



Constraint hiding constrained PRF for NC1 from LWE

Ran Canetti, Yilei Chen, from Boston University



Constraint hiding constrained PRF for NC1 from LWE

Ran Canetti, Yilei Chen, from Boston University



Once upon a time, a Swede, a Dane, and a Norwegian found themselves on a small island.



There's a cannibal tribe on the island. They imprison the three man. Each of the three man is allowed to make a final wish.



Wife

Norwegian: I want to meet my wife.



Norwegian: I want to meet my wife.
The cannibals agree. Finally they eat the Norwegian and turn
his skin into a canoe.

Cigarette



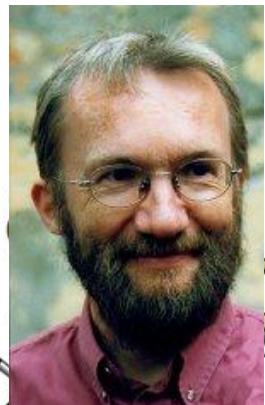
Swede: I want to have another cigarette.



Swede: I want to have another cigarette.
The cannibals agree. Finally they eat the Swede and turn his skin into a canoe.

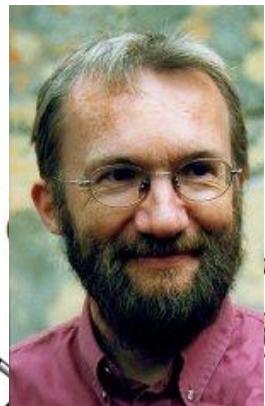


Dane: ...



Puncturable
PRF!

Dane: I want a puncturable PRF!



yay!

Dane: I want a puncturable PRF!
Then you cannot turning my skin into a canoe!!!!!!

Puncturable/constrained PRF

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

K

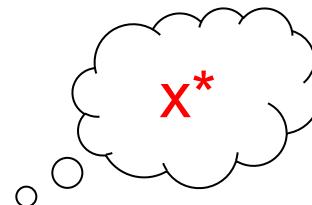
original key

Puncturable/constrained PRF

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

K

original key



$$F_{K\{x^*\}}(x) = \begin{cases} ? , & \text{if } x=x^* \\ F_K(x), & \text{else} \end{cases}$$

punctured key

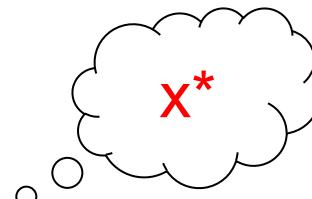
$\text{Puncture}(K, x^*) \Rightarrow K\{x^*\}$ s.t. $F_k(x^*)$ is pseudorandom, give the $K\{x^*\}$ that preserve the original outputs elsewhere.

Puncturable/constrained PRF

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

K

original key



$$F_{K\{x^*\}}(x) = \begin{cases} ? , & \text{if } x=x^* \\ F_K(x), & \text{else} \end{cases}$$

punctured key

$\text{Puncture}(K, x^*) \Rightarrow K\{x^*\}$ s.t. $F_k(x^*)$ is pseudorandom, give the $K\{x^*\}$ that preserve the original outputs elsewhere.

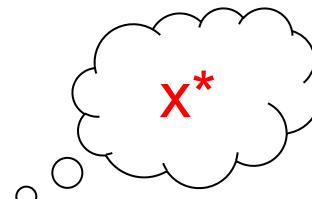
In general: $\text{Constrain}(K, C) \Rightarrow K\{C\}$

Puncturable/constrained PRF

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

K

original key



$$F_{K\{x^*\}}(x) = \begin{cases} ? , & \text{if } x=x^* \\ F_K(x), & \text{else} \end{cases}$$

punctured key

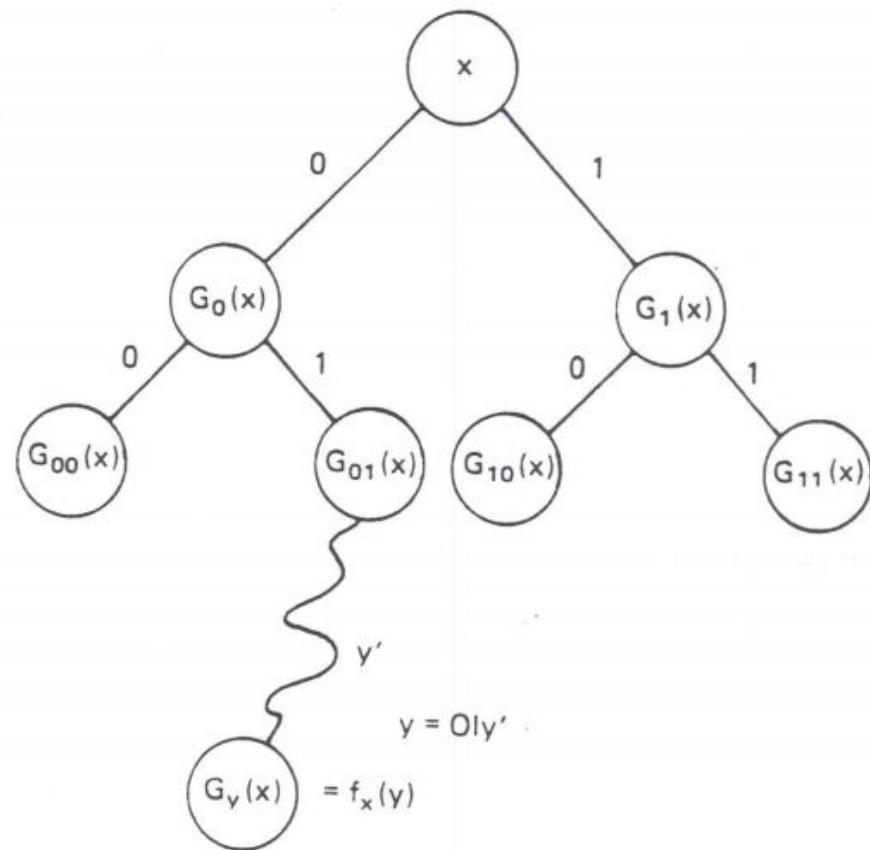
$\text{Puncture}(K, x^*) \Rightarrow K\{x^*\}$ s.t. $F_k(x^*)$ is pseudorandom, give the $K\{x^*\}$ that preserve the original outputs elsewhere.

In general: $\text{Constrain}(K, C) \Rightarrow K\{C\}$

They have many applications (delegate PRF, broadcast encryption, identity-based KE, ...) best known for being good friends of iO

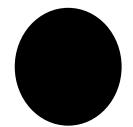
Puncturable PRF from GGM

[Goldreich, Goldwasser, Micali 84]

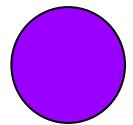


Puncturable PRF from GGM

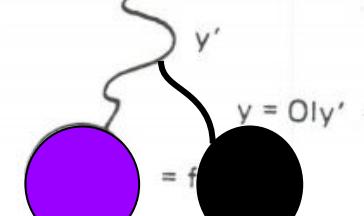
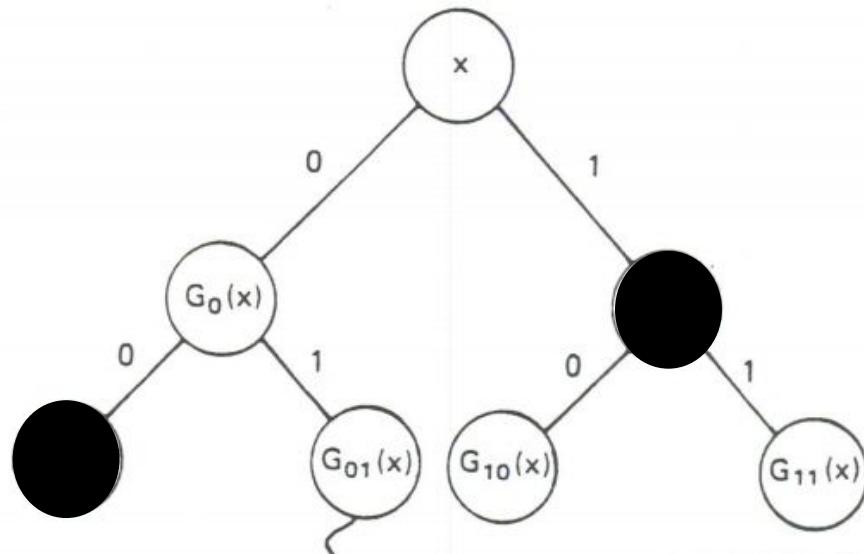
[Goldreich, Goldwasser, Micali 84]



original

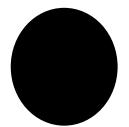


fresh random

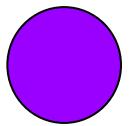


Puncturable PRF from GGM

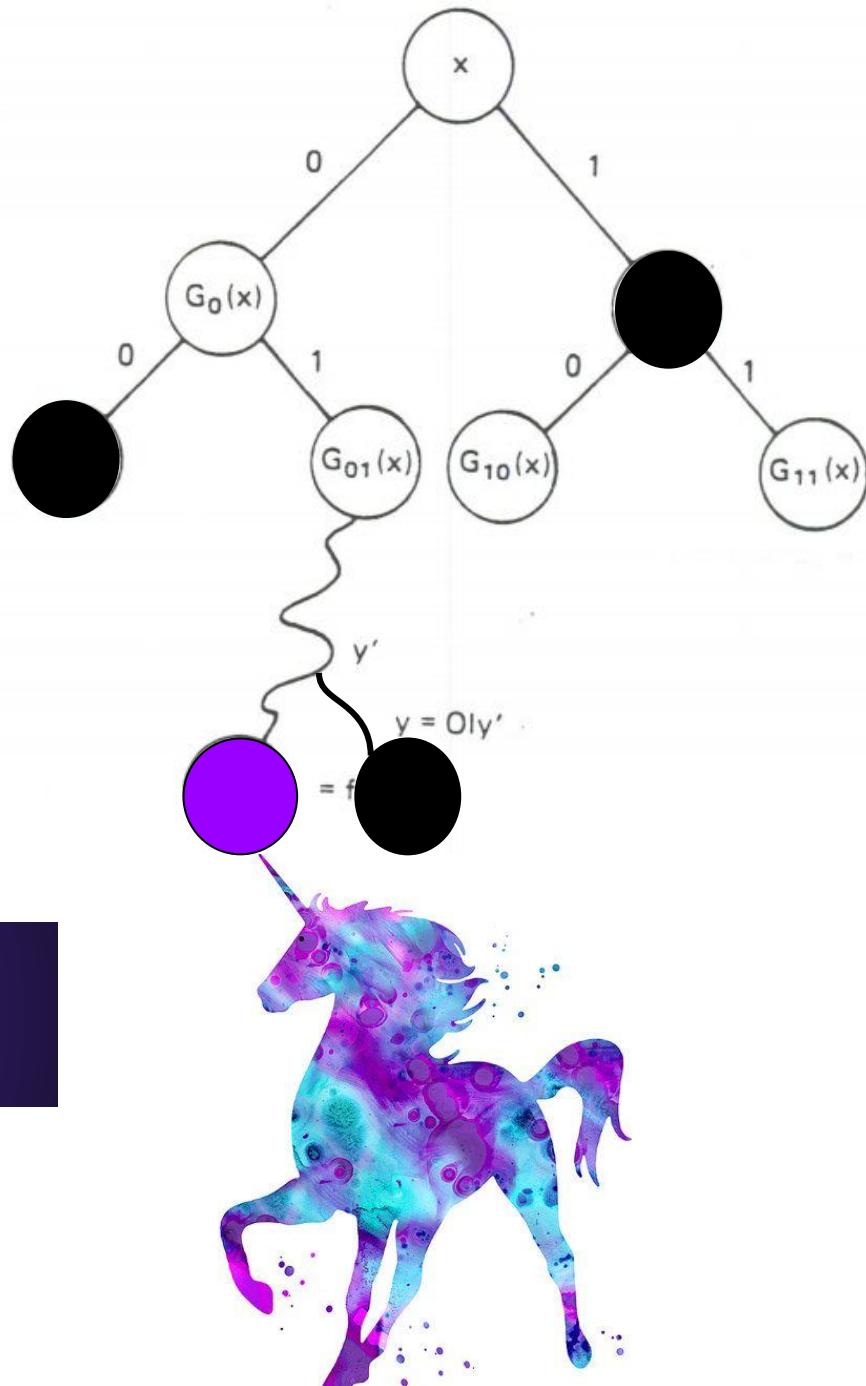
[Goldreich, Goldwasser, Micali 84]



original



fresh random



The constrained key reveals
the point x

What about hiding
the constraint?





yay!

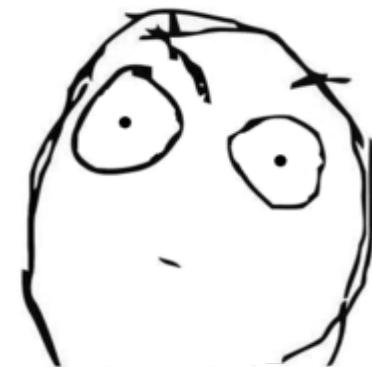
Dane: I want a puncturable PRF!
Then you cannot turning my skin into a canoe!!!!!!



yay!

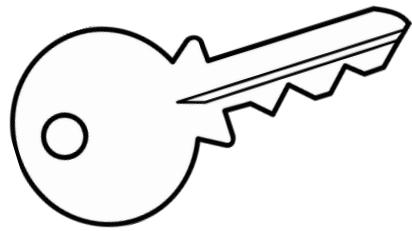
Dane: I want a puncturable PRF!
Then you cannot turning my skin into a canoe!!!!!!
YOU DON'T EVEN KNOW HOW I PUNCTURED

Some motivation scenario:
“Tricky” encryption key

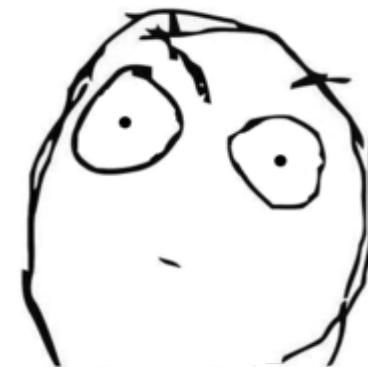


Some 3-letter agent

Some motivation scenario: “Tricky” encryption key

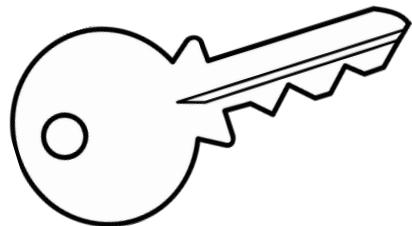


full key

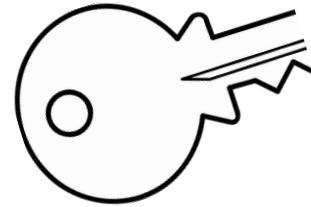


Some 3-letter agent

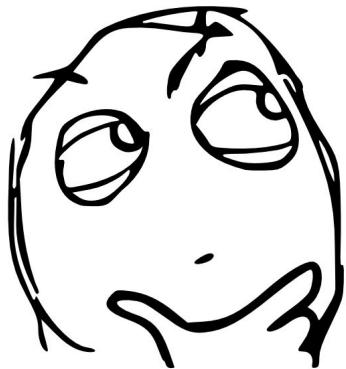
Some motivation scenario: “Tricky” encryption key



full key



corrupted key
(changed on some values)



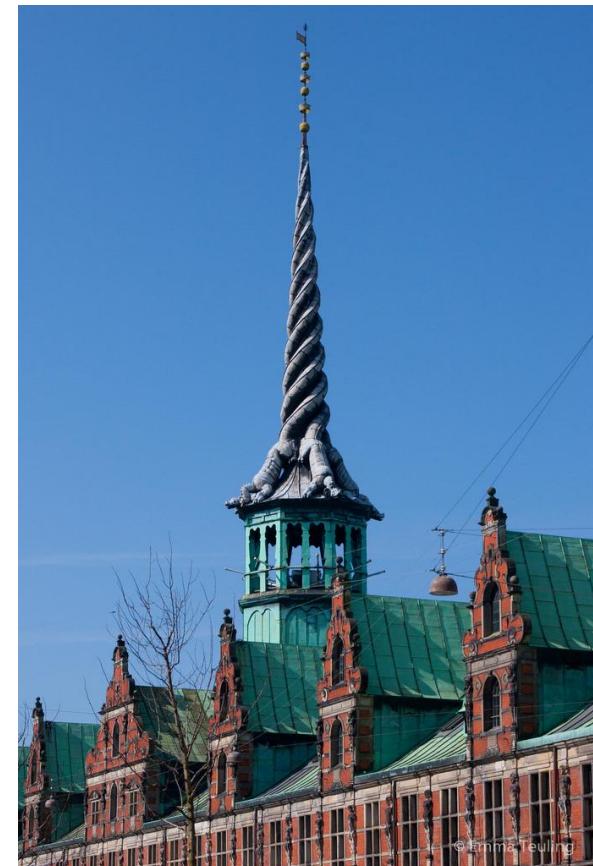
Some 3-letter agent

Boneh, Lewi, Wu (PKC17, eprint 2015/1167)

What

Where

How

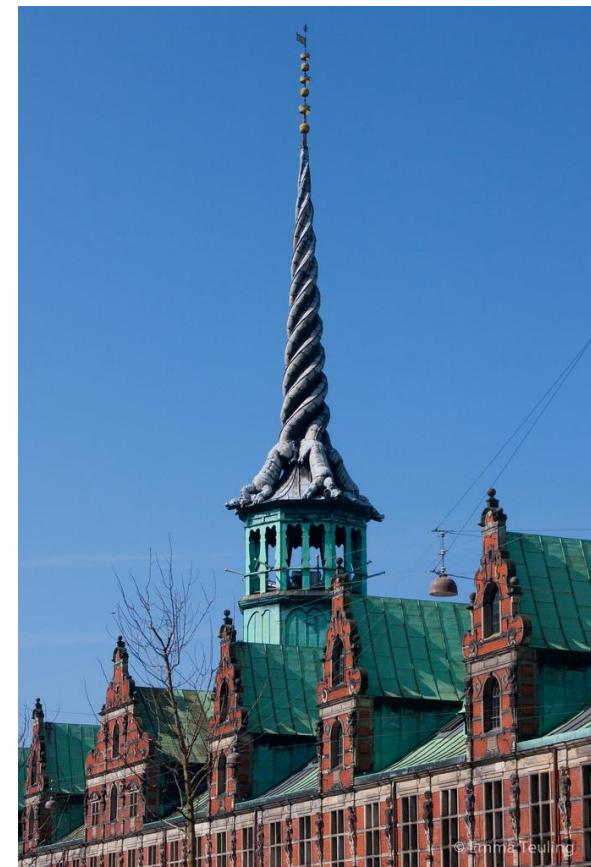


Boneh, Lewi, Wu (PKC17, eprint 2015/1167)

What are Constraint-Hiding CPRFs:
an indistinguishability-based definition

Where

How



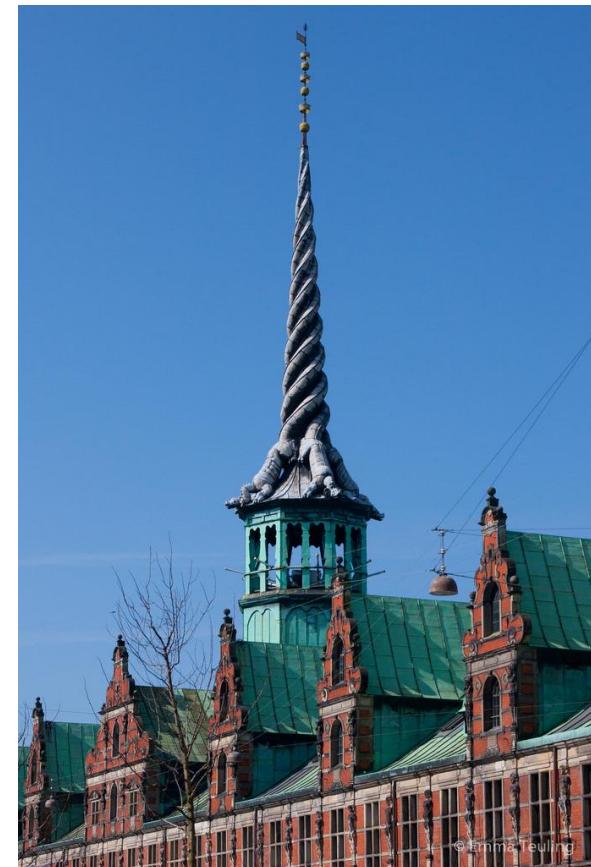
Boneh, Lewi, Wu (PKC17, eprint 2015/1167)

What are Constraint-Hiding CPRFs:
an indistinguishability-based definition

Where to find them (secure for many keys):

- iO(PPRF) is CHCPRF
- Can achieve bit-fixing, puncturing under multilinear DDH, subgroup-hiding

How



Boneh, Lewi, Wu (PKC17, eprint 2015/1167)

