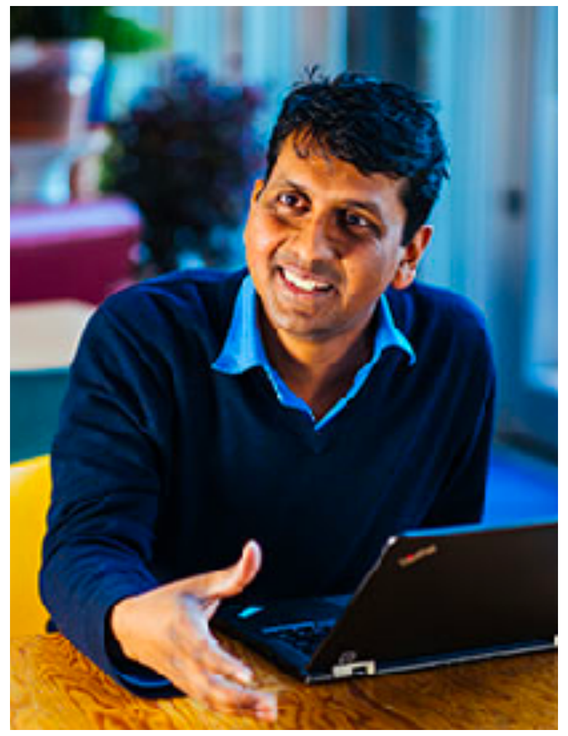
Pseudorandom Obfuscation

And applications









Pedro Branco
Bocconi

Nico Döttling
CISPA

Abhishek Jain
JHU and NTT Research

Giulio Malavolta
Bocconi

Surya Mathialagan
MIT → NTT Research

Spencer Peters
Cornell → Meta

Vinod Vaikuntanathan
MIT

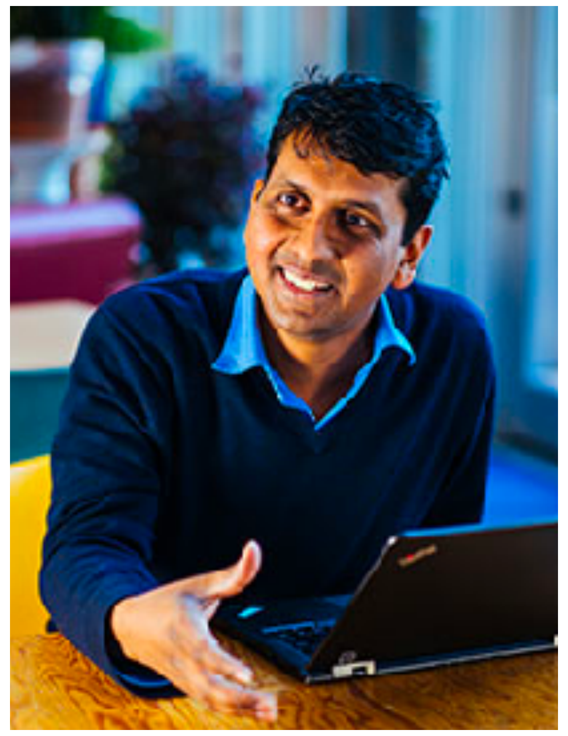
Pseudorandom Obfuscation

And applications









Pedro Branco
Bocconi

Nico Döttling
CISPA

Abhishek Jain
JHU and NTT Research

Giulio Malavolta
Bocconi

Surya Mathialagan
MIT → NTT Research

Spencer Peters
Cornell → Meta

Vinod Vaikuntanathan
MIT

Thank you Nico for many of these slides!

Indistinguishability Obfuscation

[BGI+01,GGH+13]

Indistinguishability Obfuscation

[BGI+01,GGH+13]

C_0 C_1

Indistinguishability Obfuscation

[BGI+01,GGH+13]

$$C_0 \equiv C_1$$

Indistinguishability Obfuscation

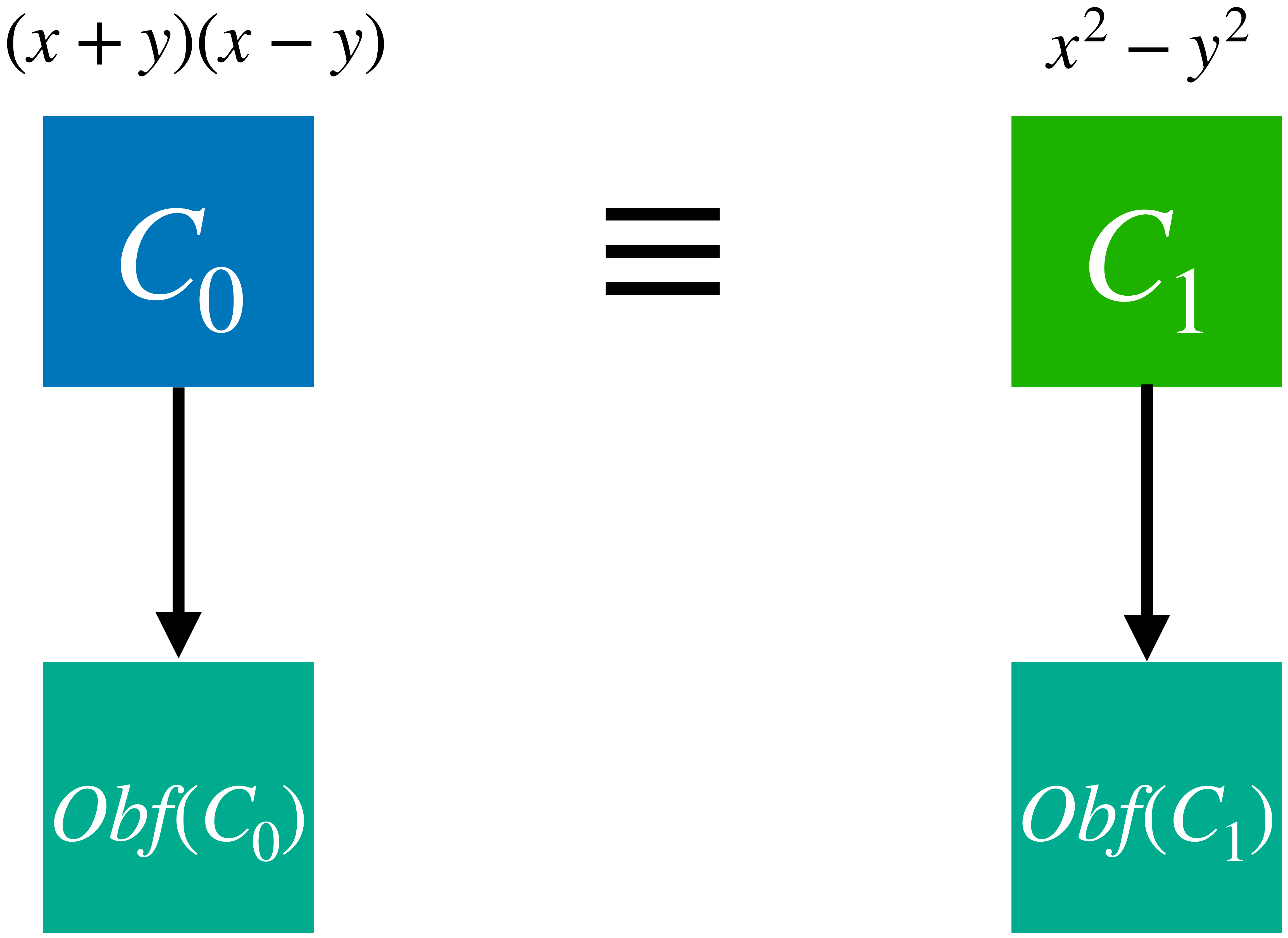
[BGI+01,GGH+13]

$(x+y)(x-y)$ $\qquad$ $x^2 - y^2$

$$C_0 \equiv C_1$$

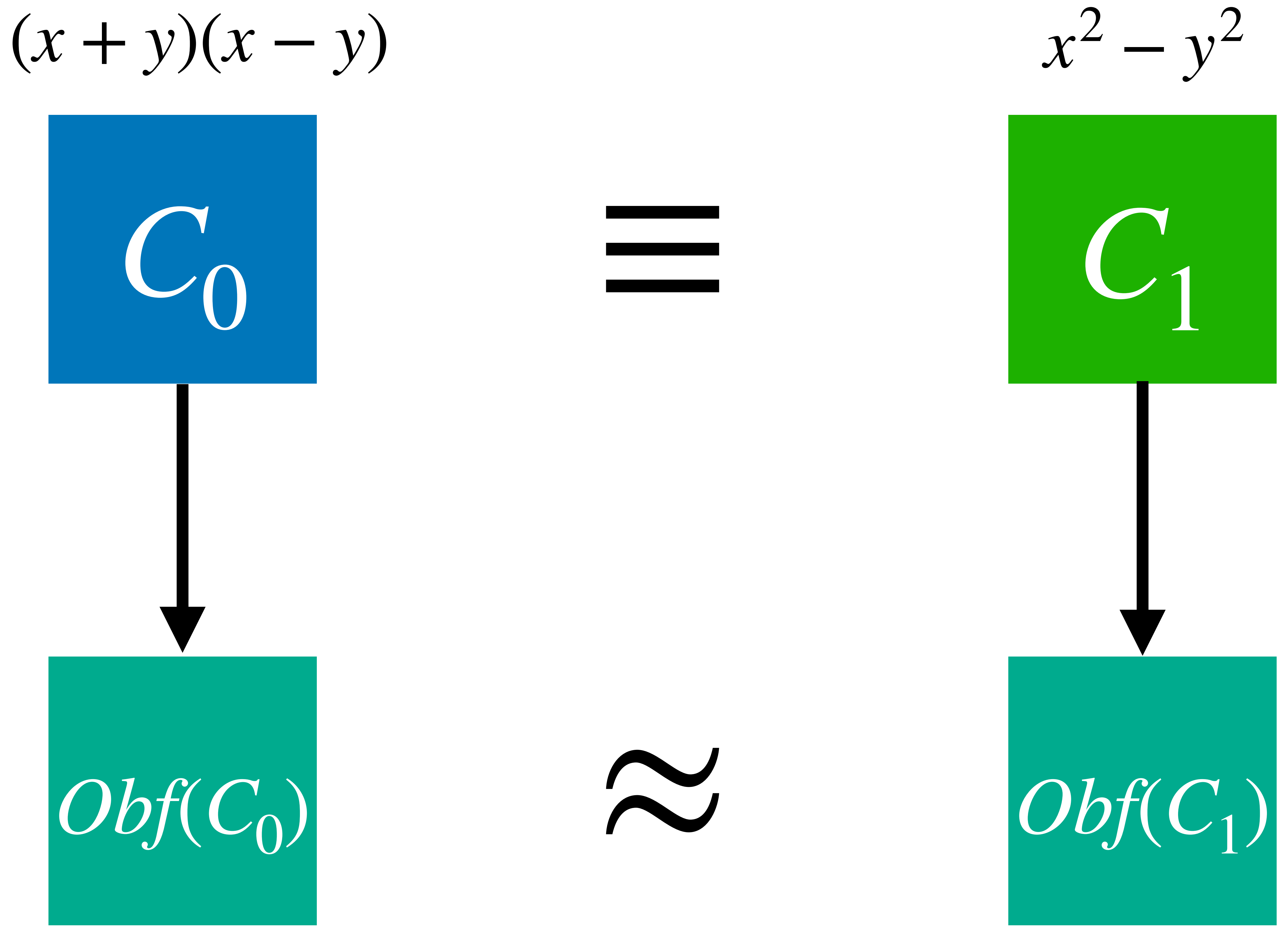
Indistinguishability Obfuscation

[BGI+01,GGH+13]



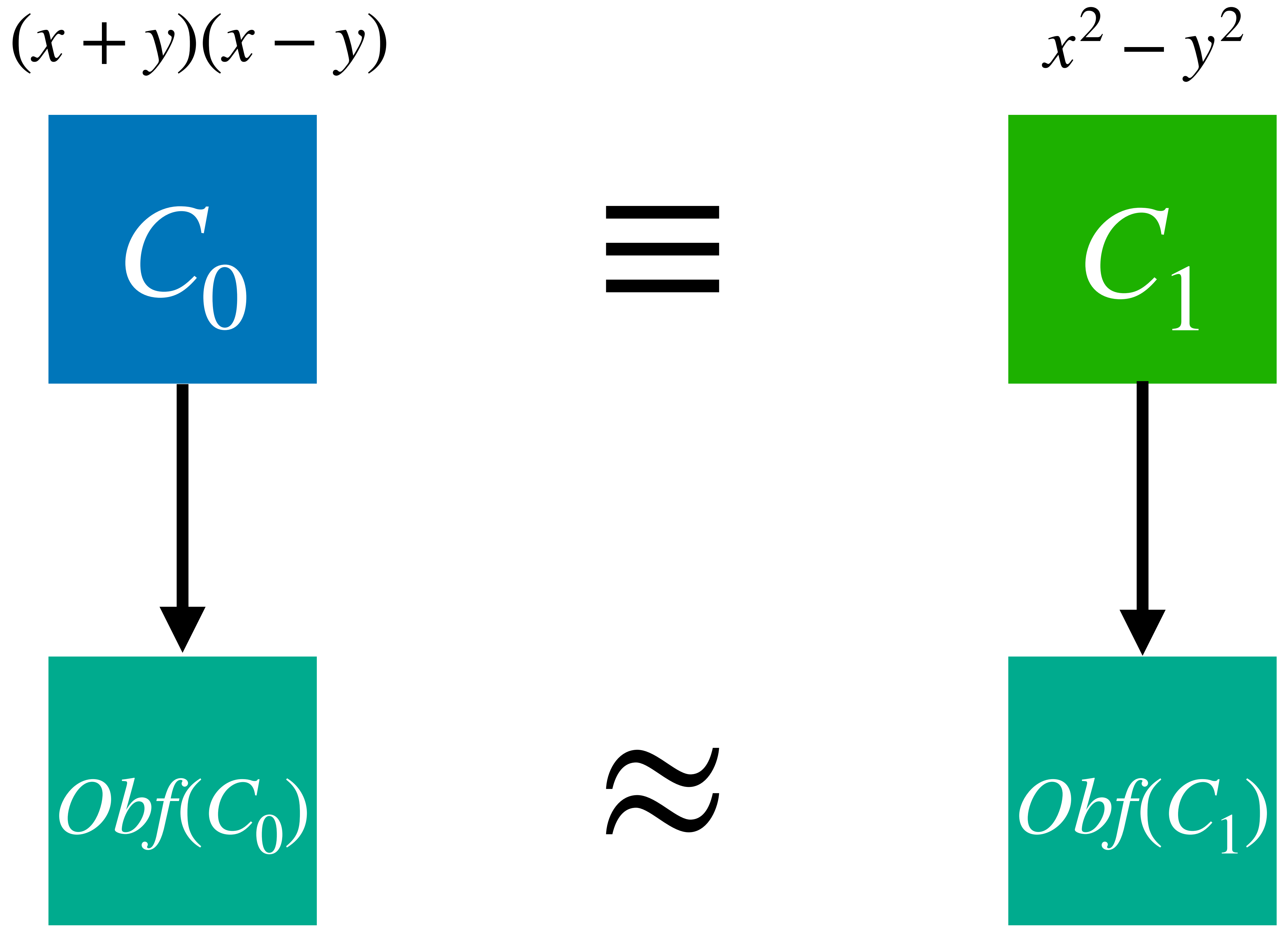
Indistinguishability Obfuscation

[BGI+01,GGH+13]



