Constraint hiding constrained PRF for NC1 from LWE

Ran Canetti, Yilei Chen, # Eurocrypt 2017 special edition

Puncture!

Puncturable/constrained PRF

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

x^*

K

original key

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

punctured key

Puncturable/constrained PRF

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

x^*

K

original key

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

punctured key

Puncture(K, x*) => 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

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

In general: Constrain(K, C) => 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

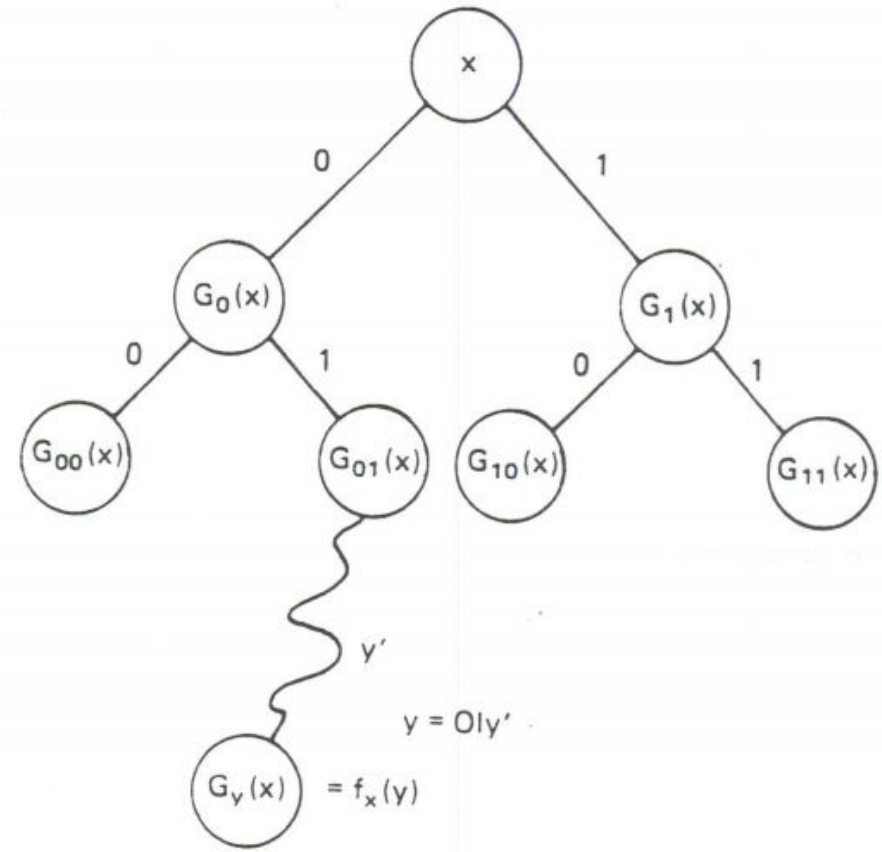
Puncture(K, x*) => K{x*} s.t. $F_k(x^*)$ is pseudorandom, give the K{x*} that preserve the original outputs elsewhere.

In general: Constrain(K, C) => K{C}

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

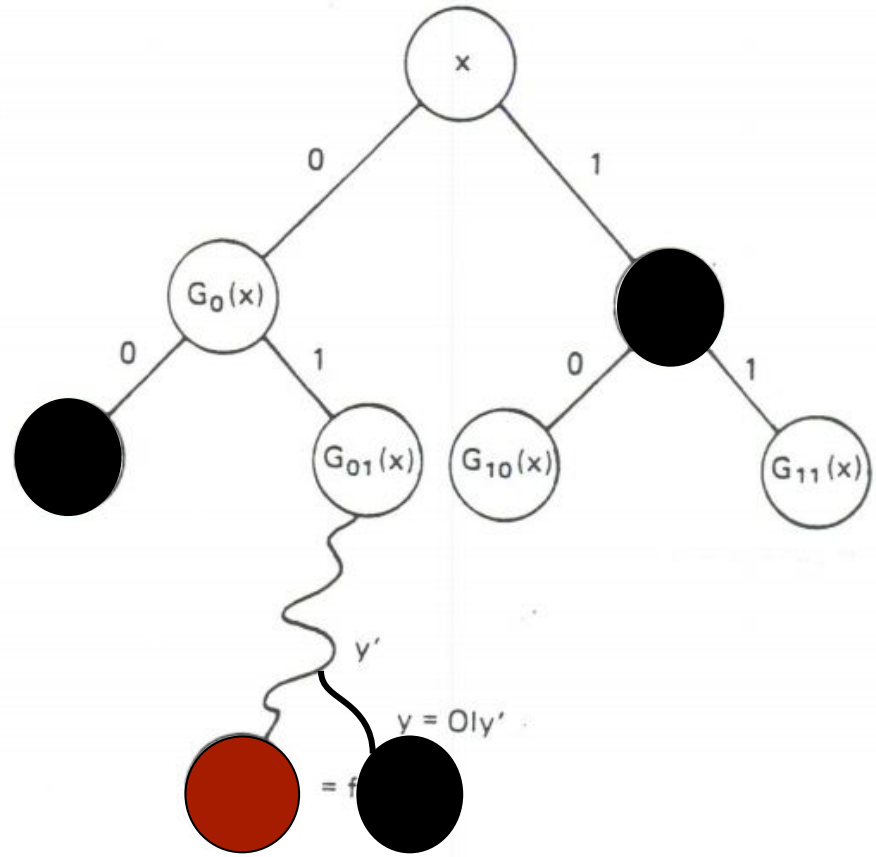
Puncturable PRF from GGM

[Goldreich, Goldwasser, Micali 84]



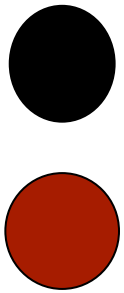
Puncturable PRF from GGM

[Goldreich, Goldwasser, Micali 84]



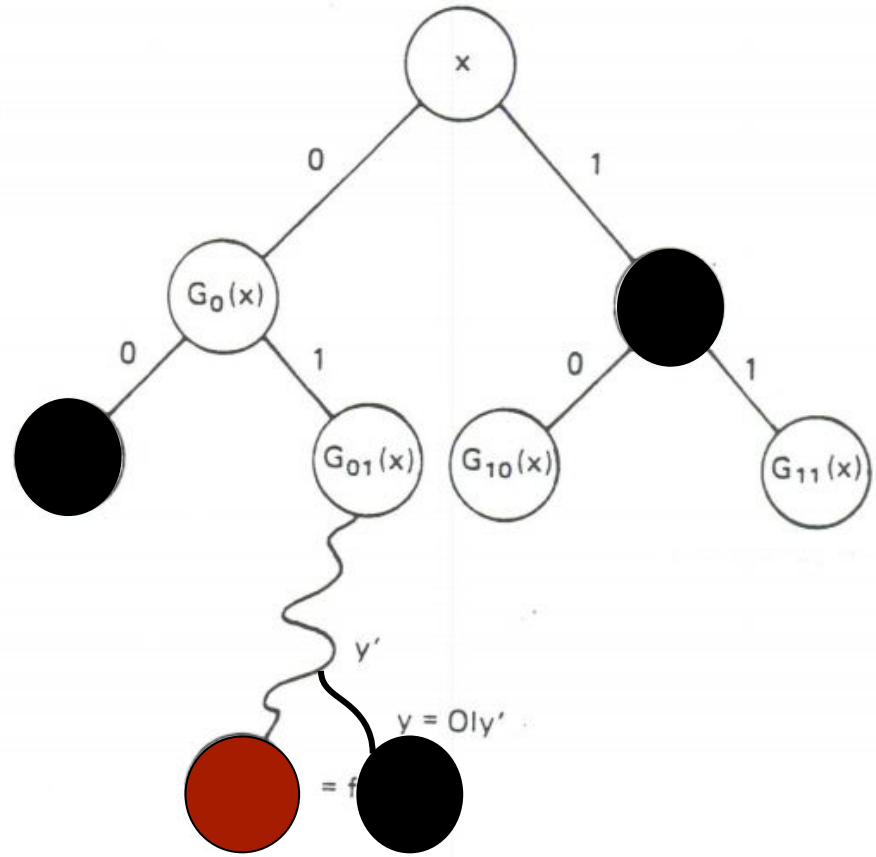
Puncturable PRF from GGM

[Goldreich, Goldwasser, Micali 84]



original

fresh random



The constrained key reveals the point x*

Constrained PRF

Constraint-hiding Constrained PRF

hide

Constraint-hiding Constrained PRF

?

Constraint-hiding Constrained PRF

? = secret?

Constraint-hiding Constrained PRF

? = secret?

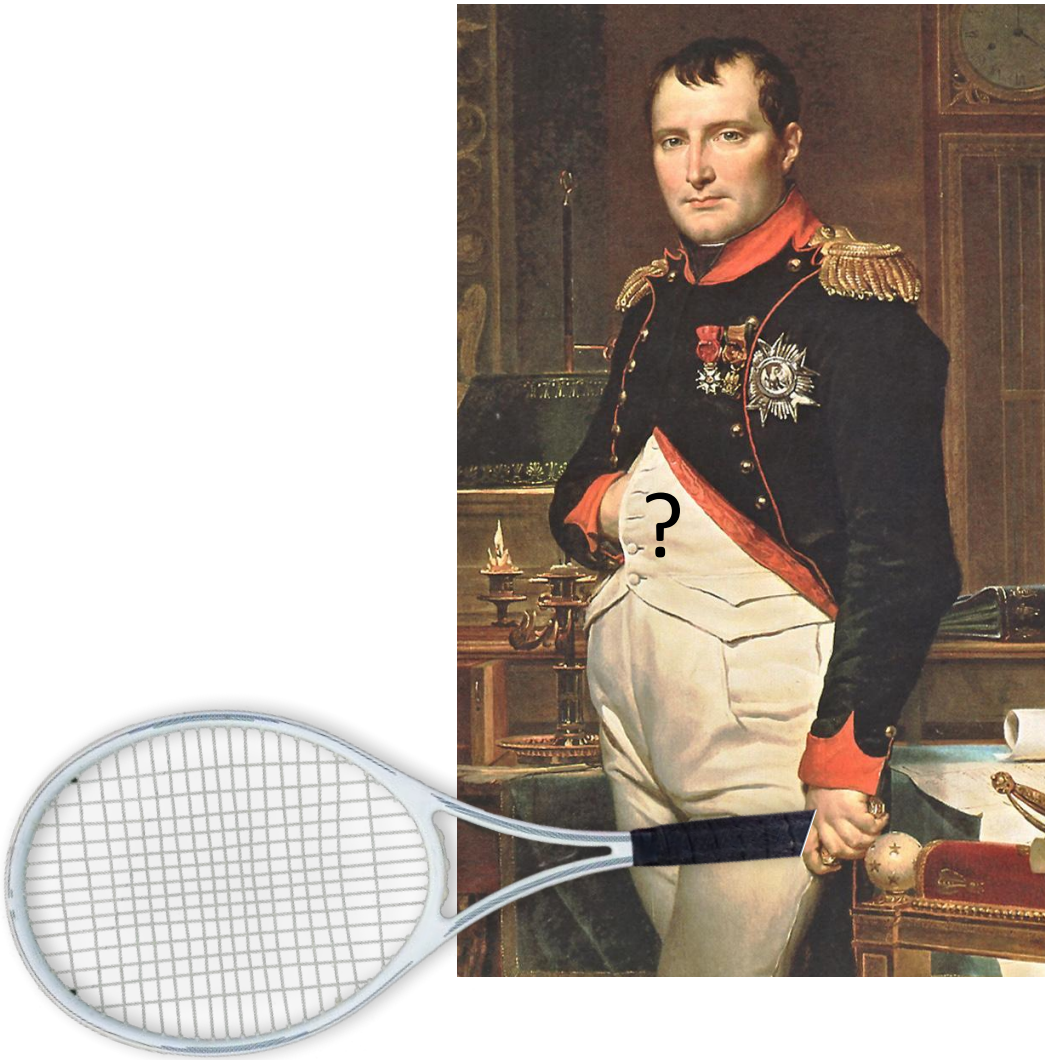
? = key?

Constraint-hiding Constrained PRF

? = secret?

? = key?

? =  ?