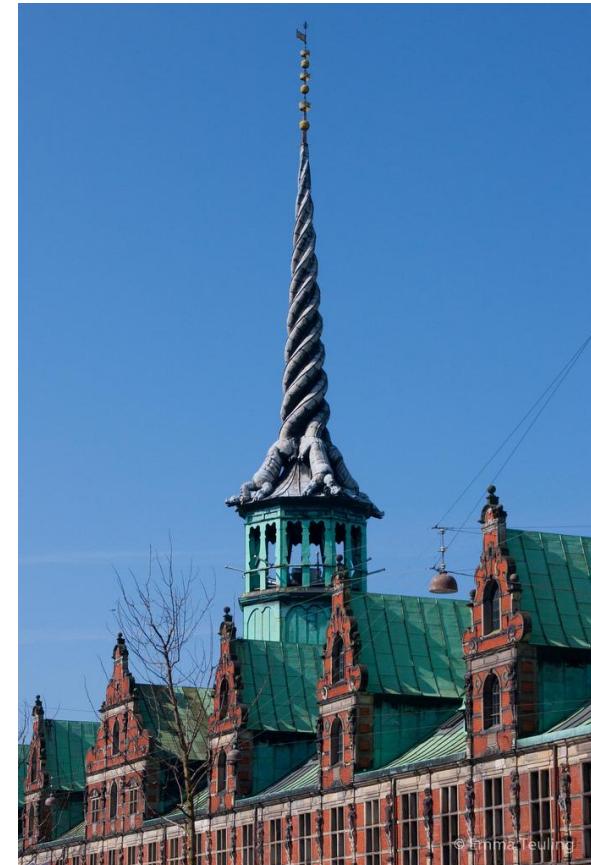
What are Constraint-Hiding CPRFs:
an indistinguishability-based definition

Where to find them (secure for many keys):

- iO(PPRF) is CHCPRF
- Can achieve bit-fixing, puncturing under multilinear DDH, subgroup-hiding

How to use them:

Private-key deniable encryption,
Privately-detectable watermarking,
Searchable encryption

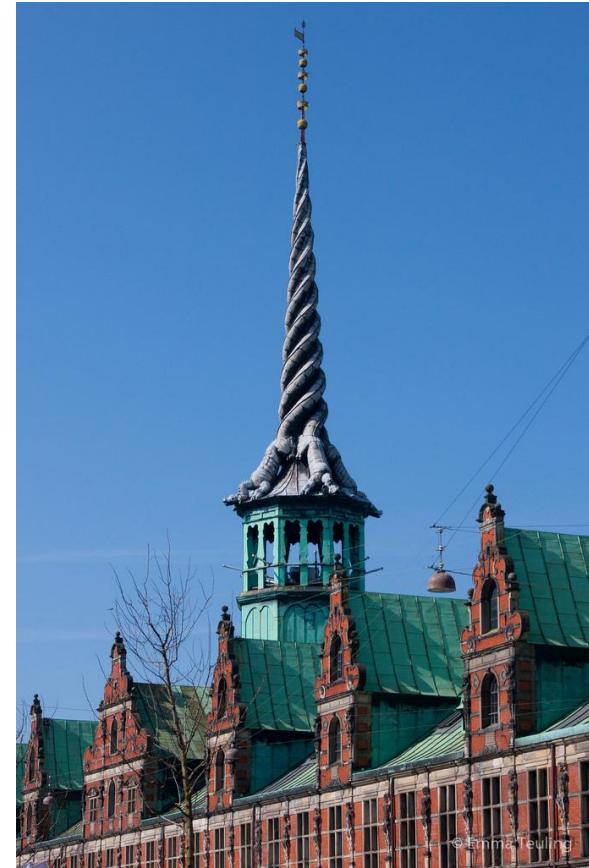


This talk:
Canetti, Chen (Eurocrypt17)

What

Where

How

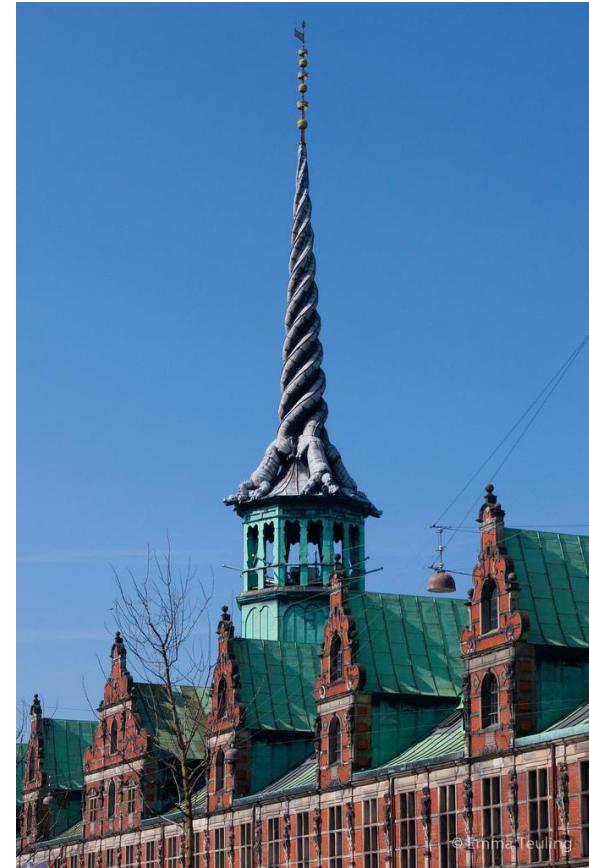


This talk:
Canetti, Chen (Eurocrypt17)

What are Constraint-Hiding CPRFs:
A simulation-based definition of CHCPRF

Where

How

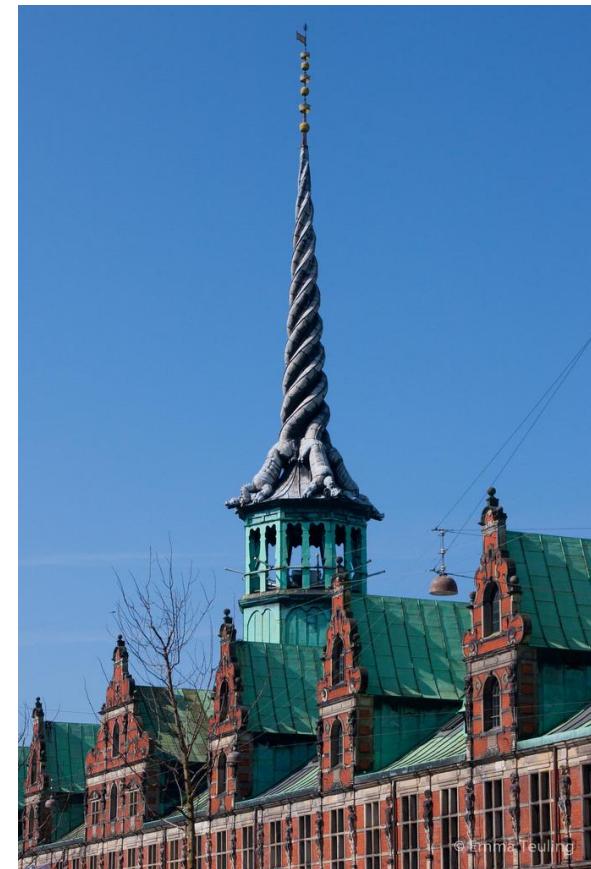


This talk:
Canetti, Chen (Eurocrypt17)

What are Constraint-Hiding CPRFs:
A simulation-based definition of CHCPRF

Where to find them:
Simulation-based 1-key CHCPRFs for NC1 from
Learning With Errors

How



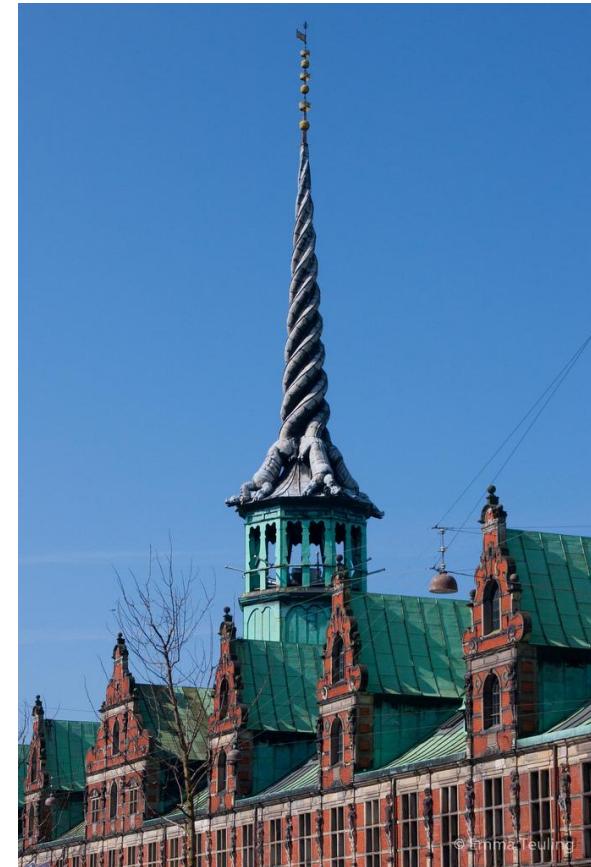
This talk:
Canetti, Chen (Eurocrypt17)

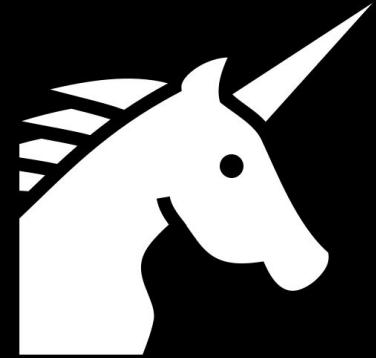
What are Constraint-Hiding CPRFs:
A simulation-based definition of CHCPRF

Where to find them:
Simulation-based 1-key CHCPRFs for NC1 from
Learning With Errors

How to use them:

- 1-key CHCPRF implies 1-key private-key functional encryption (reusable garbled circuits)
- 2-key CHCPRF implies obfuscation*





Plan for the talk:

Part 1: Definition, relation to obfuscation, functional encryption

Part 2: How to construct CHCPRFs, more on GGH15 mmaps

Defining constraint-hiding constraint PRF (CHCPRF)



Master_KeyGen -> MSK

definition of CHCPRF

Master_KeyGen -> MSK

Cons(MSK, C) -> K[C]

definition of CHCPRF

Master_KeyGen -> MSK

Cons(MSK, C) -> K[C]

Eval(K, x) -> F_K(x)

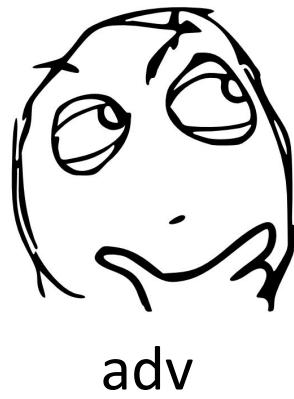
definition of CHCPRF

Simulation-based CHCPRF [CC 17]

for all p.p.t. adv, there's a simulator, such that the outputs of the real and simulated distributions are indistinguishable.



Real



Simulator

Master_KeyGen -> MSK

Cons(MSK, C) -> K[C]

Eval(K, x) -> F_K(x)

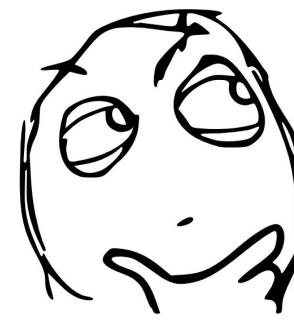
MSK

Master KeyGen

MSK^S



Real



adv



Simulator

Simulation-based definition of CHCPRF

Master_KeyGen \rightarrow MSK

Cons(MSK, C) \rightarrow K[C]

Eval(K, x) \rightarrow F_K(x)

MSK

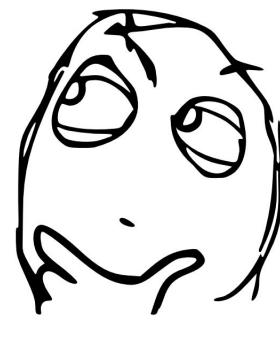
Master KeyGen

MSK^S

Constraint query C



Real



Simulator

Simulation-based definition of CHCPRF

Master_KeyGen \rightarrow MSK

Cons(MSK, C) \rightarrow K[C]

Eval(K, x) \rightarrow F_K(x)

MSK

Master KeyGen

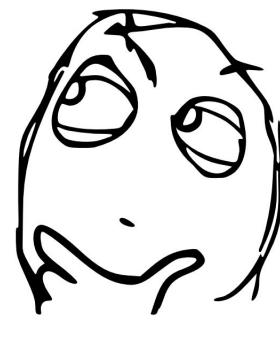
MSK^S

Cons(MSK, C) \rightarrow K[C]

Constraint query C



Real



adv



Simulator

Simulation-based definition of CHCPRF

Master_KeyGen -> MSK

Cons(MSK, C) -> K[C]

Eval(K, x) -> F_K(x)

MSK

Master KeyGen

MSK^S

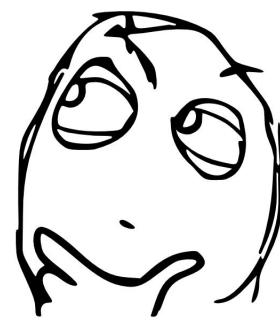
Cons(MSK, C) -> K[C]

Constraint query C

Input query x



Real



adv



Simulator

Simulation-based definition of CHCPRF

Master_KeyGen -> MSK

Cons(MSK, C) -> K[C]

Eval(K, x) -> F_K(x)

MSK

Master KeyGen

MSK^S

Cons(MSK, C) -> K[C]

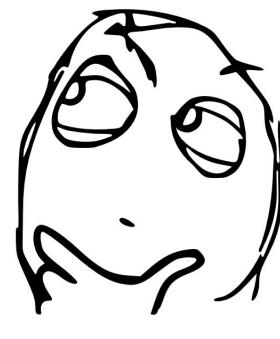
Constraint query C

Eval(MSK, x) -> F_K(x)

Input query x



Real



Simulator

Simulation-based definition of CHCPRF

Master_KeyGen -> MSK

Cons(MSK, C) -> K[C]

Eval(K, x) -> F_K(x)

MSK

Master KeyGen

MSK^S

Cons(MSK, C) -> K[C]

Constraint query C

K^S <- Sim(MSK^S, 1^{|C|})

Eval(MSK, x) -> F_K(x)

Input query x



Real



adv



Simulator

Simulation-based definition of CHCPRF

Master_KeyGen -> MSK

Cons(MSK, C) -> K[C]

Eval(K, x) -> F_K(x)

MSK

Master KeyGen

MSK^S

Cons(MSK, C) -> K[C]

Constraint query C

K^S <- Sim(MSK^S, 1^{|C|})

Eval(MSK, x) -> F_K(x)

Input query x

y^S <- Sim(MSK^S, x, C(x))



Real



adv



Simulator

Simulation-based definition of CHCPRF

Correctness: for x s.t. $C(x)=1$, $\Pr [F_K(x) = F_{K[C]}(x)] > 1-\text{negl.}$

MSK

Master KeyGen

MSK^S

$\text{Cons}(\text{MSK}, C) \rightarrow K[C]$

Constraint query C

$K^S \leftarrow \text{Sim}(MSK^S, 1^{|C|})$

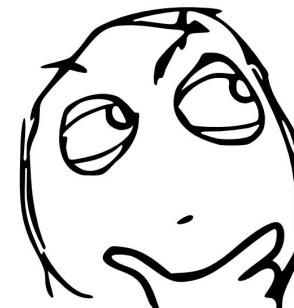
$\text{Eval}(\text{MSK}, x) \rightarrow F_K(x)$

Input query x

$y^S \leftarrow \text{Sim}(MSK^S, x, C(x))$



Real



adv



Simulator

Simulation-based definition of CHCPRF

Correctness: for x s.t. $C(x)=1$, $\Pr [F_K(x) = F_{K[C]}(x)] > 1-\text{negl}.$

Pseudorandom & Constraint-hiding:

$$K[C], F_K(x) \approx_c K^S, y^S \quad (\text{when } C(x)=0, y^S \text{ is from random})$$

MSK

Master KeyGen

MSK^S

Cons(MSK, C) -> K[C]

Constraint query C

$K^S \leftarrow \text{Sim}(\text{MSK}^S, 1^{|C|})$

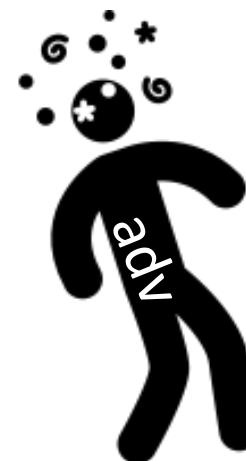
Eval(MSK, x) -> F_K(x)

Input query x

$y^S \leftarrow \text{Sim}(\text{MSK}^S, x, C(x))$



Real



Simulator

Simulation-based definition of CHCPRF

Theorem [CC17]

For 1-constrained key in the selective setting
sim-based = ind-based



Sim-based definition for **many**
constrained keys



Real



Simulator

Correctness: for x s.t. $C(x)=1$, $\Pr [F_K(x) = F_{K[C]}(x)] > 1-\text{negl}.$

Constraint-hiding: $K[C_1], K[C_2] \approx_c K_1^S, K_2^S$

MSK

Master KeyGen

MSK^S

$\text{Cons}(\text{MSK}, C_1) \rightarrow K[C_1]$

Constraint query C_1

$K_1^S \leftarrow \text{Sim}(MSK^S, 1^{|C|})$

$\text{Cons}(\text{MSK}, C_2) \rightarrow K[C_2]$

Constraint query C_2

$K_2^S \leftarrow \text{Sim}(MSK^S, 1^{|C|})$



Real



Simulator

Sim-based definition for many keys

Correctness: for x s.t. $C(x)=1$, $\Pr [F_K(x) = F_{K[C]}(x)] > 1-\text{negl}.$

Constraint-hiding:

$$K[C_1], K[C_2] \approx_c K_1^S, K_2^S$$

MSK

Master KeyGen

MSK^S

$$\text{Cons}(\text{MSK}, C_1) \rightarrow K[C_1]$$

$$\text{Constraint query } C_1$$

$$K_1^S \leftarrow \text{Sim}(\text{MSK}^S, 1^{|C|})$$

$$\text{Cons}(\text{MSK}, C_2) \rightarrow K[C_2]$$

$$\text{Constraint query } C_2$$

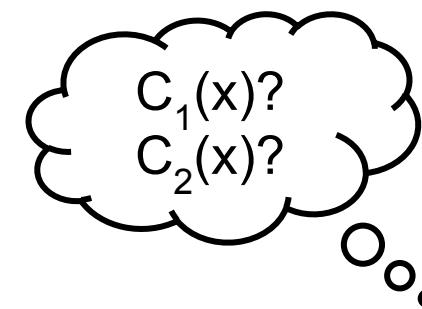
$$K_2^S \leftarrow \text{Sim}(\text{MSK}^S, 1^{|C|})$$



Real



adv



Simulator

Relaxed Sim-based definition for many keys

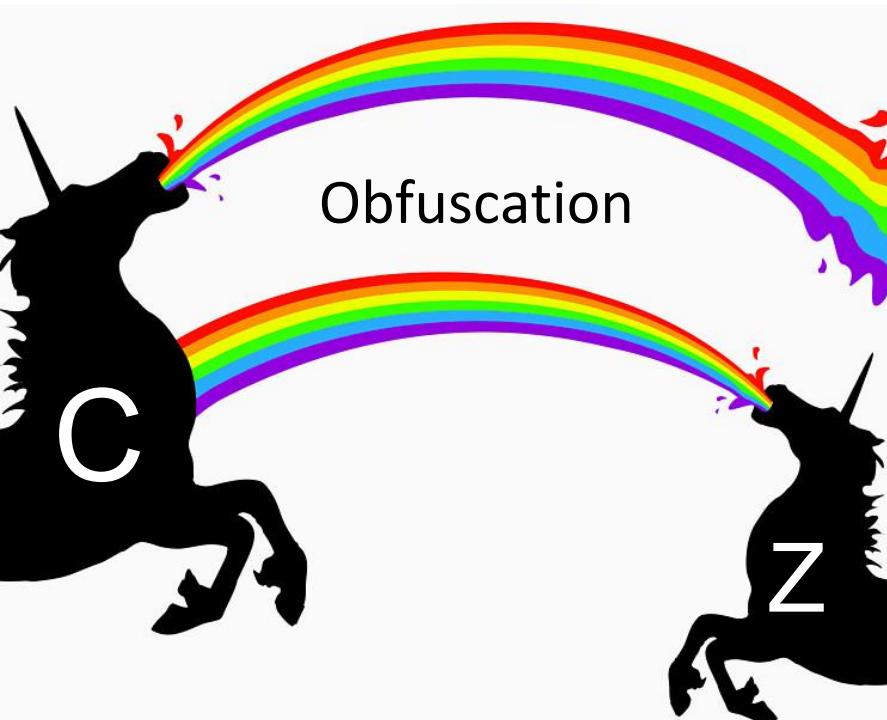


*Hide the program
in the constraint*

Reminiscent of obfuscation ...