Indistinguishability Obfuscation

[BGI+01,GGH+13]



- Currently only candidates [JLS21, JLS22, RVV24] from standard assumptions follows the **[JLS20]** mould

iO: Swiss Army knife of Cryptography

(subexponential) iO (+ standard assumptions) is “crypto complete”

iO: Swiss Army knife of Cryptography

(subexponential) iO (+ standard assumptions) is “crypto complete”

- PKE, Short Sigs, Perfect NIZKs (non-adaptive SNARGs), OT, Deniable Enc [SW’14]

iO: Swiss Army knife of Cryptography

(subexponential) iO (+ standard assumptions) is “crypto complete”

- PKE, Short Sigs, Perfect NIZKs (non-adaptive SNARGs), OT, Deniable Enc [SW’14]
- FHE [CLTV’15]

iO: Swiss Army knife of Cryptography

(subexponential) iO (+ standard assumptions) is “crypto complete”

- PKE, Short Sigs, Perfect NIZKs (non-adaptive SNARGs), OT, Deniable Enc [SW’14]
- FHE [CLTV’15]
- WE [GGHRSW’14]

iO: Swiss Army knife of Cryptography

(subexponential) iO (+ standard assumptions) is “crypto complete”

- PKE, Short Sigs, Perfect NIZKs (non-adaptive SNARGs), OT, Deniable Enc [SW’14]
- FHE [CLTV’15]
- WE [GGHRSW’14]
- Adaptive SNARGs [WW24, 25, WZ24]

iO: Swiss Army knife of Cryptography

(subexponential) iO (+ standard assumptions) is "crypto complete"

- PKE, Short Sigs, Perfect NIZKs (non-adaptive SNARGs), OT, Deniable Enc [SW'14]
- FHE [CLTV'15]
- WE [GGHRSW'14]
- Adaptive SNARGs [WW24, 25, WZ24]
- Succinct Garbling [KLW'15]

iO: Swiss Army knife of Cryptography

(subexponential) iO (+ standard assumptions) is "crypto complete"

- PKE, Short Sigs, Perfect NIZKs (non-adaptive SNARGs), OT, Deniable Enc [SW'14]
- FHE [CLTV'15]
- WE [GGHRSW'14]
- Adaptive SNARGs [WW24, 25, WZ24]
- Succinct Garbling [KLW'15]

Many of these applications involve obfuscating a cryptographic program. Can we leverage this?

Is there a *different notion* of obfuscation that suffices for these applications?

Fully Homomorphic Encryption

a la [CLTV'15]

Fully Homomorphic Encryption

a la [CLTV'15]

- Leveled FHE: pk contains key chain of length N to support depth N computation.

$Enc_{pk_2}(sk_1)$ $Enc_{pk_3}(sk_2)$ $\cdots$ $Enc_{pk_{N-1}}(sk_{N-2})$ $Enc_{pk_N}(sk_{N-1})$

Fully Homomorphic Encryption

a la [CLTV'15]

- Leveled FHE: pk contains key chain of length N to support depth N computation.

$Enc_{pk_2}(sk_1)$ $Enc_{pk_3}(sk_2)$ $\cdots$ $Enc_{pk_{N-1}}(sk_{N-2})$ $Enc_{pk_N}(sk_{N-1})$

$Enc_{pk_1}(m)$

Fully Homomorphic Encryption

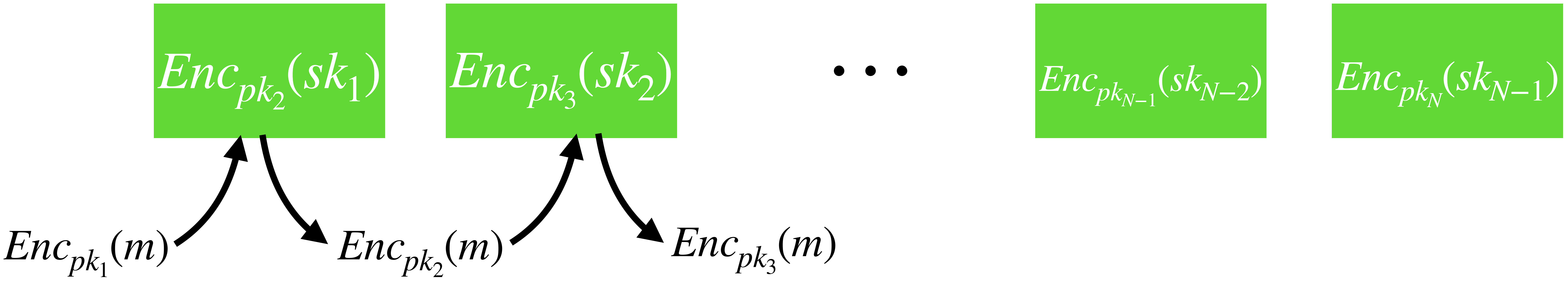
a la [CLTV'15]

- Leveled FHE: pk contains key chain of length N to support depth N computation.

Fully Homomorphic Encryption

a la [CLTV'15]

- Leveled FHE: pk contains key chain of length N to support depth N computation.



Fully Homomorphic Encryption

a la [CLTV'15]

- Leveled FHE: pk contains key chain of length N to support depth N computation.

$Enc_{pk_2}(sk_1)$ $Enc_{pk_3}(sk_2)$ $\cdots$ $Enc_{pk_{N-1}}(sk_{N-2})$ $Enc_{pk_N}(sk_{N-1})$

Fully Homomorphic Encryption

a la [CLTV’15]

- Leveled FHE: pk contains key chain of length N to support depth N computation.
- Consider a small program that computes this chain.

$$Enc_{pk_2}(sk_1) \quad Enc_{pk_3}(sk_2) \quad \cdots \quad Enc_{pk_{N-1}}(sk_{N-2}) \quad Enc_{pk_N}(sk_{N-1})$$

Fully Homomorphic Encryption

a la [CLTV'15]

- Leveled FHE: pk contains key chain of length N to support depth N computation.
- Consider a small program that computes this chain.

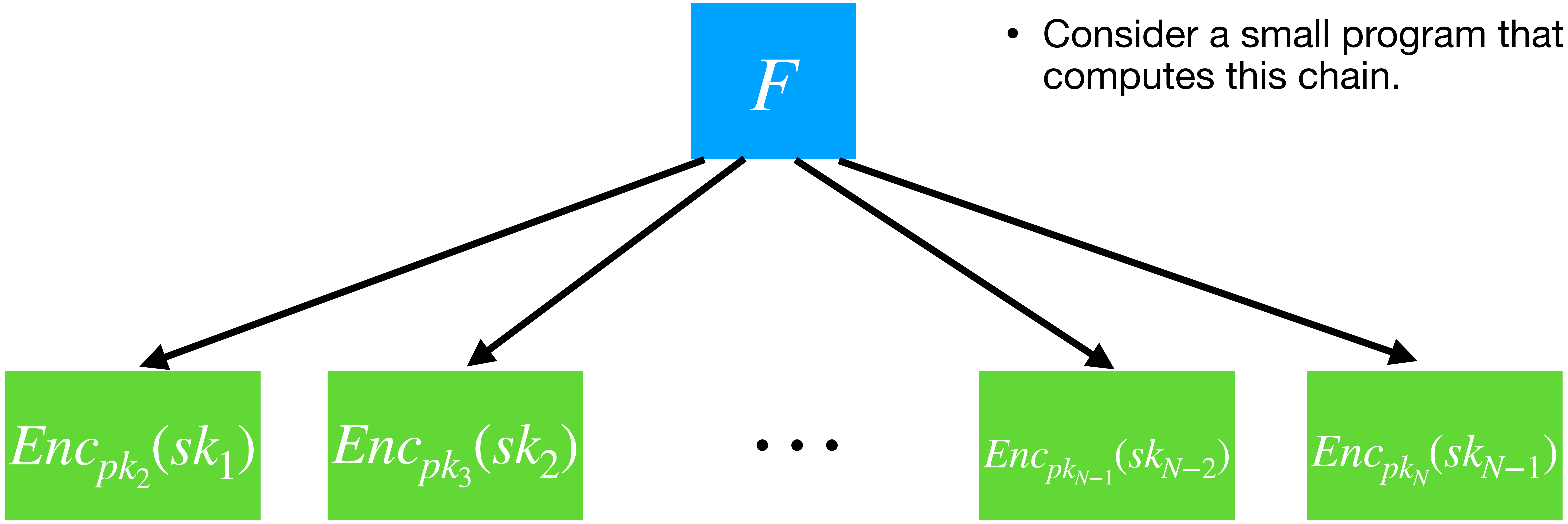
F

$Enc_{pk_2}(sk_1)$ $\quad$ $Enc_{pk_3}(sk_2)$ $\quad \cdots \quad$ $Enc_{pk_{N-1}}(sk_{N-2})$ $\quad$ $Enc_{pk_N}(sk_{N-1})$

Fully Homomorphic Encryption

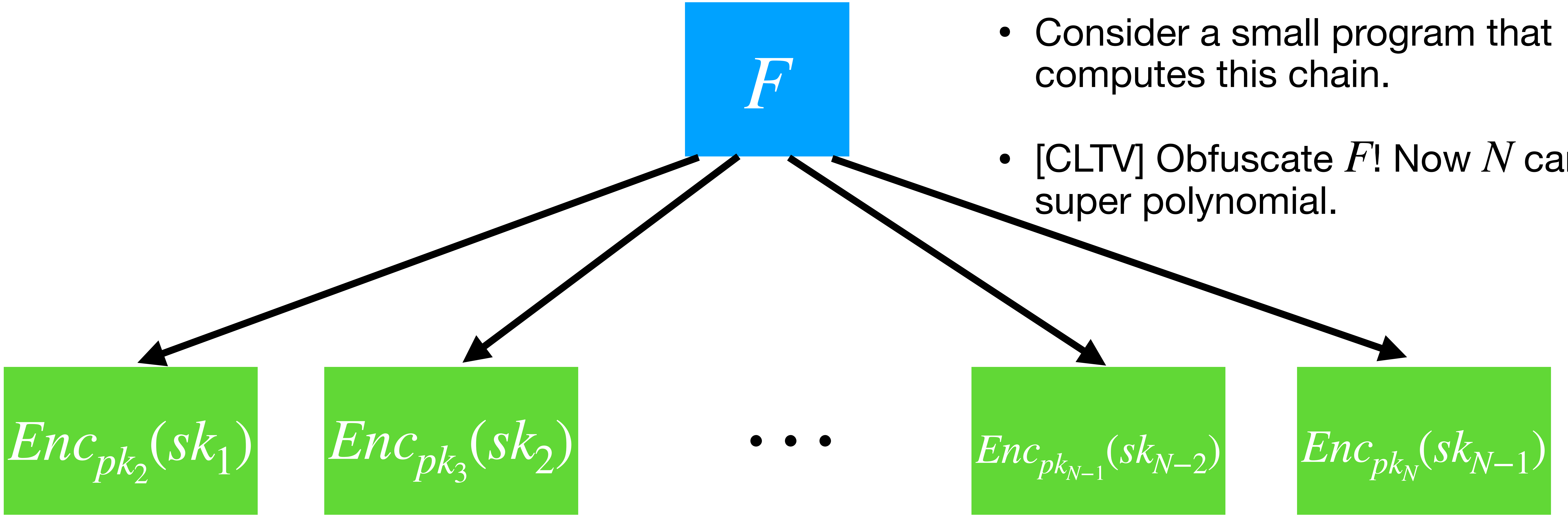
a la [CLTV'15]

- Leveled FHE: pk contains key chain of length N to support depth N computation.
- Consider a small program that computes this chain.



