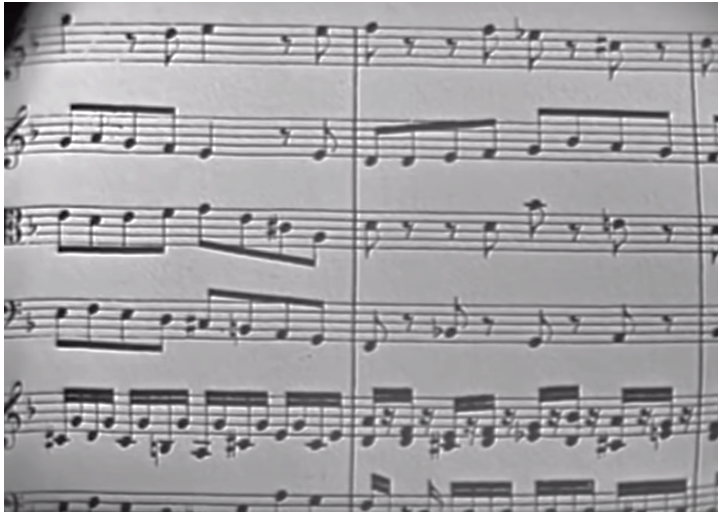
Example: how to represent an AND gate 1 and 1 $PQP^{-1}Q^{-1} = C \neq I$

1



0

Input wire 1 Input wire 2 Input wire 1 Input wire 2



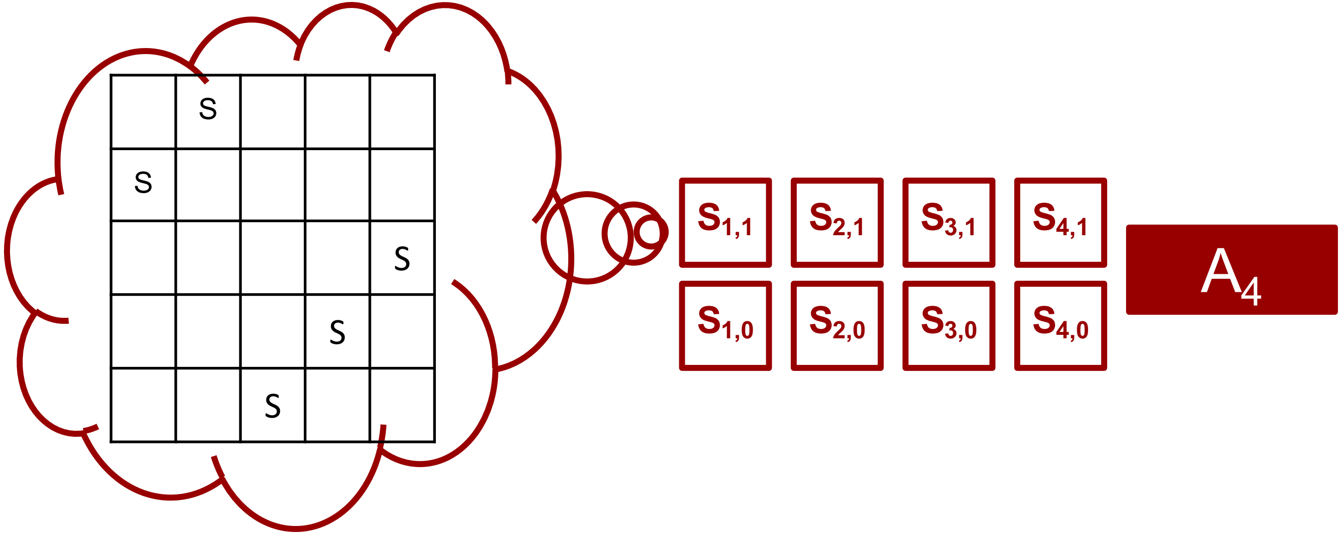
Private Constrained PRFs from Lattices?

Step 1: Start from a lattice PRF.
[Banerjee, Peikert, Rosen 12]

Step 2: Embed a constraint.
[Barrington 86]

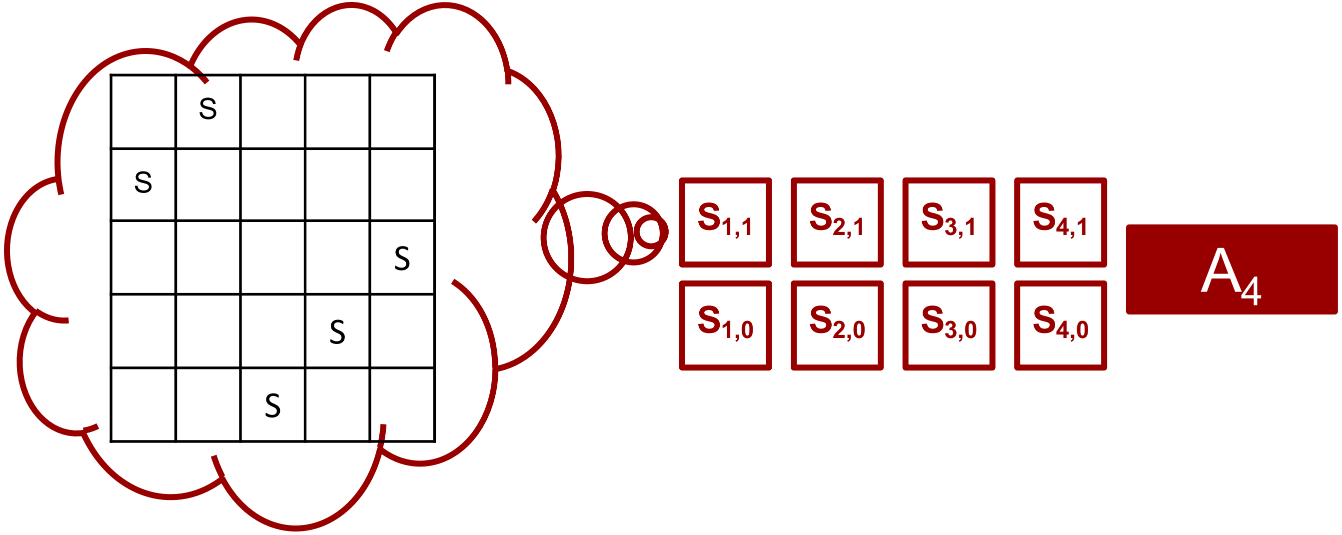
Step 3: Do Step 2 privately.
[GGH15]