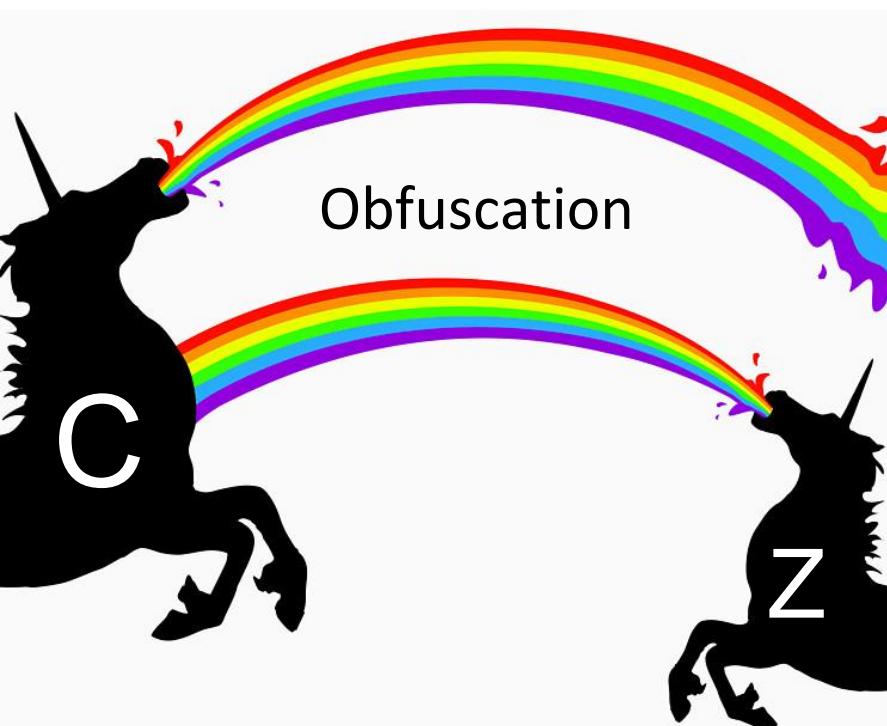
Theorem [CC 17]: Two-key CHCPRF (for function class C) implies obfuscation (for C)

- Two-key relaxed sim-CHCPRF implies strong VBB obfuscation
- Two-key ind-CHCPRF implies iO



Theorem [CC 17]: Two-key CHCPRF (for function class C) implies obfuscation (for C)

- Two-key relaxed sim-CHCPRF implies strong VBB obfuscation
- Two-key ind-CHCPRF implies iO

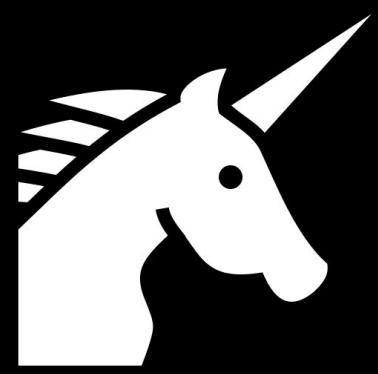


Construction:

$$\text{Obf} = (K[C], K[Z])$$

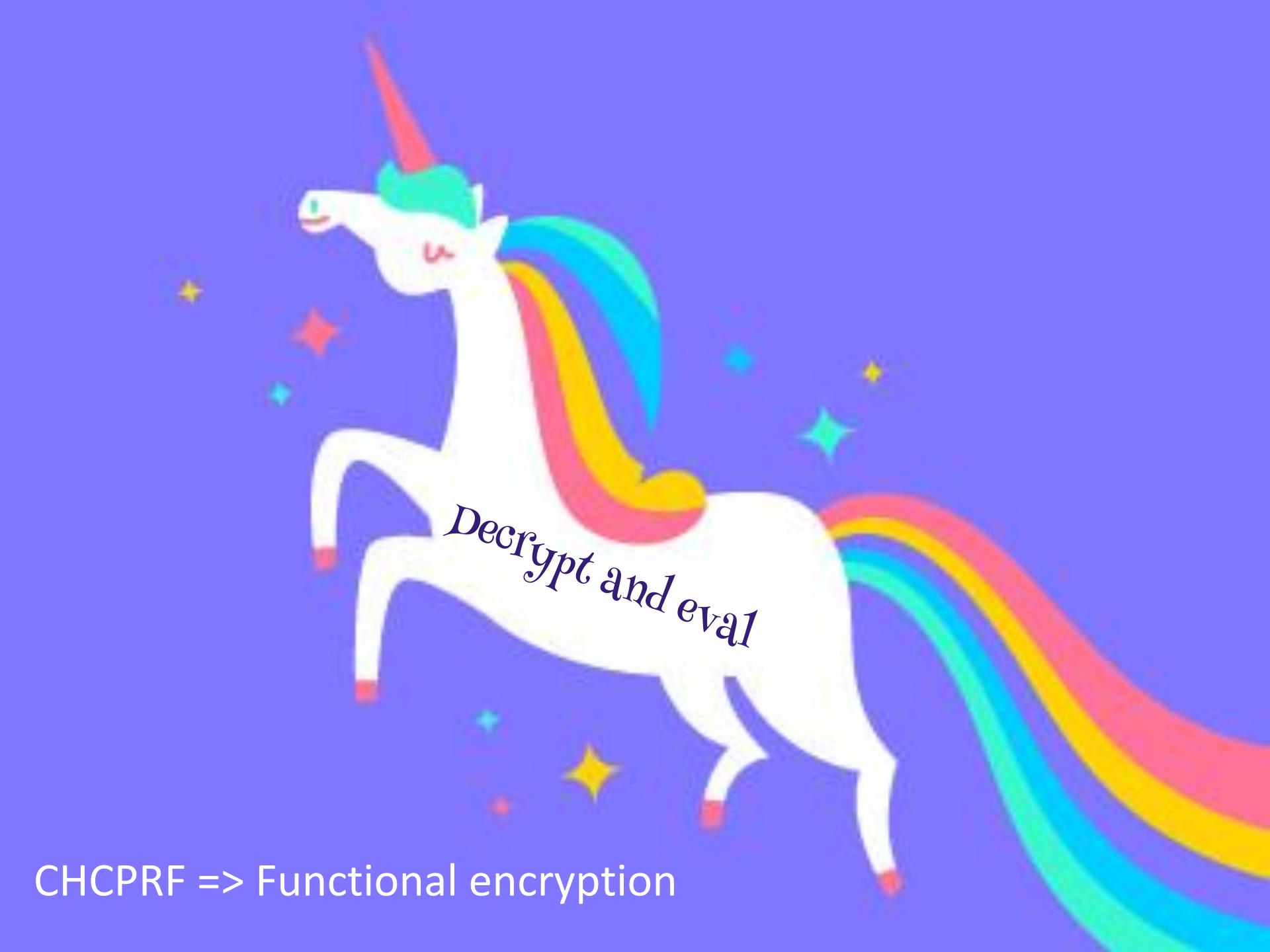
$\text{Eval}(x)$: check consistency
 $\text{Eval}(K[C], x) \stackrel{?}{=} \text{Eval}(K[Z], x)$

Idea implicit from the [GGHRSW13] candidate obfuscation



In the rest of the talk, we will focus on:

1-key simulation-based definition for CHCPRF.



Decrypt and eval

CHCPRF => Functional encryption

Theorem [CC 17] 1-key sim-based CHCPRF implies 1-key private-key functional encryption (reusable garbled circuits).



Theorem [CC 17] 1-key sim-based CHCPRF implies 1-key private-key functional encryption (reusable garbled circuits).

Construction: from normal encryption Sym and CHCPRF E

$\text{Enc}(m;r)$: $\text{ct} = \text{Enc}_{\text{Sym},K}(m;r); \quad \text{tag} = F[K](\text{ct})$

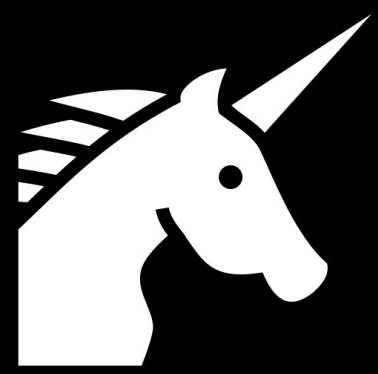
$\text{FSK}[\text{Sym}, K, F.K, C]$: constrained key for the “decryption and eval” functionality $C(\text{Dec}_{\text{Sym},K}(\cdot))$

Eval: compute $F[C(\text{Dec}_{\text{Sym},K}(\cdot))](\text{ct})$, and compare with tag





Where to find it?



Main construction:

1-key sim-based CHCPRFs for NC1 from [Learning With Errors](#), based on the multilinear maps by Gentry, Gorbunov, Halevi (GGH15)

Combine:

- Lattices-based PRFs
- Barrington's theorem to embed functionality
- GGH15 encoding to provide a public constrained mode

Demonstrate a proof methodology of GGH15-based applications.

Short intro to BPR12

[Banerjee, Peikert, Rosen 12]
-- the first LWE-based PRF

Learning with errors

Uniform Small Unspecified

A

$$Y = S \times A + E \pmod{q}$$

Secret coefficient/mask noise/error

A is n-by-m in Z_q^n (n is the lattice dimension, $m > n \log q$)

Search LWE: Given A, $y=sA+E$, find s

Decisional LWE: distinguish y from random

As hard as worst-case approx-SIVP (Quantumly) [Regev 05]
(classically for subexponential q) [Peikert 09, BLPRS 13]

Learning with errors

Uniform Small Unspecified

A

Y

s

A

E

Secret

coefficient/mask

noise/error

$\text{mod } q$

A is n-by-m in Z_q^n (n is the lattice dimension, $m > n \log q$)

Search LWE: Given A, $y=sA+E$, find s

Decisional LWE: distinguish y from random

As hard as worst-case approx-SIVP (Quantumly) [Regev 05]
(classically for subexponential q) [Peikert 09, BLPRS 13]

LWE (with small secrets)

Uniform Small Unspecified

A

Y

s

A

E

Secret

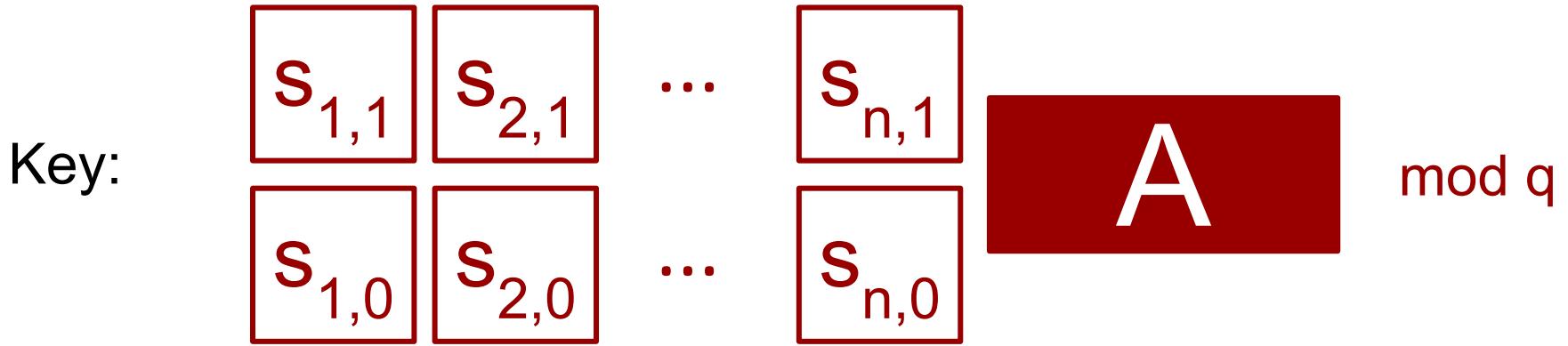
coefficient/mask

noise/error

$\text{mod } q$

Entries of S are small (e.g. from the error distribution)

As hard as normal LWE [Applebaum, Cash, Peikert, Sahai 09]



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

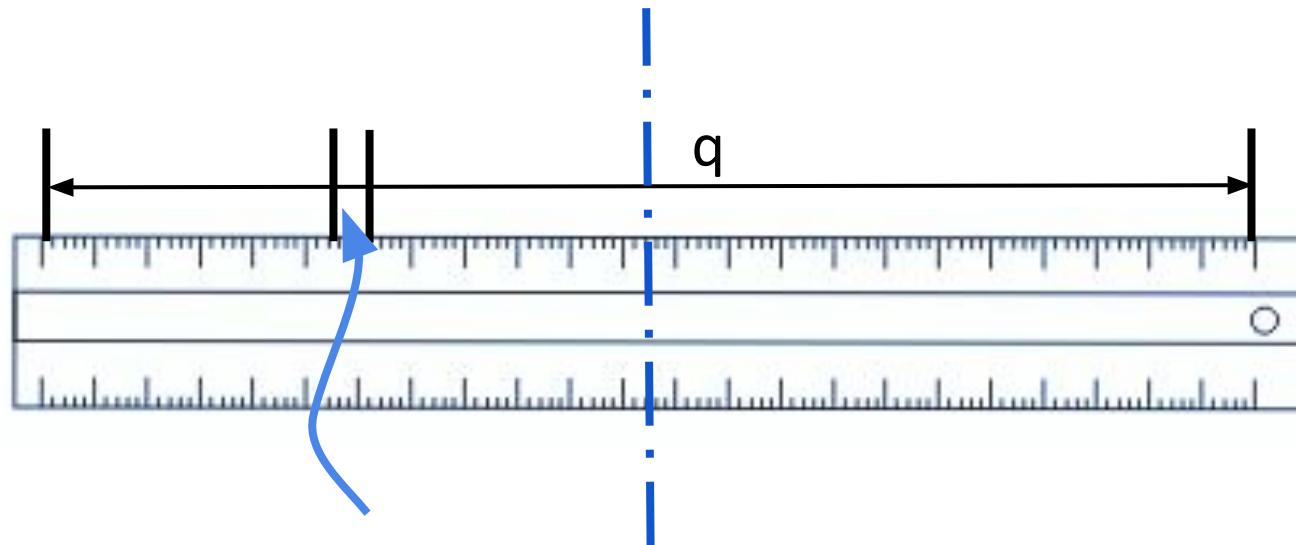
$s_{i,b}$

are LWE secrets from low-norm distributions

Rounding: $\{t\}_p : \mathbb{Z}_q \rightarrow \mathbb{Z}_p$

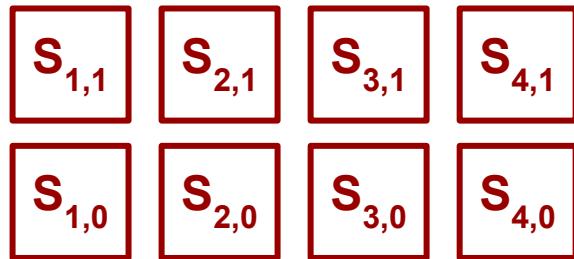
Compute t^*p/q , then round to the nearest integer

In this talk, $p=2$, $q/p > \exp(L)$, $q/p \sim \text{super-polynomial}$



Amount of noise

A is public, S_{i,x_i} are secret

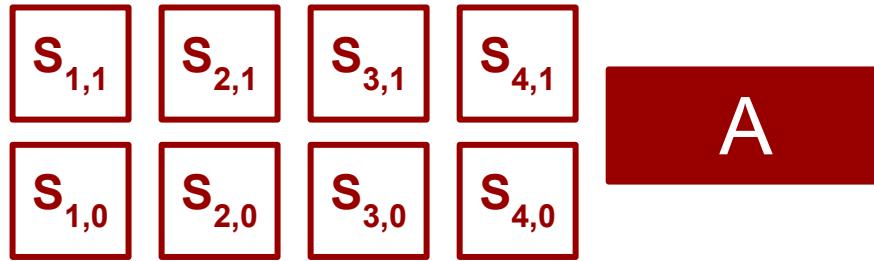


A mod q

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

Main observation: After rounding, can inject noises without changing functionality whp.

A is public, S_{i,x_i} are secret

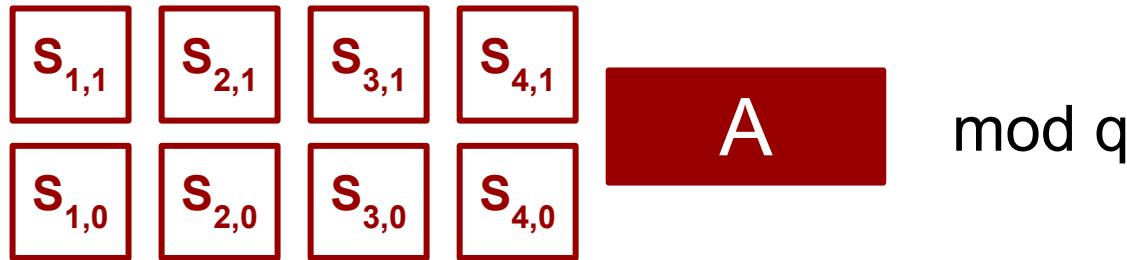


mod q

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

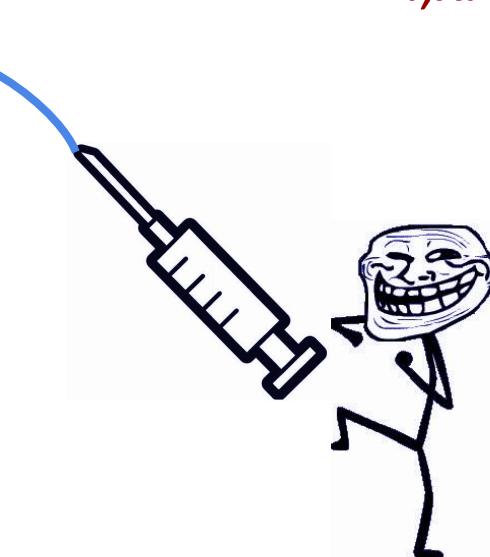
$$F(x) = \{ \prod s_{i,x_i} A \}_2$$

A is public, S_{i,x_i} are secret

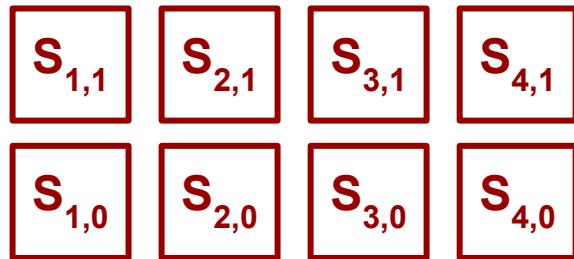


$$\begin{aligned} F(0110) &= \{ s_{1,0} s_{2,1} s_{3,1} s_{4,0} A \}_2 \\ &\approx_s \{ s_{1,0} s_{2,1} s_{3,1} (s_{4,0} A + E_{4,0}) \}_2 \end{aligned}$$

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



A is public, S_{i,x_i} are secret



A

$\mod q$

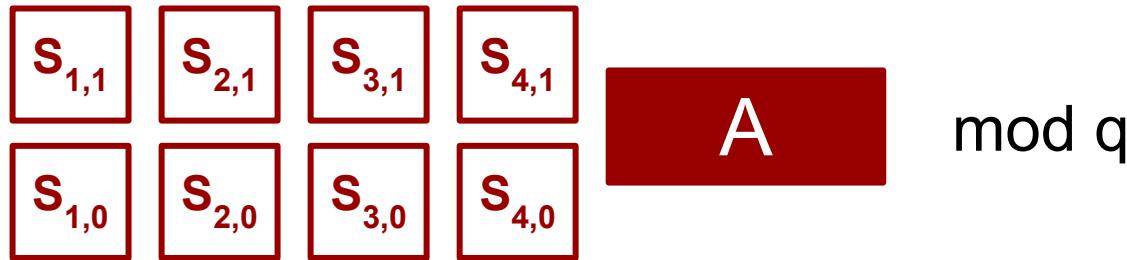
$F(0110)$

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

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



A is public, S_{i,x_i} are secret



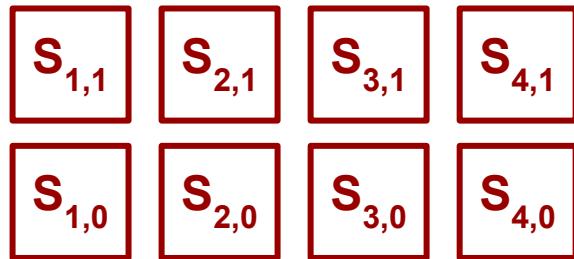
$F(0110)$

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

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



A is public, S_{i,x_i} are secret



A

$\mod q$

$F(0110)$

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

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



Key:

$s_{1,1}$	$s_{2,1}$...	$s_{n,1}$		A	$\text{mod } q$
$s_{1,0}$	$s_{2,0}$...	$s_{n,0}$		A	$\text{mod } q$

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

What we need in addition to build a CHCPRF:

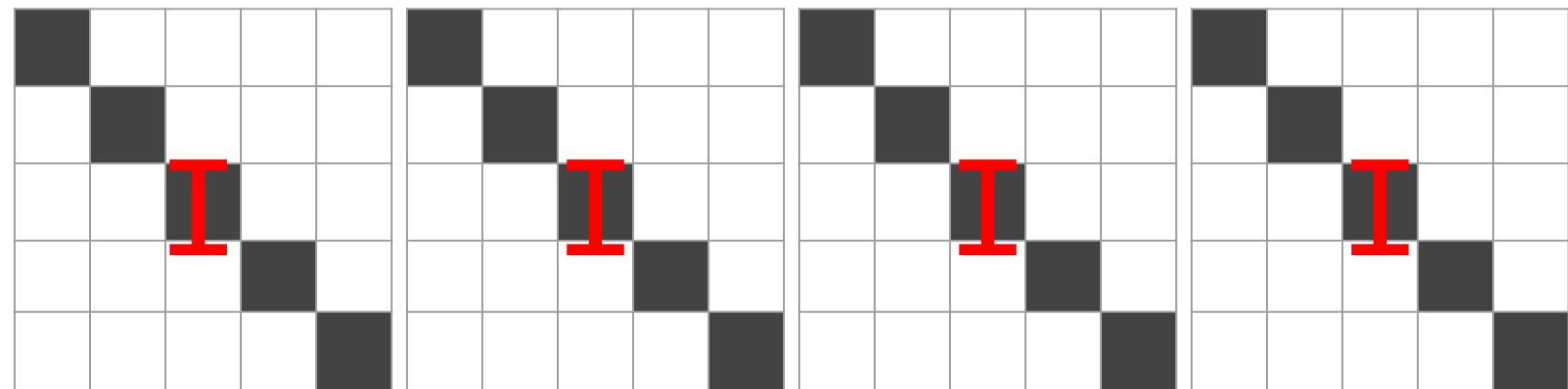
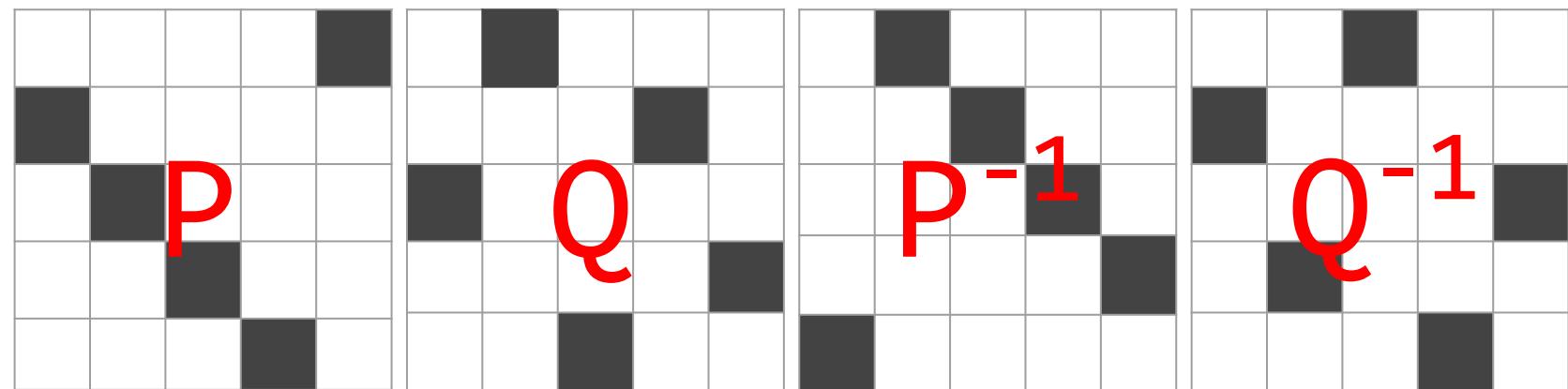
- + Embed **structures** in the secret terms to perform functionality (Barrington's theorem)
- + A proper **public mode** of the function (GGH15 encoding)



Barrington's theorem
(used to embed a circuit into the key)

Barrington 1986: log-depth boolean circuits can be recognized by subset products of permutation matrices of width 5.