Fully Homomorphic Encryption

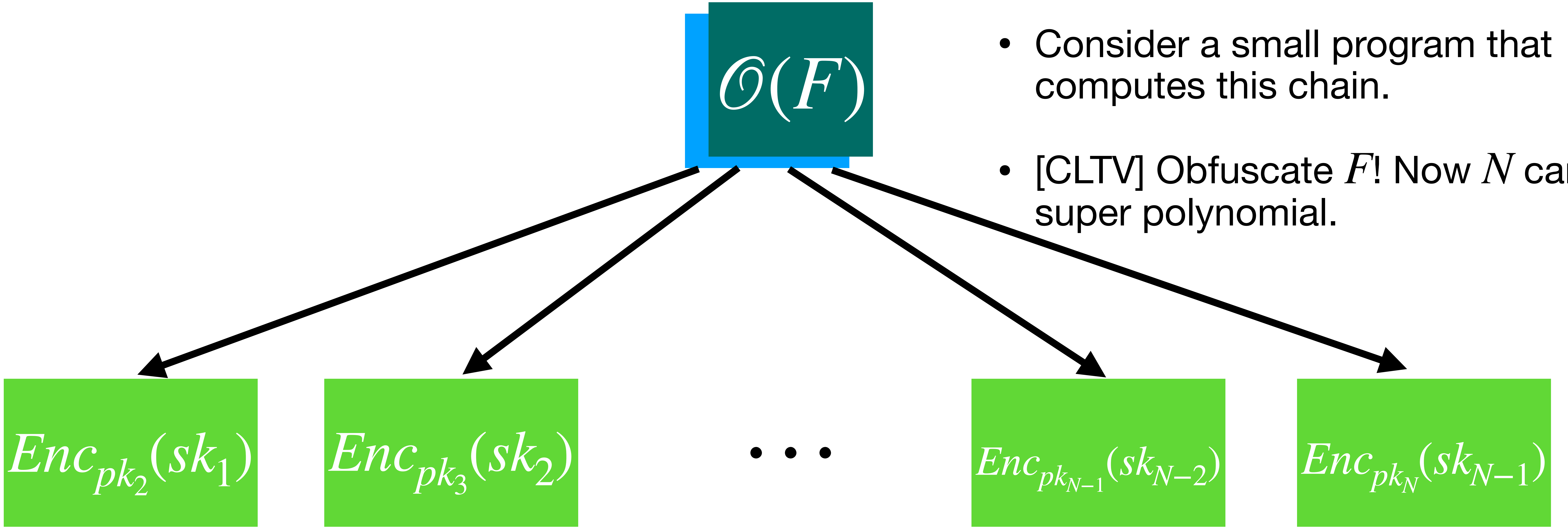
a la [CLTV'15]



- Leveled FHE: pk contains key chain of length N to support depth N computation.
- Consider a small program that computes this chain.
- [CLTV] Obfuscate F! Now N can be super polynomial.

Fully Homomorphic Encryption

a la [CLTV'15]



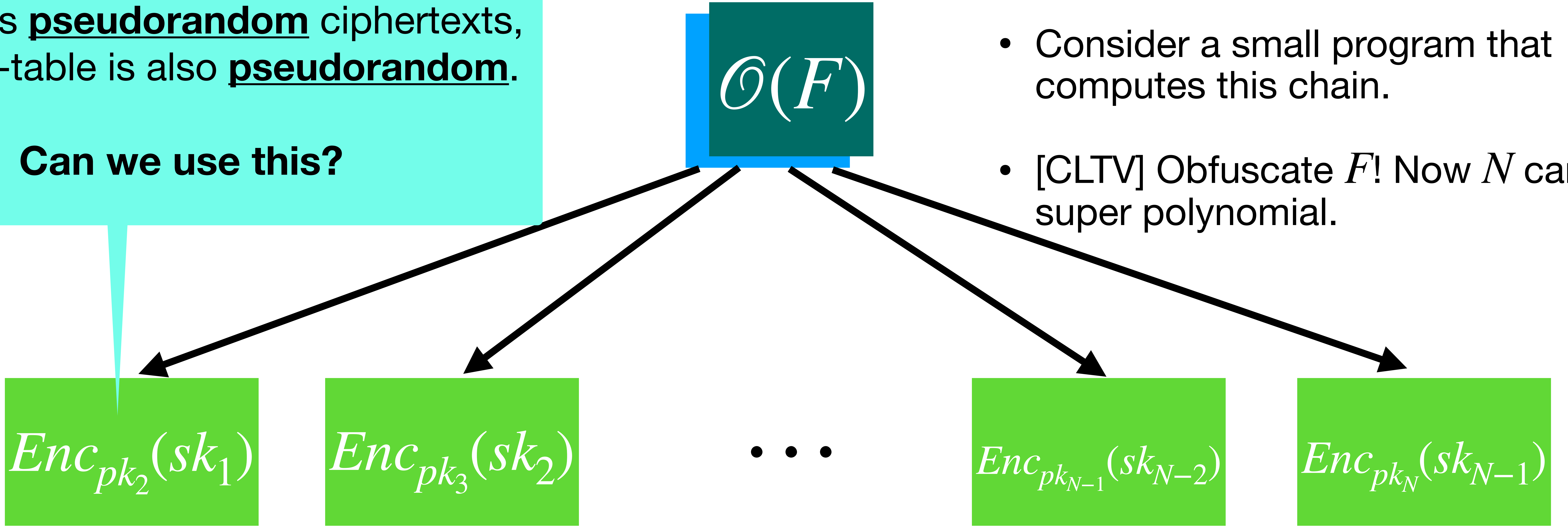
- Leveled FHE: pk contains key chain of length N to support depth N computation.
- Consider a small program that computes this chain.
- [CLTV] Obfuscate F! Now N can be super polynomial.

Fully Homomorphic Encryption

a la [CLTV'15]

If FHE has **pseudorandom** ciphertexts, this truth-table is also **pseudorandom**.

Can we use this?



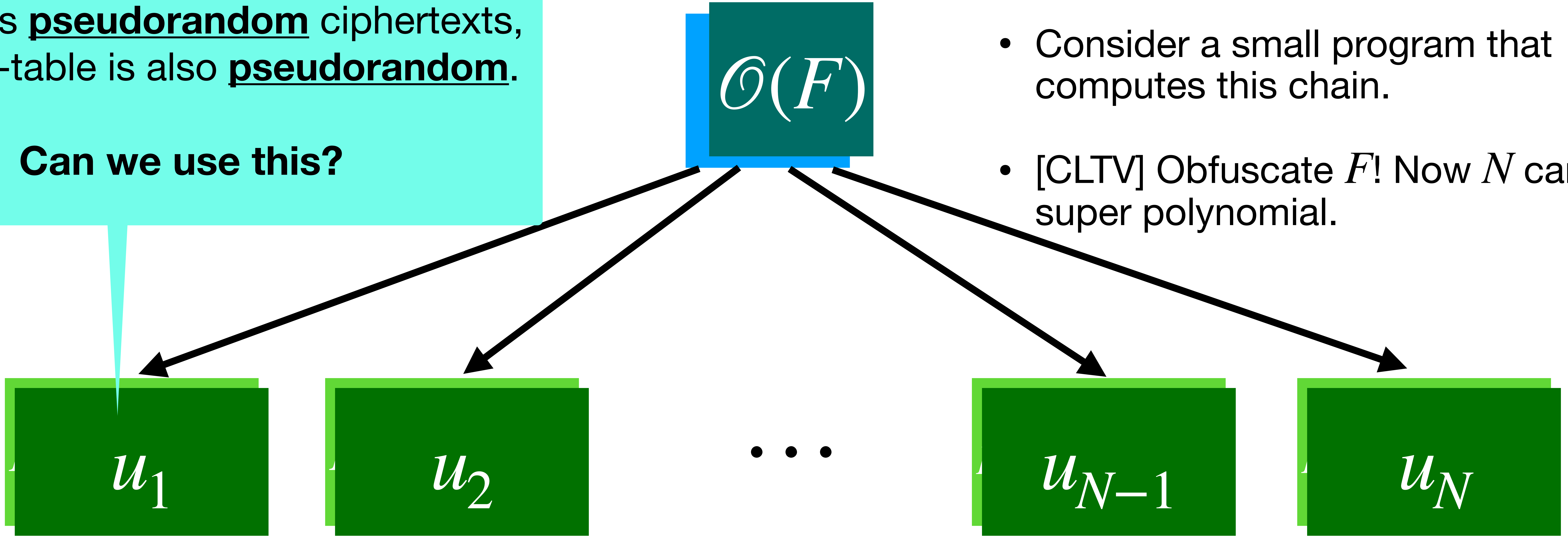
- Leveled FHE: pk contains key chain of length N to support depth N computation.
- Consider a small program that computes this chain.
- [CLTV] Obfuscate F! Now N can be super polynomial.

Fully Homomorphic Encryption

a la [CLTV'15]

If FHE has **pseudorandom** ciphertexts, this truth-table is also **pseudorandom**.

Can we use this?



- Leveled FHE: pk contains key chain of length N to support depth N computation.
- Consider a small program that computes this chain.
- [CLTV] Obfuscate F! Now N can be super polynomial.

Pseudorandom Obfuscation

TLDR

TLDR

- This work is a **systematic study** of the various notions of pseudorandom obfuscation (PRO).

TLDR

- This work is a **systematic study** of the various notions of pseudorandom obfuscation (PRO).
  - 3 notions of PRO

TLDR

- This work is a **systematic study** of the various notions of pseudorandom obfuscation (PRO).
    - 3 notions of PRO
    - Possibilities and impossibilities

TLDR

- This work is a **systematic study** of the various notions of pseudorandom obfuscation (PRO).
  - 3 notions of PRO
  - Possibilities and impossibilities
  - PRO + Bilinear Maps = iO

TLDR

- This work is a **systematic study** of the various notions of pseudorandom obfuscation (PRO).
    - 3 notions of PRO
    - Possibilities and impossibilities
    - PRO + Bilinear Maps = iO
- (Not in talk) The full version of this paper additionally includes a candidate construction of pseudorandom obfuscation from the evasive LWE heuristic.

Pseudorandom Obfuscation

Strongest Notion: Double Pseudorandomness (dPRO)

Pseudorandom Obfuscation

Strongest Notion: Double Pseudorandomness (dPRO)

$TT(C)$

Pseudorandom Obfuscation

Strongest Notion: Double Pseudorandomness (dPRO)

$TT(C)$ $\approx$ U

Pseudorandom Obfuscation

Strongest Notion: Double Pseudorandomness (dPRO)

$$TT(C) \approx U$$

$$\Downarrow$$

Pseudorandom Obfuscation

Strongest Notion: <u>Double</u> Pseudorandomness (dPRO)

$$TT(C) \approx U$$

$$\Downarrow$$

$$PRO(C) \approx u$$

Pseudorandom Obfuscation

Strongest Notion: Double Pseudorandomness (dPRO)

$TT(C) \approx U$ given aux(C)

$\Downarrow$

$PRO(C) \approx u$ given aux(C)

Pseudorandom Obfuscation

Strongest Notion: <u>Double</u> Pseudorandomness (dPRO)

$TT(C) \approx U$ given aux(C)

$\Downarrow$

$PRO(C) \approx u$ given aux(C)

Pseudorandom Obfuscation

Strongest Notion: Double Pseudorandomness (dPRO)

$TT(C)$ $\approx$ U given aux(C)

$\Downarrow$

$PRO(C)$ $\approx$ u given aux(C)

xPRO: $|PRO(C)| = |TT(C)|^{1-\epsilon}$

Pseudorandom Obfuscation

Strongest Notion: Double Pseudorandomness (dPRO)

Precondition

$TT(C)$ $\approx$ U given aux(C)

$\Downarrow$

$PRO(C)$ $\approx$ u given aux(C)

xPRO: $|PRO(C)| = |TT(C)|^{1-\epsilon}$

Pseudorandom Obfuscation

Strongest Notion: Double Pseudorandomness (dPRO)

Precondition

$TT(C) \approx U$ given aux(C)

$\Downarrow$

Postcondition

$PRO(C) \approx u$ given aux(C)

xPRO: $|PRO(C)| = |TT(C)|^{1-\epsilon}$

Pseudorandom Obfuscation

Precondition

$TT(C)$ $\approx$ U given $aux(C_b)$

$\Downarrow$

Postcondition

$PRO(C)$ $\approx$ u given $aux(C_b)$

Pseudorandom Obfuscation

Medium notion

Precondition

$TT(C) \approx U$ given $aux(C_b)$

$\Downarrow$

Postcondition

$PRO(C) \approx u$ given $aux(C_b)$

Pseudorandom Obfuscation

Medium notion

Precondition

$TT(C_b) \approx U$ given $aux(C_b)$

$\Downarrow$

Postcondition

$PRO(C) \approx u$ given $aux(C_b)$

Pseudorandom Obfuscation

Medium notion

Precondition

$TT(C_b) \approx U$ given $aux(C_b)$

$\Downarrow$

Postcondition

$PRO(C_0) \approx PRO(C_1)$ given $aux(C_b)$

Pseudorandom Obfuscation

Medium notion

Precondition

$TT(C_b)$ $\approx$ U given $aux(C_b)$

Obfuscation itself doesn't have to be pseudorandom

Postcondition

$PRO(C_0)$ $\approx$ $PRO(C_1)$ given $aux(C_b)$

The notions and applications

The notions and applications

		$TT(f_K)$ is pseudorandom

The notions and applications

		$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{U}$		

The notions and applications

		$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{U}$		**dPRO**

The notions and applications

		$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$		
$\mathcal{O}(f_K) \approx_c \mathcal{U}$		**dPRO**

The notions and applications

		$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$		**PRO**
$\mathcal{O}(f_K) \approx_c \mathcal{U}$		**dPRO**

The notions and applications

	$f_K \equiv f_{K'}$	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**PRO**
$\mathcal{O}(f_K) \approx_c \mathcal{U}$		**dPRO**

The notions and applications

	$f_K \equiv f_{K'}$	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**PRO** Null-iO [VWW22] SNARK for UP [**M**PV24] SNARG for NP [JKL**M**25]
$\mathcal{O}(f_K) \approx_c \mathcal{U}$		**dPRO**

The notions and applications

	$f_K \equiv f_{K'}$	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**PRO** Null-iO [VWW22] SNARK for UP [**M**PV24] SNARG for NP [JKL**M**25] **This work:** (+LWE) FHE Succinct Garbling Succinct witness encryption
$\mathcal{O}(f_K) \approx_c \mathcal{U}$		**dPRO**

The notions and applications

	$f_K \equiv f_{K'}$	$TT(f_K)$ is pseudorandom	
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**PRO** Null-iO [VWW22] SNARK for UP [**M**PV24] SNARG for NP [JKL**M**25]	**This work:** (+LWE) FHE Succinct Garbling Succinct witness encryption
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**	**dPRO**	

The notions and applications

	$f_K \equiv f_{K'}$	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**PRO** Null-iO [VWW22] SNARK for UP [**M**PV24] SNARG for NP [JKL**M**25] **This work:** (+LWE) FHE ✓ Succinct Garbling Succinct witness encryption
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**	**dPRO**

The notions and applications

	$f_K \equiv f_{K'}$	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**PRO** Null-iO [VWW22] SNARK for UP [**M**PV24] SNARG for NP [JKL**M**25] **This work:** (+LWE) FHE ✓ Succinct Garbling ← Succinct witness encryption
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**	**dPRO**