Embed the permutation matrices in the LWE secret $B_{i,b} \otimes s_{i,b}$

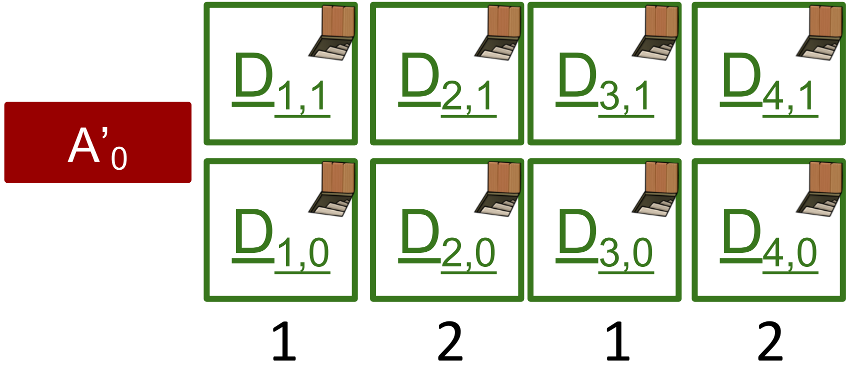


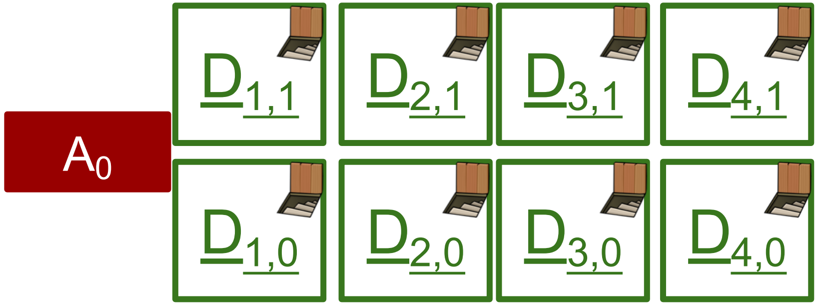
e.g. $I \otimes s = \begin{pmatrix} s & & & & \\ & s & & & \\ & & s & & \\ & & & s & \\ & & & & s \end{pmatrix}$ $\quad P \otimes s = \begin{pmatrix} & s & & & \\ s & & & & \\ & & & & s \\ & & & s & \\ & & s & & \end{pmatrix}$

Embed the permutation matrices in the LWE secret $B_{i,b} \otimes s_{i,b}$

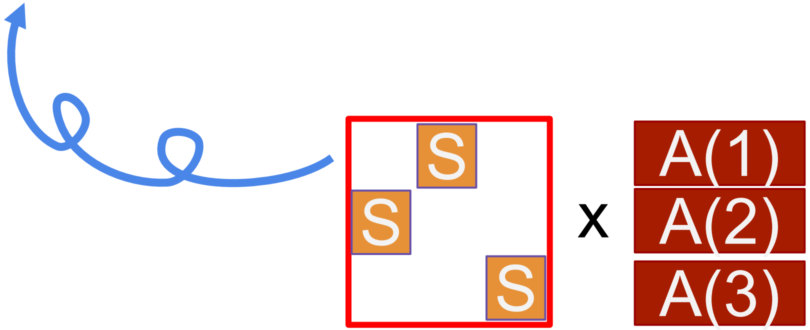


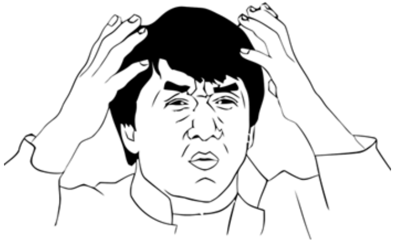
Constrained key:
the GGH15 encoding





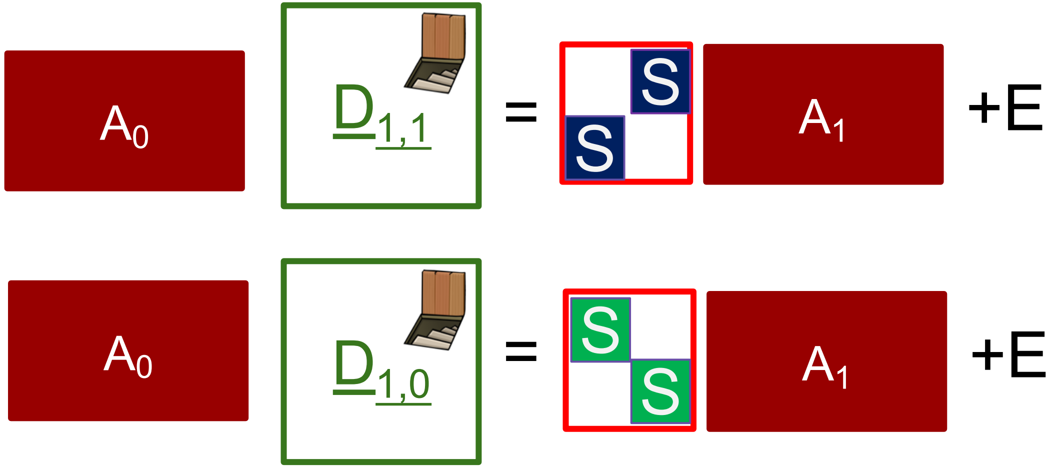
The real constrained key

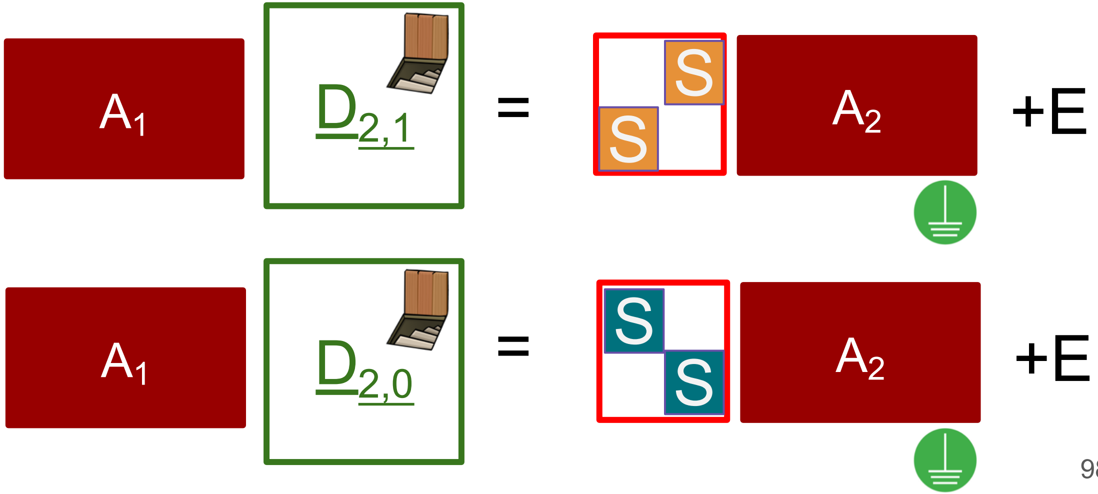




How to prove the branching program is hidden by GGH15 encoding?