Example: how to represent an AND gate



Input wire 1

Input wire 2

Input wire 1

Input wire 2

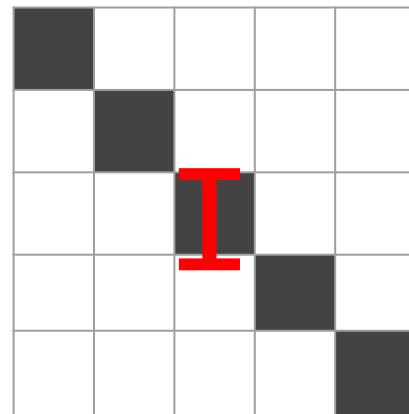


Our construction only work for certain representation of Barrington (e.g. S_5)

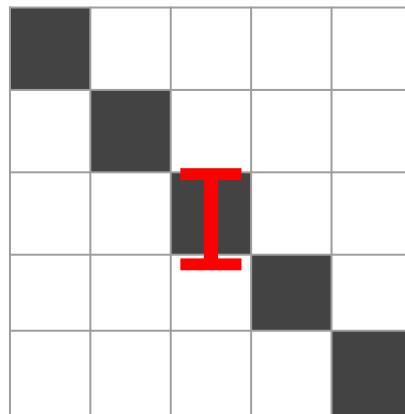
Barrington 1986: log-depth boolean circuits can be recognized by subset products of permutation matrices of width 5.

Example: how to represent an AND gate 0 and 0

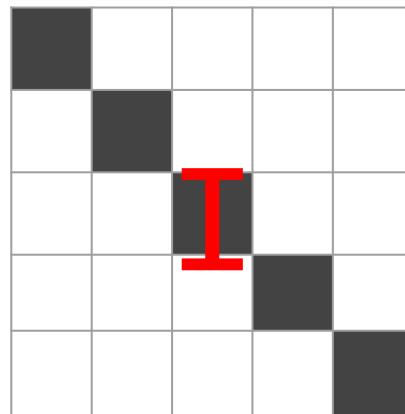
1



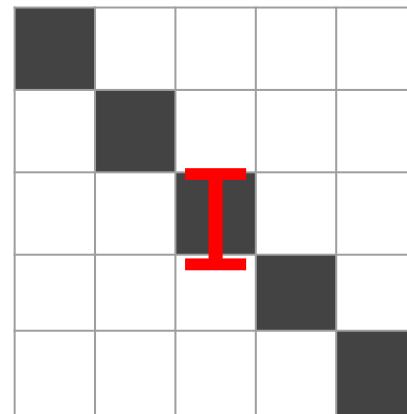
Input wire 1



Input wire 2



Input wire 1



Input wire 2

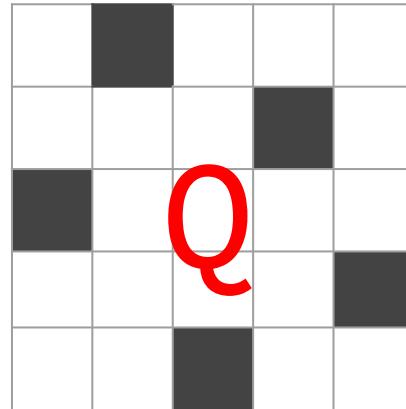


Our construction only work for certain representation of Barrington (e.g. S_5)

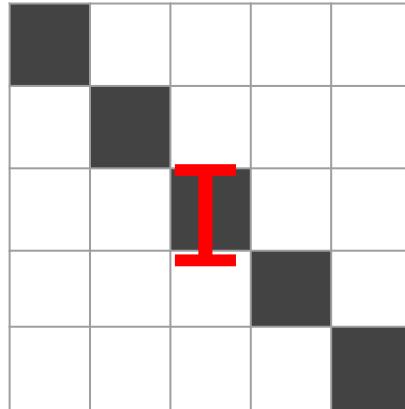
Barrington 1986: log-depth boolean circuits can be recognized by subset products of permutation matrices of width 5.

Example: how to represent an AND gate 0 and 1

1

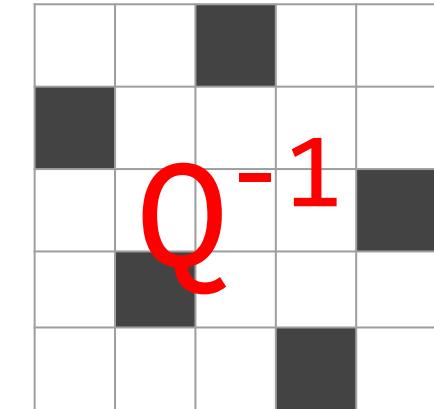


0

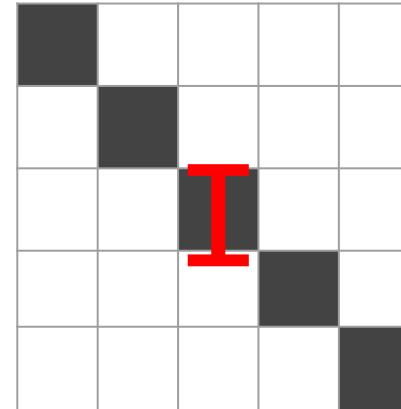


Input wire 1

Input wire 2



Q^{-1}



Input wire 1

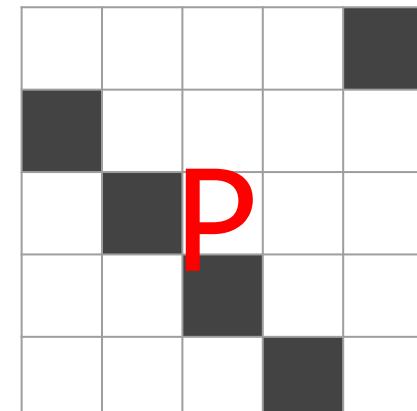
Input wire 2



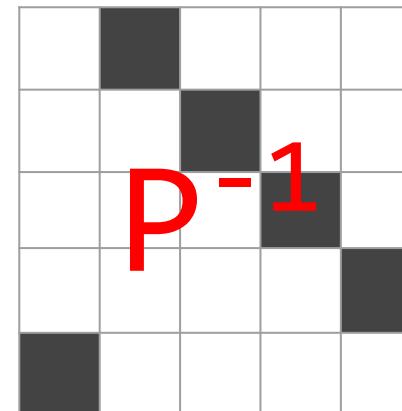
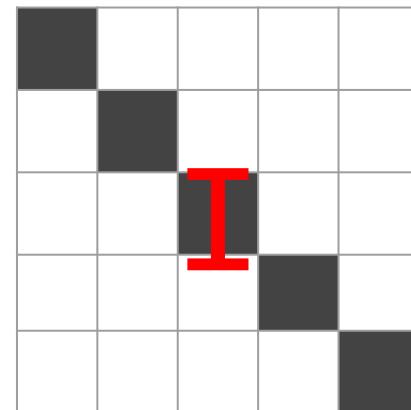
Our construction only work for certain representation of Barrington (e.g. S_5)

Barrington 1986: log-depth boolean circuits can be recognized by subset products of permutation matrices of width 5.

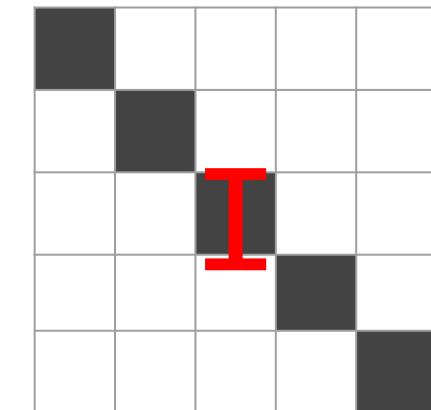
Example: how to represent an AND gate **1 and 0**



1

**P⁻¹**

0



Input wire 1

Input wire 2

Input wire 1

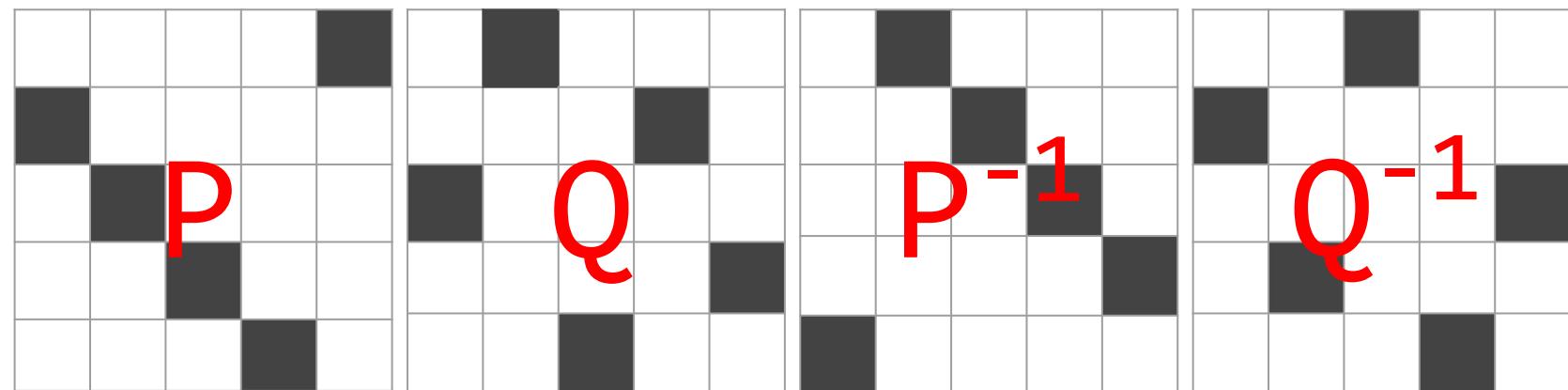
Input wire 2



Our construction only work for certain representation of Barrington (e.g. S_5)

Barrington 1986: log-depth boolean circuits can be recognized by subset products of permutation matrices of width 5.

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



0

Input wire 1

Input wire 2

Input wire 1

Input wire 2



Our construction only work for certain representation of Barrington (e.g. S_5)

Representation of the constraint predicate: branching program

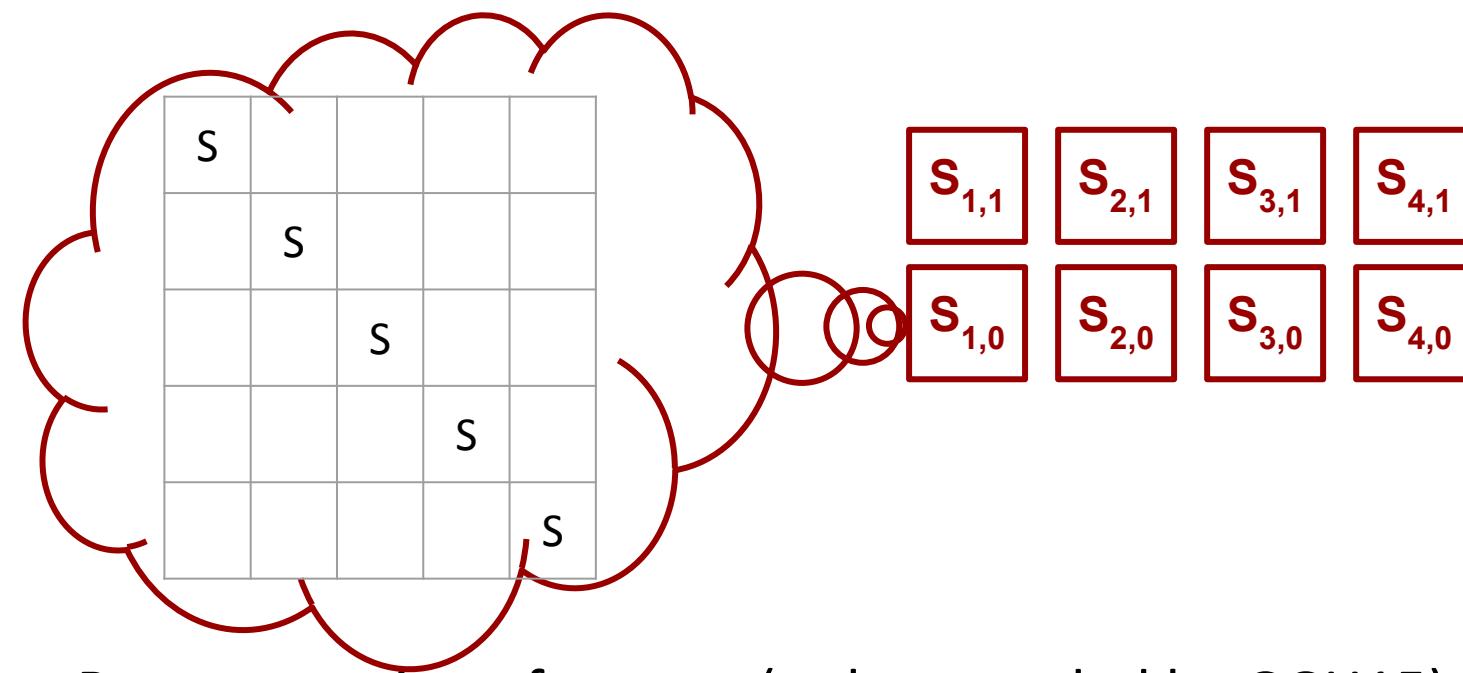
1 $B_{1,1}$ $B_{2,1}$ $B_{3,1}$... $B_{L,1}$

0 $B_{1,0}$ $B_{2,0}$ $B_{3,0}$... $B_{L,0}$

Steps 1 2 3 ... L

Eval: $\prod B_{z(i), x_z(i)} = I \text{ or } C$

We set the secrets like:



Representation of secrets (to be encoded by GGH15): $B_{i,b} \otimes s_{i,b}$

e.g. $I \otimes s =$

S			
	S		
		S	
			S

$P \otimes s =$

				S
S				
	S			
		S		



GGH15 encoding [Gentry, Gorbunov, Halevi 15]

GGH15 “graph-induced multilinear maps”

Multilinear maps perspective:

$$g, g^{S_1}, \dots g^{S_k}, g^{\prod S}$$

Normal hard group for DLOG

$$A, S_1 A + E_1, \dots, S_k A + E_k, \prod S A + E$$

GGH15: (Ring)LWE analogy

The “plaintexts” are encoded in the **secret** terms of LWE

A

with

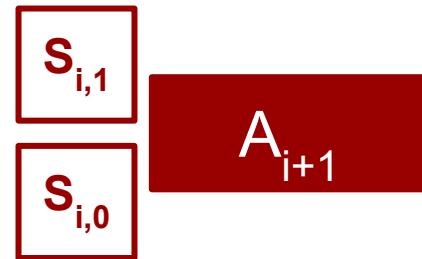


Trapdoor

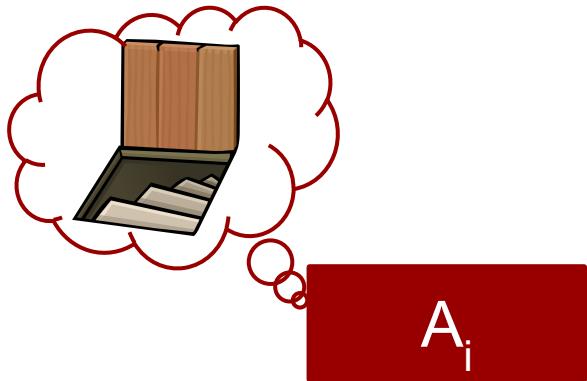
Trapdoor [Ajtai 99, Alwen, Peikert 09, Micciancio, Peikert 12]
Can sample A with a trapdoor T.