Over-simplified Sketch of Succinct Garbling

Via standard iO

Over-simplified Sketch of Succinct Garbling

Via standard iO

Size of program is independent of runtime of TM

Over-simplified Sketch of Succinct Garbling

Via standard iO

Over-simplified Sketch of Succinct Garbling

Via standard iO

- **Idea:** Obfuscate the machine which:

Over-simplified Sketch of Succinct Garbling

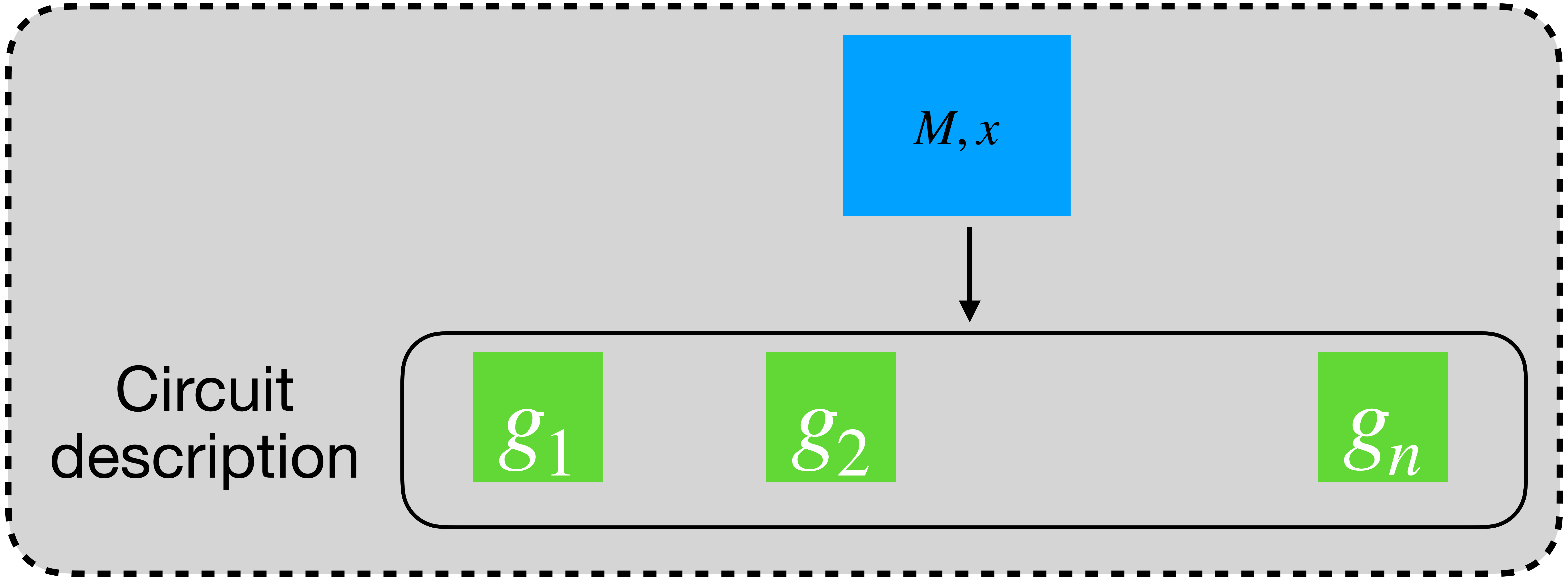
Via standard iO

- **Idea:** Obfuscate the machine which:
  - Expands (M, x) into a circuit $C_{M,x}$.

Over-simplified Sketch of Succinct Garbling

Via standard iO

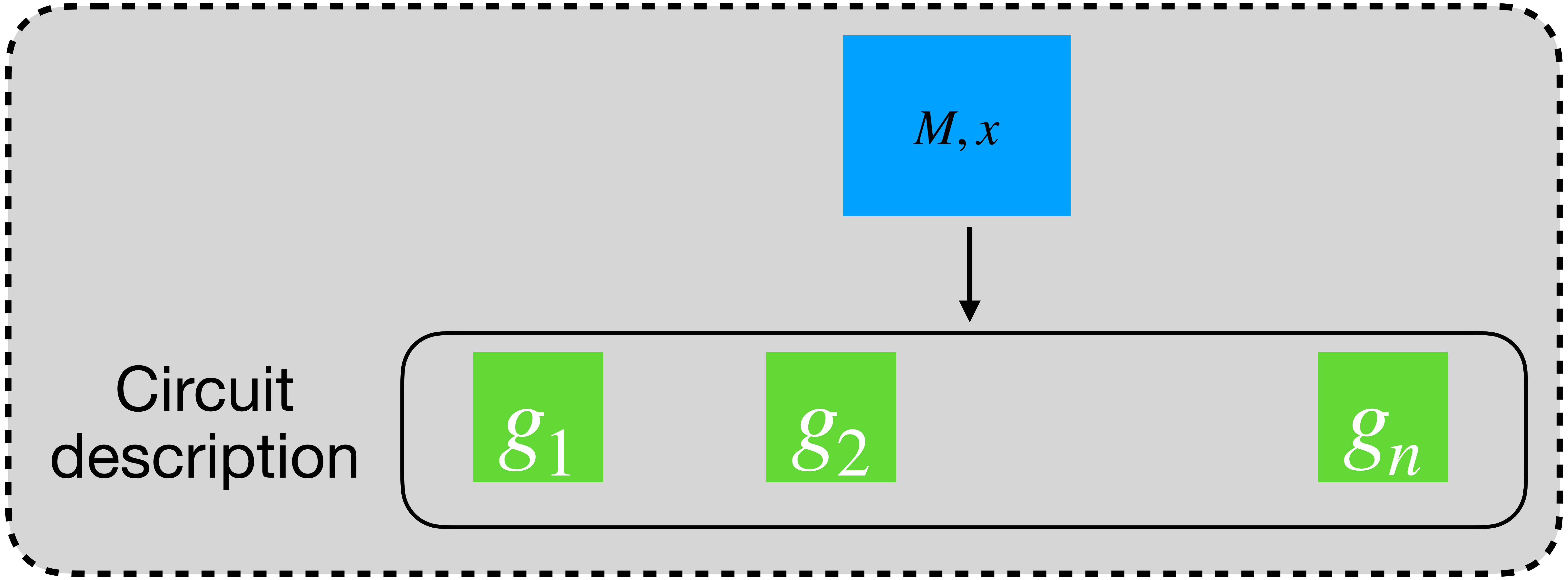
- **Idea:** Obfuscate the machine which:
  - Expands (M, x) into a circuit $C_{M,x}$.



Over-simplified Sketch of Succinct Garbling

Via standard iO

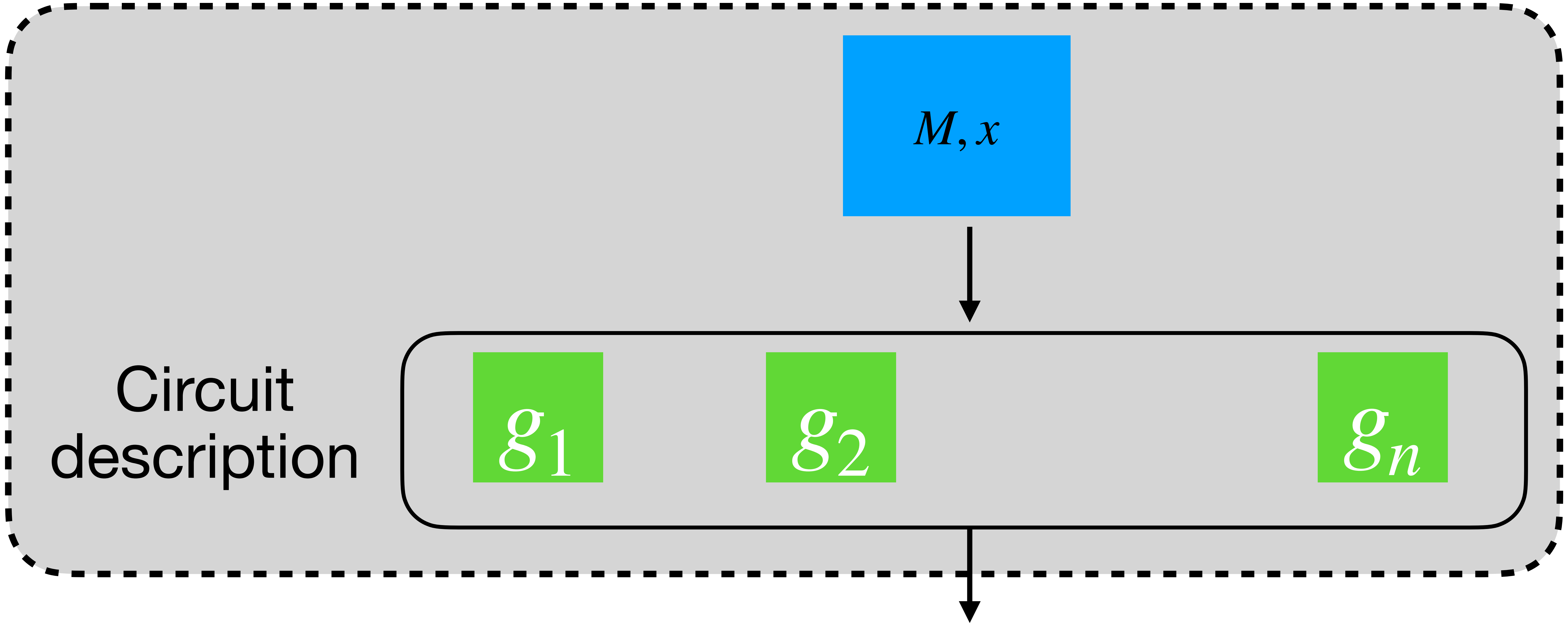
- **Idea:** Obfuscate the machine which:
  - Expands (M, x) into a circuit $C_{M,x}$.
  - Outputs a garbling of $C_{M,x}$.



Over-simplified Sketch of Succinct Garbling

Via standard iO

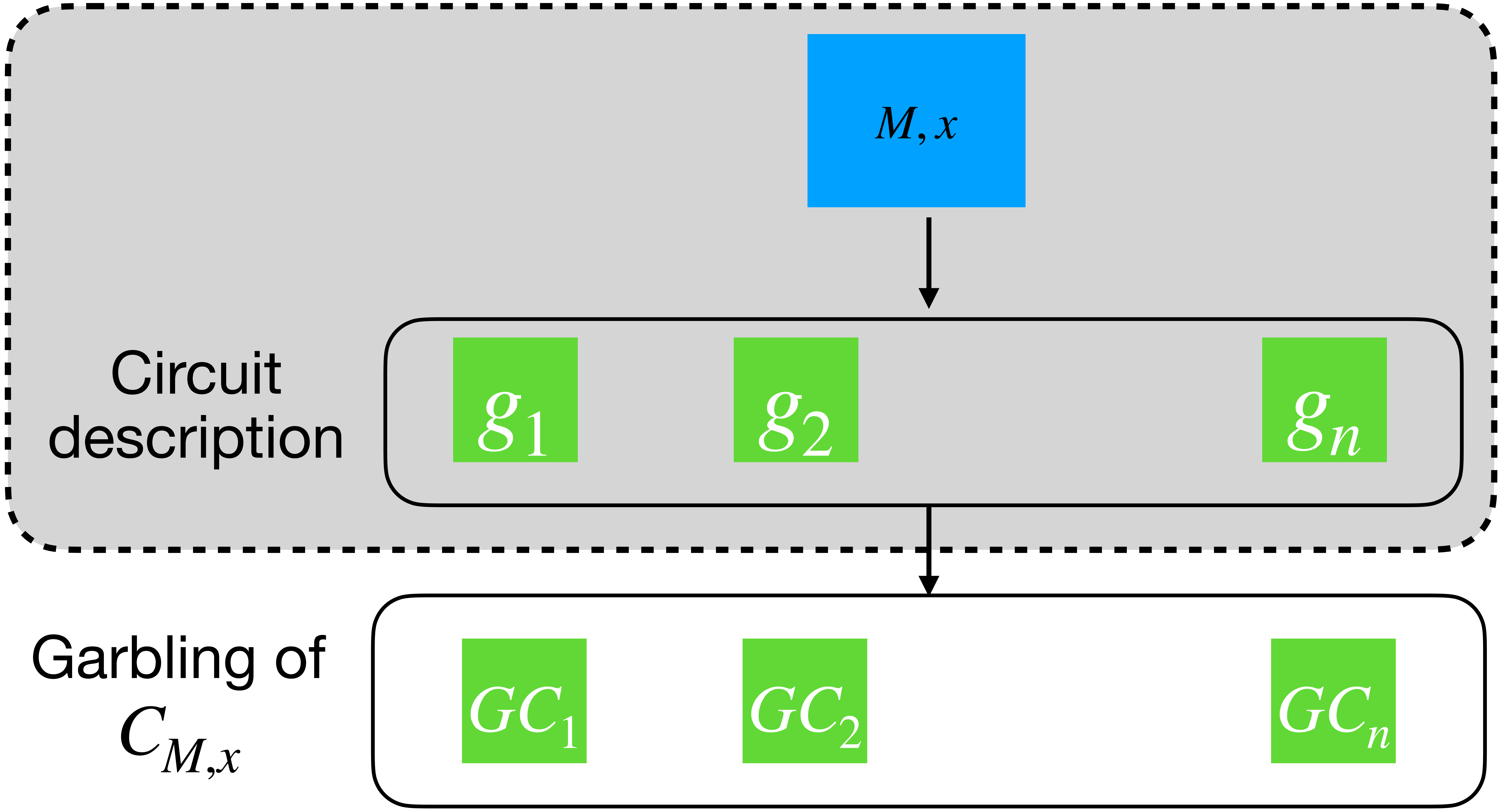
- **Idea:** Obfuscate the machine which:
  - Expands (M, x) into a circuit $C_{M,x}$.
  - Outputs a garbling of $C_{M,x}$.



Over-simplified Sketch of Succinct Garbling

Via standard iO

- **Idea:** Obfuscate the machine which:
  - Expands (M, x) into a circuit $C_{M,x}$.
  - Outputs a garbling of $C_{M,x}$.