#fakeTennisCourtOath

Boneh, Lewi, Wu (PKC 17, eprint 2015/1167)

What are Constraint-Hiding CPRFs:

- An indistinguishability-based definition

Boneh, Lewi, Wu (PKC 17, eprint 2015/1167)

What are Constraint-Hiding CPRFs:

- An indistinguishability-based definition

Where to find them (secure for many keys):

- iO(PPRF) is CHCPRF
- Bit-fixing from multilinear DDH, puncturing from multilinear subgroup-hiding

Boneh, Lewi, Wu (PKC 17, eprint 2015/1167)

What are Constraint-Hiding CPRFs:

- An indistinguishability-based definition

Where to find them (secure for many keys):

- iO(PPRF) is CHCPRF
- Bit-fixing from multilinear DDH, puncturing from multilinear subgroup-hiding

How to use them:

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

This work:
Canetti, Chen (Eurocrypt 17)

What are Constraint-Hiding CPRFs:

- A simulation-based definition of CHCPRF

This work:
Canetti, Chen (Eurocrypt 17)

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

This work:
Canetti, Chen (Eurocrypt 17)

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*

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

1-key <u>puncturable</u> CHCPRFs from LWE.

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

Plan for the talk:

Part 1: Definition, implications to obfuscation, functional encryption

Part 2: How to construct CHCPRFs for NC1

Defining constraint-hiding constraint PRF (CHCPRF)

Master_KeyGen -> MSK

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

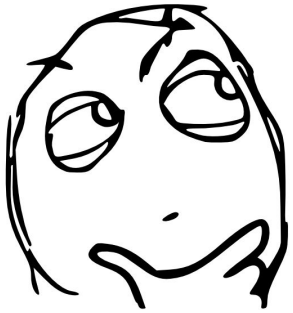
Eval(K, x) -> $F_K(x)$

definition of CHCPRF

Simulation-based CHCPRF [CC 17]

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

Real



adv

Simulator

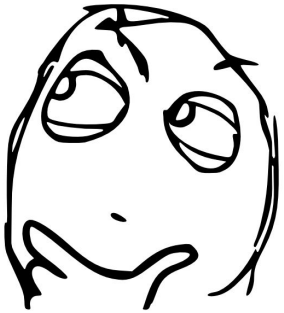
Master_KeyGen -> MSK

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

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

Constraint query C

Real



adv

Simulation-based definition of CHCPRF

Master_KeyGen -> MSK

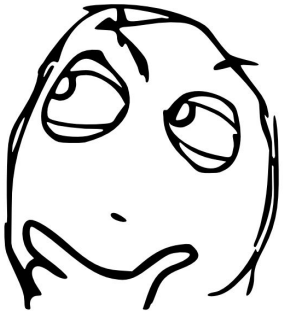
Cons(MSK, C) -> K[C]

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

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

Constraint query C

Real



adv

Simulation-based definition of CHCPRF

Master_KeyGen -> MSK

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

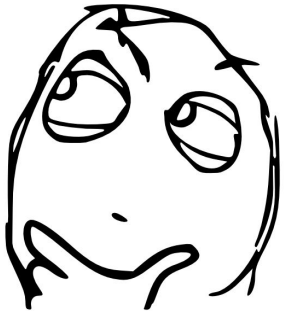
Eval(K, x) -> $F_K(x)$

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

Constraint query C

Input query x

Real



adv

Simulation-based definition of CHCPRF

Master_KeyGen -> MSK

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

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

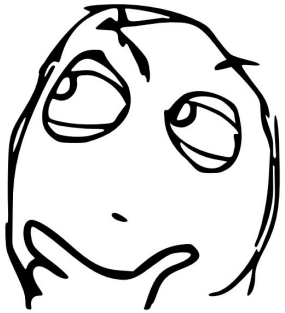
Cons(MSK, C) -> K[C]

Constraint query C

Input query x

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

Real



adv

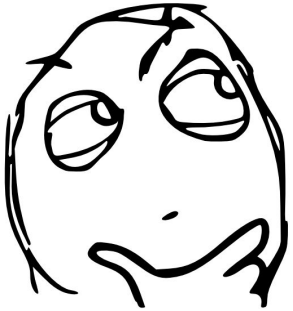
Simulation-based definition of CHCPRF

Master_KeyGen -> MSK

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

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

Constraint query C →



adv

Simulator

Simulation-based definition of CHCPRF

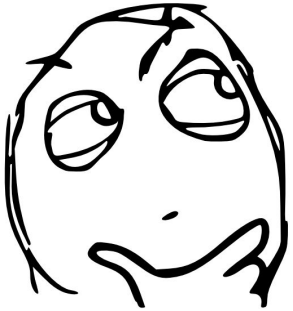
Master_KeyGen -> MSK

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

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

Constraint query C

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



adv

Simulator

Simulation-based definition of CHCPRF

Master_KeyGen -> MSK

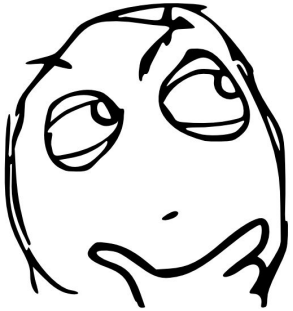
Cons(MSK, C) -> K[C]

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

Constraint query C

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

Input query x



adv

Simulator

Simulation-based definition of CHCPRF

Master_KeyGen -> MSK

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

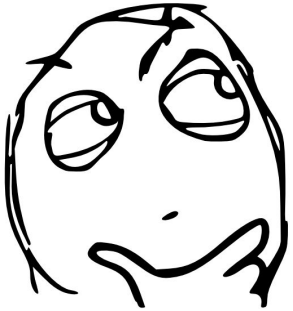
Eval(K, x) -> $F_K(x)$

Constraint query C

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

Input query x

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



adv

Simulator

Simulation-based definition of CHCPRF

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

Constraint query C

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

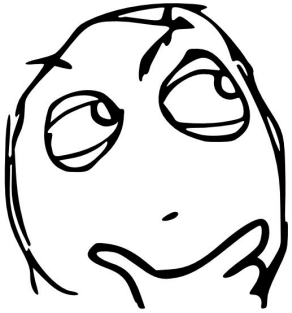
$K^S \leftarrow \mathrm{Sim}(MSK^S, 1^{|C|})$

Input query x

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

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

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:

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

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

Constraint query C

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

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

Input query x

y^S <- $\mathrm{Sim}(\mathrm{MSK}^S, x, C(x))$

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
$Cons(MSK, C_1) \to K[C_1]$	Constraint query C_1	$K_1^S \leftarrow Sim(MSK^S, 1^{\lvert C \rvert})$
$Cons(MSK, C_2) \to K[C_2]$	Constraint query C_2	$K_2^S \leftarrow Sim(MSK^S, 1^{\lvert C \rvert})$

Real

Simulator

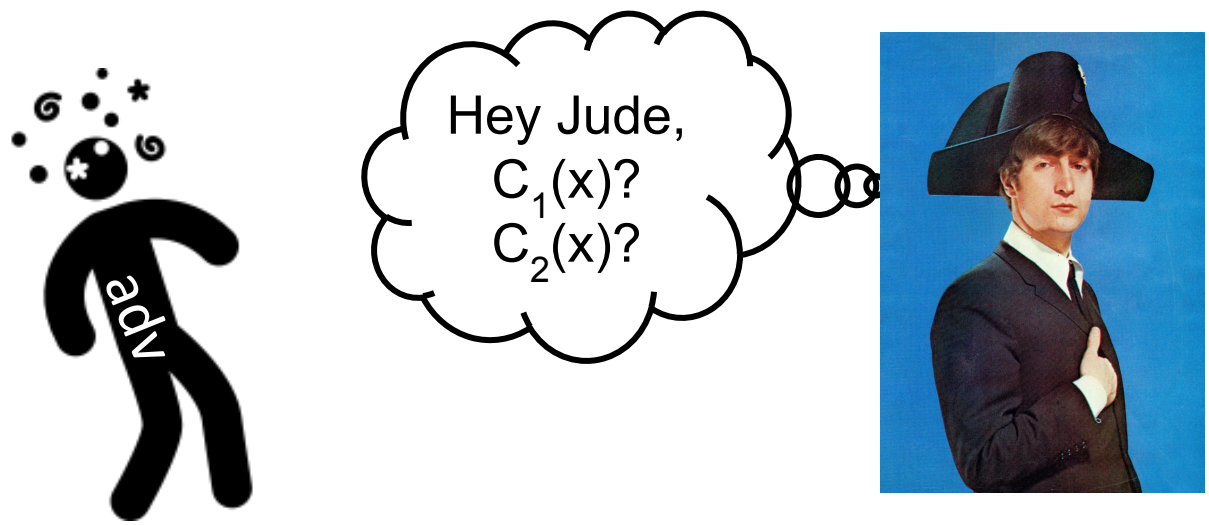
Sim-based definition for 2 keys

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
$Cons(MSK, C_1) \to K[C_1]$	Constraint query C_1	$K_1^S \leftarrow Sim(MSK^S, 1^{\lvert C \rvert})$
$Cons(MSK, C_2) \to K[C_2]$	Constraint query C_2	$K_2^S \leftarrow Sim(MSK^S, 1^{\lvert C \rvert})$

Real



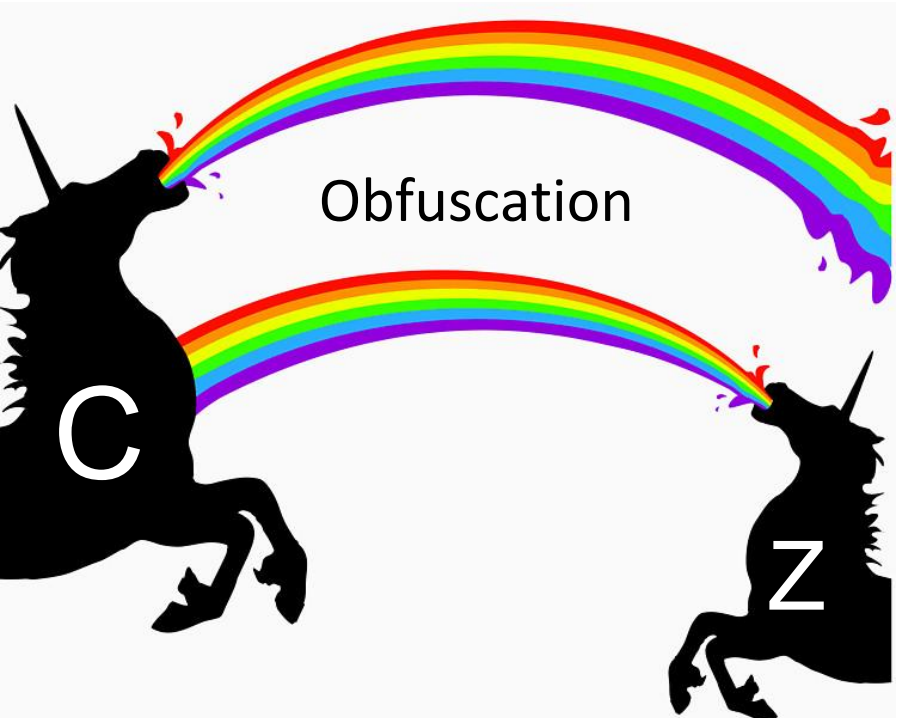
Simulator

Relaxed Sim-based definition for 2 keys

Reminiscent of obfuscation ...