Can sample small preimage from **Gaussian** [Klein '00, GPV'08]

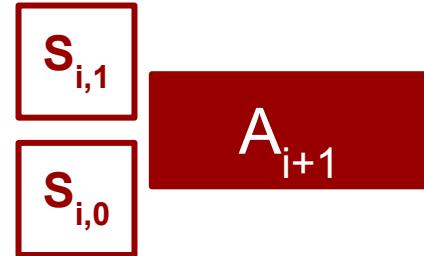
GGH15 encoding for the i^{th} hop:



GGH15 encoding for the i^{th} hop:



$$Y_{i,1} = s_{i,1} A_{i+1} + E_{i,1}$$

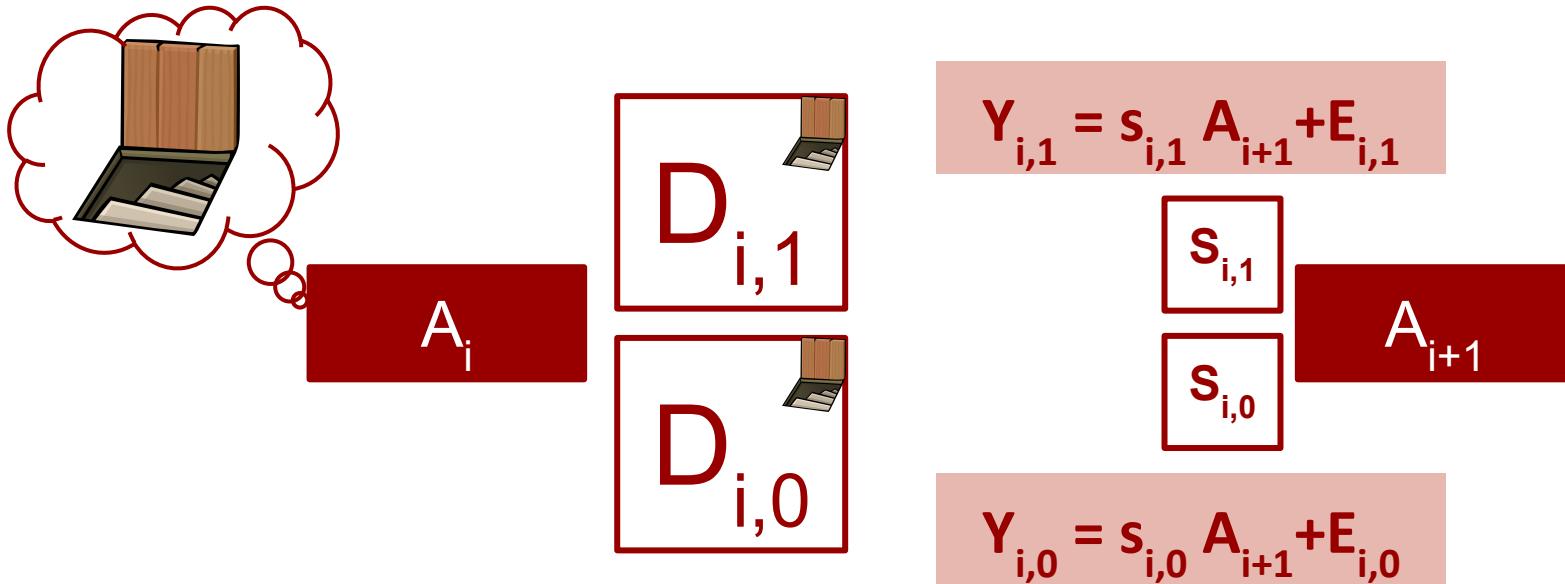


$$Y_{i,0} = s_{i,0} A_{i+1} + E_{i,0}$$

Encode($s_{i,b}$): 2 steps

1. $Y_{i,b} = s_{i,b} A_{i+1} + E_{i,b}$

GGH15 encoding for the i^{th} hop:

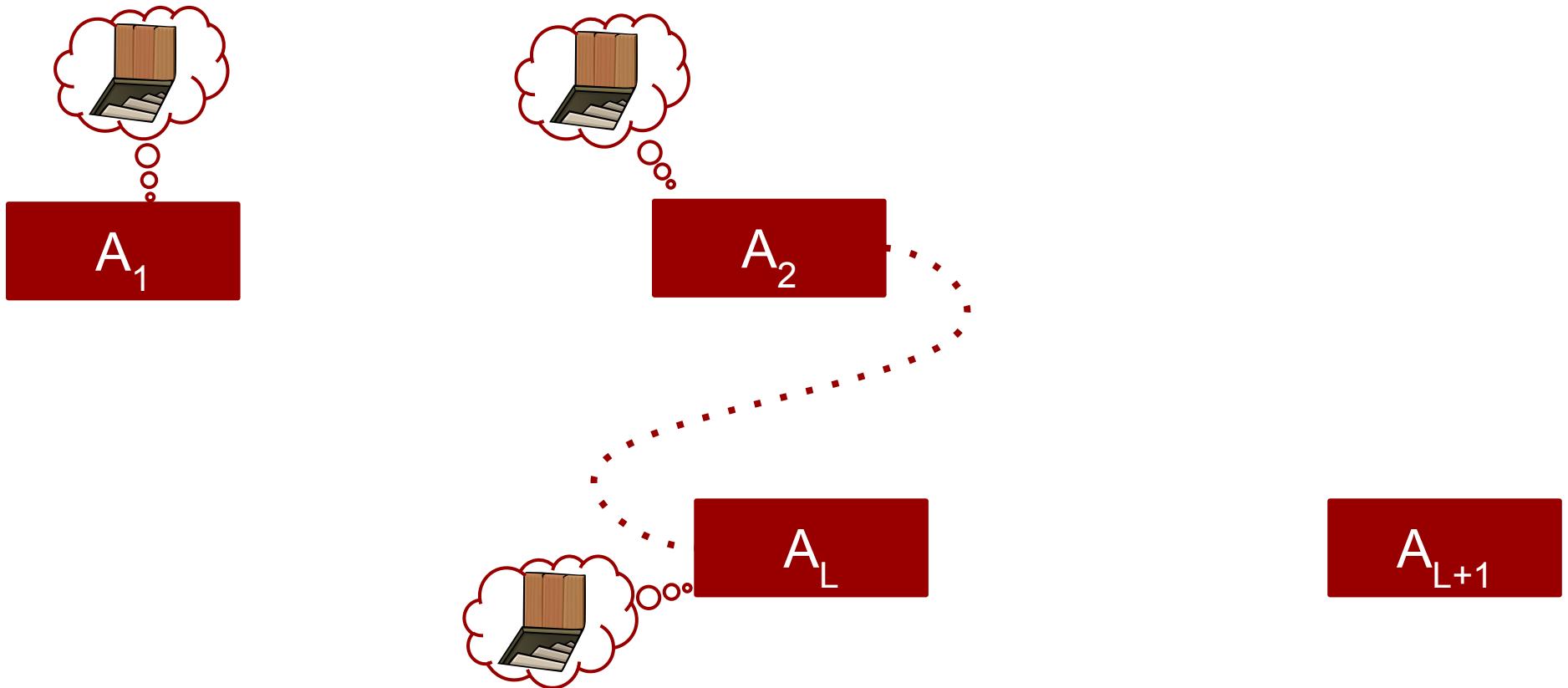


Encode($s_{i,b}$): 2 steps

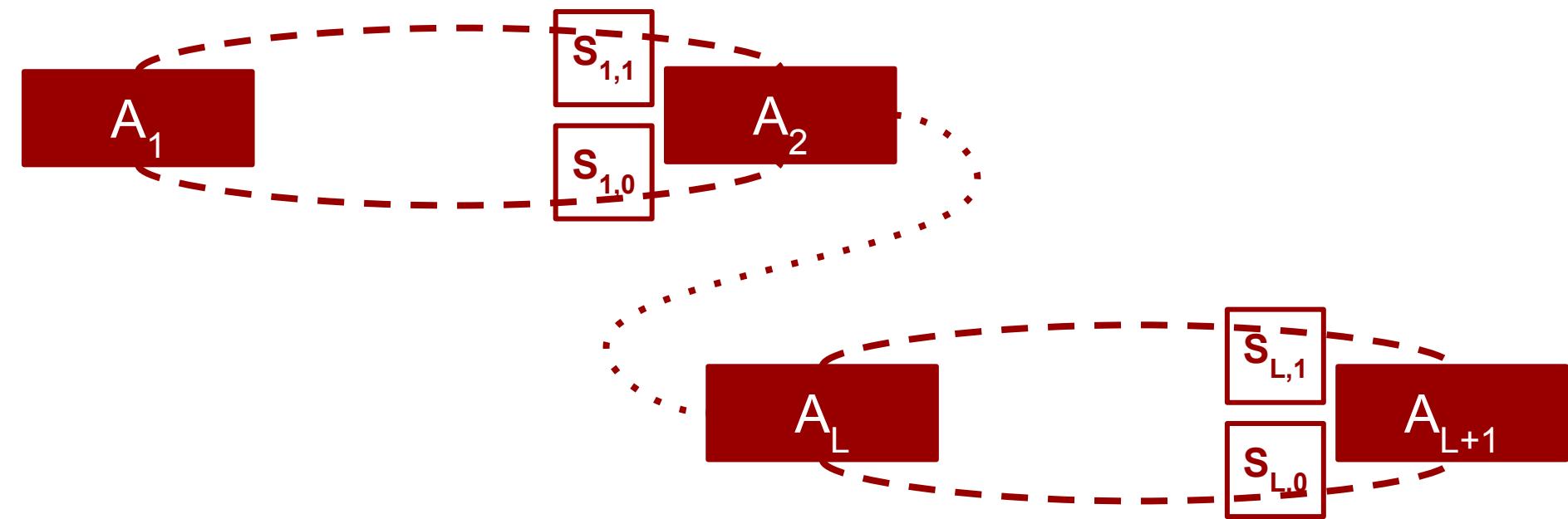
1. $Y_{i,b} = s_{i,b} A_{i+1} + E_{i,b}$
2. Sample (by the trapdoor of A_i) small $D_{i,b}$ s.t. $A_i D_{i,b} = Y_{i,b}$

$$D_{i,b} = \text{Encoding}(s_{i,b})$$

GGH15 for L hops:

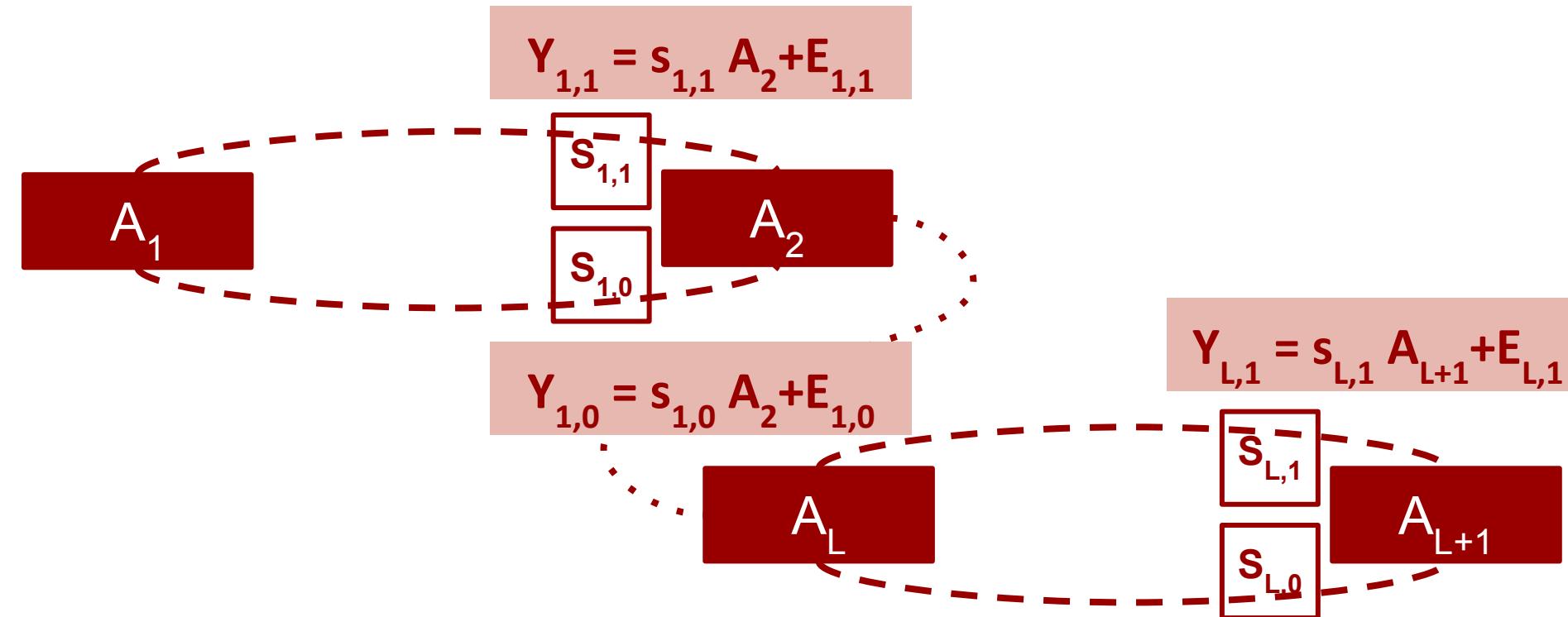


GGH15 for L hops:



$\text{Encode}(s_{i,b})$: 2 steps

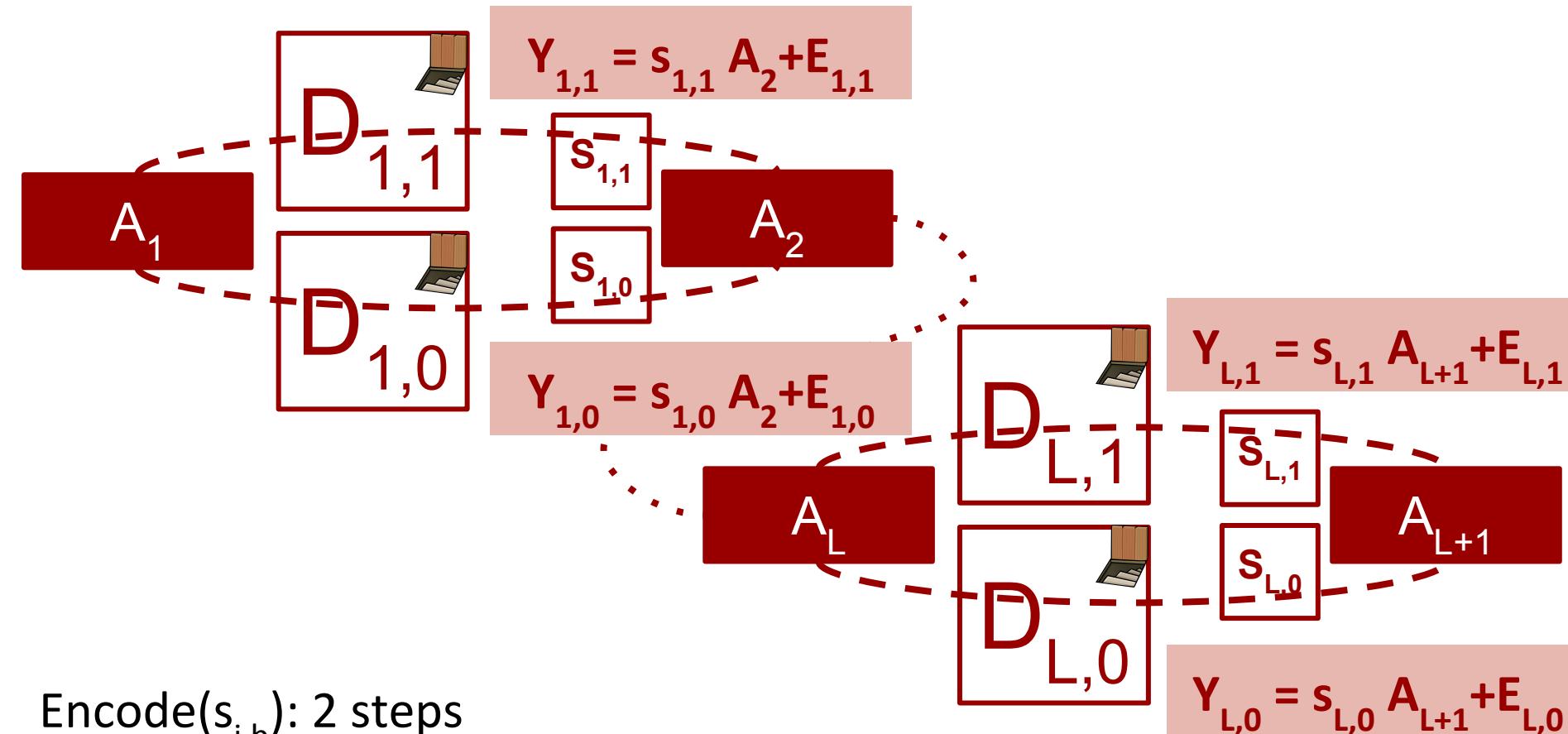
GGH15 for L hops:



Encode($s_{i,b}$): 2 steps

1. $Y_{i,b} = s_{i,b} A_{i+1} + E_{i,b}$

GGH15 for L hops:

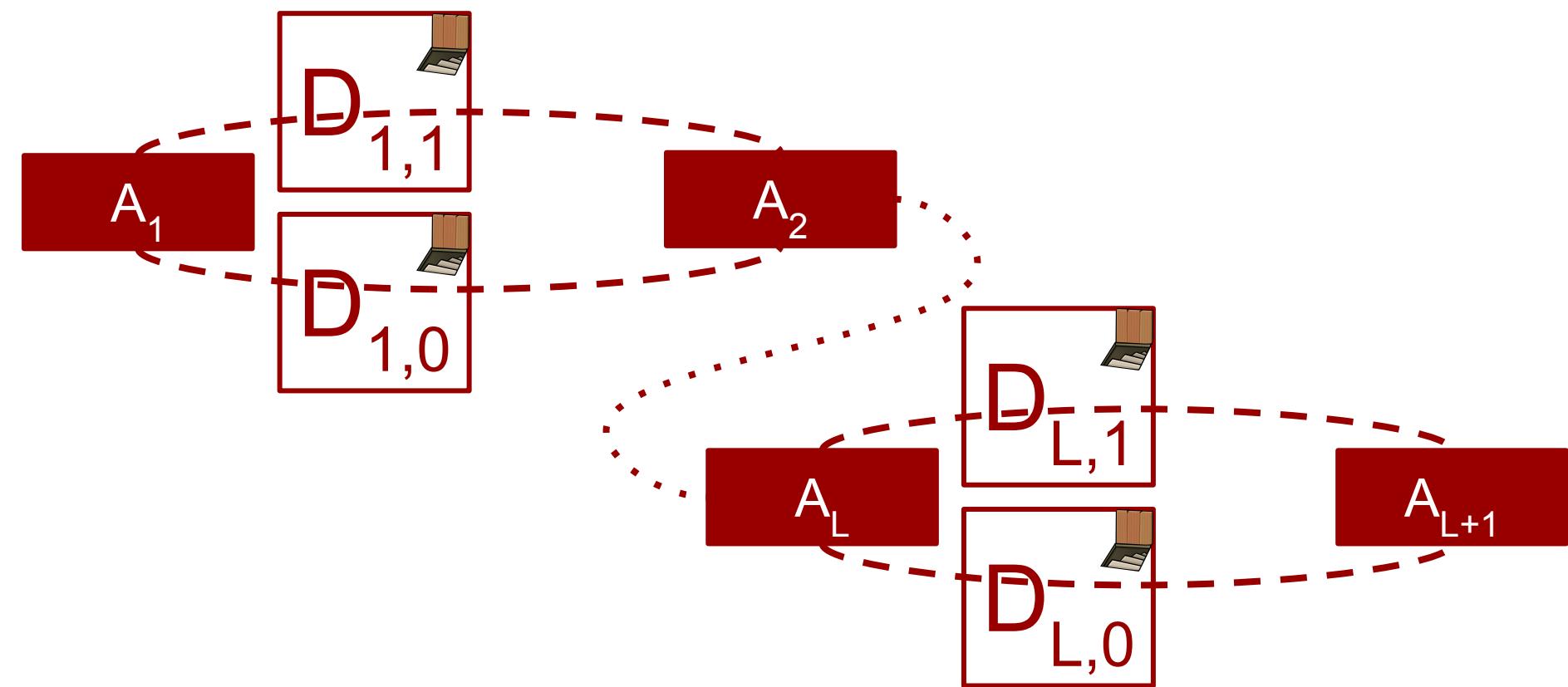


$\text{Encode}(s_{i,b})$: 2 steps

1. $Y_{i,b} = s_{i,b} A_{i+1} + E_{i,b}$
2. Sample (by the trapdoor of A_i) small $D_{i,b}$ s.t. $A_i D_{i,b} = Y_{i,b}$

Let $D_{i,b}$ be $\text{Encoding}(s_{i,b})$

GGH15 for L hops:

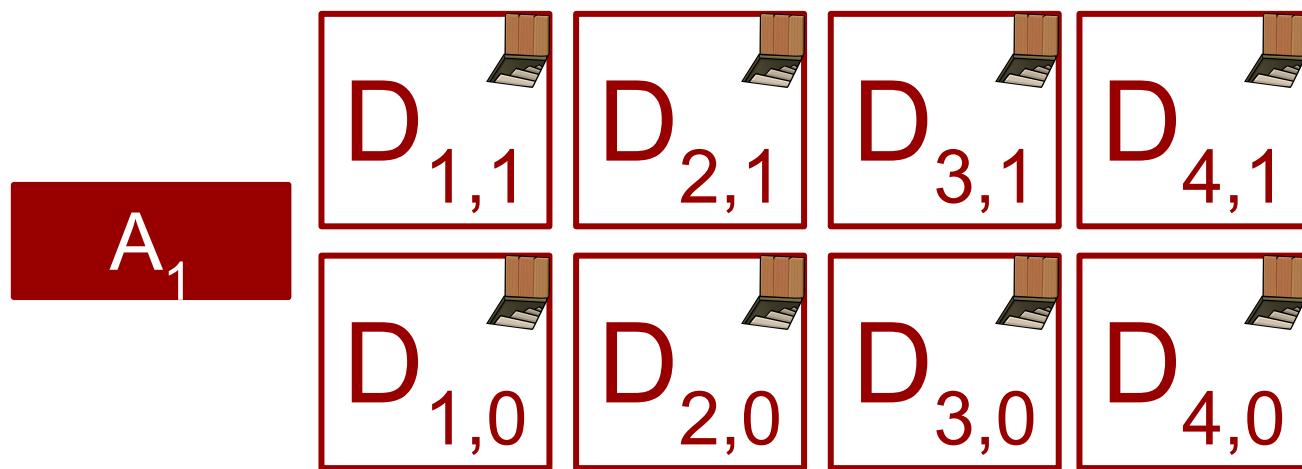


Review: What are public



Understanding the functionality of GGH15

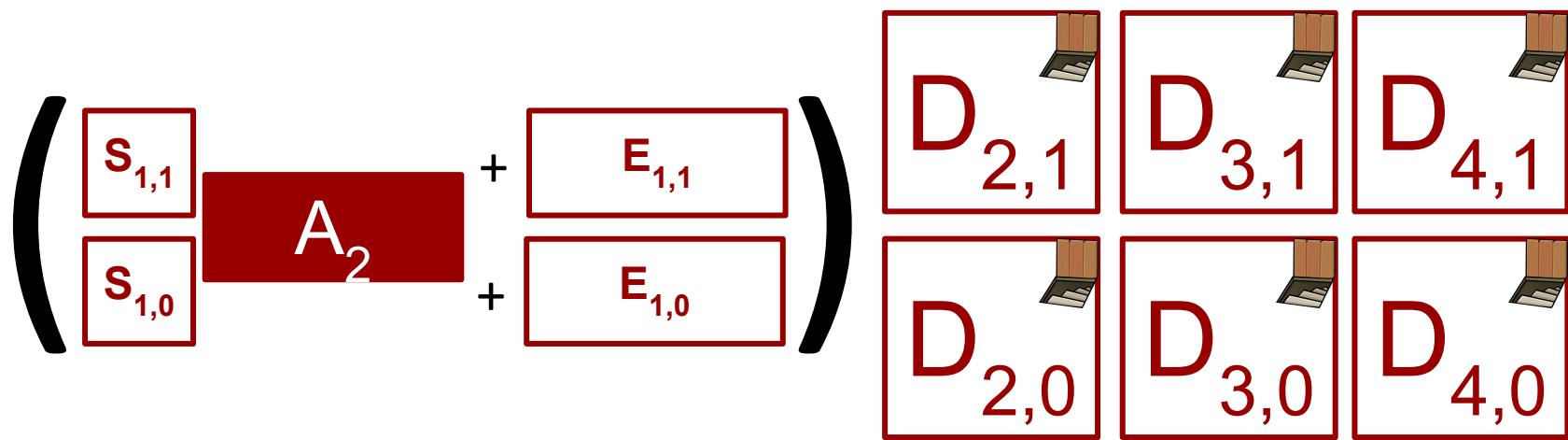
Evaluation of GGH15 (prove by example):