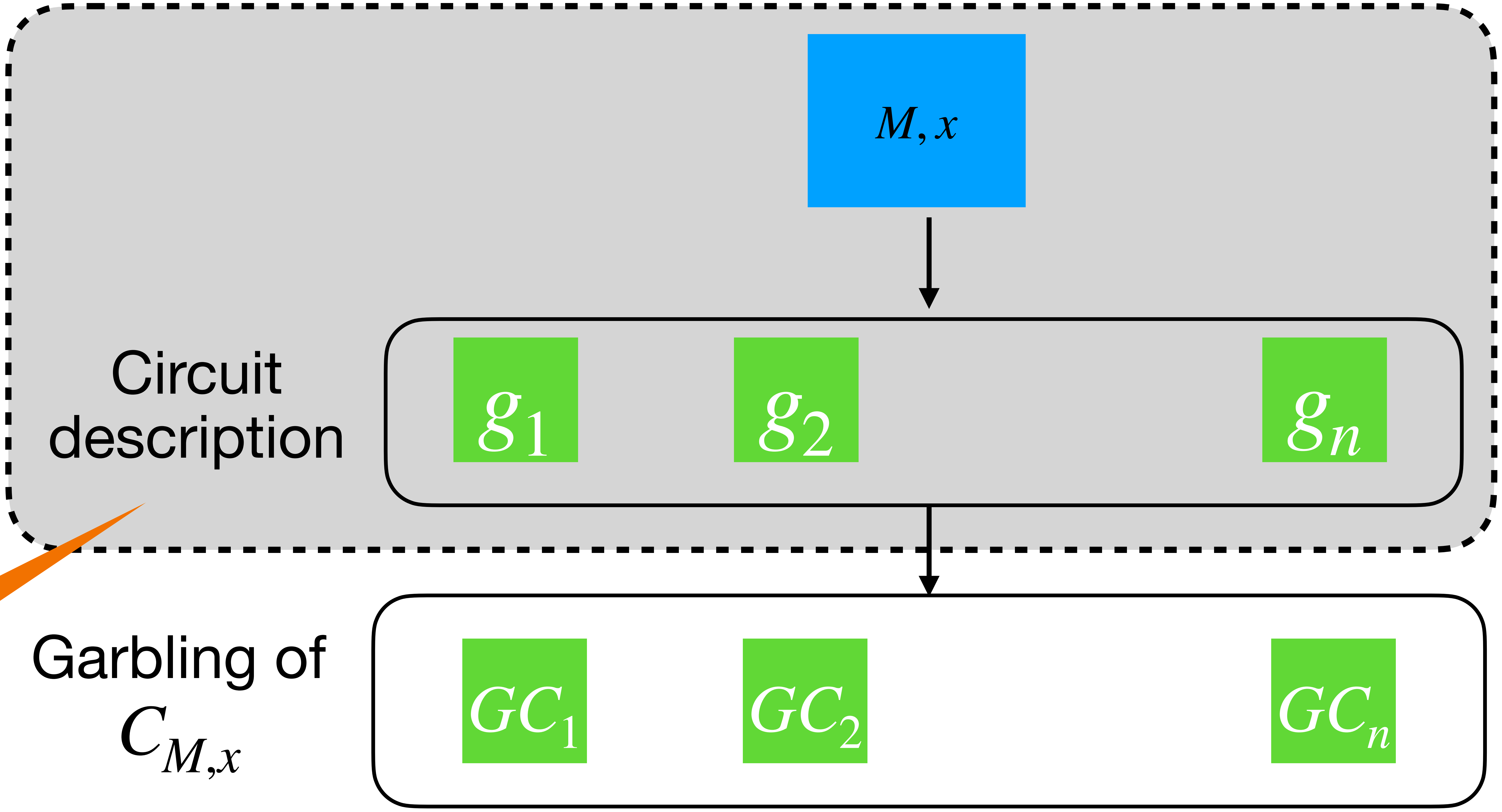


Over-simplified Sketch of Succinct Garbling

Via standard iO

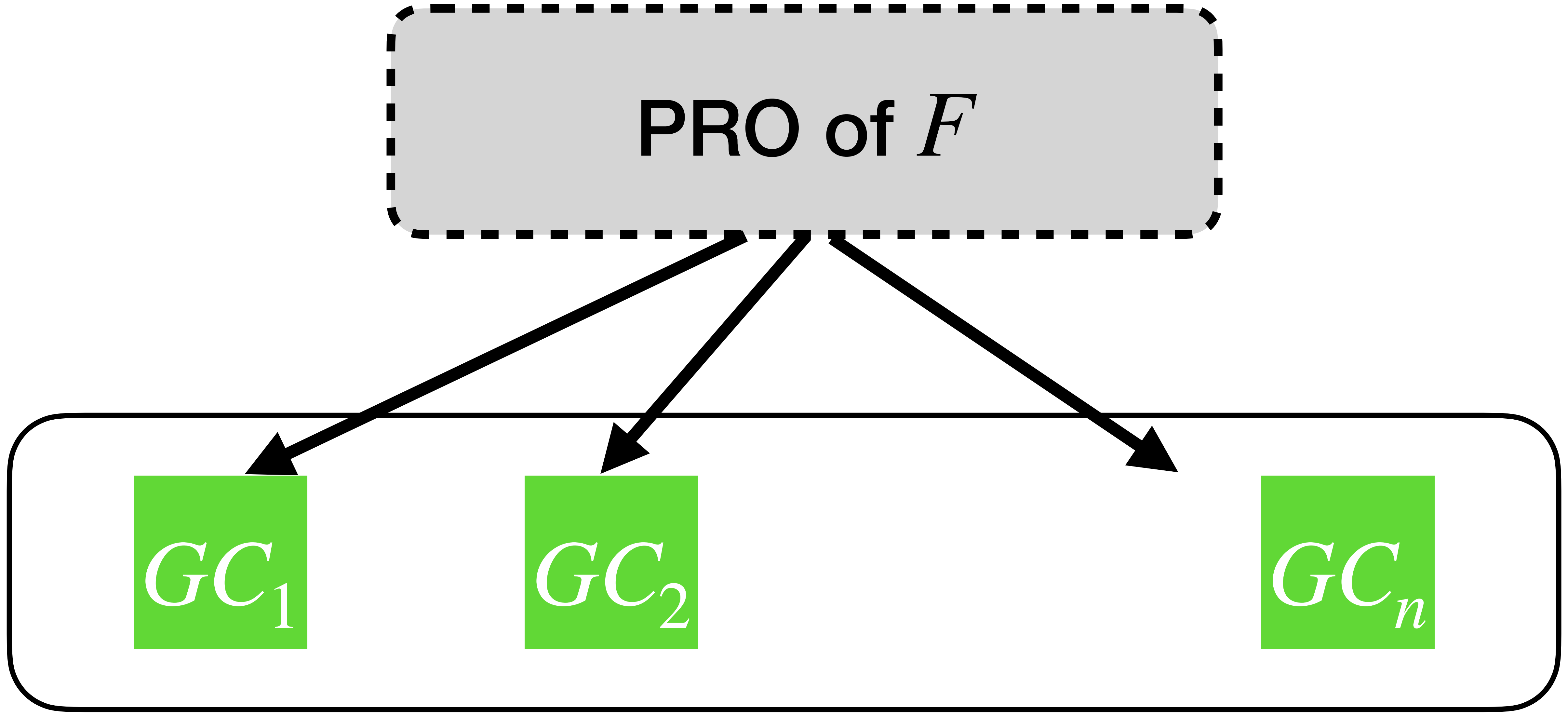
- **Idea:** Obfuscate the machine which:
  - Expands (M, x) into a circuit $C_{M,x}$.
  - Outputs a garbling of $C_{M,x}$.

Garbling = Obfuscation of this program!