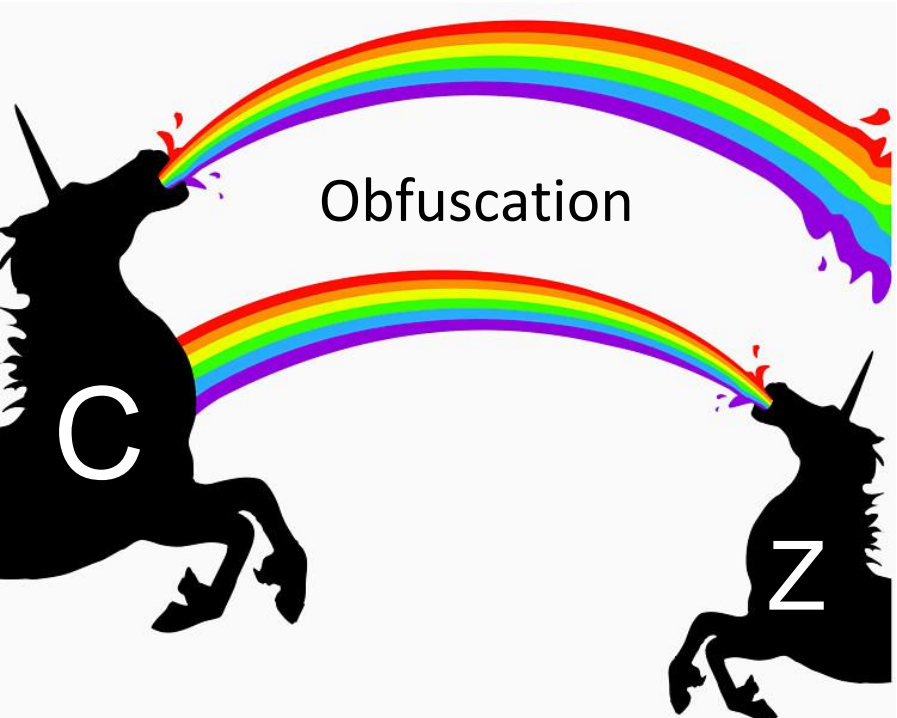
Theorem [CC 17]: Two-key CHCPRF (for function class C) implies obfuscation (for C)

- 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:

Obf = (K[C], K[Z])

Eval(x) = Eval(K[C], x) - Eval(K[Z], x)

Idea implicit from the [GGHRSW13] candidate obfuscation

Pope: wait … you haven’t achieved 2-key security

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

1-key simulation-based definition for CHCPRF.

CHCPRF => Functional encryption

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

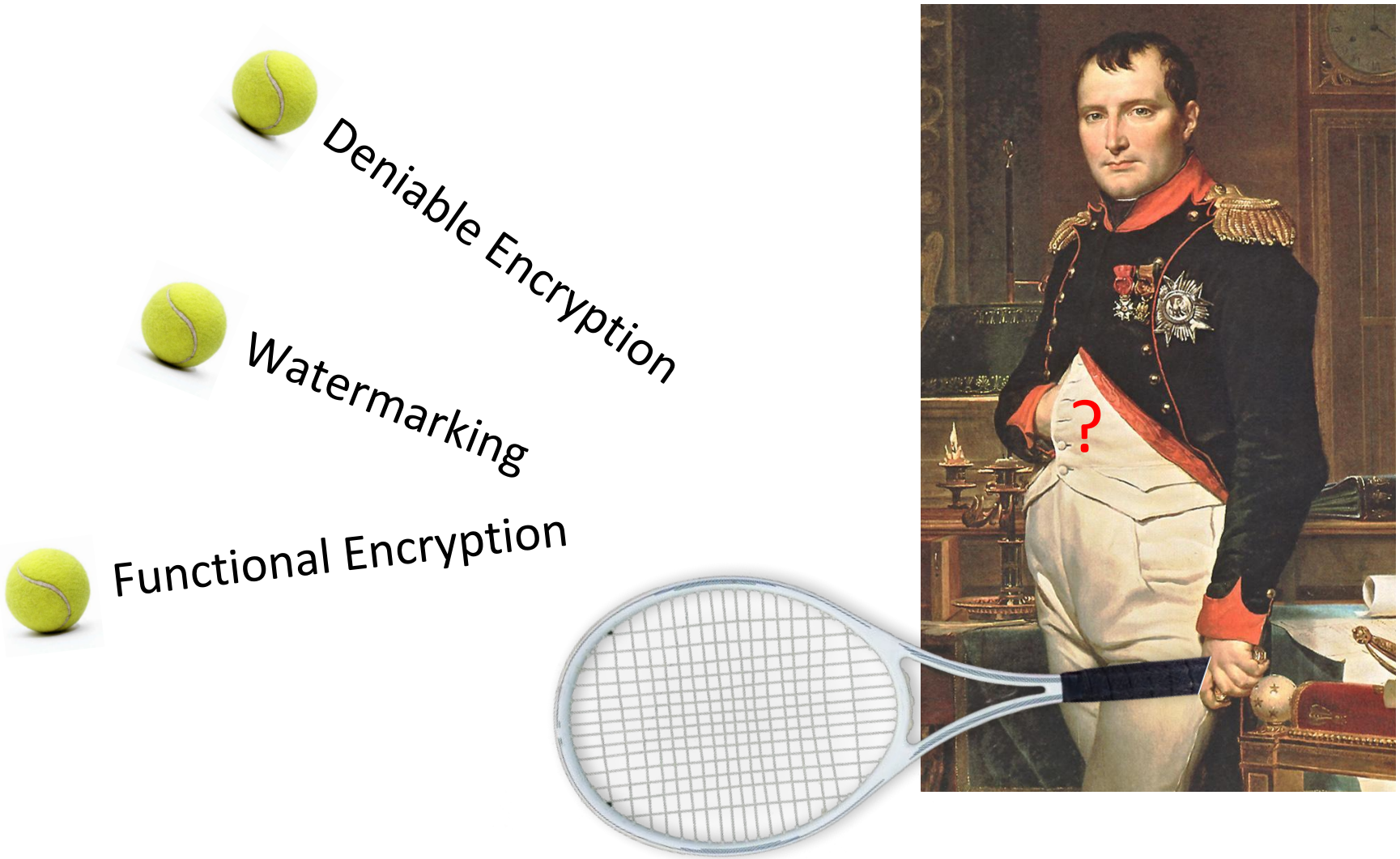
Theorem [CC 17] 1-key sim-based CHCPRF implies 1-key private-key functional encryption (reusable garbled circuits).

Construction: from normal encryption <u>Sym</u> and CHCPRF <u>F</u>

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

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

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



How to construct CHCPRF?

Main construction:

1-key sim-based CHCPRFs for NC1 from Learning With Errors.

Main construction:

1-key sim-based CHCPRFs for NC1 from Learning With Errors.

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.

Banerjee, Peikert, Rosen '12
Subset-product & rounding

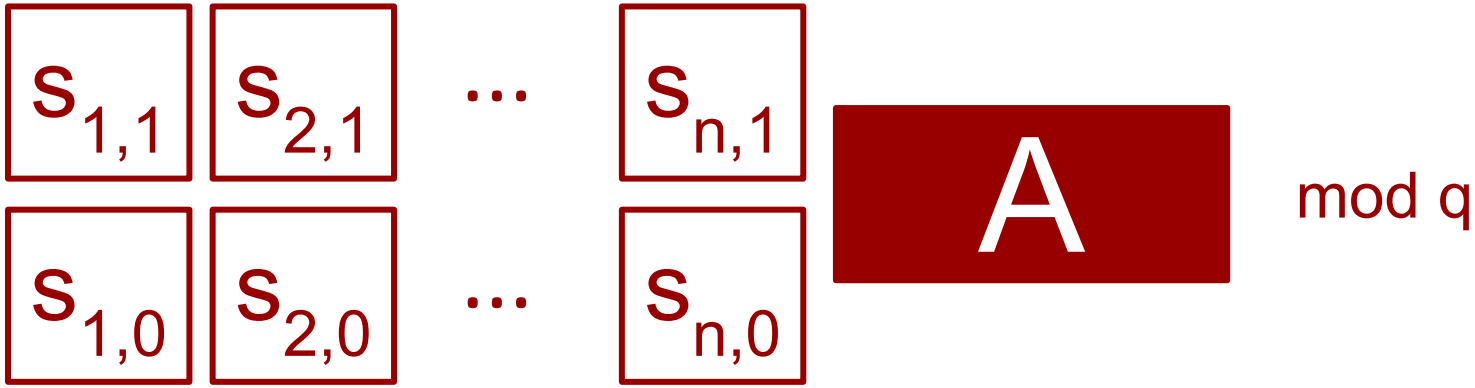
Key: $s_{1,1}$ $s_{2,1}$ … $s_{n,1}$ $s_{1,0}$ $s_{2,0}$ … $s_{n,0}$ A mod q

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

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

Banerjee, Peikert, Rosen '12
Subset-product & rounding

Key:



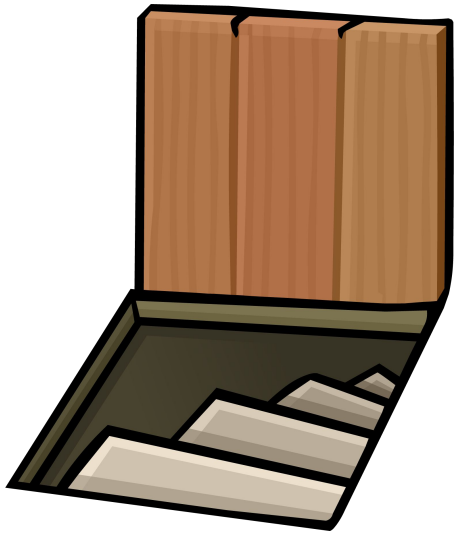
Eval: $F(x) = \{ \prod s_{i,xi} A \}_2$

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

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

Tool 1: GGH15 encoding [Gentry, Gorbunov, Halevi 15]

A with

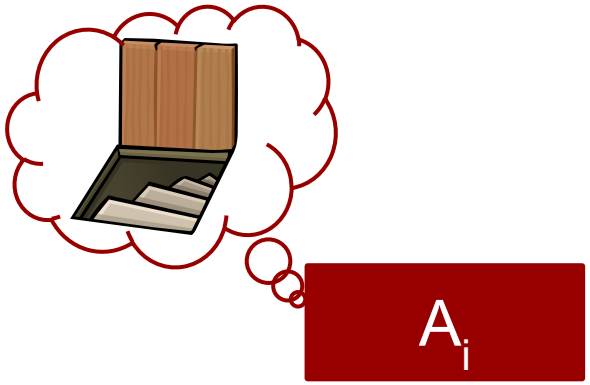


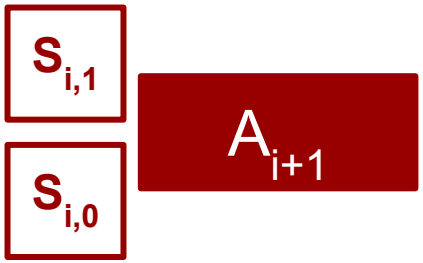
Trapdoor

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

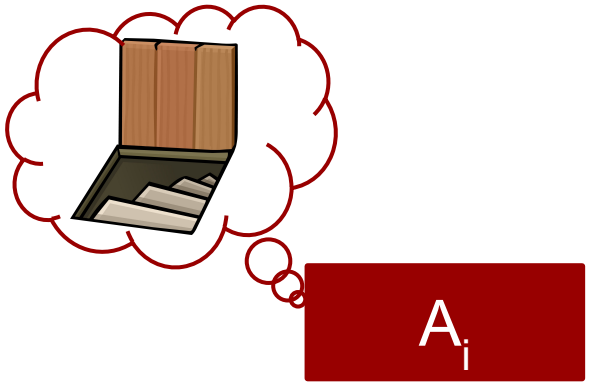
Can sample small preimage from Gaussian [Klein ‘00, GPV’08]

GGH15 encoding for the ith hop:





GGH15 encoding for the ith hop:



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

$s_{i,1}$

$s_{i,0}$

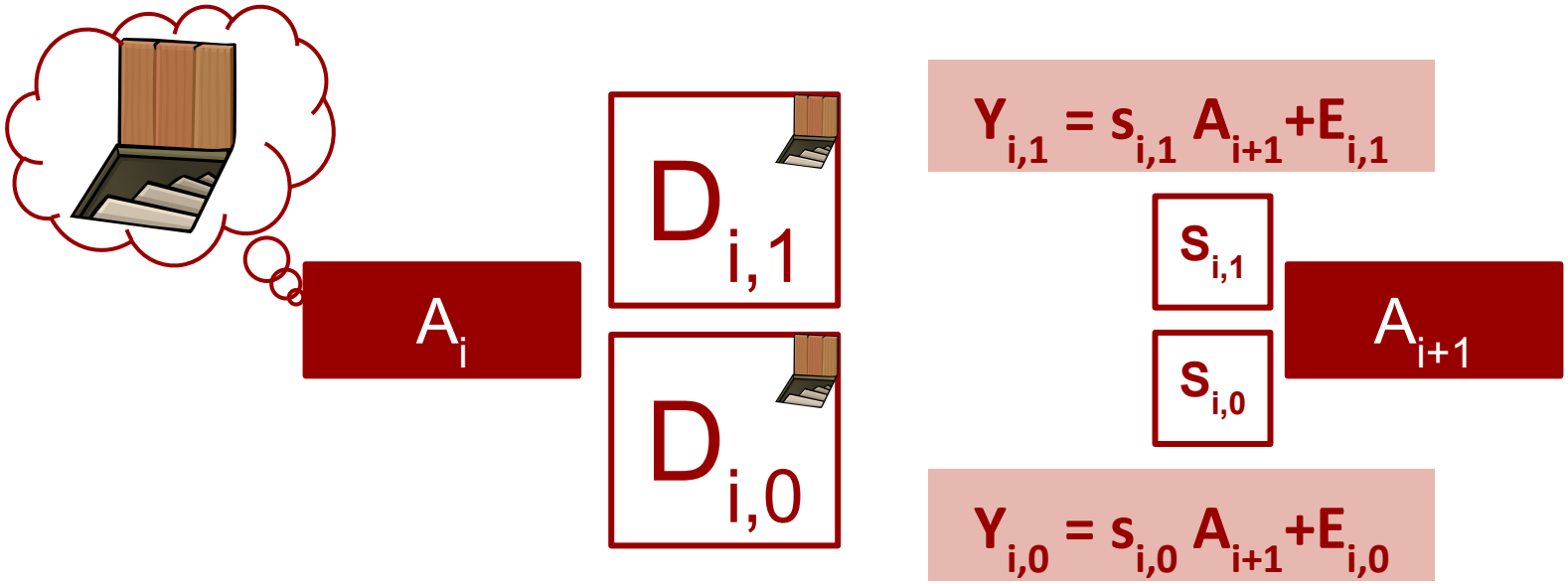
A_{i+1}

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

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

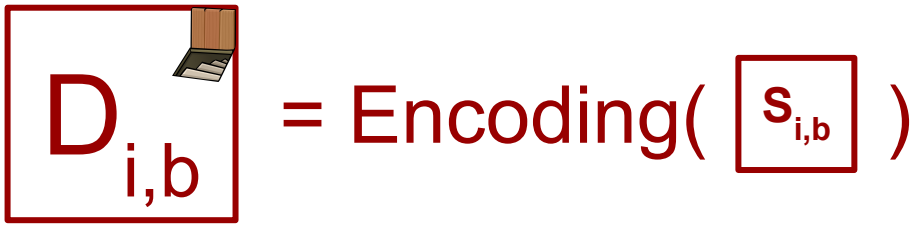
1. $Y_{i,b} = s_{i,b} A_{i+1} + E_{i,b}$

GGH15 encoding for the ith hop:

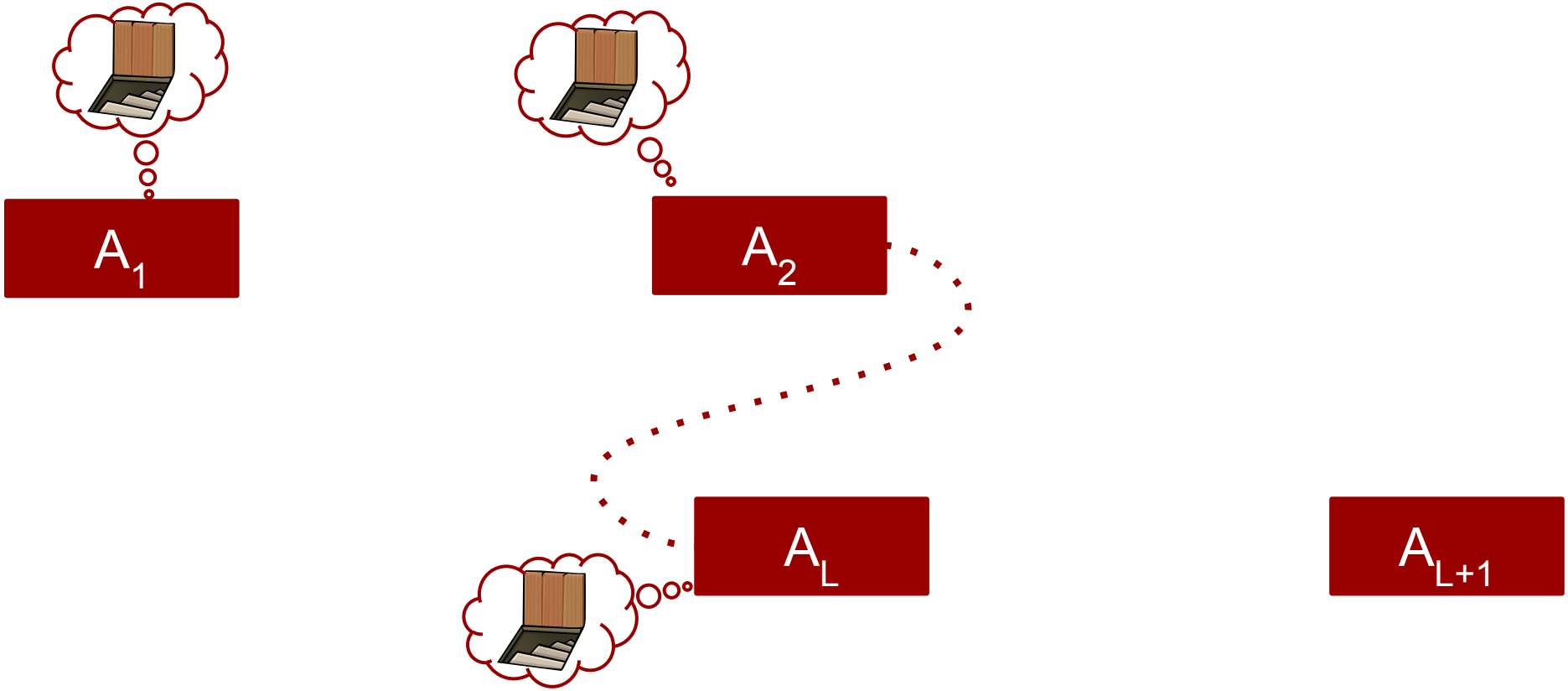


Encode($s_{i,b}$): 2 steps

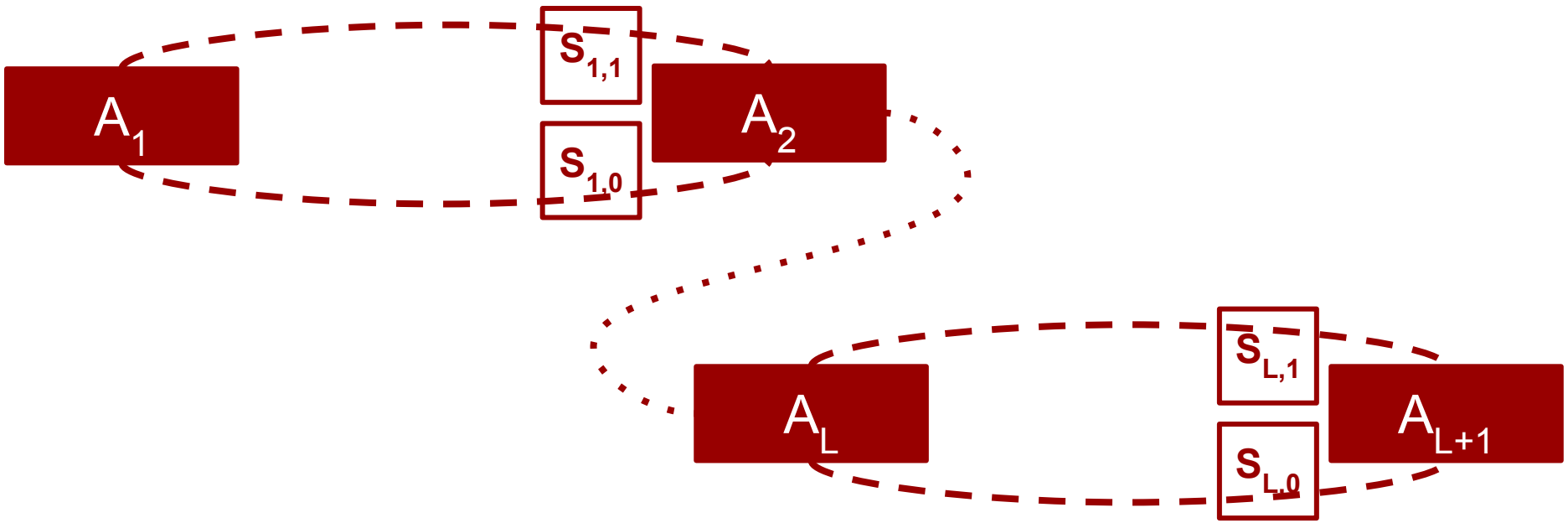
1. $Y_{i,b} = s_{i,b} A_{i+1} + E_{i,b}$
2. Sample (by the trapdoor of A_i) small $D_{i,b}$ s.t. $A_i D_{i,b} = Y_{i,b}$



GGH15 for L hops:

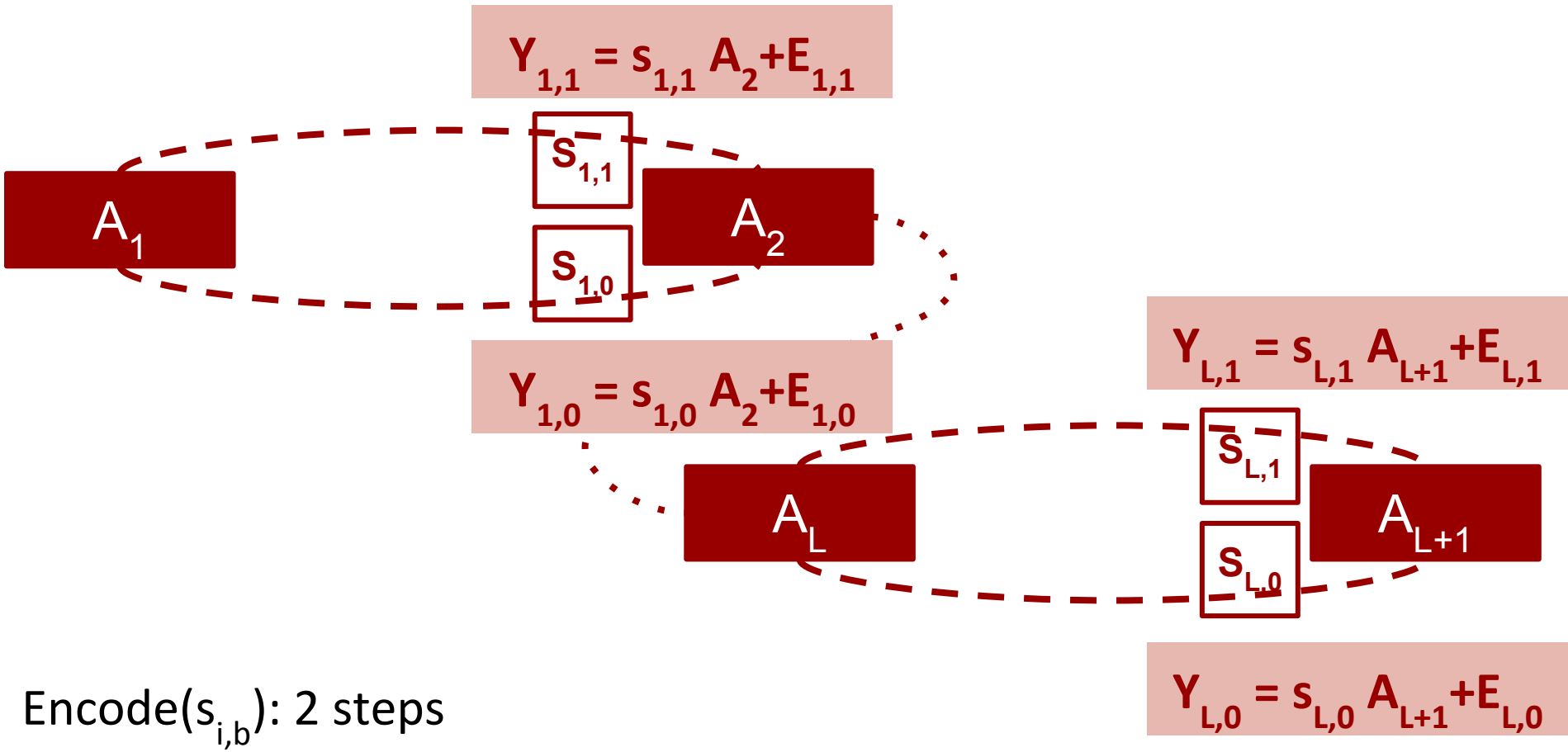


GGH15 for L hops:



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

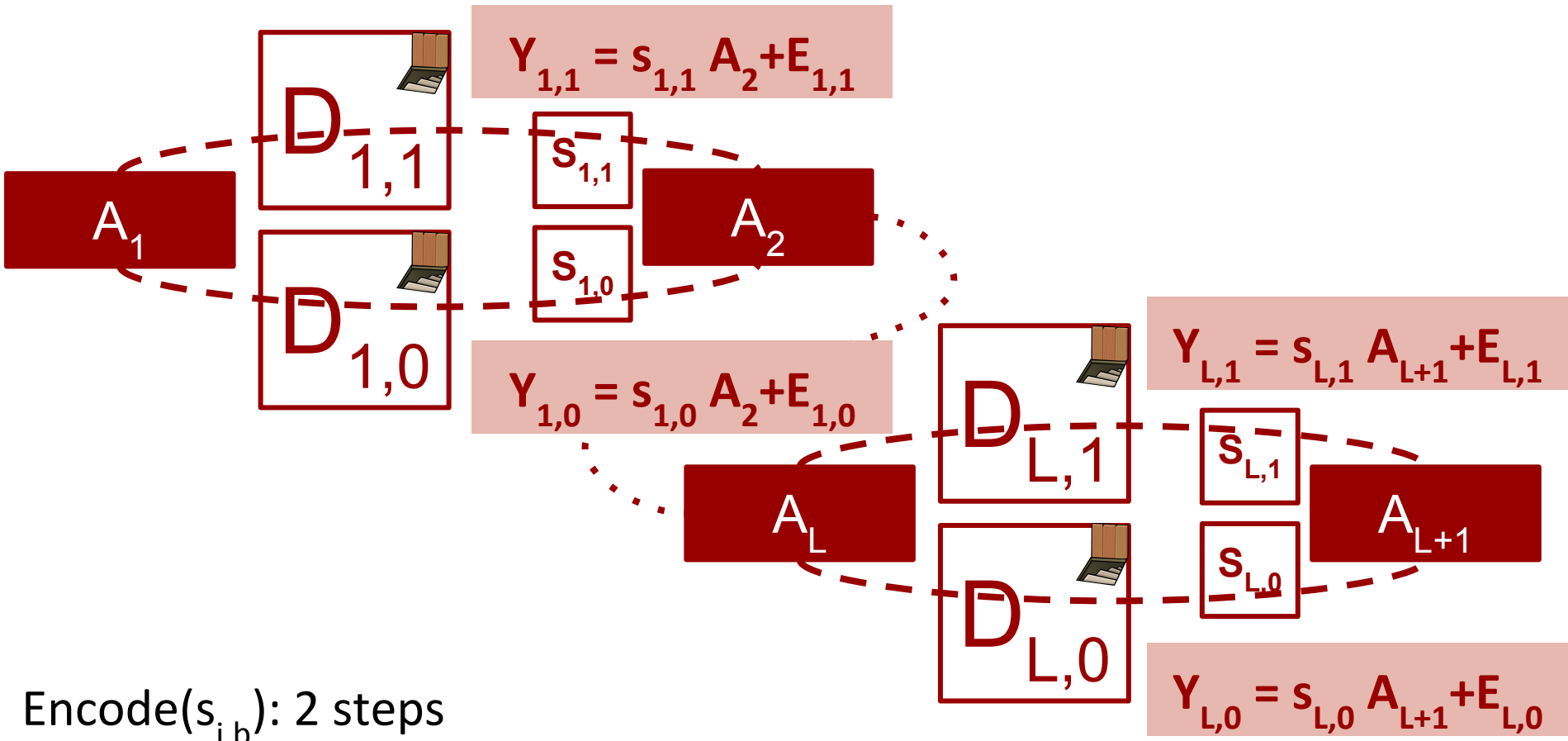
GGH15 for L hops:



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

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

GGH15 for L hops:

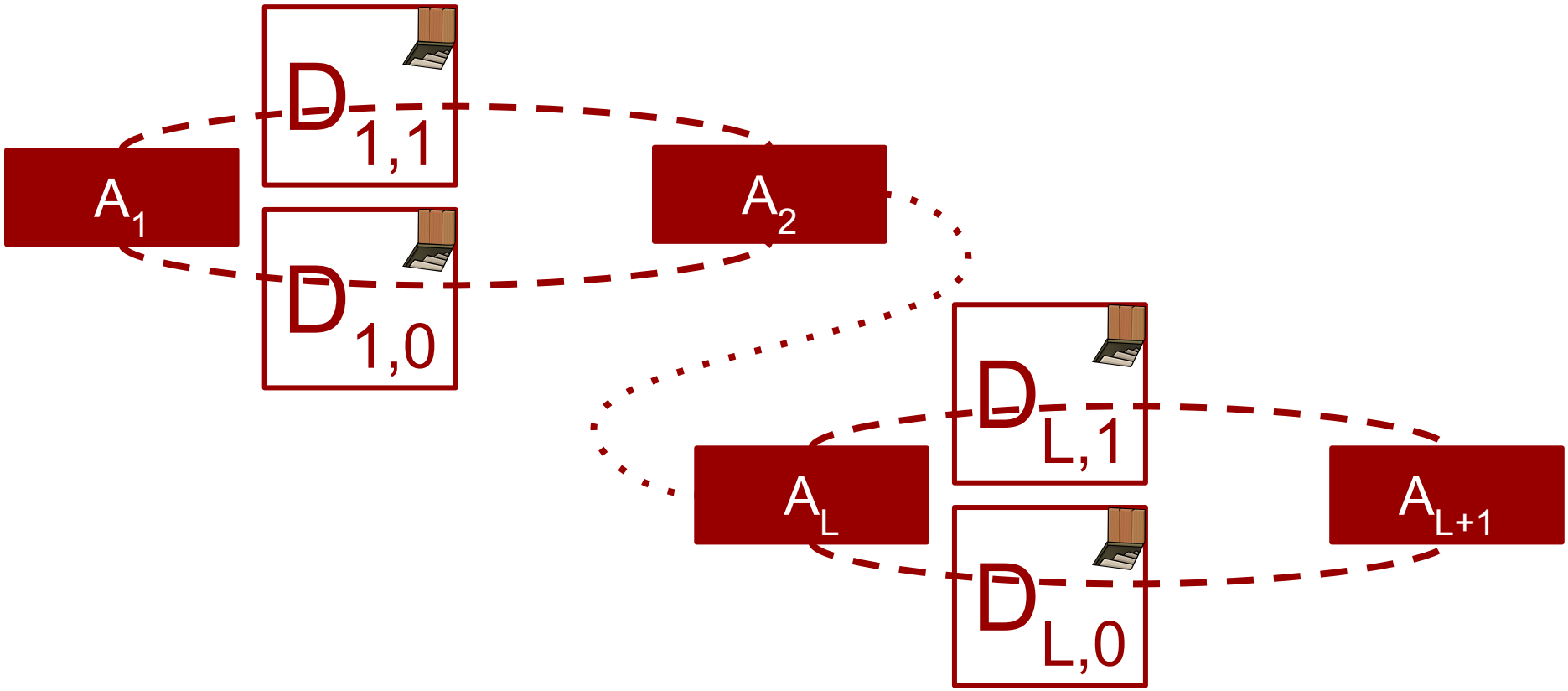


Encode($s_{i,b}$): 2 steps

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 Encoding($s_{i,b}$)

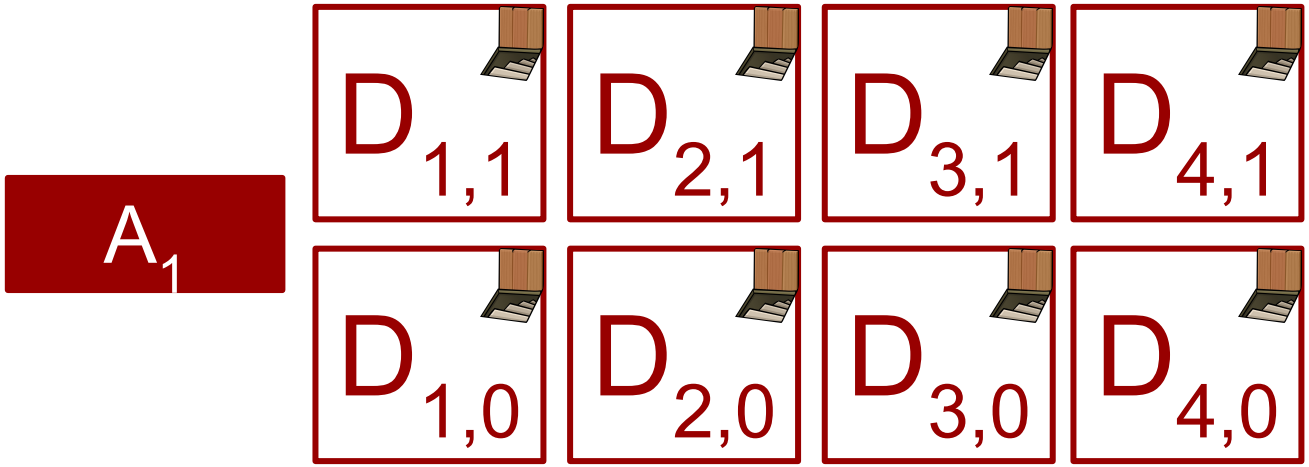
GGH15 for L hops:



Review: What are public

Understanding the functionality of GGH15

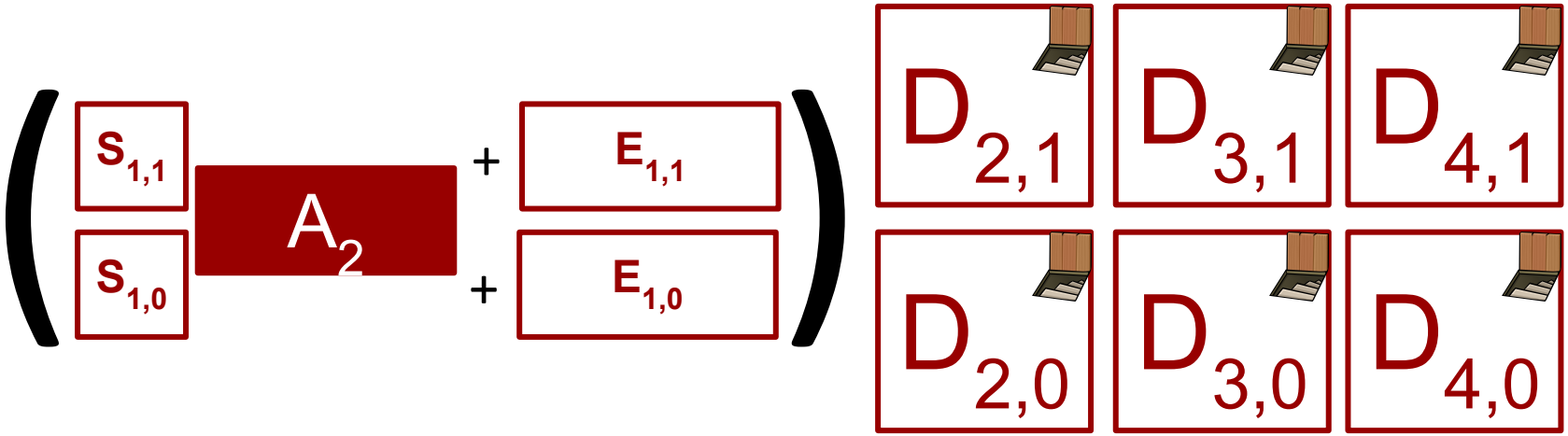
Evaluation of GGH15 (prove by example):



Eval(0110)

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

Evaluation of GGH15 (prove by example):