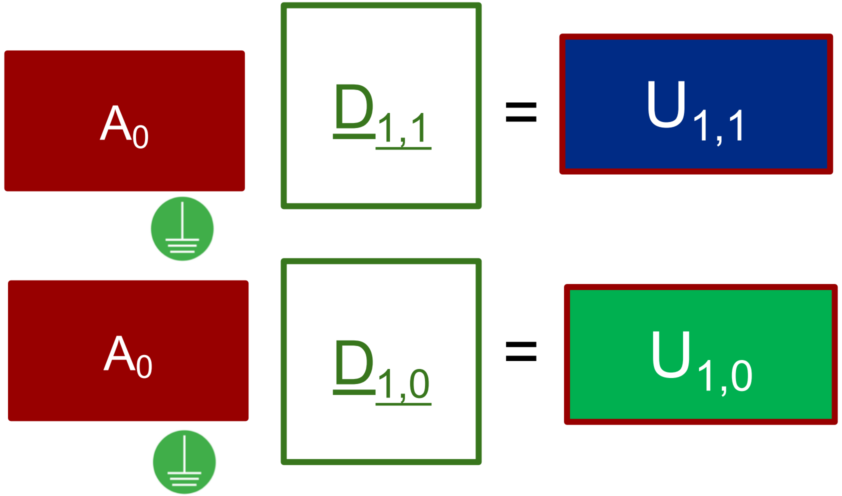
Recall Toy example 2

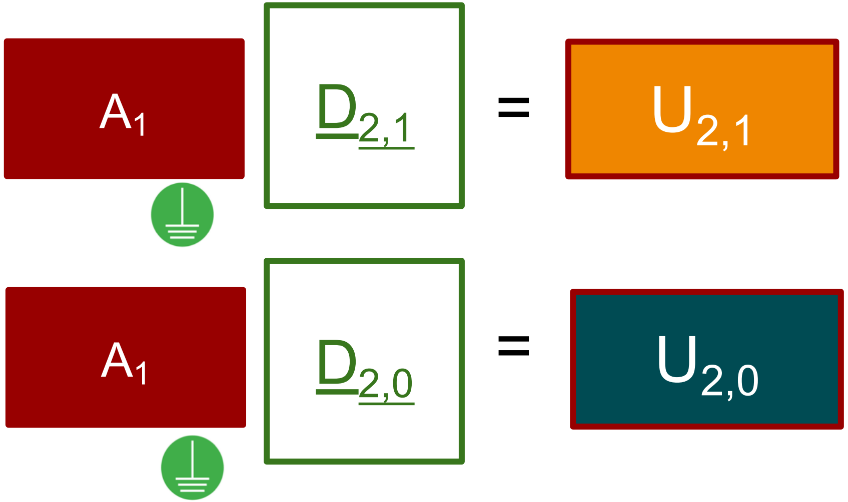




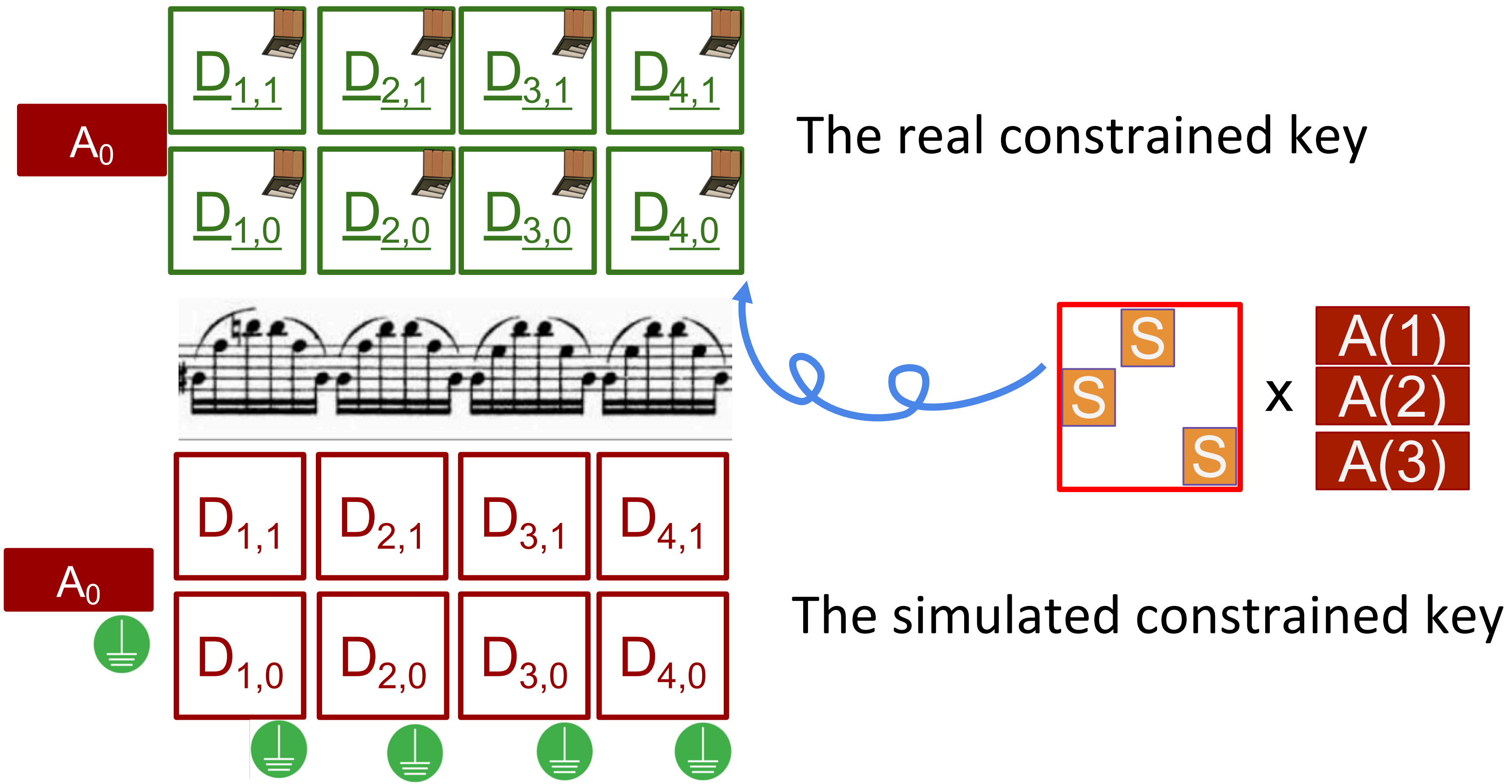
Claim: this construction hides all the structures in the S matrices.

Recall Toy example 2



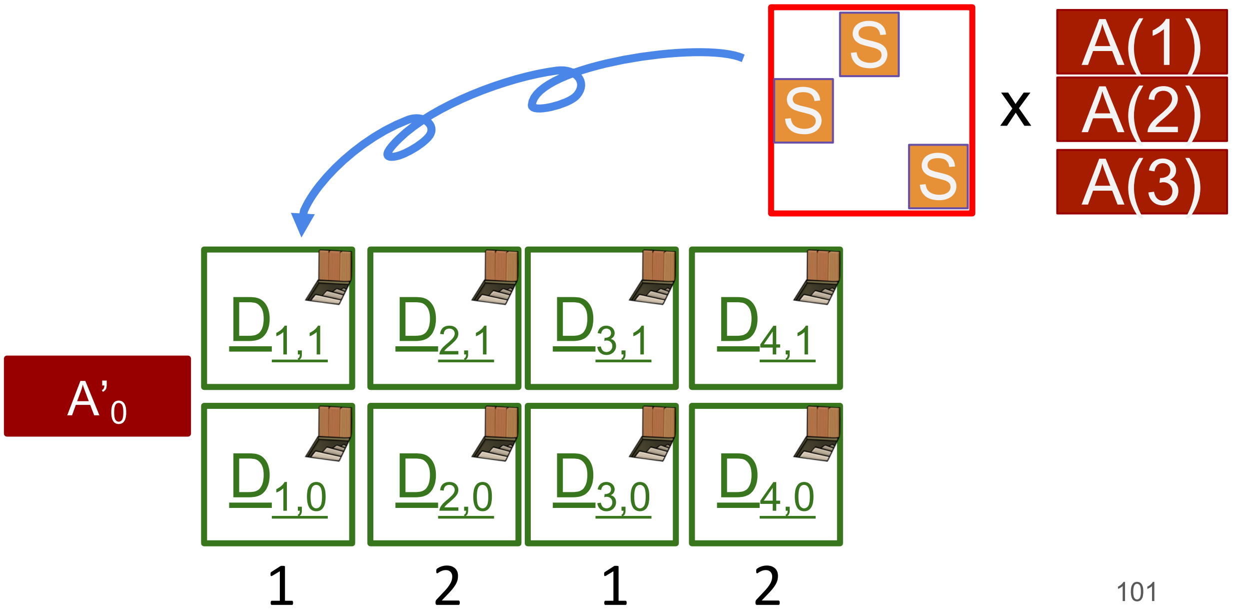


Perm-LWE + Turning off the trapdoor using GPV


A_0
$D_{1,1}$ $D_{2,1}$ $D_{3,1}$ $D_{4,1}$
$D_{1,0}$ $D_{2,0}$ $D_{3,0}$ $D_{4,0}$
The real constrained key
S
S
S
x
A(1)
A(2)
A(3)
A_0
$D_{1,1}$ $D_{2,1}$ $D_{3,1}$ $D_{4,1}$
$D_{1,0}$ $D_{2,0}$ $D_{3,0}$ $D_{4,0}$
The simulated constrained key