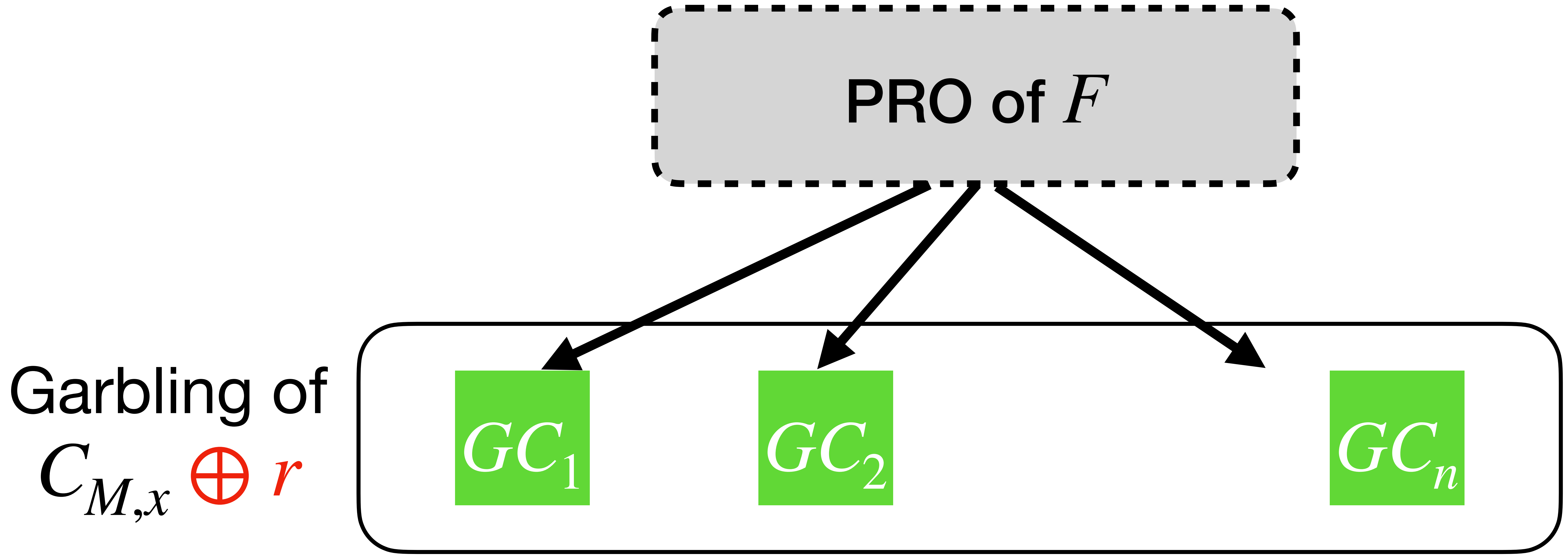


Succinct Garbling from PRO

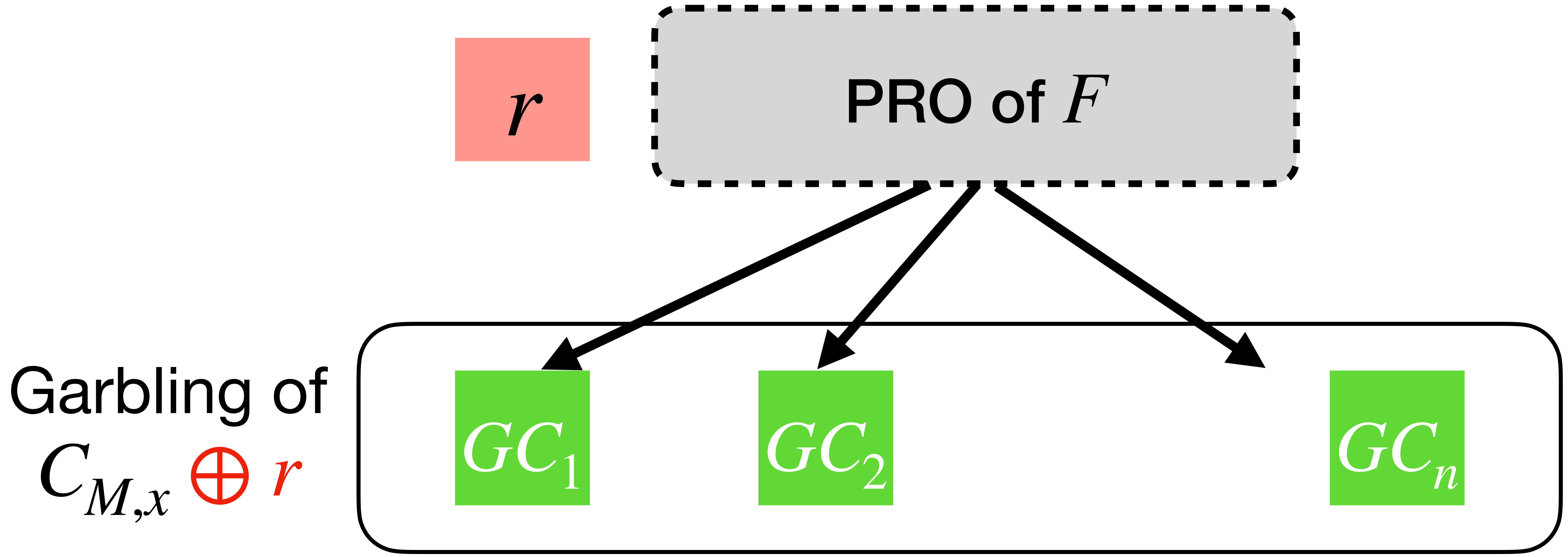
Succinct Garbling from PRO



Succinct Garbling from PRO

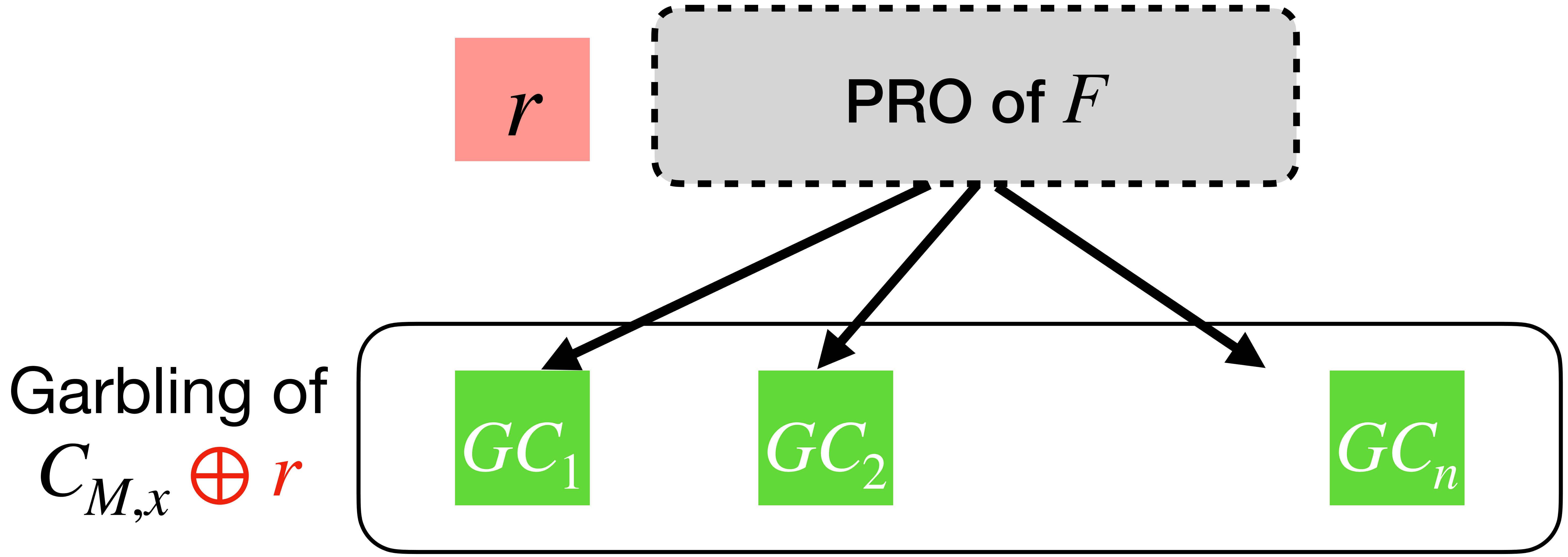


Succinct Garbling from PRO



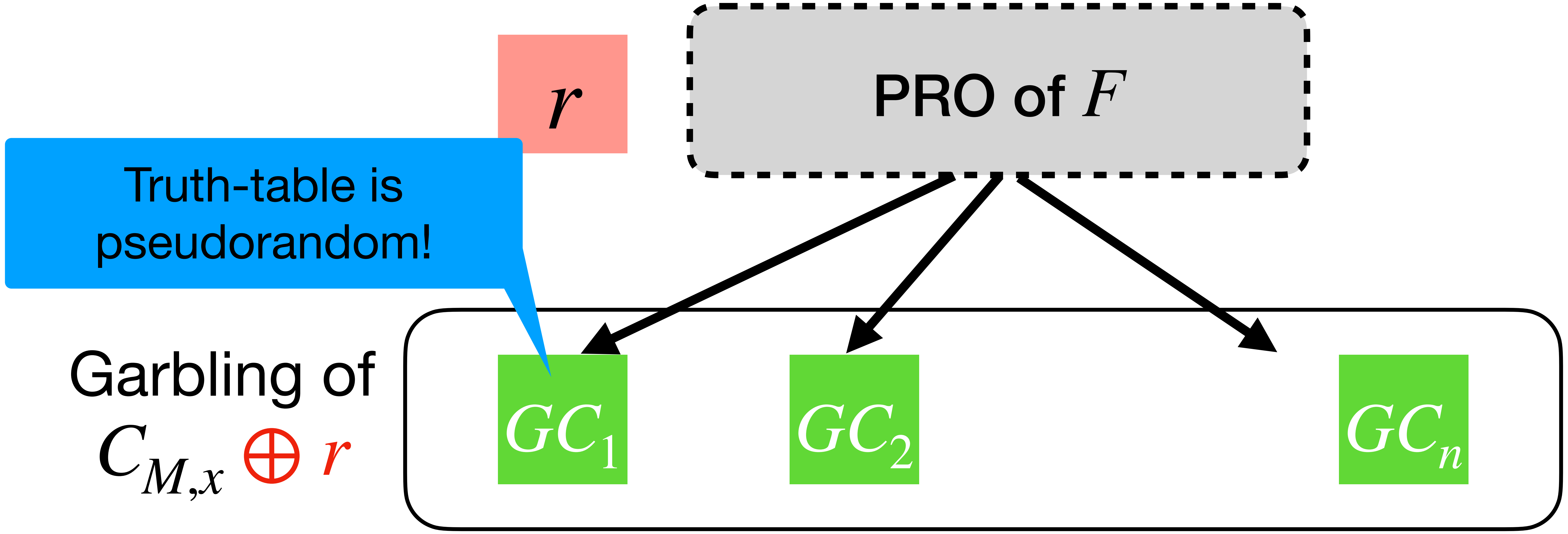
Succinct Garbling from PRO

- Pick a garbling scheme with ***pseudorandom*** garbling [BLSV '18].



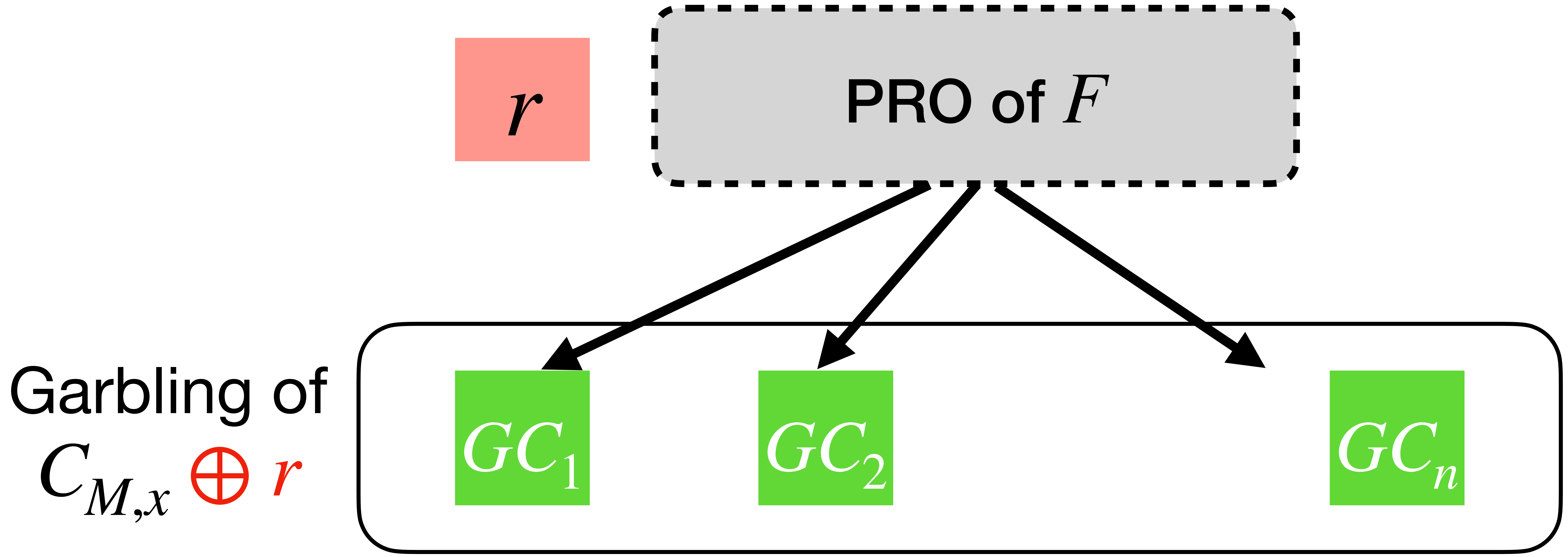
Succinct Garbling from PRO

- Pick a garbling scheme with ***pseudorandom*** garbling [BLSV '18].



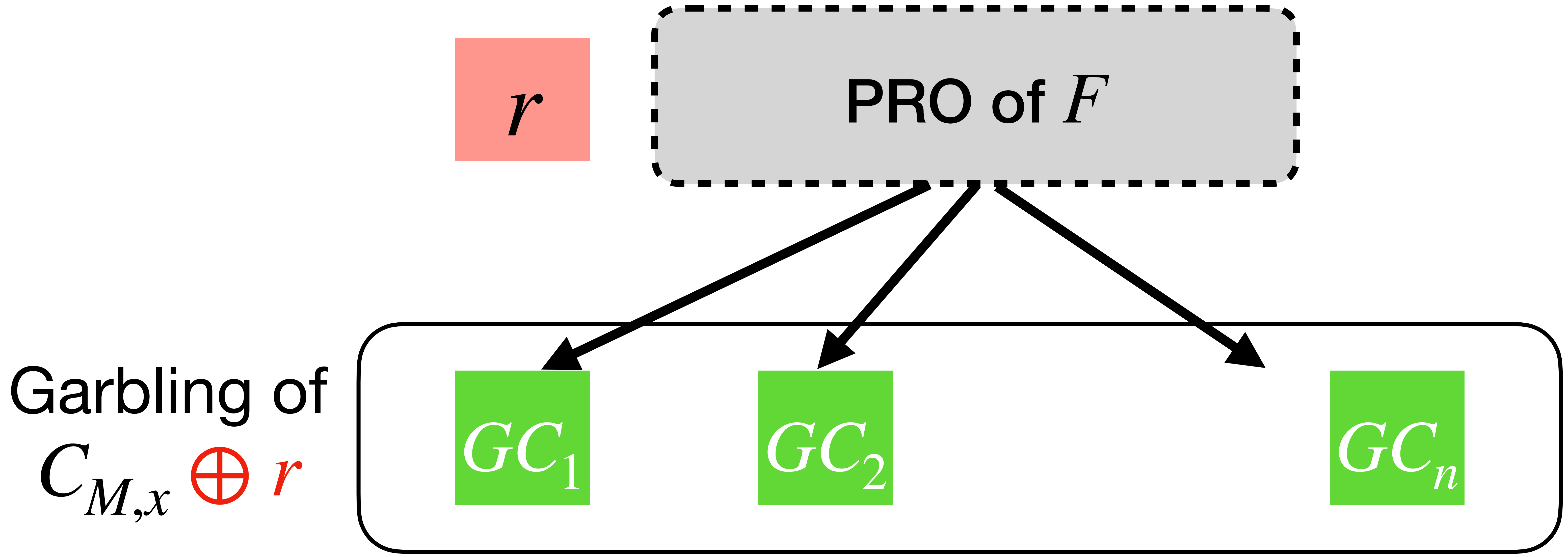
Succinct Garbling from PRO

- Pick a garbling scheme with ***pseudorandom*** garbling [BLSV '18].



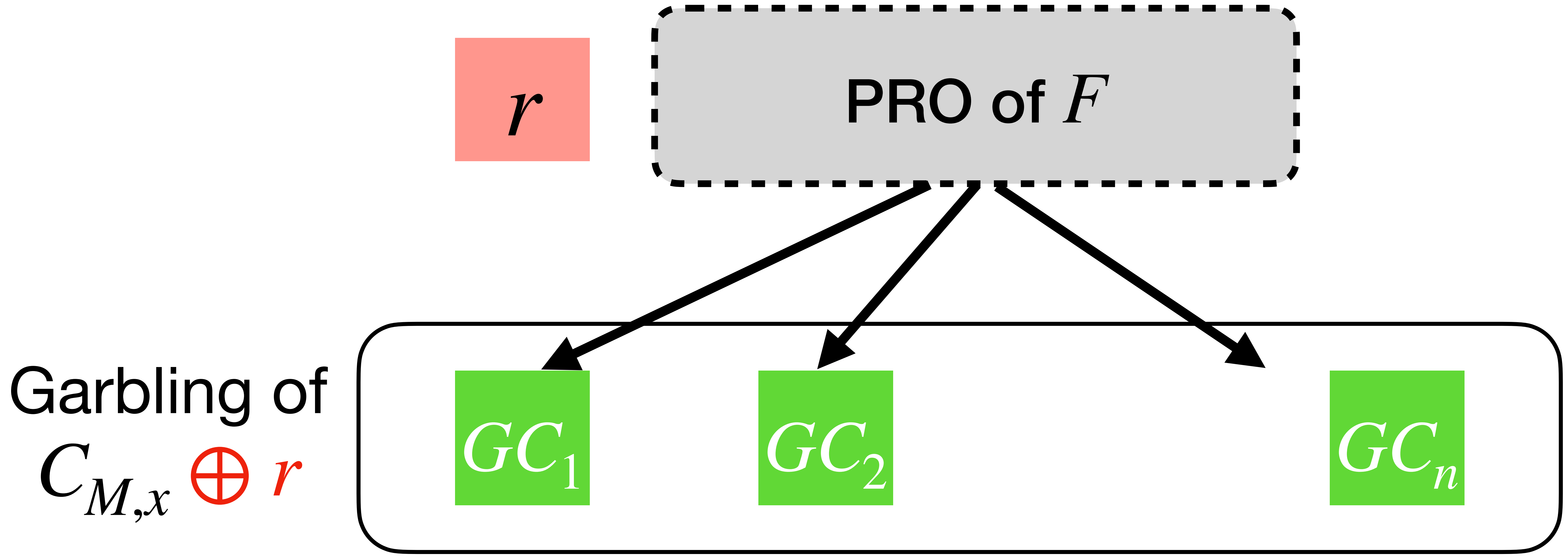
Succinct Garbling from PRO

- Pick a garbling scheme with ***pseudorandom*** garbling [BLSV '18].
- GC . Garb $= (r, PRO(F))$