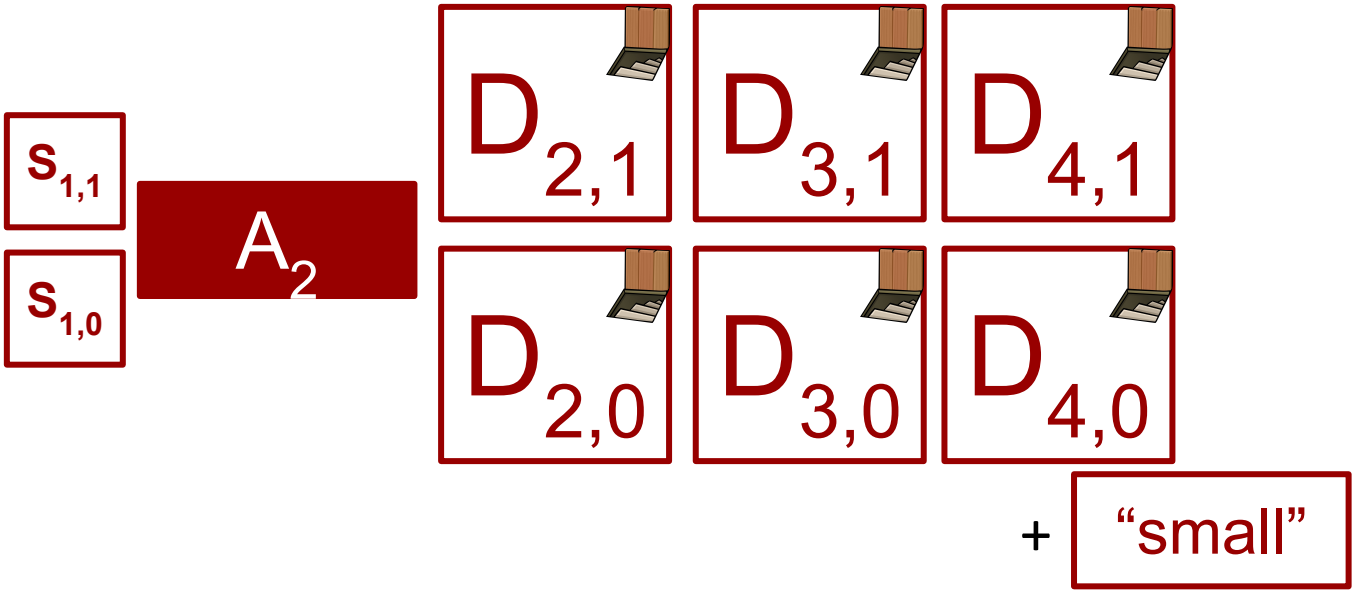


Eval(0110)

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

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

Evaluation of GGH15 (prove by example):



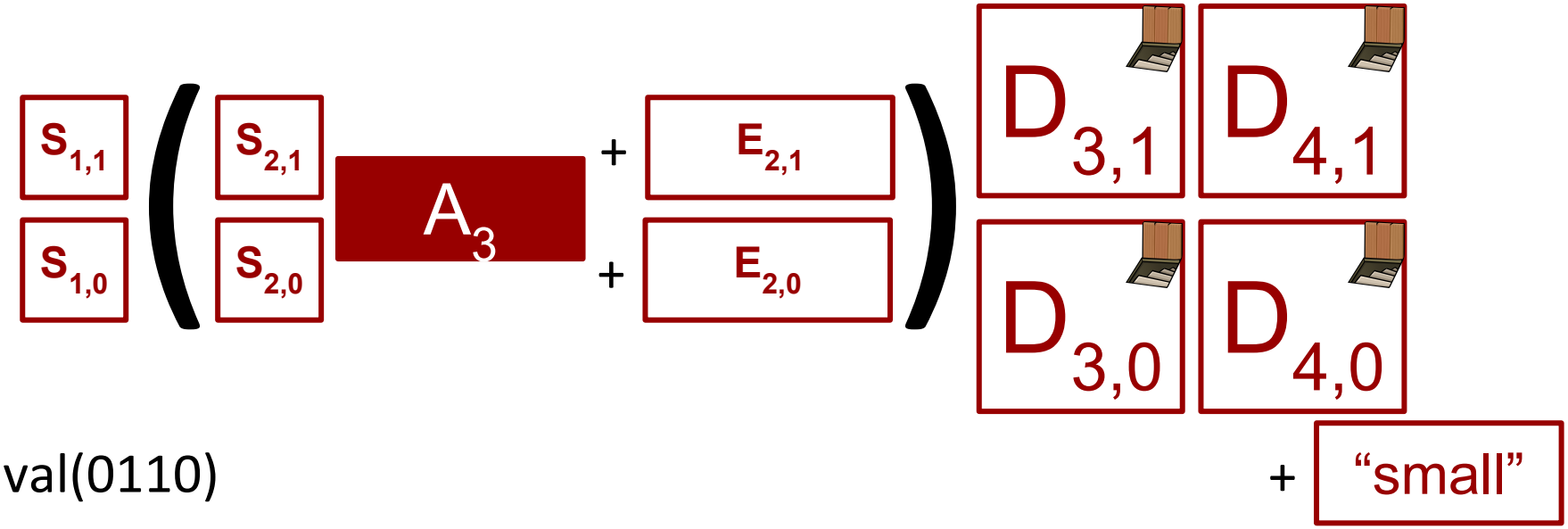
Eval(0110)

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

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

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

Evaluation of GGH15 (prove by example):



Eval(0110)

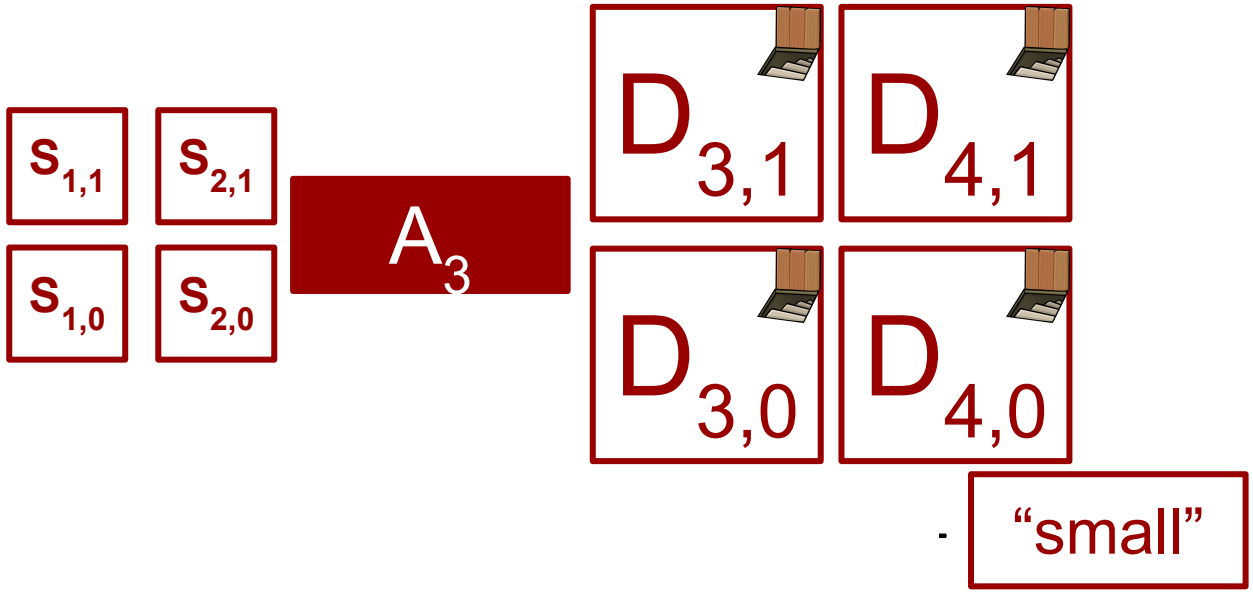
$= A_1 D_{1,0} D_{2,1} D_{3,1} D_{4,0}$

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

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

$= s_{1,0} (s_{2,1} A_3 + E_{2,1}) D_{3,1} D_{4,0}$ + "small"

Evaluation of GGH15 (prove by example):



Eval(0110)

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

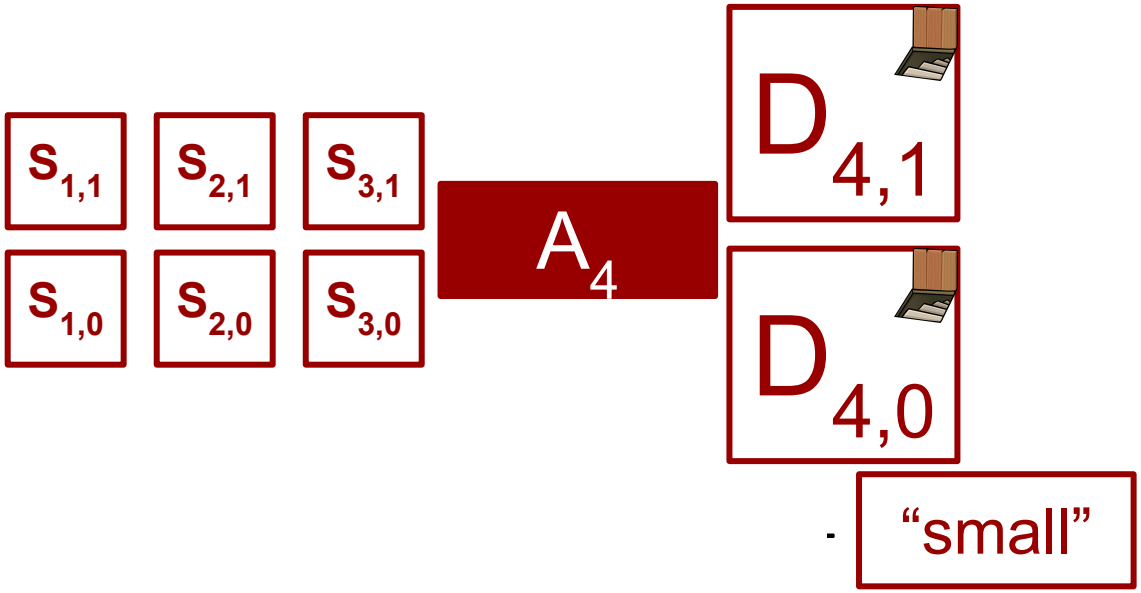
$= (s_{1,0} A_2 + E_{1,0}) D_{2,1} D_{3,1} D_{4,0}$

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

$= s_{1,0} (s_{2,1} A_3 + E_{2,1}) D_{3,1} D_{4,0}$ + "small"

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

Evaluation of GGH15 (prove by example):



Eval(0110)

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

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

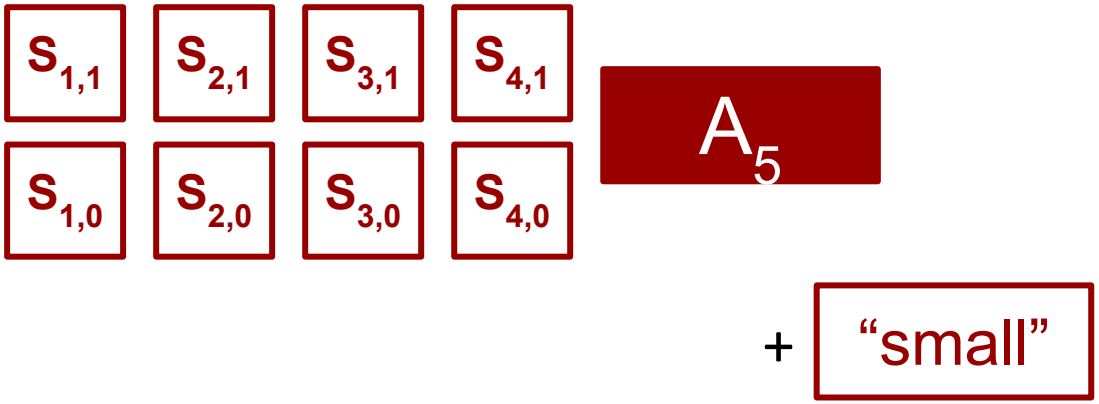
$= s_{1,0} A_2 D_{2,1} D_{3,1} D_{4,0}$ + "small"

$= s_{1,0} (s_{2,1} A_3 + E_{2,1}) D_{3,1} D_{4,0}$ + "small"

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

$= s_{1,0} s_{2,1} s_{3,1} A_4 D_{4,0}$ + "still smallish"

Evaluation of GGH15 (prove by example):



Eval(0110)