$\text{Eval}(0110)$

$$= A_1 D_{1,0} D_{2,1} D_{3,1} D_{4,0}$$

Evaluation of GGH15 (prove by example):

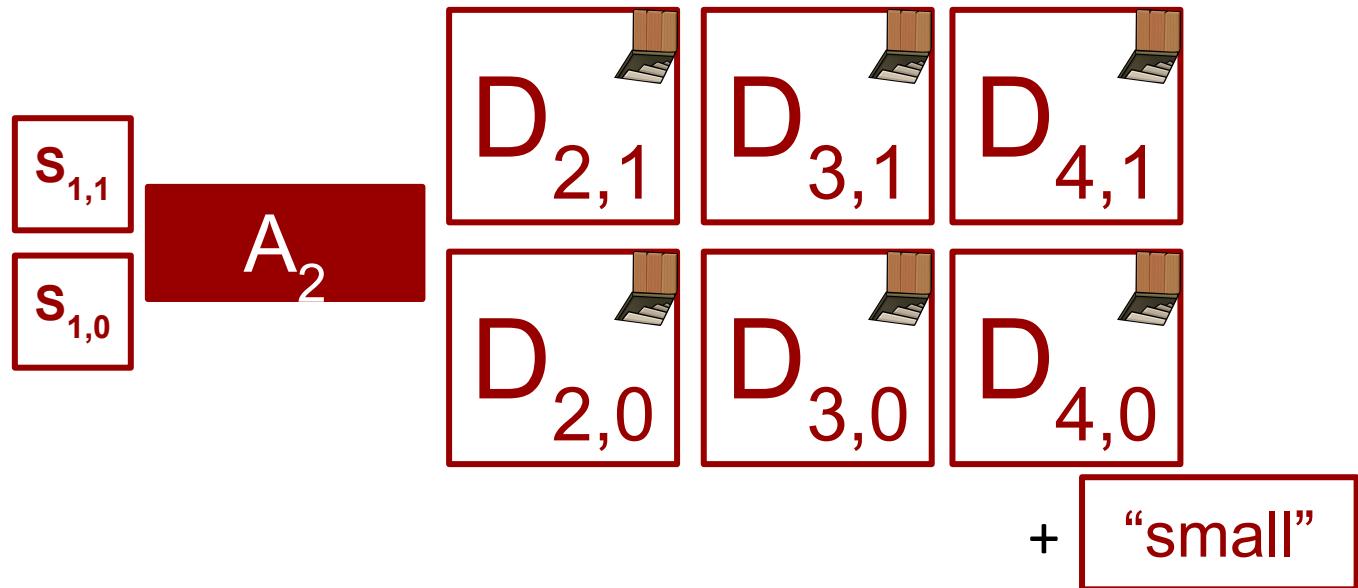


Eval(0110)

$$= A_1 D_{1,0} D_{2,1} D_{3,1} D_{4,0}$$

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

Evaluation of GGH15 (prove by example):



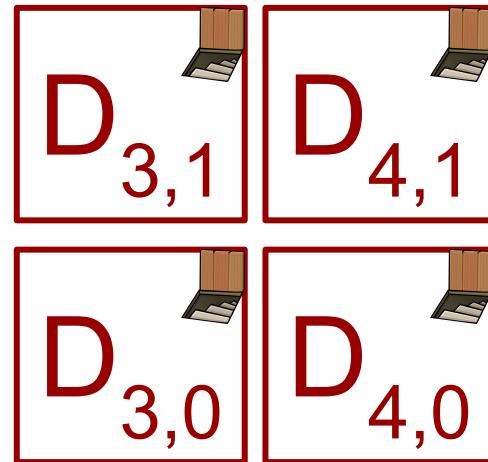
$$= A_1 D_{1,0} D_{2,1} D_{3,1} D_{4,0}$$

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

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

Evaluation of GGH15 (prove by example):

$$\left(\begin{array}{c} S_{1,1} \\ S_{1,0} \end{array} \right) \left(\begin{array}{c} S_{2,1} \\ S_{2,0} \end{array} \right) A_3 + \begin{array}{c} E_{2,1} \\ E_{2,0} \end{array}$$



+ “small”

Eval(0110)

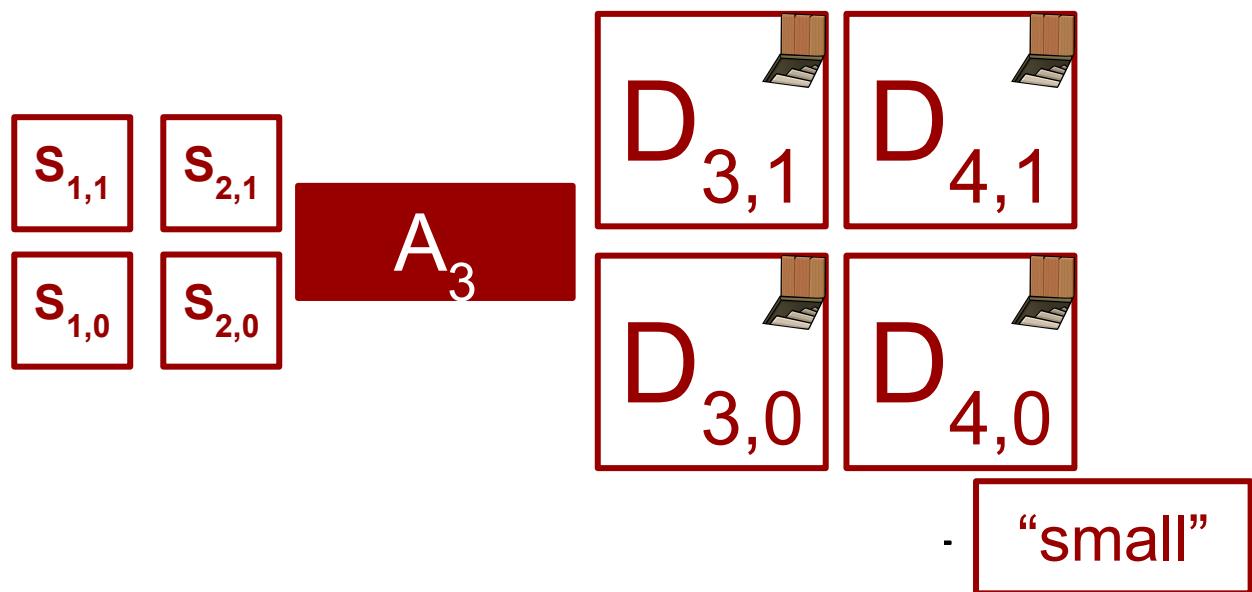
$$= A_1 D_{1,0} D_{2,1} D_{3,1} D_{4,0}$$

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

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

$$= s_{1,0} (s_{2,1} A_3 + E_{2,1}) D_{3,1} D_{4,0} + \text{“small”}$$

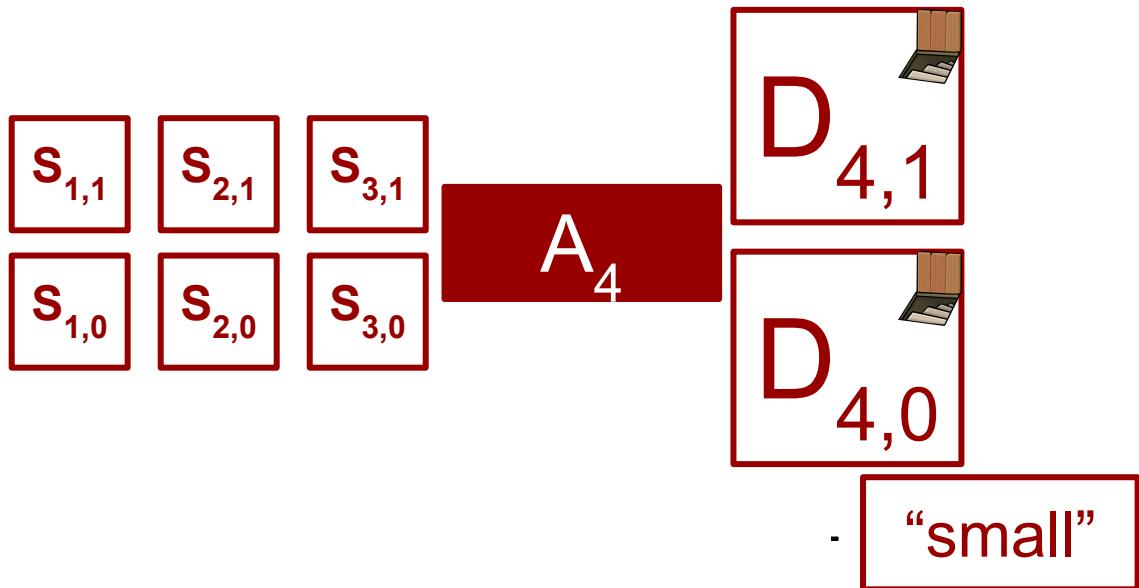
Evaluation of GGH15 (prove by example):



Eval(0110)

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

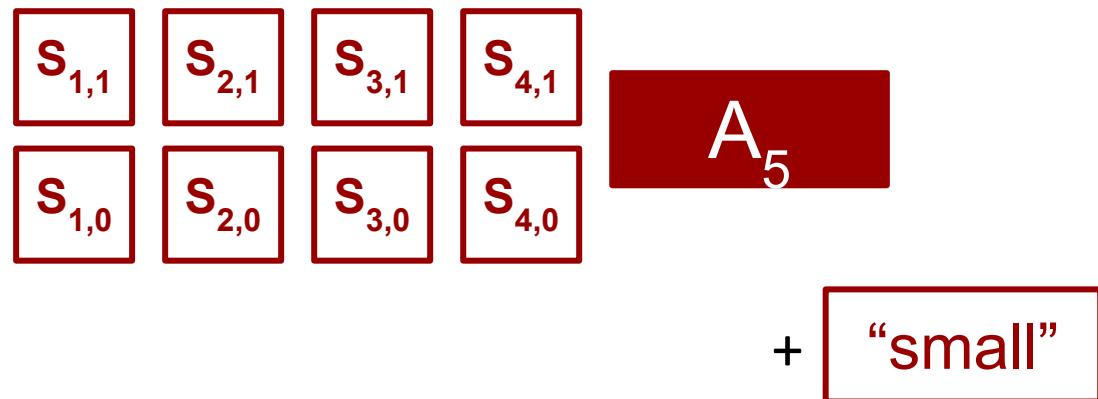
Evaluation of GGH15 (prove by example):



Eval(0110)

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

Evaluation of GGH15 (prove by example):



Eval(0110)

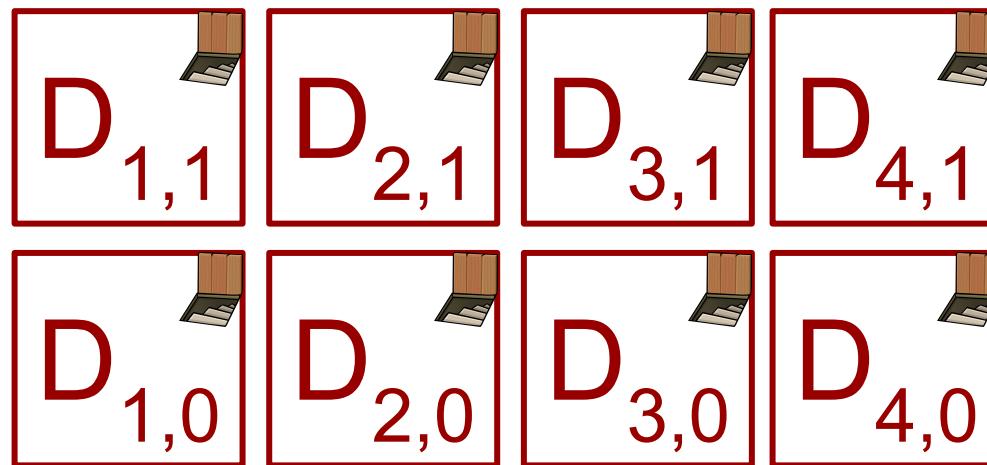
$$\begin{aligned}&= A_1 D_{1,0} D_{2,1} D_{3,1} D_{4,0} \\&= (s_{1,0} A_2 + E_{1,0}) D_{2,1} D_{3,1} D_{4,0} \\&= s_{1,0} A_2 D_{2,1} D_{3,1} D_{4,0} + \text{"small"} \\&= s_{1,0} (s_{2,1} A_3 + E_{2,1}) D_{3,1} D_{4,0} + \text{"small"} \\&= s_{1,0} s_{2,1} A_3 D_{3,1} D_{4,0} + \text{"still small"} \\&= s_{1,0} s_{2,1} s_{3,1} A_4 D_{4,0} + \text{"still smallish"} \\&= s_{1,0} s_{2,1} s_{3,1} s_{4,0} A_5 + \text{"small"}\end{aligned}$$

Evaluation of GGH15 (prove by example):



A_5
+ “small”

Evaluate



A_1



CHCPRF for NC1 constraint

NC1-CHCPRF from GGH15

Master public key: $A_1 \dots A_{L+1}$ ($L = \# \text{steps in BP}$)

Master secret key: trapdoors of $A_1 \dots A_L$, $s_{1,0}, s_{1,1}, \dots, s_{L,0}, s_{L,1}$, & J

NC1-CHCPRF from GGH15

Master public key: $A_1 \dots A_{L+1}$ ($L = \# \text{steps in BP}$)

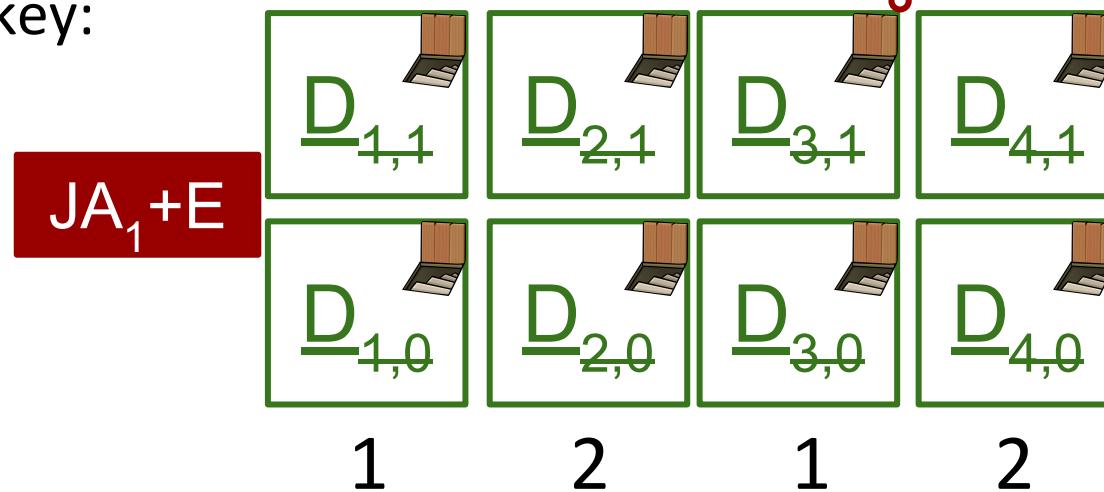
Master secret key: trapdoors of $A_1 \dots A_L$, $s_{1,0}, s_{1,1}, \dots, s_{L,0}, s_{L,1}$, & J

Constrained key gen: let $S_{i,b} := B_{i,b} \otimes s_{i,b}$, sample GGH15 encodings for $S_{i,b}$

Eval: $F(x) = \{ JA_1 \prod D_{i,x_z(i)} \}_2$ ($z: [L] \rightarrow [n]$ is the step-to-input mapping)



Constrained key:



NC1-CHCPRF from GGH15

Master public key: $A_1 \dots A_{L+1}$ ($L = \# \text{steps in BP}$)

Master secret key: trapdoors of $A_1 \dots A_L, s_{1,0}, s_{1,1}, \dots, s_{L,0}, s_{L,1}$, & J

Constrained key gen: let $S_{i,b} := B_{i,b} \otimes s_{i,b}$, sample GGH15 encodings for $S_{i,b}$

Eval: $F(x) = \{ JA_1 \prod D_{i,x_z(i)} \}_2$ ($z: [L] \rightarrow [n]$ is the step-to-input mapping)

Functionality check:

when $C(x)=1$,

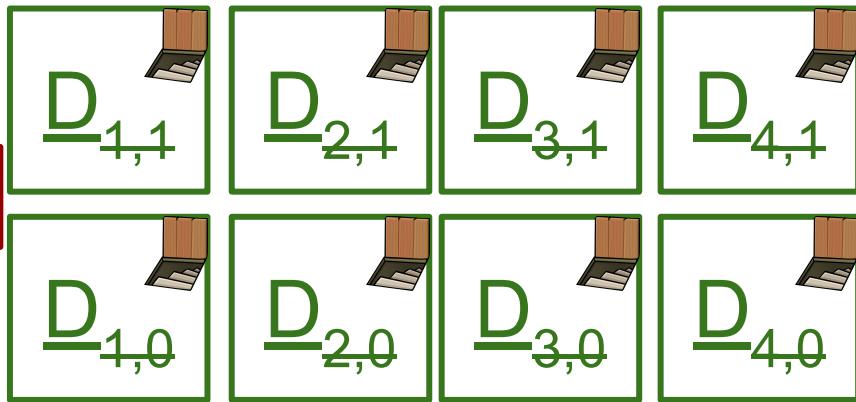
$$\begin{aligned} F(x) &= \{ JA_1 \prod D_{i,x_z(i)} \}_2 \\ &= \{ J(I \otimes \prod S_{i,x_z(i)})^2 A_{L+1} + \text{small noise} \}_2 \\ &\approx_s \{ J(I \otimes \prod S_{i,x_z(i)}) A_{L+1} \}_2 \end{aligned}$$

when $C(x)=0$,

$$\begin{aligned} F(x) &= \{ JA_1 \prod D_{i,x_z(i)} \}_2 \\ &= \{ J(C \otimes \prod S_{i,x_z(i)})^2 A_{L+1} + \text{small noise} \}_2 \\ &\approx_s \{ J(C \otimes \prod S_{i,x_z(i)}) A_{L+1} \}_2 \end{aligned}$$