Takeaway from the Private Constrained PRF:

It is safe to use GGH15 to encode permutation matrices, and make it useful.



Genealogy of Lattices-based PRFs

Open Problem 3

[BPR12] -- the first lattice-based PRF
[BLMR13] -- key homomorphic
*[BP14] -- better key homomorphic, embed a tree
*[BFPPS15] -- [BP14] is puncturable
*[BV15] -- embed a circuit, constrained for P
*[BKM17] -- puncture privately, built from [BV15]
[CC17] -- constrained privately for NC1, influenced by GGH15 mmaps
*[BTVW17] -- constrained privately for all P, built from [BV15]
*[PS18] -- constrained and program privately for all P, built from [BV15]
[CVW18] -- constrained privately for BP, influenced by GGH15 mmaps

* uses gadget matrix G, adapted from the lattices-based FHE, ABE, PE

Open Question: Is there a transformation between Dual-Regev-based homomorphic schemes and GGH15-based ones?

Multilinear maps

GGH13, CLT13, GGH15

1. Private Constrained PRFs

[Canetti, Chen 17]

2. General-purpose obfuscation

[Gentry, Gorbunov, Halevi 15], …

Recall [Canetti Chen 17]

"Obfuscation is almost private constrained PRF with two keys:
One for the constraint C, the other one for all 1."

Recall [Canetti Chen 17]

"Private constrained PRF is almost
[GGHRSW 13] + [GGH 15] obfuscator with only one branch."

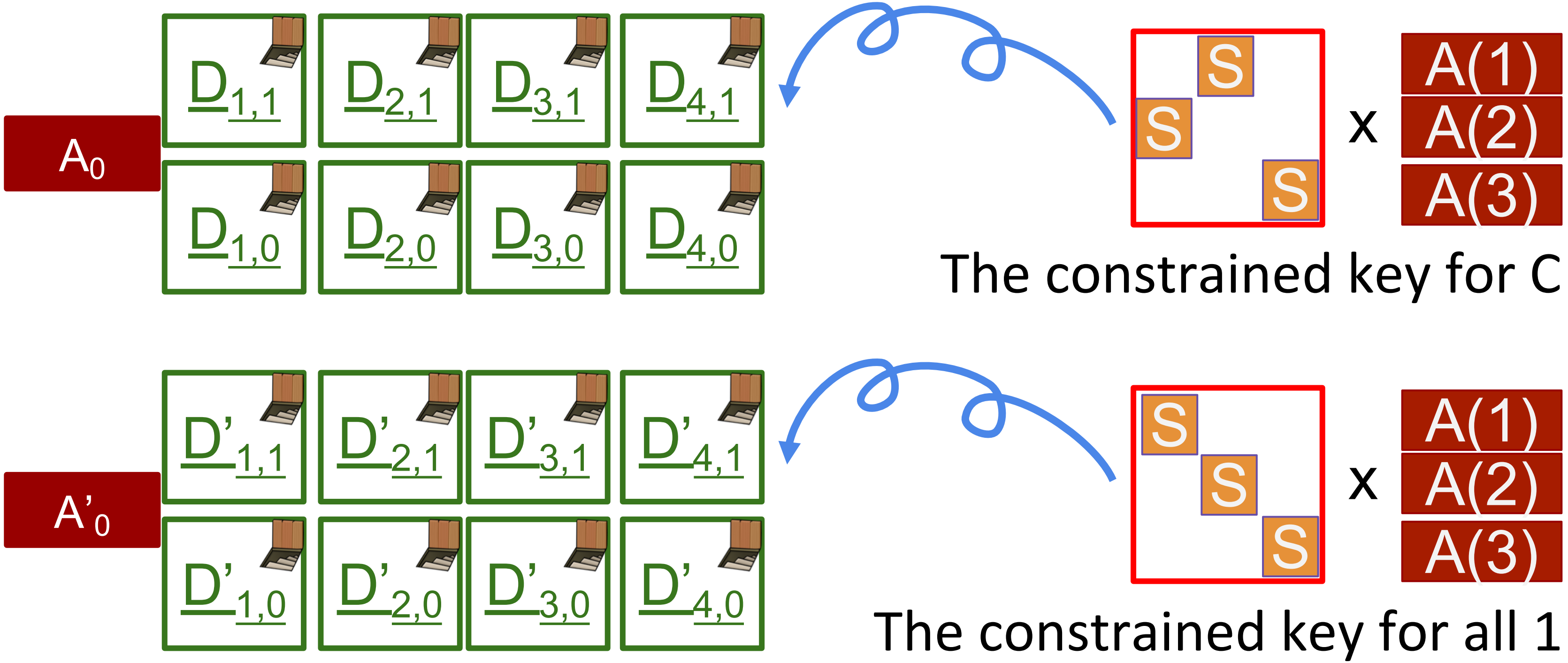
↑ The more "historically correct" view

Recall [Canetti Chen 17]

"Obfuscation is almost private constrained PRF with two keys:
One for the constraint C, the other one for all 1."