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

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

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

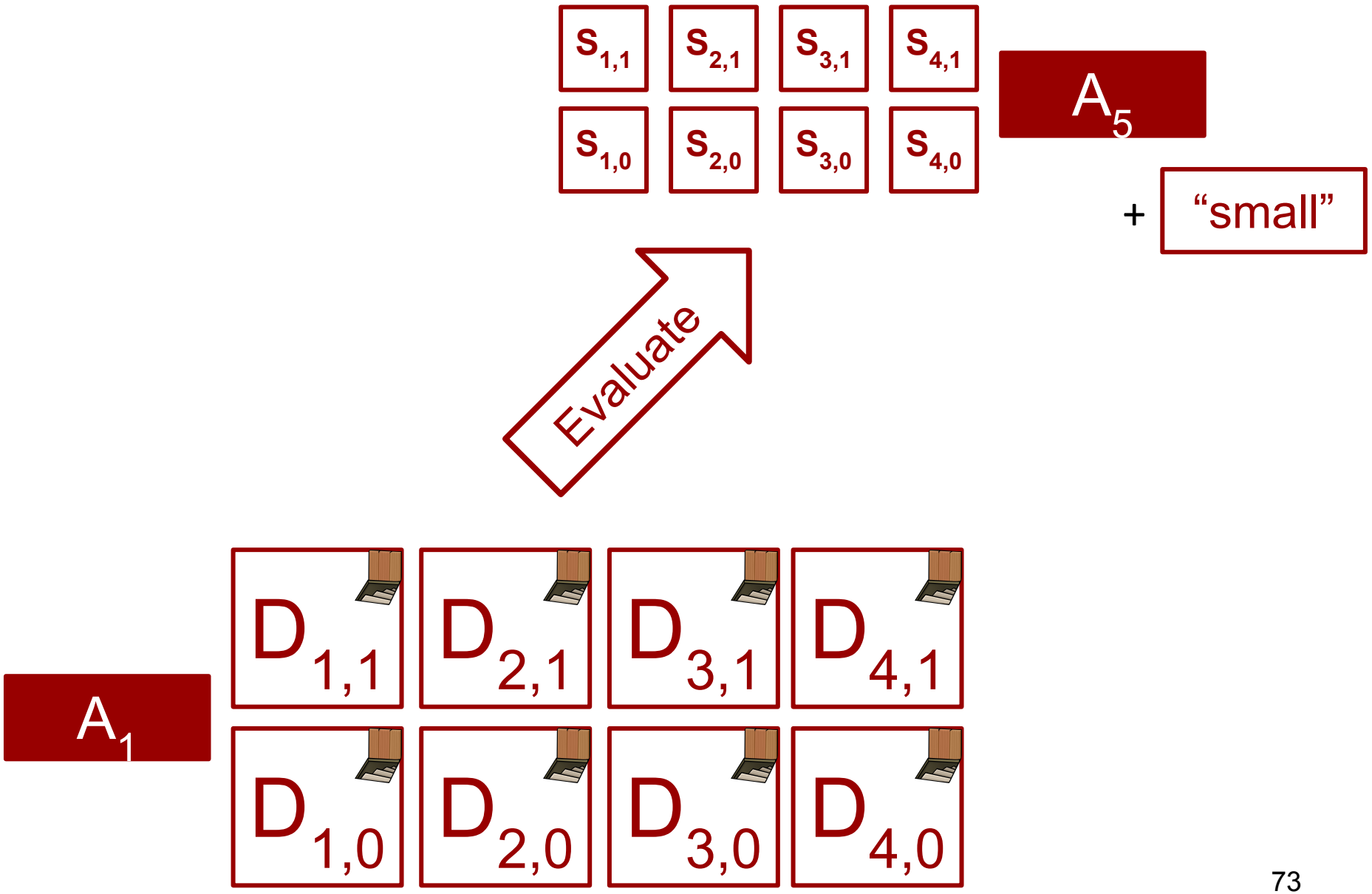
$= s_{1,0} (s_{2,1} A_3 + E_{2,1}) D_{3,1} D_{4,0}$ + "small"

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

$= s_{1,0} s_{2,1} s_{3,1} A_4 D_{4,0}$ + "still smallish"

$= s_{1,0} s_{2,1} s_{3,1} s_{4,0} A_5$ + "small"

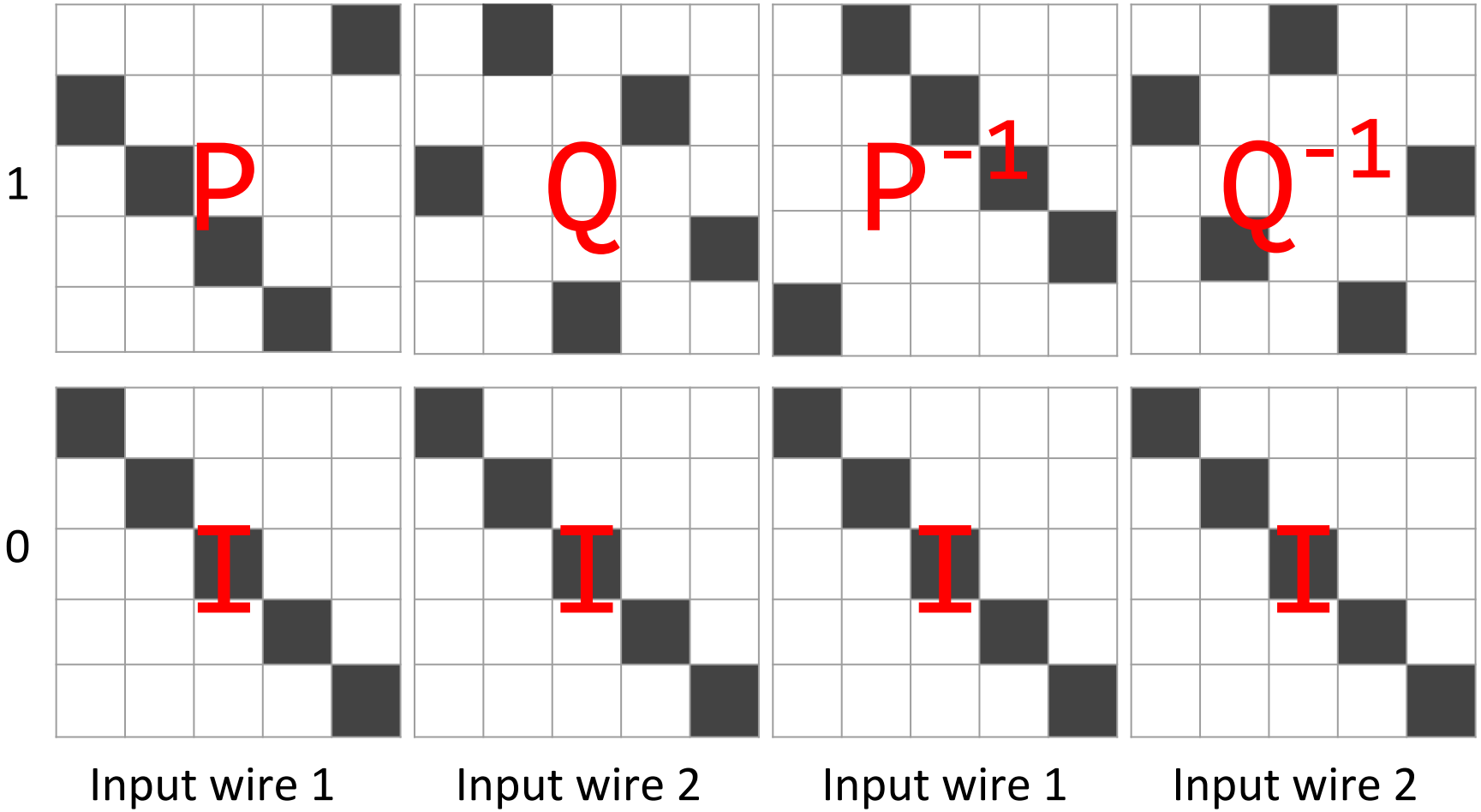
Evaluation of GGH15 (prove by example):



Tool 2: 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



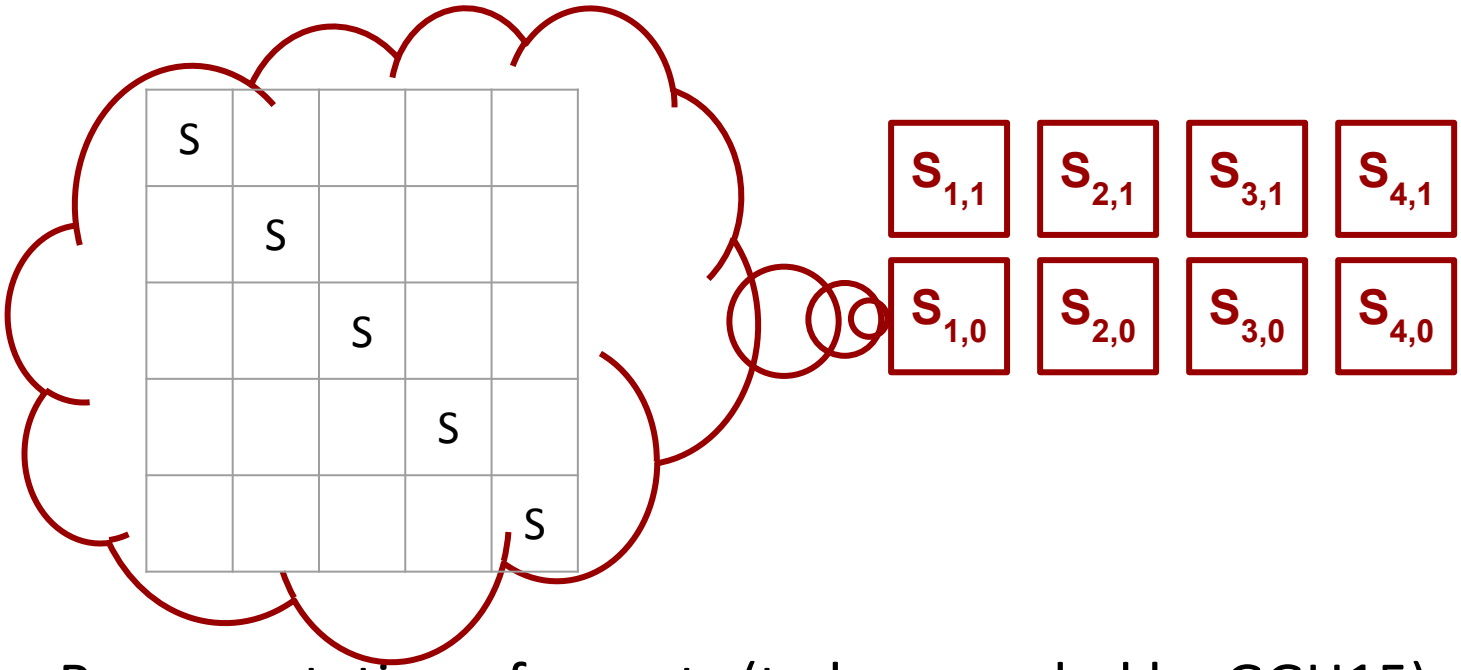
Our construction only work for certain representation of Barrington (e.g. S_5)

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: $\prod B_{z(i),x\_z(i)} = I$ 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	
				s