NC1-CHCPRF from GGH15 *

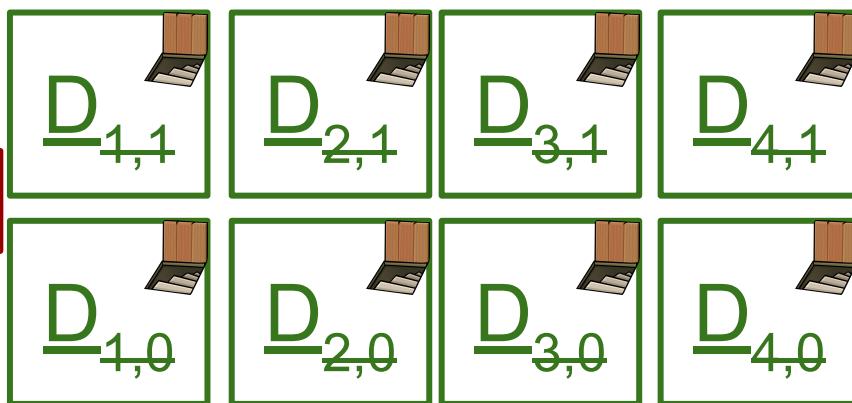
JA₁+E



Compare to GGM

NC1-CHCPRF from GGH15 *

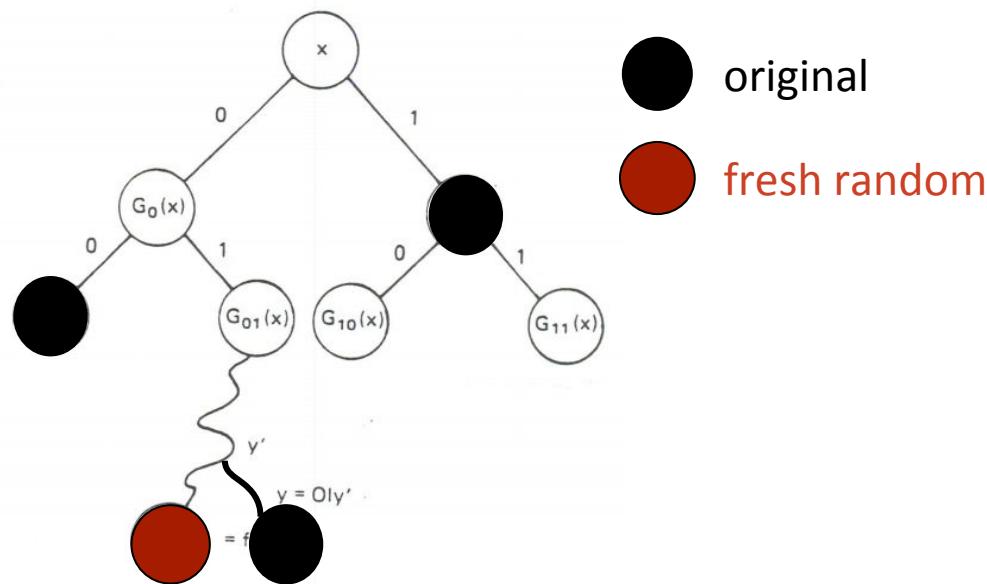
$JA_1 + E$



$B_{i,b} \otimes S_{i,b}$

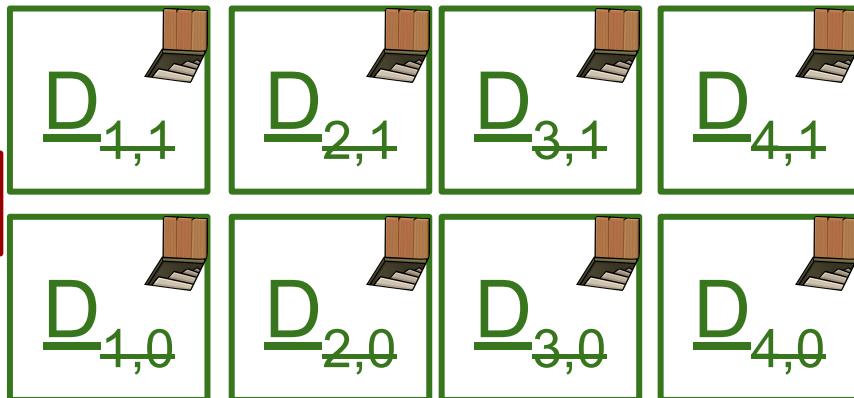
Our CHCPRF

Compare to GGM



Example: $C(x)=0$ iff $x_1=x_2=1$ query $x=11$

$s_{i,xi}$ are secret, $A_i, D_{i,xi}$ are public



$$\{ I^{\otimes(s_{1,1}s_{2,1}s_{3,1}s_{4,1})} A_5 \}_2$$



Real

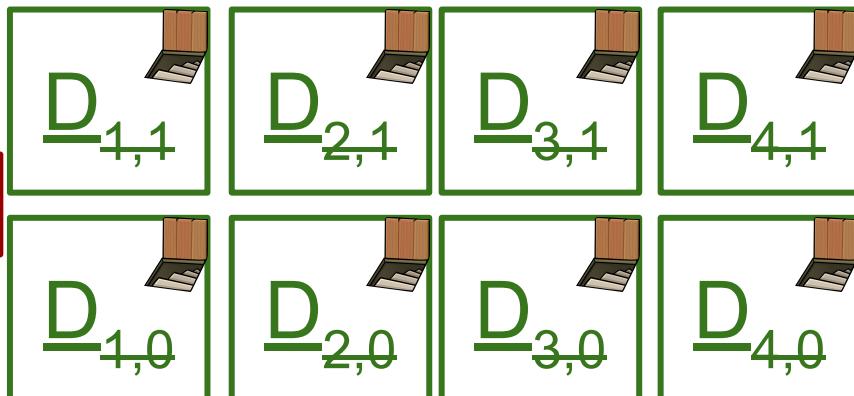
What are we trying to simulate?



Simulator

Example: $C(x)=0$ iff $x_1=x_2=1$ query $x=11$

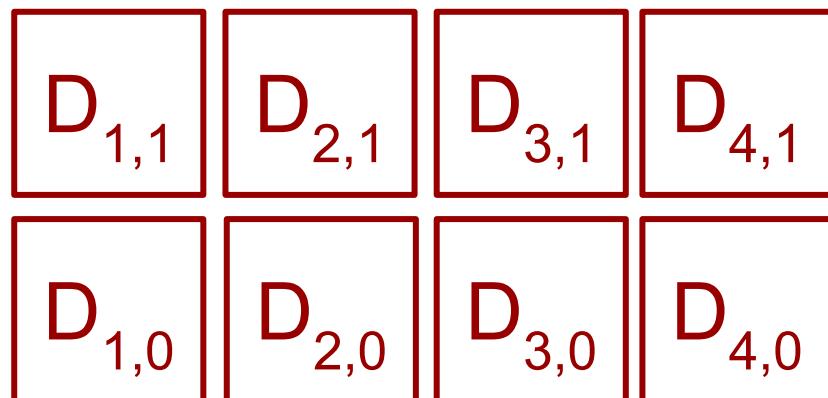
$s_{i,xi}$ are secret, $A_i, D_{i,xi}$ are public



$$\{ I^{\otimes(s_{1,1}s_{2,1}s_{3,1}s_{4,1})} A_5 \}_2$$



Real



$$\{ \text{Uniform} \}_2$$



Simulator

Proof by example with 1 input query

Example: $C(x)=0$ iff $x_1=x_2=1$ query $x=11$

$s_{i,xi}$ are secret, $A_i, D_{i,xi}$ are public



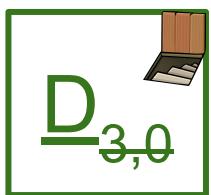
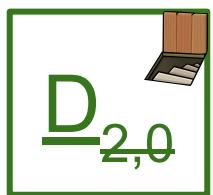
A_1

A_2

A_3

A_4

A_5



Real

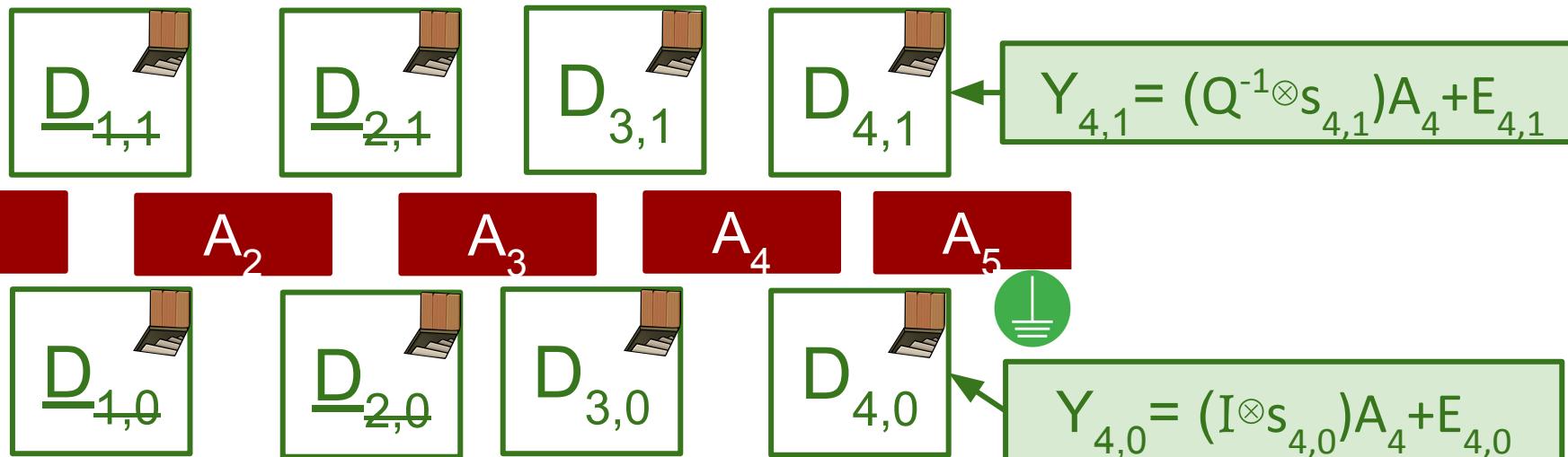
$$\text{Eval}(11) = \{ I^{\otimes(s_{1,1}s_{2,1}s_{3,1}s_{4,1})} A_5 \}_2$$



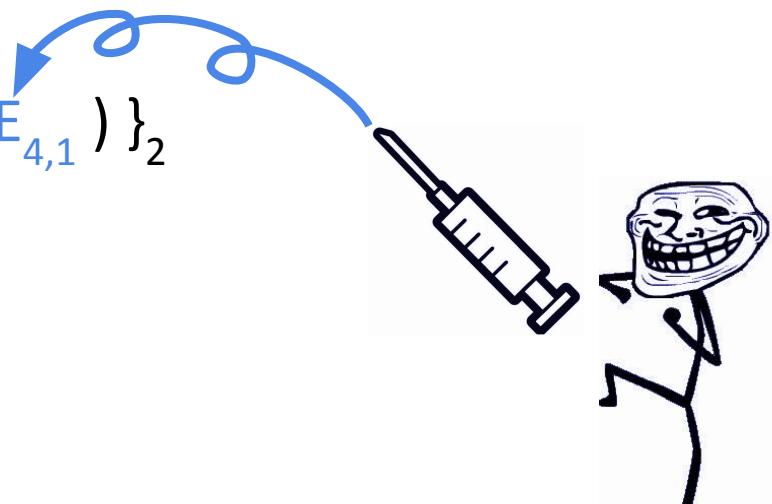
Whats up

Example: $C(x)=0$ iff $x_1=x_2=1$ query $x=11$

$s_{i,xi}$ are secret, $A_i, D_{i,xi}$ are public

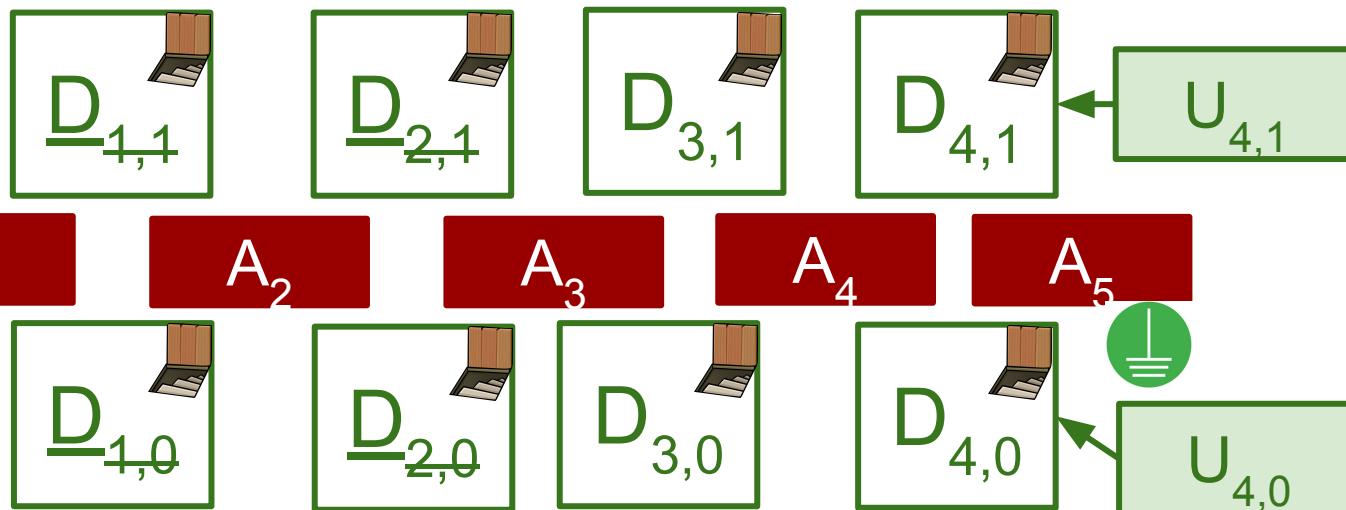


$$\begin{aligned} \text{Eval}(11) &= \{ I \otimes (s_{1,1} s_{2,1} s_{3,1} s_{4,1}) A_5 \}_2 \\ &\approx_s \{ (Q \otimes (s_{1,1} s_{2,1} s_{3,1})) ((Q^{-1} \otimes s_{4,1}) A_5 + E_{4,1}) \}_2 \end{aligned}$$



Example: $C(x)=0$ iff $x_1=x_2=1$ query $x=11$

$s_{i,xi}$ are secret, $A_i, D_{i,xi}$ are public

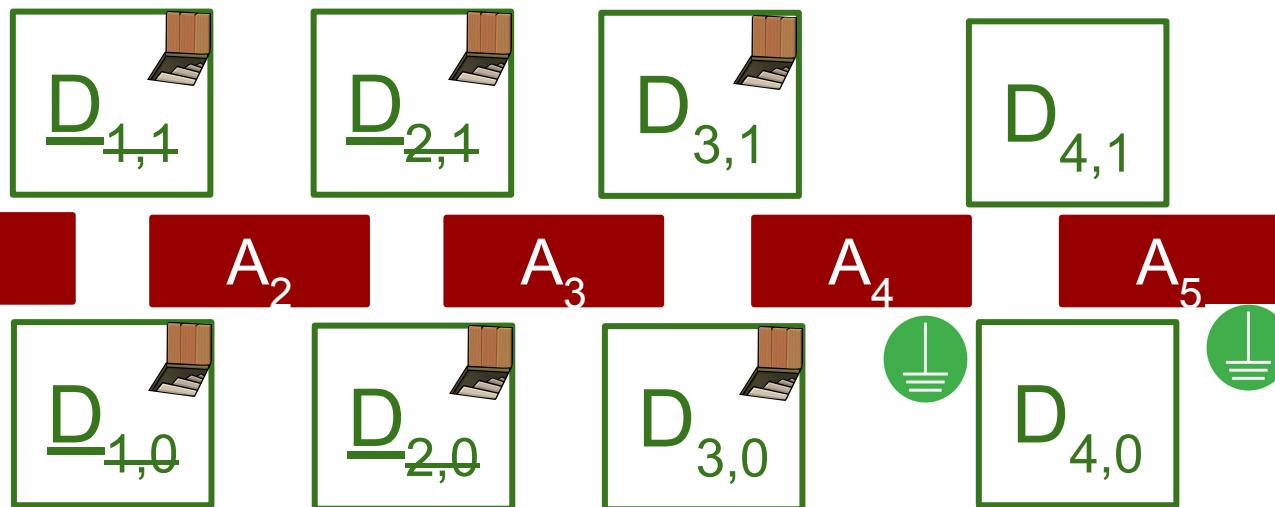


$$\begin{aligned} \text{Eval}(11) &= \{ I^{\otimes(s_{1,1}s_{2,1}s_{3,1}s_{4,1})} A_5 \}_2 \\ &\approx_s \{ (Q^{\otimes(s_{1,1}s_{2,1}s_{3,1})}) U_{4,1} \}_2 \end{aligned}$$



Example: $C(x)=0$ iff $x_1=x_2=1$ query $x=11$

$s_{i,xi}$ are secret, $A_i, D_{i,xi}$ are public

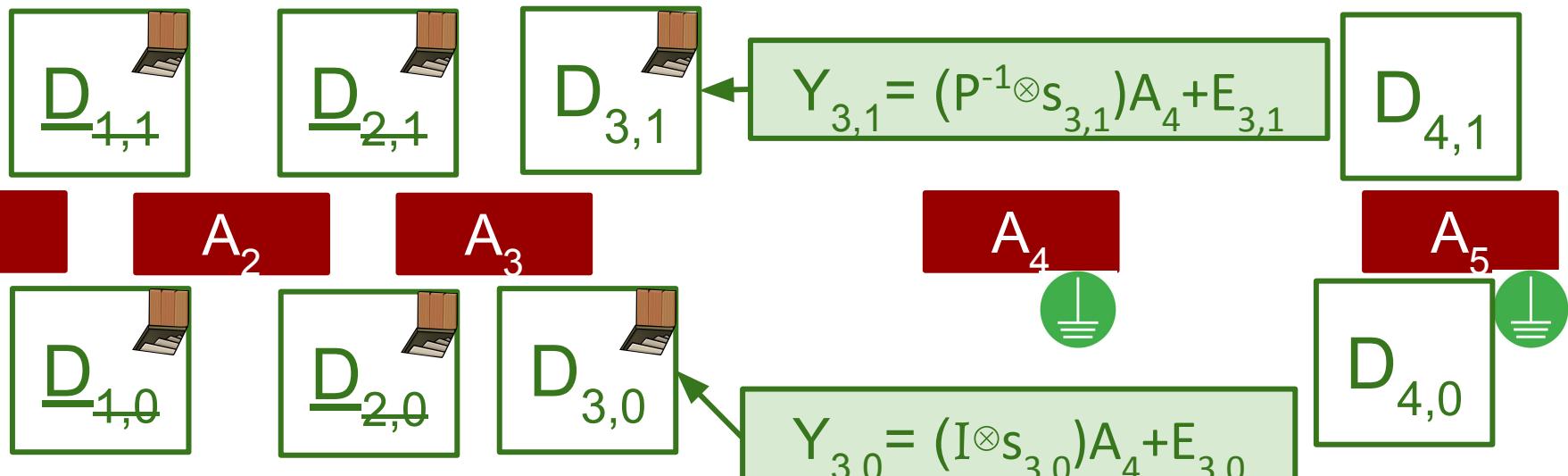


$$\begin{aligned}
 \text{Eval}(11) &= \{ I^{\otimes(s_{1,1}s_{2,1}s_{3,1}s_{4,1})} A_5 \}_2 \\
 &\approx_s \{ (Q^{\otimes(s_{1,1}s_{2,1}s_{3,1})})((Q^{-1}\otimes s_{4,1})A_5 + E_{4,1}) \}_2 \\
 &\approx_c \{ (Q^{\otimes(s_{1,1}s_{2,1}s_{3,1})}) \textcolor{red}{A}_4 D_{4,1} \}_2
 \end{aligned}$$



Example: $C(x)=0$ iff $x_1=x_2=1$ query $x=11$

$s_{i,xi}$ are secret, $A_i, D_{i,xi}$ are public

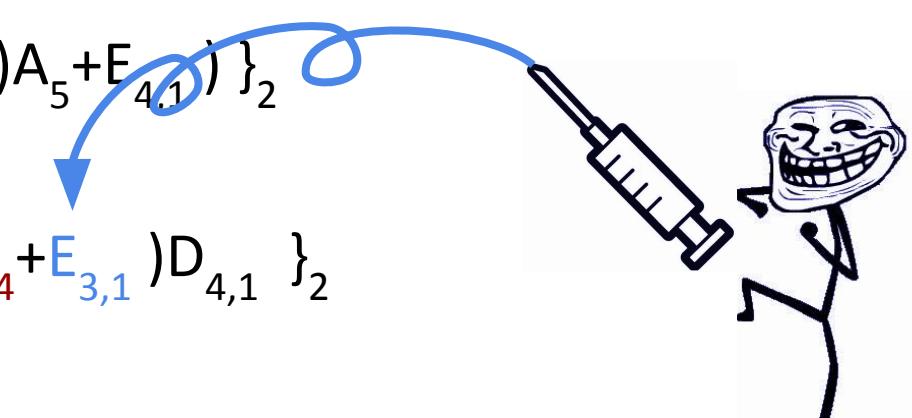


$$\text{Eval}(11) = \{ I \otimes (s_{1,1} s_{2,1} s_{3,1} s_{4,1}) A_5 \}_2$$

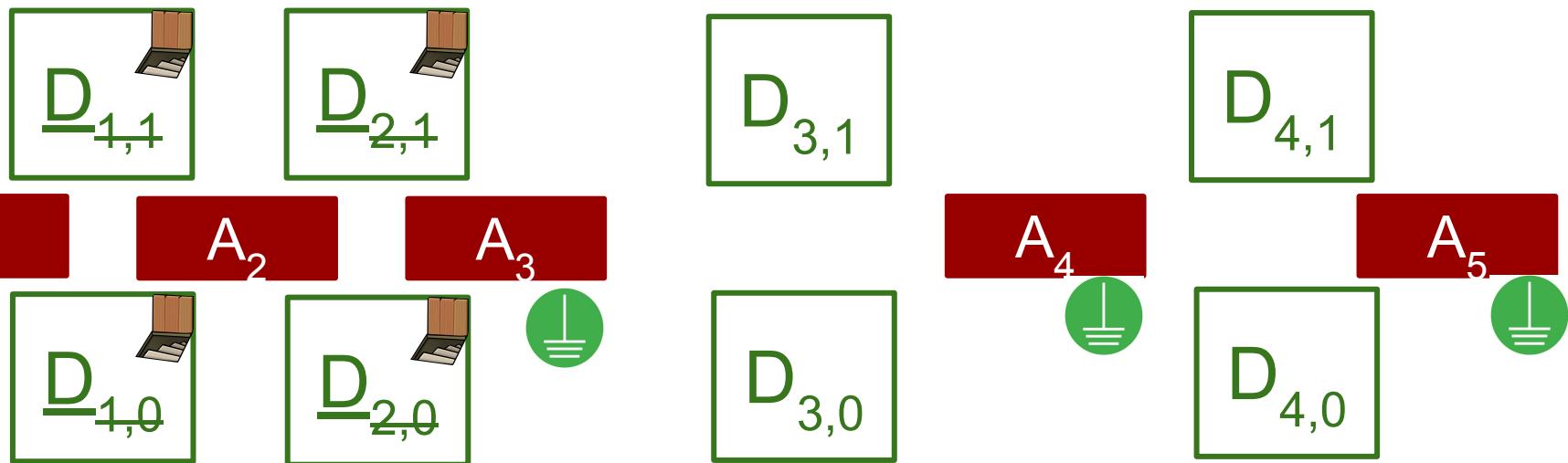
$$\approx_s \{ (Q \otimes (s_{1,1} s_{2,1} s_{3,1})) ((Q^{-1} \otimes s_{4,1}) A_5 + E_{4,1}) \}_2$$

$$\approx_c \{ (Q \otimes (s_{1,1} s_{2,1} s_{3,1})) A_4 D_{4,1} \}_2$$

$$\approx_s \{ (QP \otimes (s_{1,1} s_{2,1})) ((P^{-1} \otimes s_{3,1}) A_4 + E_{3,1}) D_{4,1} \}_2$$