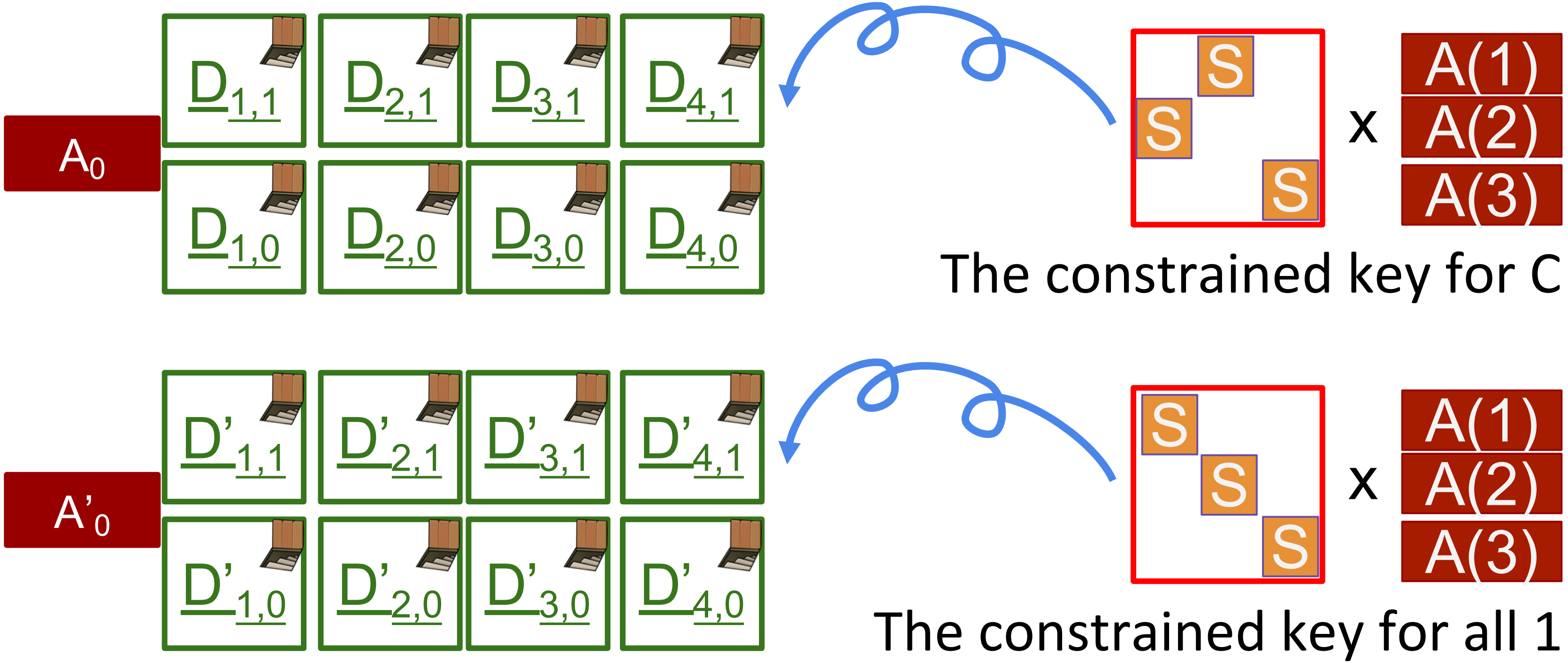
Recall [Canetti Chen 17]

"Private constrained PRF is almost
[GGHRSW 13] + [GGH 15] obfuscator with only one branch."



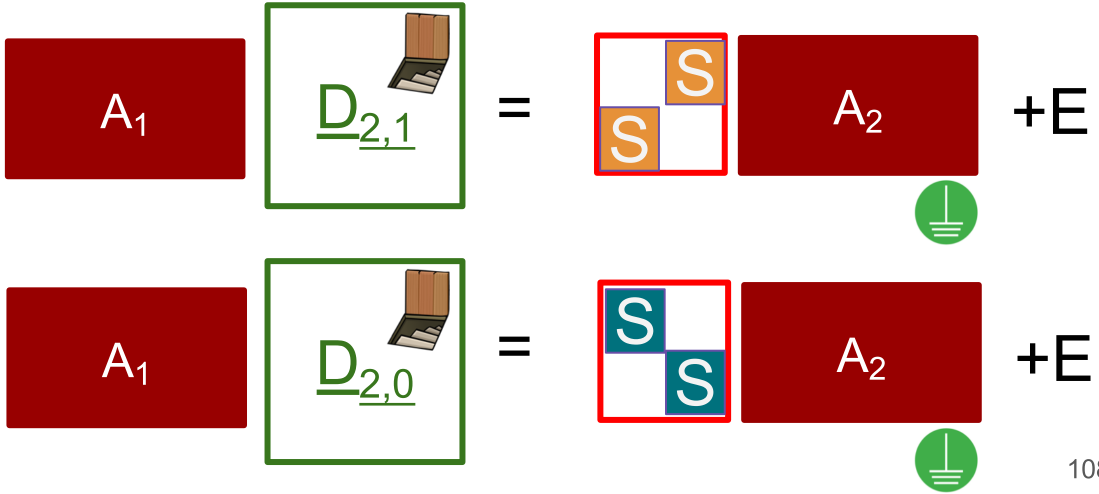
Claim 1: the proof strategy mentioned does not work.

Claim 2: when a sufficient amount of evaluation-to-0 is available, we can break the obfuscation scheme.



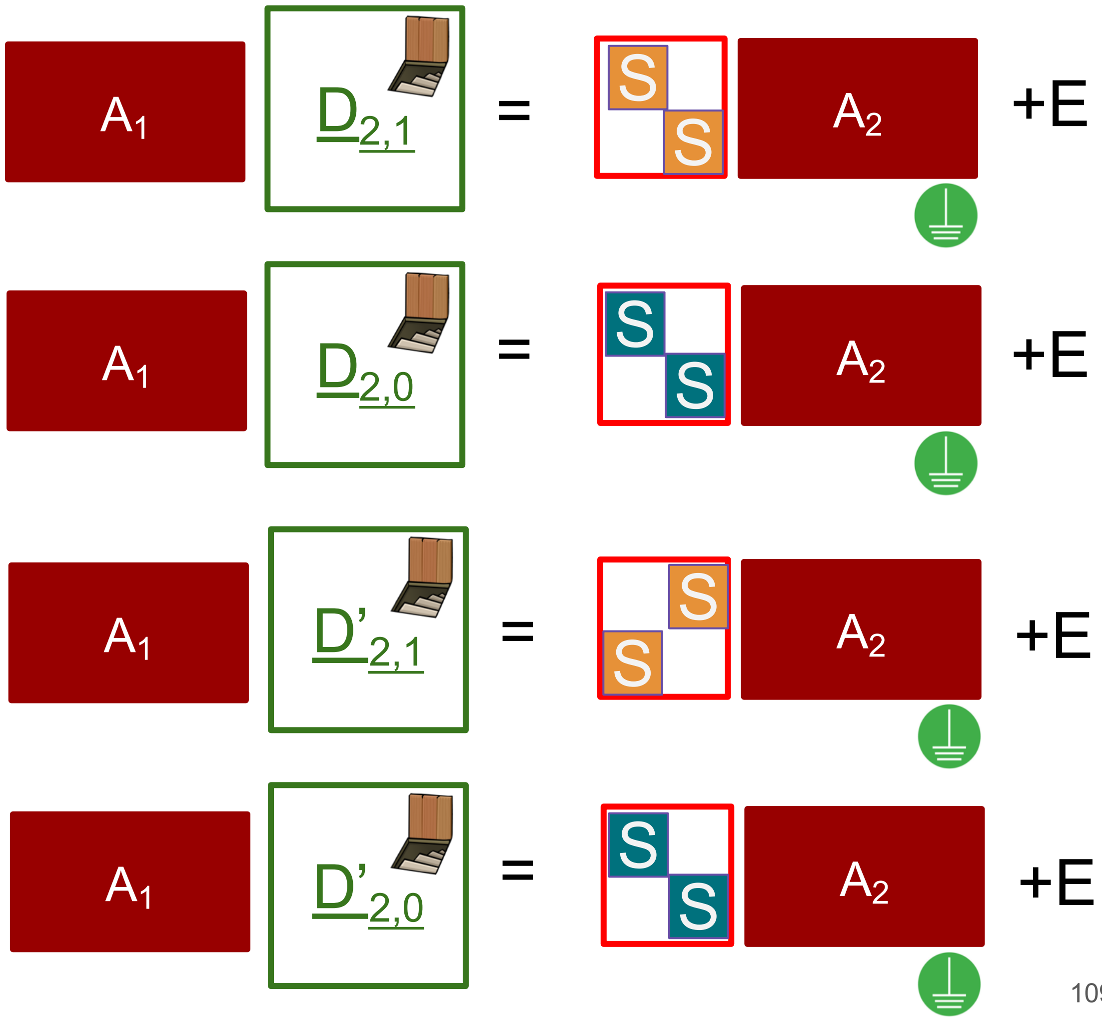
Claim 1: the proof strategy mentioned does not work.