$P \otimes s =$

				s
s				
	s			
		s		
			s	

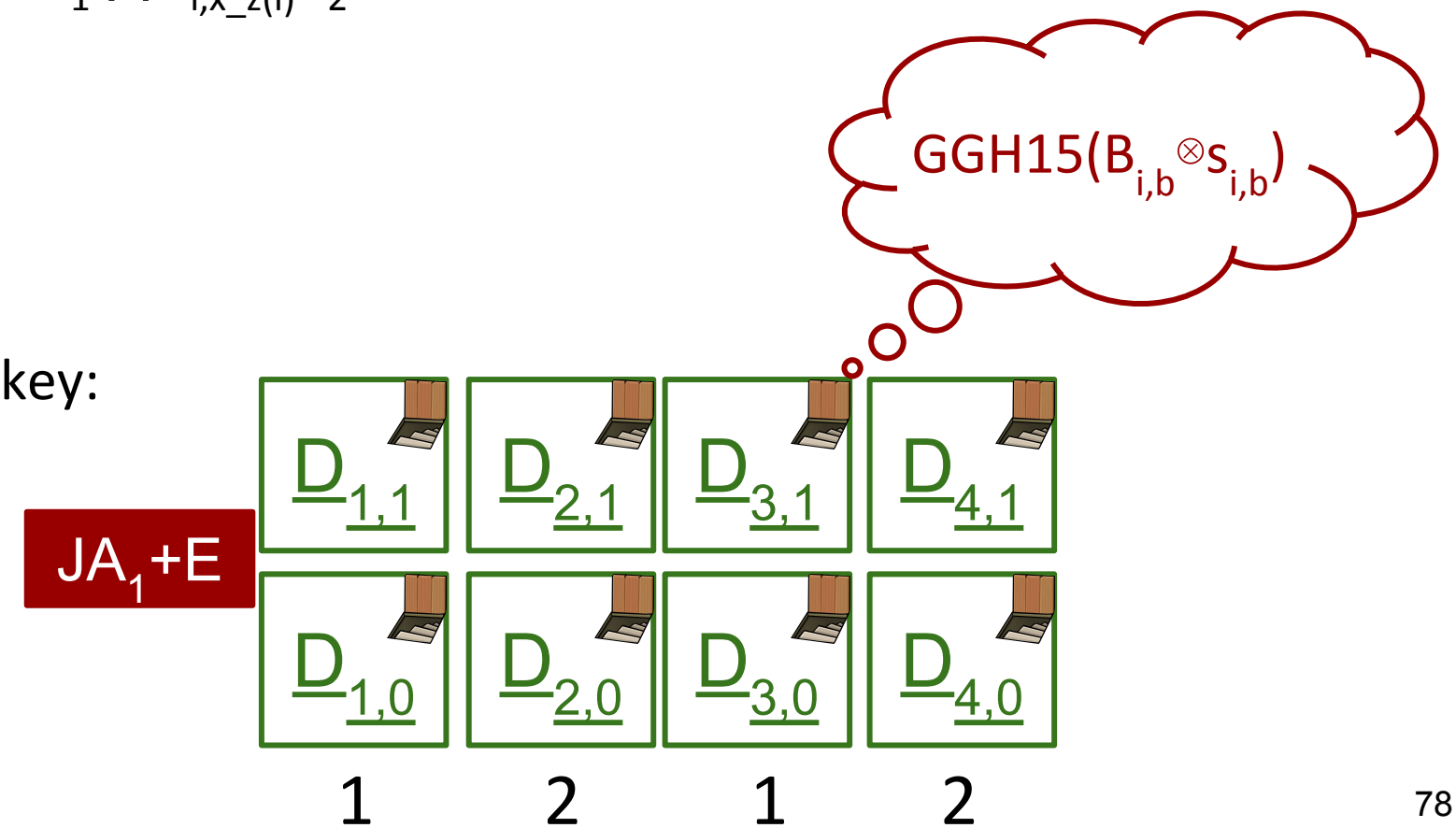
NC1-CHCPRF from GGH15

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

Master secret key: trapdoors of $A_1 \ldots A_L$, $s_{1,0}$, $s_{1,1}$, ..., $s_{L,0}$, $s_{L,1}$, & 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]->[n] is the step-to-input mapping)



NC1-CHCPRF from GGH15

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

Master secret key: trapdoors of $A_1 \ldots A_L$, $s_{1,0}$, $s_{1,1}$, ..., $s_{L,0}$, $s_{L,1}$, & 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]->[n] is the step-to-input mapping)

Functionality check:

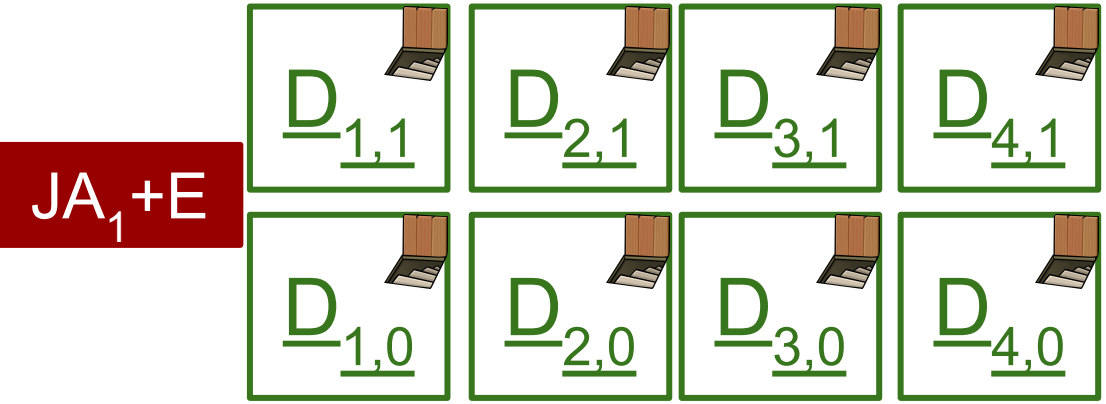
when C(x)=1,

$$F(x) = \{ JA_1 \prod D_{i,x\_z(i)} \}_2$$
$$= \{ J(I \otimes \prod s_{i,x\_z(i)}) A_{L+1} + \text{small noise} \}_2$$
$$\approx_s \{ J(I \otimes \prod s_{i,x\_z(i)}) A_{L+1} \}_2$$

when C(x)=0,

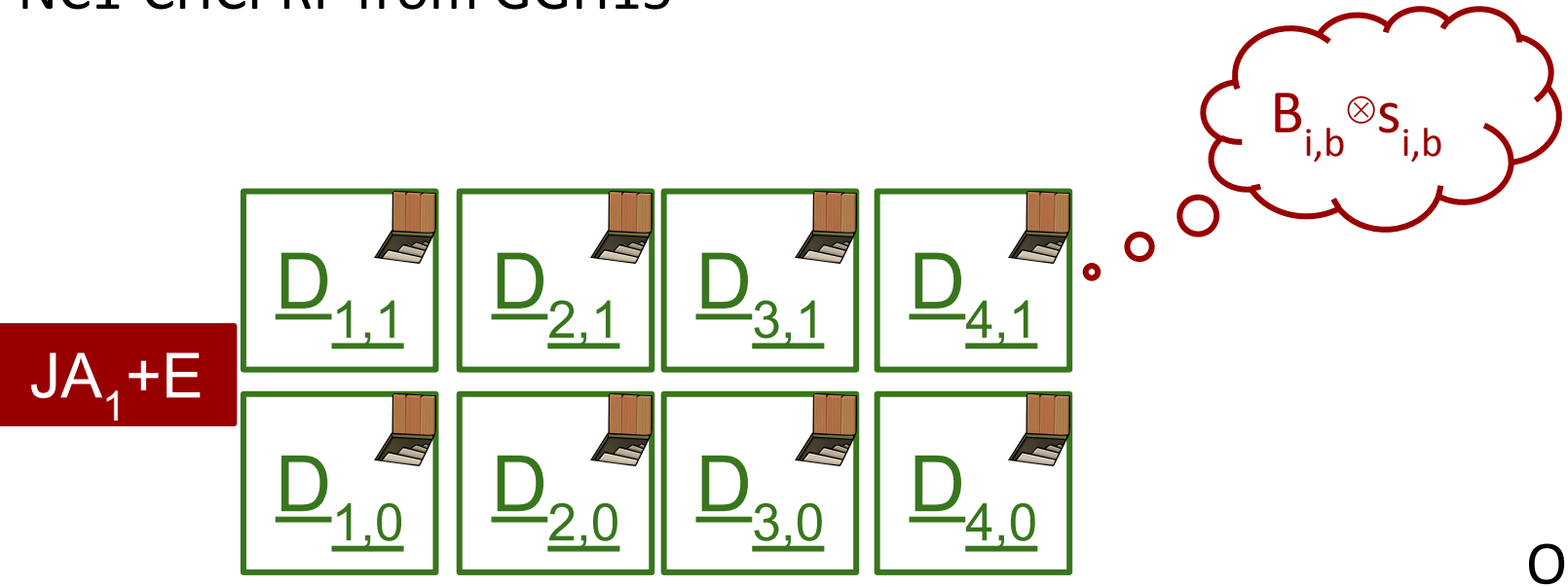
$$F(x) = \{ JA_1 \prod D_{i,x\_z(i)} \}_2$$
$$= \{ J(C \otimes \prod s_{i,x\_z(i)}) A_{L+1} + \text{small noise} \}_2$$
$$\approx_s \{ J(C \otimes \prod s_{i,x\_z(i)}) A_{L+1} \}_2$$

NC1-CHCPRF from GGH15 *



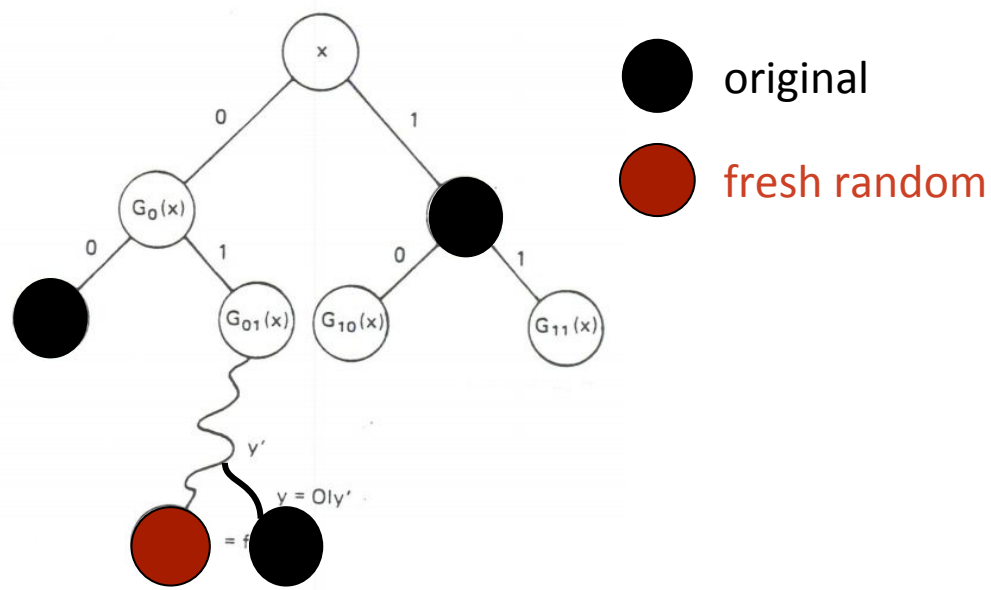
Compare to GGM

NC1-CHCPRF from GGH15 *



Our CHCPRF

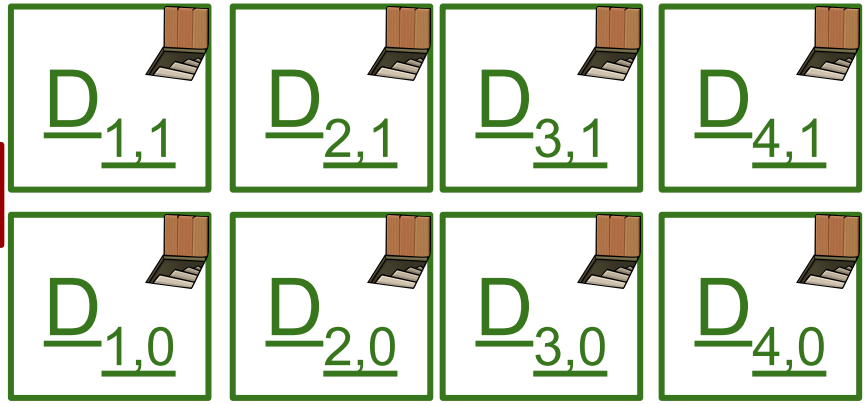
Compare to GGM



NC1-CHCPRF from GGH15 *

Example: C(x)=0 iff x1=x2=1 query x=11

JA_1+E



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

Real

What are we trying to simulate?

Simulator

NC1-CHCPRF from GGH15 *

Example: C(x)=0 iff x1=x2=1 query x=11

JA_1+E

$D_{1,1}$ $D_{2,1}$ $D_{3,1}$ $D_{4,1}$

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

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

Real

A_J

$D_{1,1}$ $D_{2,1}$ $D_{3,1}$ $D_{4,1}$

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

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

Simulator

Proof by example with 1 input query

JA_1+E

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

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

Real

A_J

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

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

Proof of NC1-CHCPRF:

- LWE with permutation-structured secrets + GPV sampling lemma
- Close the trapdoor from the right to the left

Thanks for your time

More in https://eprint.iacr.org/2017/143