Example: $C(x)=0$ iff $x_1=x_2=1$ query $x=11$ $s_{i,xi}$ are secret, $A_i, D_{i,xi}$ are public

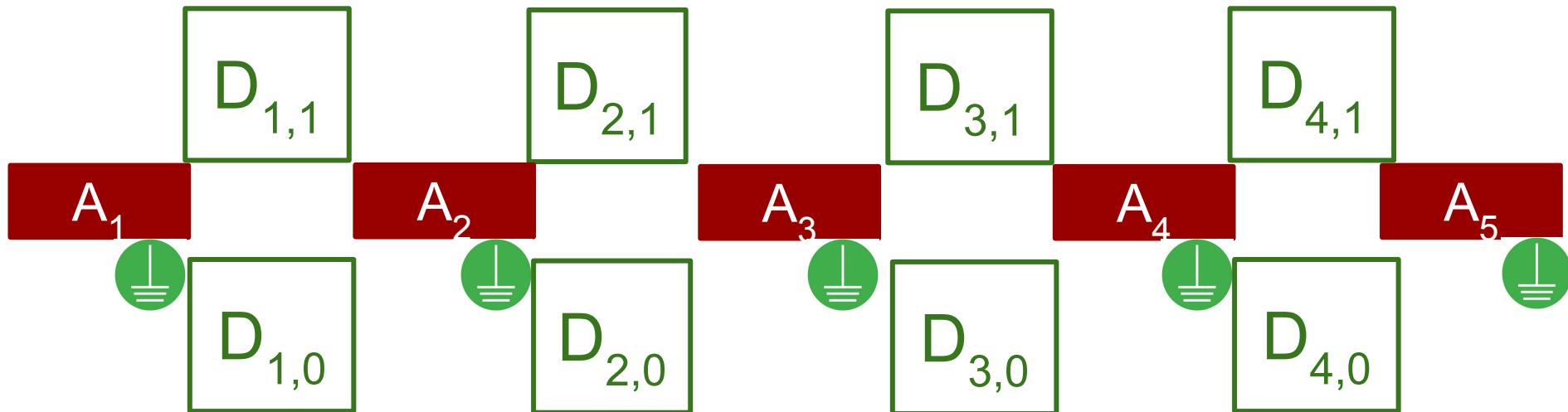


$$\begin{aligned}
 \text{Eval}(11) &= \{ I^{\otimes(s_{1,1}s_{2,1}s_{3,1}s_{4,1})} A_5 \}_2 \\
 &\approx_s \{ (Q^{\otimes(s_{1,1}s_{2,1}s_{3,1})})((Q^{-1} \otimes s_{4,1})A_5 + E_{4,1}) \}_2 \\
 &\approx_c \{ (Q^{\otimes(s_{1,1}s_{2,1}s_{3,1})})A_4 D_{4,1} \}_2 \\
 &\approx_s \{ (QP^{\otimes(s_{1,1}s_{2,1})})((P^{-1} \otimes s_{3,1})A_4 + E_{3,1})D_{4,1} \}_2 \\
 &\approx_c \{ (QP^{\otimes(s_{1,1}s_{2,1})})\mathbf{A}_3 D_{3,1} D_{4,1} \}_2
 \end{aligned}$$



Example: $C(x)=0$ iff $x_1=x_2=1$ query $x=11$

$s_{i,xi}$ are secret, $A_i, D_{i,xi}$ are public

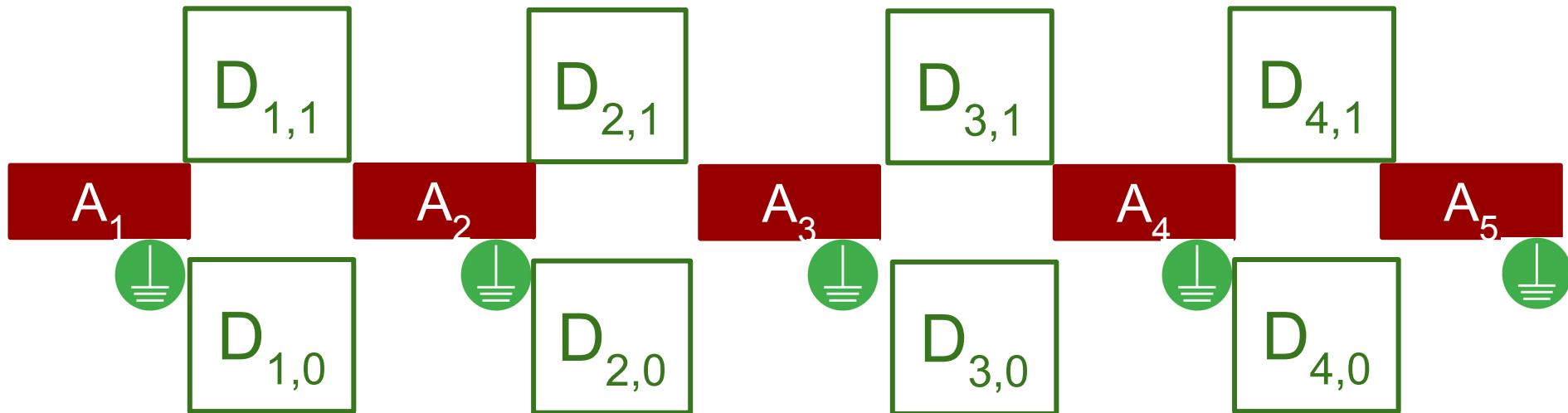


$$\begin{aligned}
 \text{Eval}(11) &= \{ I^{\otimes(s_{1,1}s_{2,1}s_{3,1}s_{4,1})} A_5 \}_2 \\
 &\approx_s \{ (Q^{\otimes(s_{1,1}s_{2,1}s_{3,1})})((Q^{-1}\otimes s_{4,1})A_5 + E_{4,1}) \}_2 \\
 &\approx_c \{ (Q^{\otimes(s_{1,1}s_{2,1}s_{3,1})})A_4 D_{4,1} \}_2 \\
 &\approx_s \{ (QP^{\otimes(s_{1,1}s_{2,1})})((P^{-1}\otimes s_{3,1})A_4 + E_{3,1})D_{4,1} \}_2 \\
 &\approx_c \{ (QP^{\otimes(s_{1,1}s_{2,1})})A_3 D_{3,1} D_{4,1} \}_2 \\
 &\approx_c \dots \approx_c \{ C^{-1} A_1 \prod D_{z(x), x_z(x)} \}_2
 \end{aligned}$$



Example: $C(x)=0$ iff $x_1=x_2=1$ query $x=11$

$s_{i,xi}$ are secret, $A_i, D_{i,xi}$ are public



$$\begin{aligned}
 \text{Eval}(11) &= \{ I^{\otimes(s_{1,1}s_{2,1}s_{3,1}s_{4,1})}A_5 \}_2 \\
 &\approx_s \{ (Q^{\otimes(s_{1,1}s_{2,1}s_{3,1})})((Q^{-1}\otimes s_{4,1})A_5 + E_{4,1}) \}_2 \\
 &\approx_c \{ (Q^{\otimes(s_{1,1}s_{2,1}s_{3,1})})A_4 D_{4,1} \}_2 \\
 &\approx_s \{ (QP^{\otimes(s_{1,1}s_{2,1})})((P^{-1}\otimes s_{3,1})A_4 + E_{3,1})D_{4,1} \}_2 \\
 &\approx_c \{ (QP^{\otimes(s_{1,1}s_{2,1})})A_3 D_{3,1} D_{4,1} \}_2 \\
 &\approx_c \dots \approx_c \{ C^{-1}A_1 \prod D_{z(x),x_z(x)} \}_2
 \end{aligned}$$

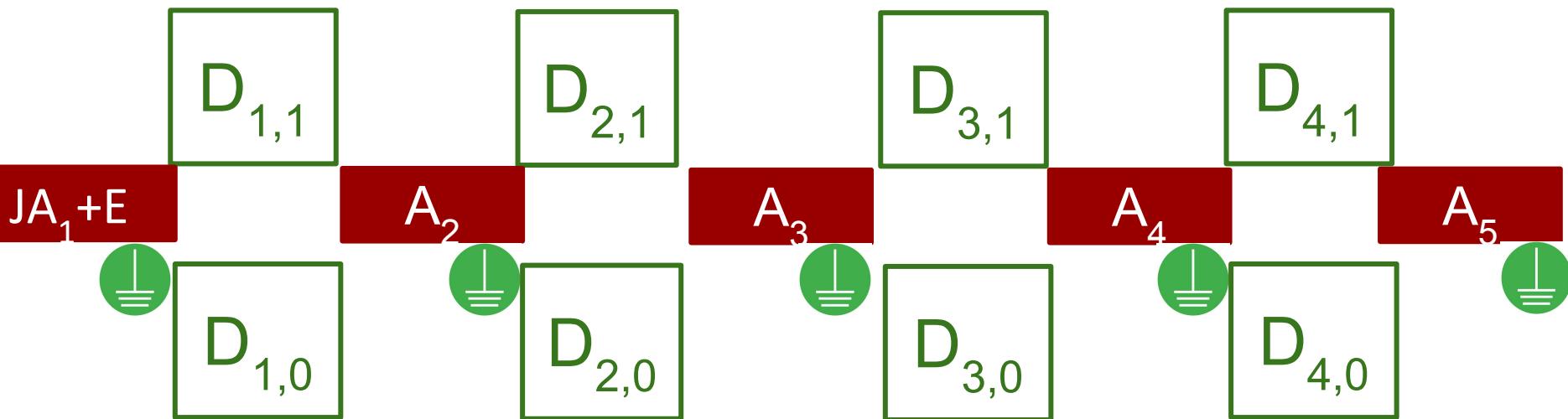
Current status:
 - CK ✓
 - randomness of the outputs ✗



cont.

Example: $C(x)=0$ iff $x_1=x_2=1$ query $x=11$

$s_{i,xi}$ are secret, $A_i, D_{i,xi}$ are public

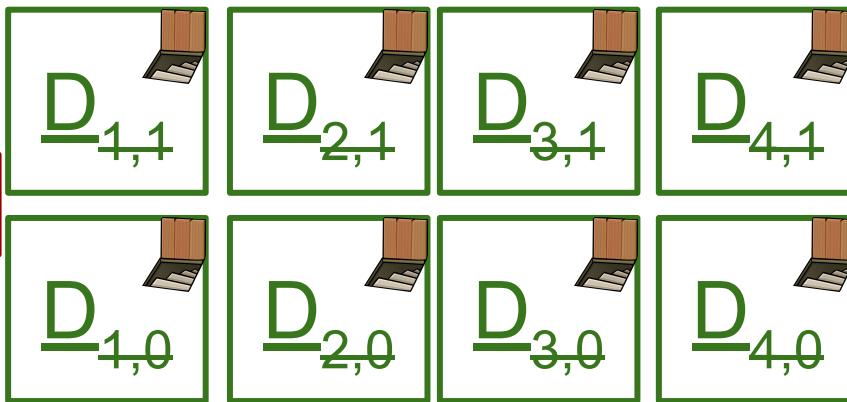


$$\begin{aligned}
 \text{Eval}(11) &= \{ I^{\otimes(s_{1,1}s_{2,1}s_{3,1}s_{4,1})}A_5 \}_2 \\
 &\approx_s \{ (Q^{\otimes(s_{1,1}s_{2,1}s_{3,1})})((Q^{-1}\otimes s_{4,1})A_5 + E_{4,1}) \}_2 \\
 &\approx_c \{ (Q^{\otimes(s_{1,1}s_{2,1}s_{3,1})})A_4 D_{4,1} \}_2 \\
 &\approx_s \{ (QP^{\otimes(s_{1,1}s_{2,1})})((P^{-1}\otimes s_{3,1})A_4 + E_{3,1})D_{4,1} \}_2 \\
 &\approx_c \{ (QP^{\otimes(s_{1,1}s_{2,1})})A_3 D_{3,1} D_{4,1} \}_2 \\
 &\approx_c \dots \approx_c \{ JC^{-1}A_1 \prod D_{z(x),x_z(x)} \}_2
 \end{aligned}$$

Current status:
 - CK ✓
 - randomness of
the outputs ✗
 Solution:
 Multiply a random
vector J on the left



cont.

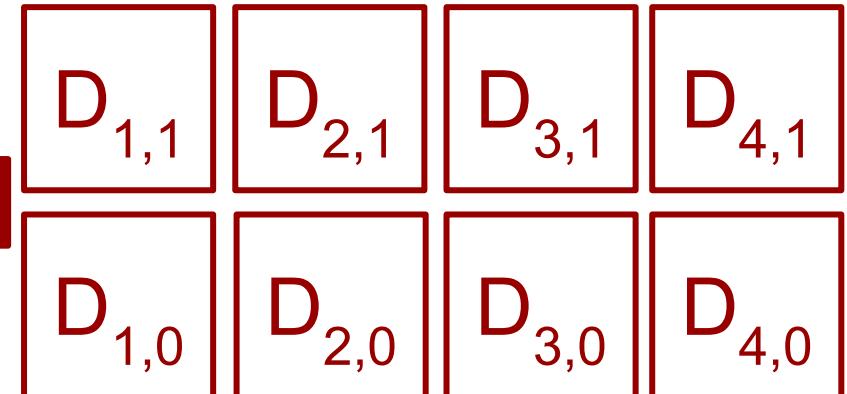


$JA_1 + E$

$$\{ J(I^{\otimes}(\prod s_{z(x), x_z(x)})) A_{L+1} \}_2$$



Real



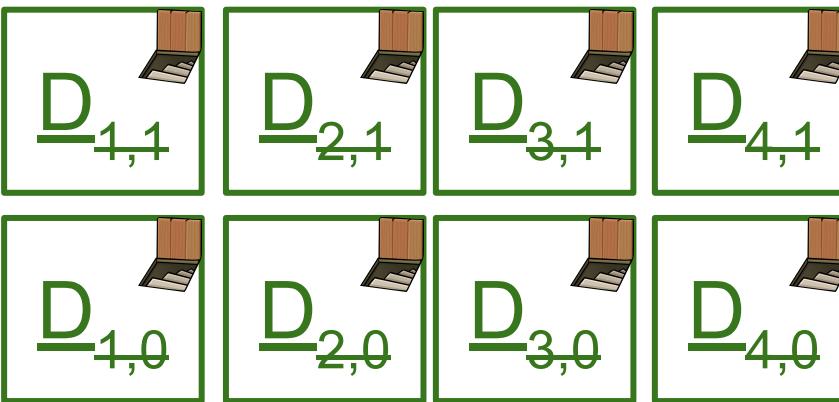
$JA_1 + E$



Eval = ...

$$\approx_c \{ (JC^{-1}A_1 + E) \prod D_{z(x), x_z(x)} \}_2$$

$JA_1 + E$

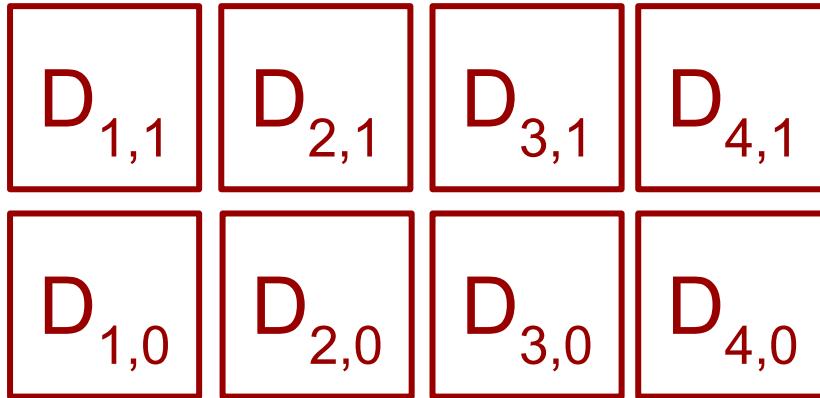


$$\{ J(I^{\otimes}(\prod s_{z(x), x_z(x)})) A_{L+1} \}_2$$



Real

A_J



Eval = ...

$$\approx_c \{ (JC^{-1}A_1 + E) \prod D_{z(x), x_z(x)} \}_2$$

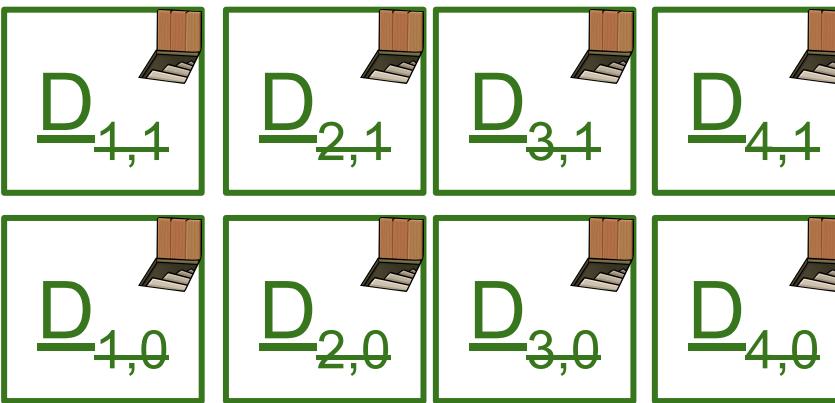
$$\approx_c \{ U \prod D_{z(x), x_z(x)} \}_2$$

$$\approx_c \{ \text{Uniform} \}_2$$



Simulator

$JA_1 + E$

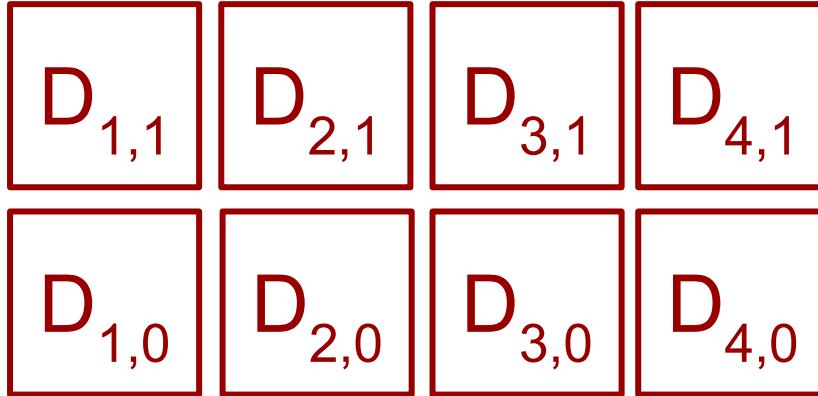


$$\{ J(I^{\otimes(\prod s_{z(x), x_z(x)})}) A_{L+1} \}_2$$



Real

A_J



Eval = ...

$$\approx_c \{ (JC^{-1}A_1 + E) \prod D_{z(x), x_z(x)} \}_2$$

$$\approx_c \{ U \prod D_{z(x), x_z(x)} \}_2$$

$$\approx_c \{ \text{Uniform} \}_2$$



Simulator

Summary: NC1 CHCPRF from GGH15

- Constraint hiding: Perm-LWE + GPV
- Outputs: need additional protection J , justified by JLWE

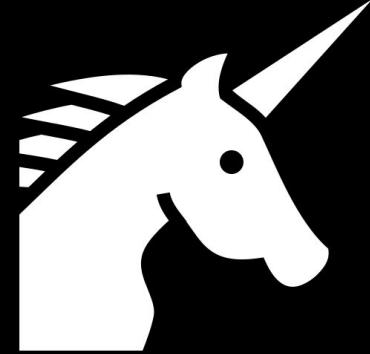
Concurrent work:
Boneh, Kim, Montgomery (Eurocrypt 17)

1-key puncturable CHCPRFs from LWE.

Both root from previous lattices-based PRFs,
but different method to constrain and hide.



Genealogy of Lattices-based PRFs



- [BPR12] -- the settler
 - [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
- * uses gadget matrix G , adapted from the lattices-based FHE, ABE, PE

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

p.s. Hoeteck asked me if there's an interpretation of [GVW13] ABE from [GGH15]. I thought for a little bit, not obvious.



More questions of GGH15

Q: What safe modes do we have confidence for GGH15?

A: With limited number/restricted form of zeros, very likely.

Q: What is weird about GGH15 (as a useful mmaps)?

A: Must prove from 1 direction (namely make sure that the trapdoor sampling is safe, from sink to source), not a desirable property of mmaps.

Q: Anything to say when the A matrices are hidden?

A: There must be something to say ... a question worth to understand

The end