Recall Toy example 2



Claim 1: the proof strategy mentioned does not work.

In the GGH15 obfuscator, it looks like …



Claim 1: the proof strategy mentioned does not work.

Correlated

Can apply LWE, but don't know how to use GPV

In the GGH15 obfuscator, it looks like …

Claim 2: when a sufficient amount of evaluation-to-0 is available, we can break the obfuscation scheme.

For x such that C(x) = 0, Eval(x) = … = $S_1E_2 + E_1D_2$ mod q

Recover $S_1E_2 + E_1D_2$ over integers, can do many things.

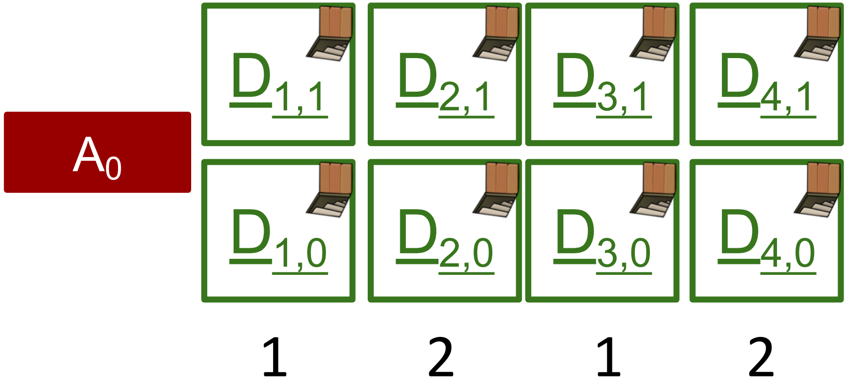
[Cheon, Han, Lee, Ryu, Stehle 15], [Coron, Lee, Lepoint, Tibouchi 16], [Chen, Gentry, Halevi 17]

Claim 2: when a sufficient amount of evaluation-to-0 is available, we can break the obfuscation scheme.

For x such that $C(x) = 0$, $\mathrm{Eval}(x) = \ldots = S_1E_2 + E_1D_2 \bmod q$

Recover $S_1E_2 + E_1D_2$ over integers, can do many things.

[Cheon, Han, Lee, Ryu, Stehle 15], [Coron, Lee, Lepoint, Tibouchi 16], [Chen, Gentry, Halevi 17]



[Chen, Vaikuntanathan, Wee 18] shows a classical polynomial attack, works as long as the inputs repeat for at most constant times.

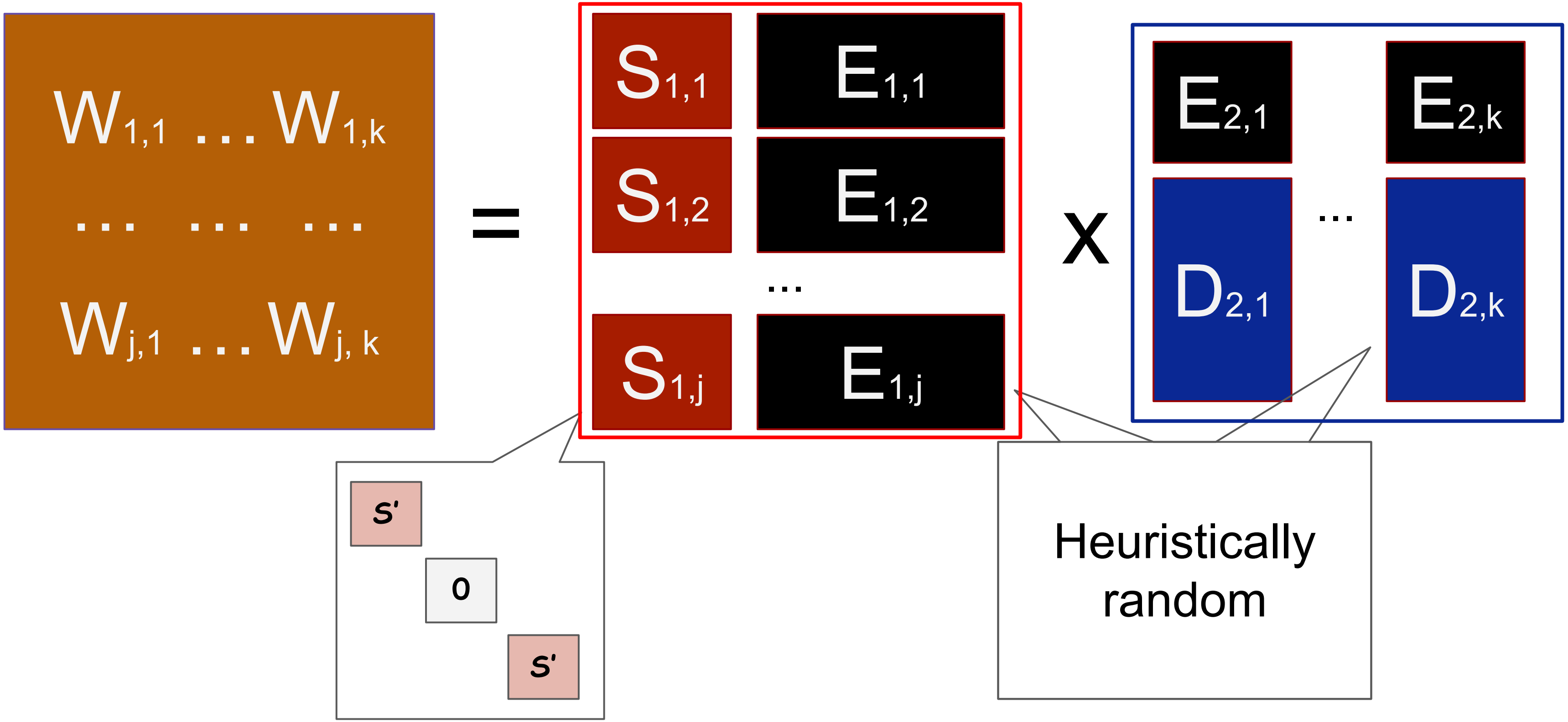
[Chen, Vaikuntanathan, Wee 18]

First compute a matrix,

$$\begin{matrix} W_{1,1} & \ldots & W_{1,k} \\ \ldots & \ldots & \ldots \\ W_{j,1} & \ldots & W_{j,k} \end{matrix} = \text{Results on many inputs that eval to small}$$

[Chen, Vaikuntanathan, Wee 18]

First compute a matrix, then compute the rank of the matrix.



Survey of iO candidates related to GGH15:

[Gentry, Gorbunov, Halevi 15]: translate GGHRSW13 into GGH15

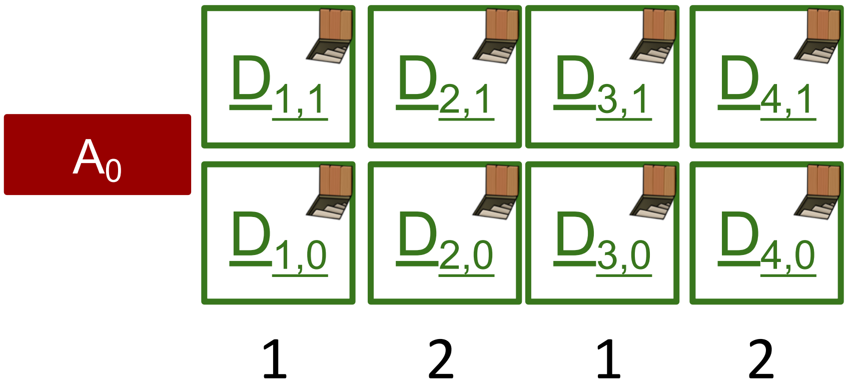
[Chen, Gentry, Halevi 17]: quantum attack for simple branching program

[Chen, Vaikuntanathan, Wee 18]: Break GGH15 with constant repetition, propose a candidate that enforce repetitions, non-commutative scalars, etc.

[Bartusek, Guan, Ma, Zhandry 18]: Another candidate, proof in the idealized model

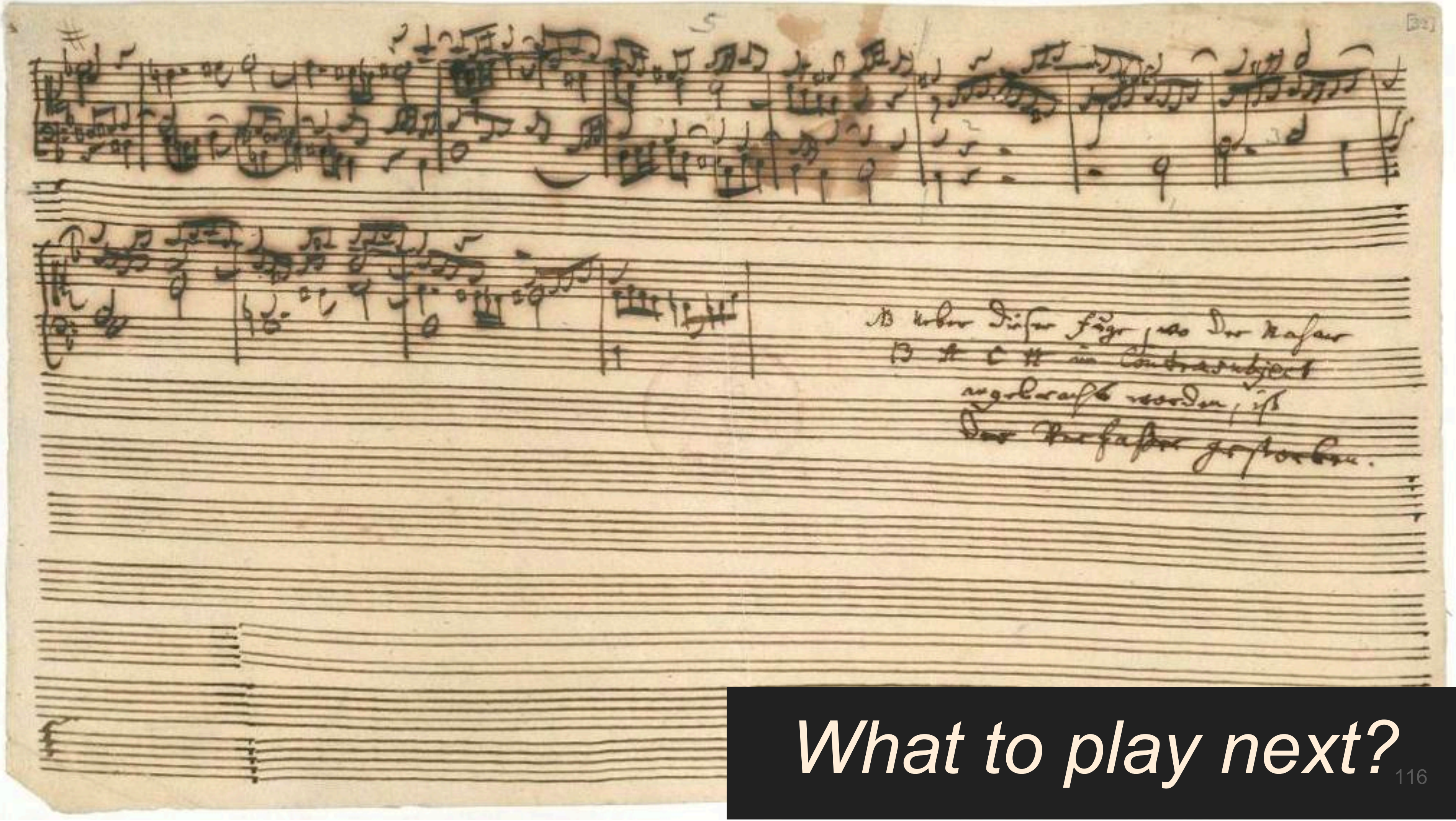
[Cheon, Cho, Hhan, Kim, Lee 19]: Statistical attack on CVW18 for polynomial noise

[Chen, Hhan, Vaikuntanathan, Wee 19]: Proof in a weaker idealized model, using super-polynomial noise.



Short summary:

Take [Gentry, Gorbunov, Halevi 15], or [Chen, Vaikuntanathan, Wee 18], using branching programs with super-constant repetitions, super-polynomial noise, no attacks are known, even quantum ones.



What to play next?

Lockable obfuscation
(Compute-then-Compare obf.)

Private constrained PRFs

Permutation branching program, almost always output 1 (random)

Witness encryption

Open Problem 4: classify

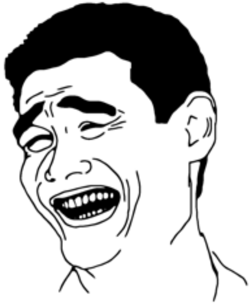
General evasive function obfuscators

Output 0 (small) very often

Multi-party key agreement

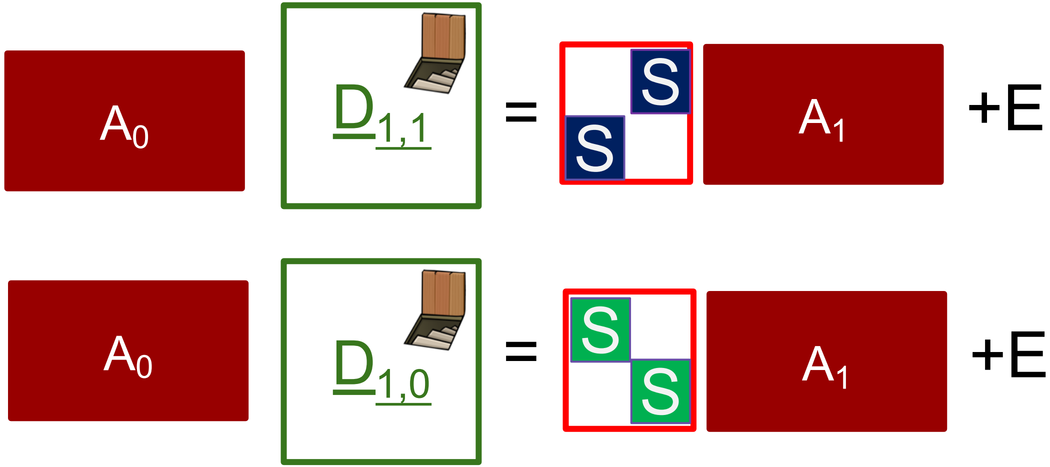
Indistinguishability obfuscation

Thought 1: on the proof technique

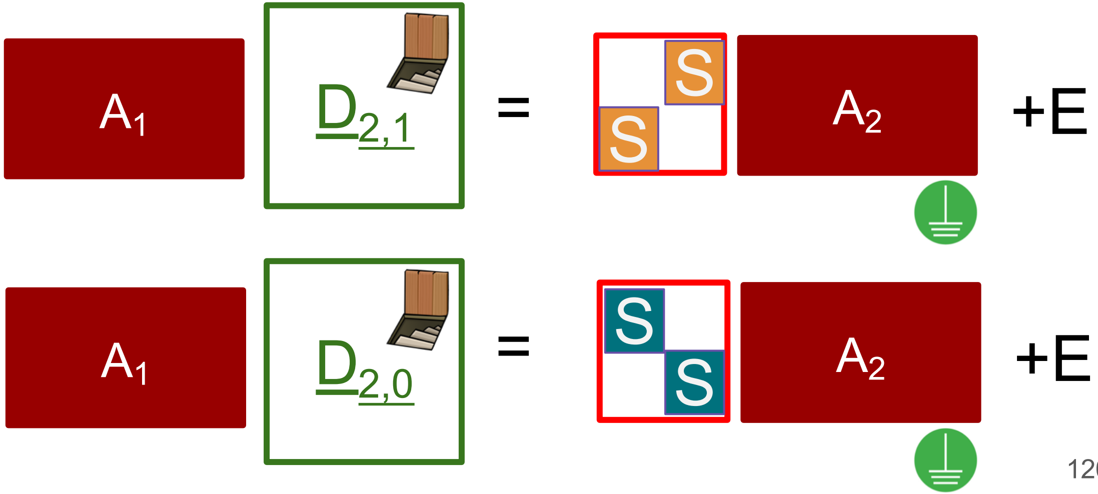


Thought 1: on the proof technique

Recall Toy example 2



Proof works when A_1 and A_2 are public, but they don’t have to be public …



Lockable obfuscation
(Compute-then-Compare obf.)

Private constrained PRFs

Permutation branching program, almost always output 1 (random)

[Chen, Vaikuntanathan, Wee 18]: proof beyond permutation BPs, using the fact that A matrices are hidden, but the S matrices are public

Still, witness encryption and general evasive function obfuscators are open

Open Problem 4

Output 0 (small) very often

Indistinguishability obfuscation

Thought 2: need new hard problems ”without mod q”

LWE + low-degree "PRG"

[Barak, Hopkins, Jain, Kothari, Sahai 19], [Jain, Lin, Matt, Sahai 19]

LWE + degree 3 functions over Z:

$$A, \quad s^T A + e^T \bmod q, \quad \{Q_i, Q_i(x, y, e)\}, i = 1 \text{ to } N$$

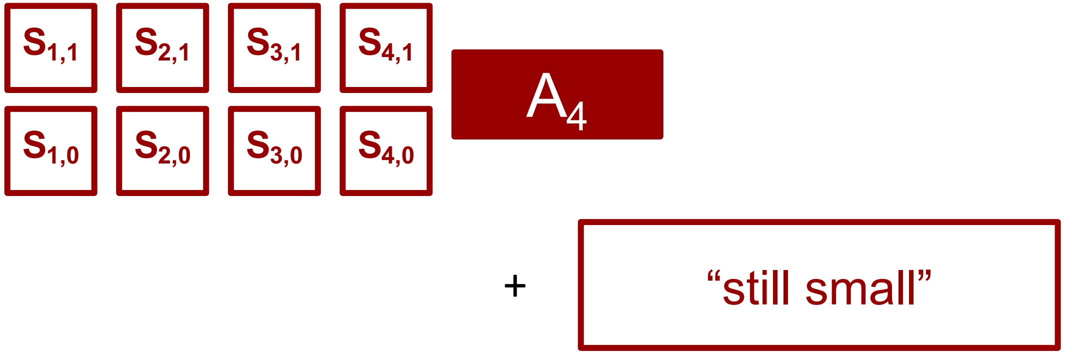
The adversary is asked to recover e. Here x, y, e are small and of dimension m, Qi are degree-3 "small" polynomials over Z, $N = m^{1.01}$

Bilinear maps + LWE + low-degree "PRG"
$\Rightarrow$ Succinct Functional Encryption for low-degree function
$\Rightarrow$ iO

Open Problem 5: break it.

Open Problem 6: if not, build iO from it directly.

The efficiency of GGH15



Eval(0110)

$= A_0 D_{1,0} D_{2,1} D_{3,1} D_{4,0}$

$= (s_{1,0} A_1 + E_{1,0}) D_{2,1} D_{3,1} D_{4,0}$

$= s_{1,0} A_1 D_{2,1} D_{3,1} D_{4,0}$ + "small"

$= s_{1,0}(s_{2,1} A_2 + E_{2,1}) D_{3,1} D_{4,0}$ + "small"

$= s_{1,0} s_{2,1} A_2 D_{3,1} D_{4,0}$ + "still small"

$= s_{1,0} s_{2,1} s_{3,1} A_3 D_{4,0}$ + "still smallish"

$= s_{1,0} s_{2,1} s_{3,1} s_{4,0} A_4$ + "small"

The "small" noise grows exponentially with #levels, becomes a problem in the efficiency.

Private Constrained PRFs

↑

Multilinear maps

GGH13, CLT13, GGH15

↓

Lockable Obfuscation

(Compute-then-Compare obf.)

Open Problem 7: construct PCPRF or LO based on GGH13 or CLT13, prove security from a concrete assumption, like NTRU or approx-gcd.

Likely to give new insights on GGH13 and CLT13, and improve efficiency.

SIMONS INSTITUTE for the Theory of Computing

LWE => iO = $100

#7 with further investigation

The Last Open Problems

Search

HOME ABOUT PEOPLE PROGRAMS & EVENTS VISITING SUPPORT WATCH / READ CALENDAR

‹ Home

Open Problems, Cryptography, Summer 2015

Below is a list of open problems proposed during the Cryptography program at the Simons Institute for the Theory of Computing, compiled by Ron Rothblum and Alessandra Scafuro. Each problem comes with a symbolic cash prize.

1. One-way permutations from a worst-case lattice assumption ($100 from Vinod Vaikuntanathan).
2. Non-interactive zero-knowledge (NIZK) proofs (or even arguments) for NP from LWE ($100 from Vinod Vaikuntanathan).
3. iO from LWE ($100 from Amit Sahai). This result would also solve problems (1) and (2). For (1) see construction and limitations and for (2) see argument system and proof system.
4. Interactive proofs for languages computable in DTISP(t,s) (time t and space s), where the prover runs in time poly(t) and the verifier runs in time poly(s). The provers in known proofs of IP = PSPACE run in time exponential in $2^{poly(s)}$ or $2^{O(s)}$ ($100 from Yael Kalai).
5. $20 per broken password challenge (from Jeremiah Blocki).
6. (Dis)prove that scrypt requires amortized (space × time) = $\Omega(n^2/polylog(n))$ per evaluation on a parallel machine ($100 from Joël).
7. A 3-linear map with unique encoding (i.e., without noise) for which “discrete log” is “plausibly hard” ($1000 from Dan Boneh).
8. SZK = PZK, or in other words, transform any statistical zero-knowledge proof (SZK) into a perfect zero-knowledge proof (PZK) ($100 from Shafi Goldwasser).

Update: During the talk, Amit raised the award for “iO from LWE” to $1000.

THE END. THANKS

Happy lunar new year!