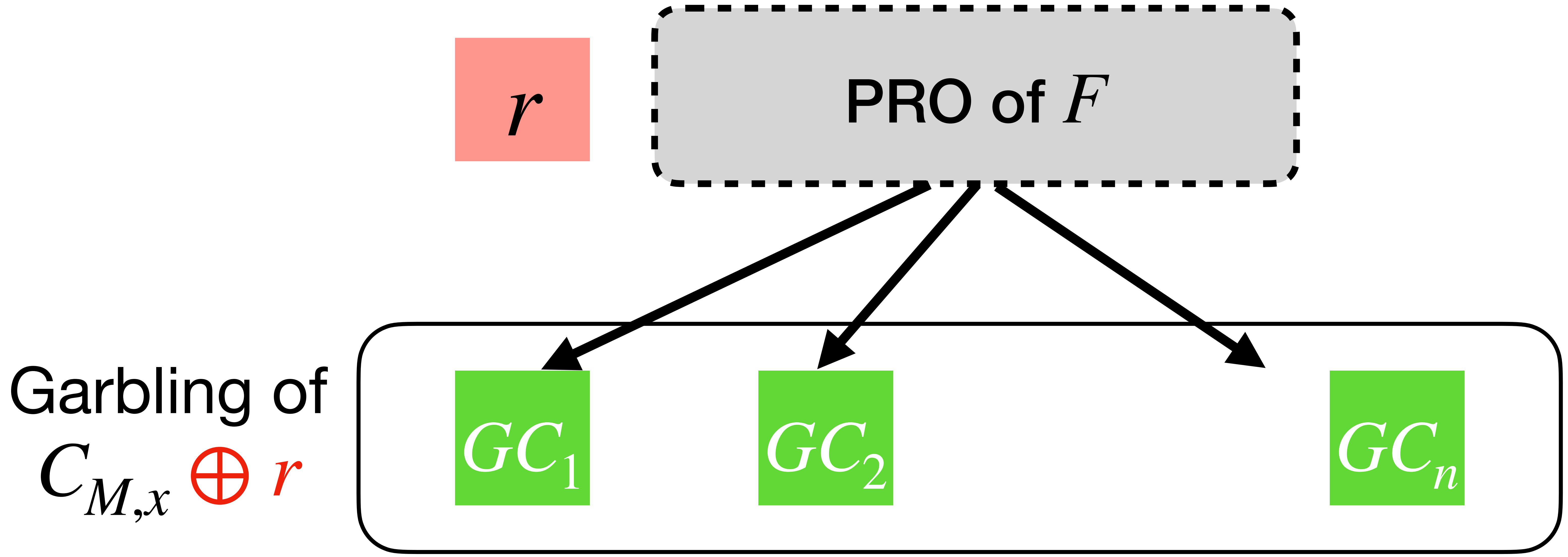
Succinct Garbling from PRO

- Pick a garbling scheme with ***pseudorandom*** garbling [BLSV '18].
- $\mathsf{GC.Garb} = (r, PRO(F))$



Succinct Garbling from PRO

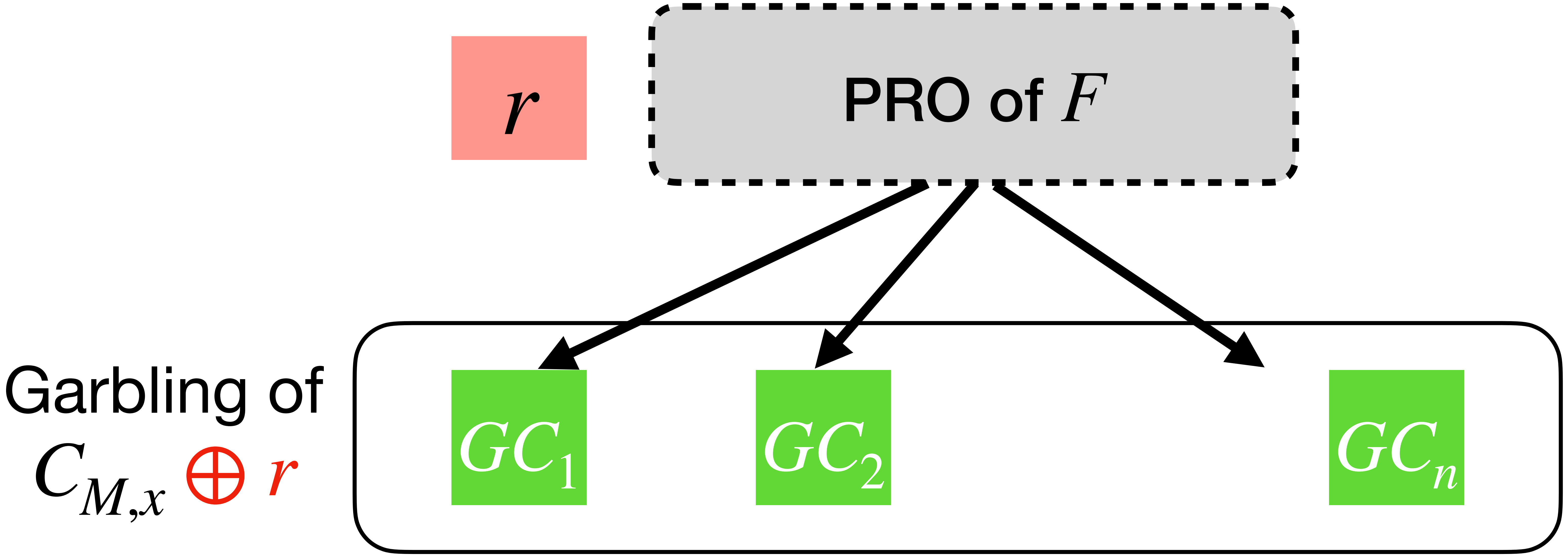
- Pick a garbling scheme with ***pseudorandom*** garbling [BLSV '18].
- $\mathsf{GC.Garb} = (r, PRO(F))$



Simulator:

Succinct Garbling from PRO

- Pick a garbling scheme with ***pseudorandom*** garbling [BLSV '18].
- $\mathrm{GC.Garb} = (r, PRO(F))$

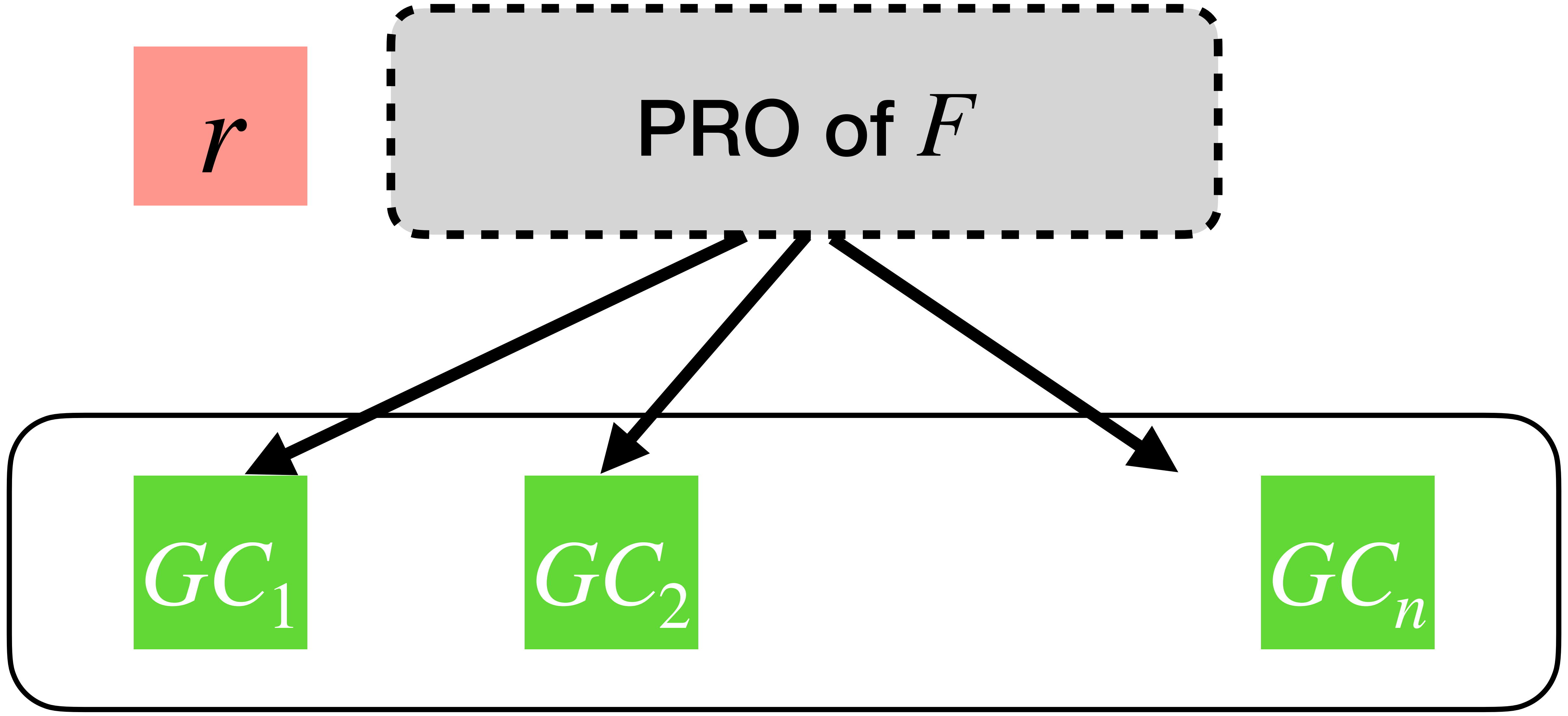


Simulator:

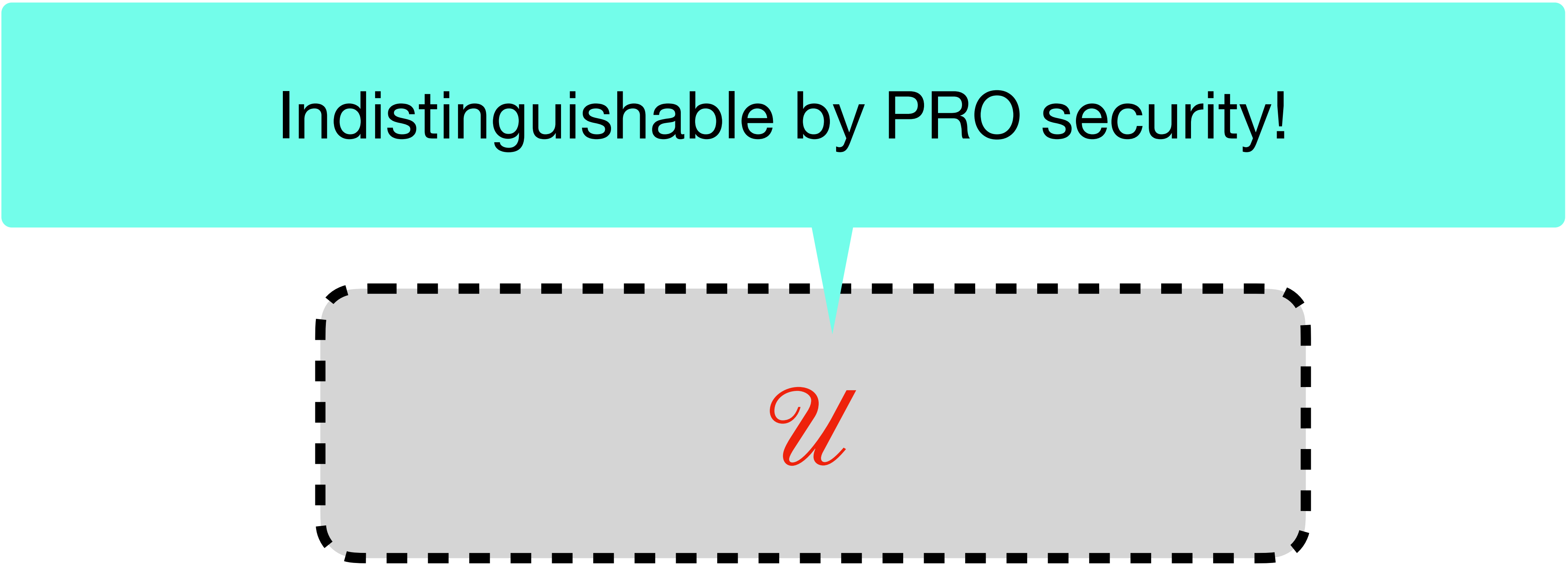
Succinct Garbling from PRO

- Pick a garbling scheme with ***pseudorandom*** garbling [BLSV '18].
- $\mathsf{GC.Garb} = (r, PRO(F))$

Garbling of $C_{M,x} \oplus r$

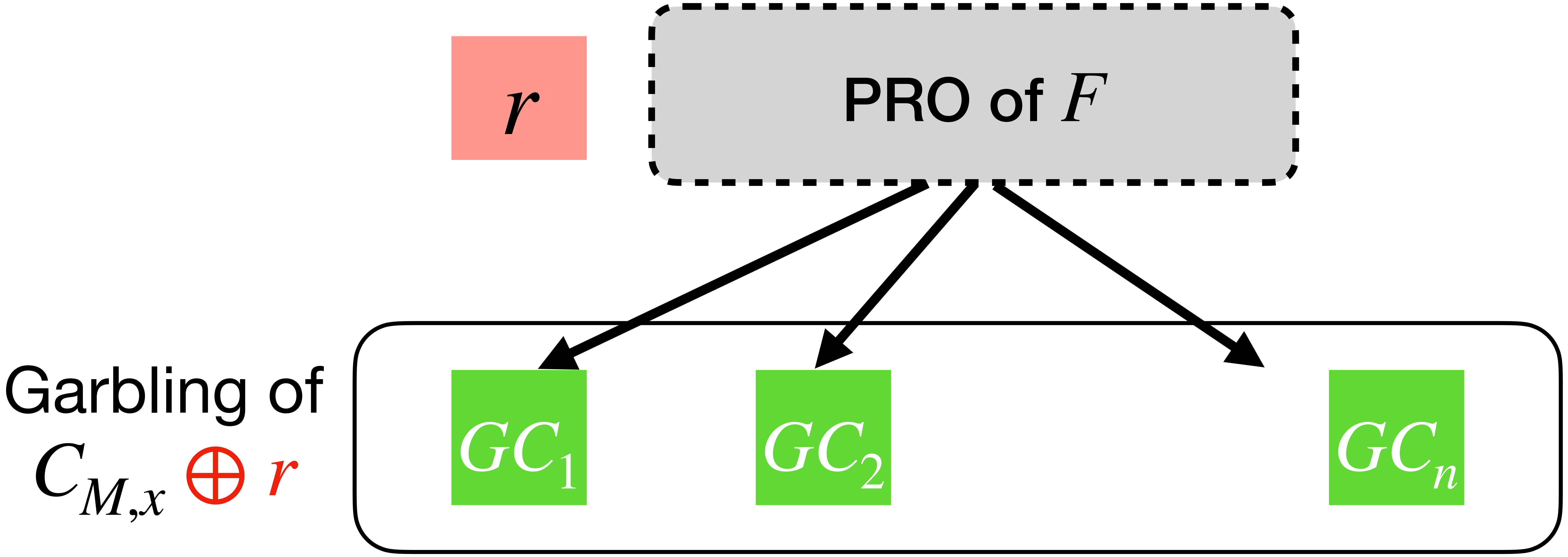


Simulator:

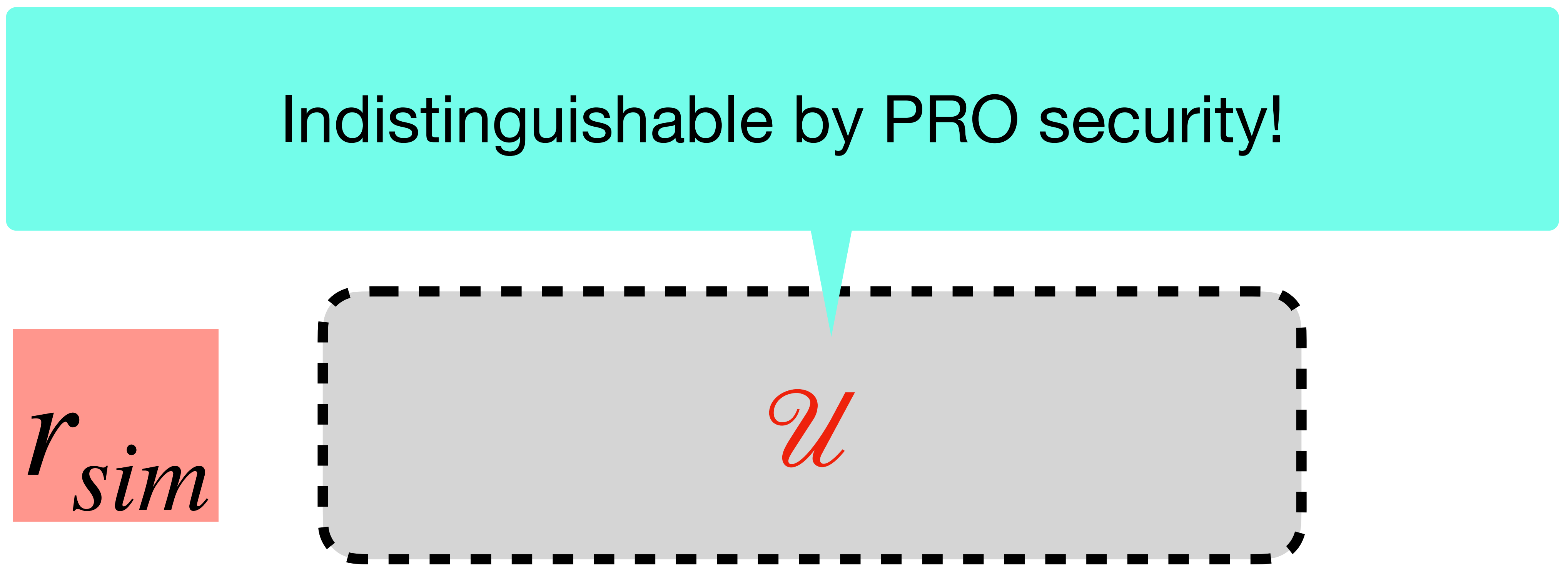


Succinct Garbling from PRO

- Pick a garbling scheme with ***pseudorandom*** garbling [BLSV '18].
- $\mathsf{GC.Garb} = (r, PRO(F))$

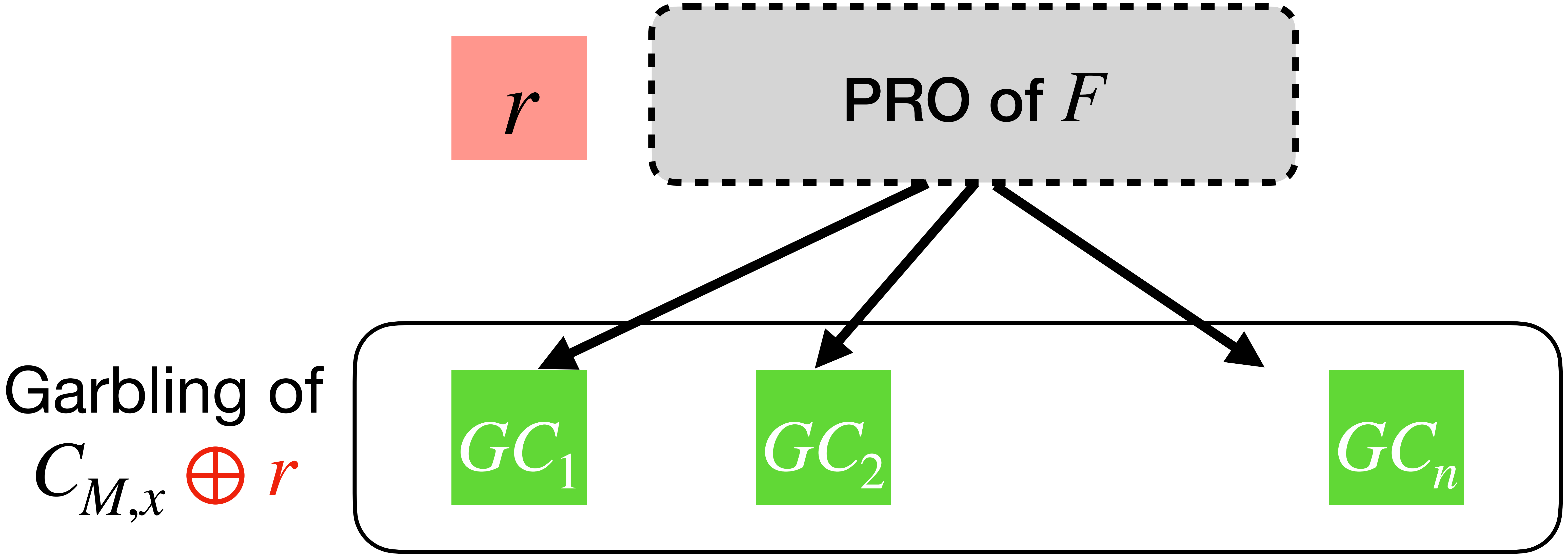


Simulator:



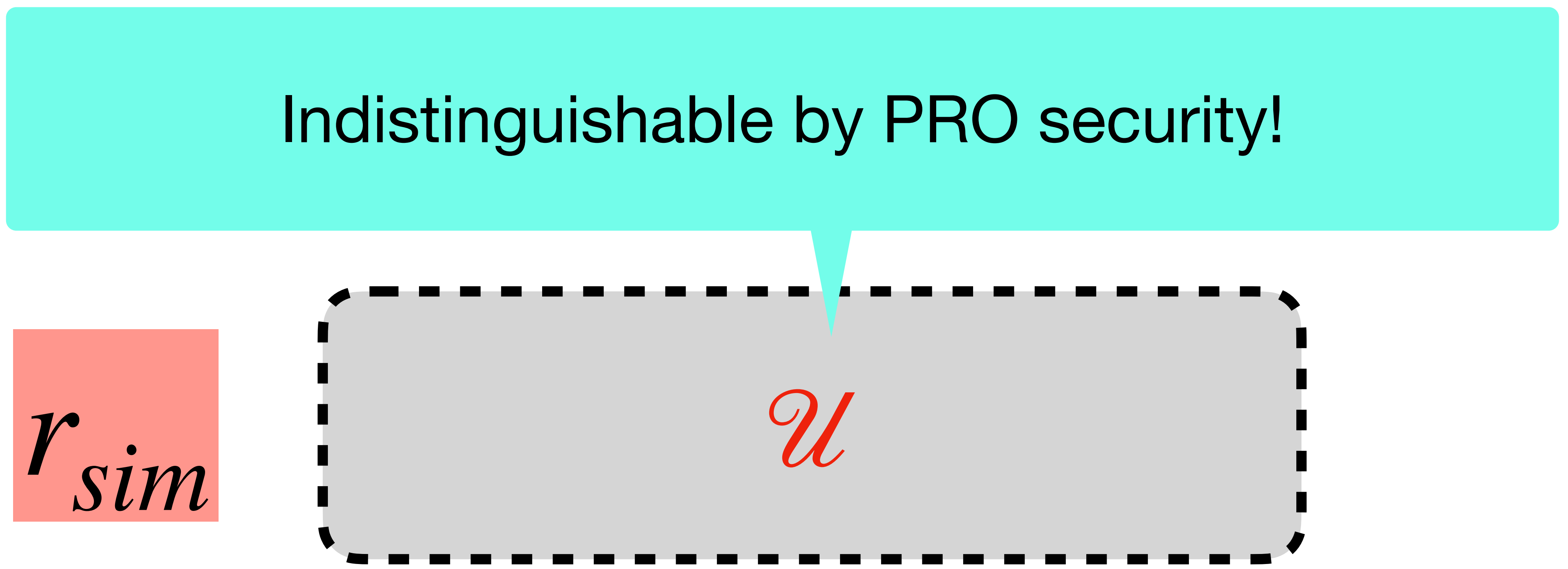
Succinct Garbling from PRO

- Pick a garbling scheme with ***pseudorandom*** garbling [BLSV '18].
- $\mathsf{GC.Garb} = (r, PRO(F))$



Simulator:

Set $r_{sim} = M(x) \oplus PRO.Eval(\mathscr{U})$



The notions and applications

	$f_K \equiv f_{K'}$	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**PRO** Null-iO [VWW22] SNARK for UP [**M**PV24] SNARG for NP [JKL**M**25] **This work:** (+LWE) FHE Succinct Garbling Succinct witness encryption
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**	**dPRO**

The notions and applications

	$f_K \equiv f_{K'}$	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**PRO** Null-iO [VWW22] SNARK for UP [**M**PV24] SNARG for NP [JKL**M**25] **This work:** (+LWE) FHE ✓ Succinct Garbling ✓ Succinct witness encryption
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**	**dPRO**

The notions and applications

Bad news: We show PRO does not exist

	$f_K \equiv f_{K'}$	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**PRO** Null-iO [VWW22] SNARK for UP [**M**PV24] SNARG for NP [JKL**M**25] **This work:** (+LWE) FHE ✓ Succinct Garbling ✓ Succinct witness encryption
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**	**dPRO**

The notions and applications

Bad news: We show PRO does not exist

	$f_K \equiv f_{K'}$	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**PRO** Null-iO [VWW22] SNARK for UP [**M**PV24] SNARG for NP [JKL**M**25] **This work:** (+LWE) FHE ✓ Succinct Garbling ✓ Succinct witness encryption
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**	**dPRO**

Go to bonus slide

A Weaker Notion

Indistinguishability PRO (iPRO)

$F_{K_0} \equiv F_{K_1},$

A Weaker Notion

Indistinguishability PRO (iPRO)

$F_{K_0} \equiv F_{K_1}$, $TT(F_K)$

A Weaker Notion

Indistinguishability PRO (iPRO)

$F_{K_0} \equiv F_{K_1},$ $TT(F_K)$ $\approx$ U

A Weaker Notion

Indistinguishability PRO (iPRO)

$F_{K_0} \equiv F_{K_1}$, $TT(F_K)$ $\approx$ U

$\approx$

A Weaker Notion

Indistinguishability PRO (iPRO)

$F_{K_0} \equiv F_{K_1}$, $TT(F_K)$ $\approx$ U

$\Downarrow$

$\approx$

A Weaker Notion

Indistinguishability PRO (iPRO)

$F_{K_0} \equiv F_{K_1}$, $TT(F_K)$ $\approx$ U

$\Downarrow$

$iPRO(F_{K_0})$ $\approx$ $iPRO(F_{K_1})$

A Weaker Notion

Indistinguishability PRO (iPRO)

$F_{K_0} \equiv F_{K_1}$, $TT(F_K) \approx U$

$\Downarrow$

$iPRO(F_{K_0}) \approx iPRO(F_{K_1})$

- iO for Pseudorandom Functions

A Weaker Notion

Indistinguishability PRO (iPRO)

$F_{K_0} \equiv F_{K_1}$, $TT(F_K)$ $\approx$ U

$\Downarrow$

$iPRO(F_{K_0})$ $\approx$ $iPRO(F_{K_1})$

- iO for Pseudorandom Functions
- Implied by iO ⇒ no counterexamples!

A Weaker Notion

Indistinguishability PRO (iPRO)

$$F_{K_0} \equiv F_{K_1}, \quad TT(F_K) \approx U$$

$$\Downarrow$$

$$iPRO(F_{K_0}) \approx iPRO(F_{K_1})$$

- iO for Pseudorandom Functions
- Implied by iO $\Rightarrow$ no counterexamples!

xiPRO: $|iPRO(C)| = |TT(C)|^{1-\epsilon}$

The notions and applications

	$f_K \equiv f_{K'}$		$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**		**PRO**
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**		**dPRO**

The notions and applications

	$f_K \equiv f_{K'}$	$f_K \equiv f_{K'}$ and pseudorandom	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**		**PRO**
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**		**dPRO**

The notions and applications

	$f_K \equiv f_{K'}$	$f_K \equiv f_{K'}$ and pseudorandom	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**iPRO**	**PRO**
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**		**dPRO**

The notions and applications

	$f_K \equiv f_{K'}$	$f_K \equiv f_{K'}$ and pseudorandom	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**iPRO**	**PRO**
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**	**Same as dPRO**	**dPRO**

The notions and applications

	$f_K \equiv f_{K'}$	$f_K \equiv f_{K'}$ and pseudorandom	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**iPRO** Null-iO [VWW22] SNARK for UP [**M**PV24] SNARG for NP [JKL**M**25] (+LWE) FHE Succinct garbling	**PRO**
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**	**Same as dPRO**	**dPRO**

The notions and applications

	$f_K \equiv f_{K'}$	$f_K \equiv f_{K'}$ and pseudorandom	$TT(f_K)$ is pseudorandom
$\mathcal{O}(f_K) \approx_c \mathcal{O}(f_{K'})$	**iO**	**iPRO** Null-iO [VWW22] SNARK for UP [**M**PV24] SNARG for NP [JKL**M**25] (+LWE) FHE Succinct garbling	**PRO** Succinct WE (This work) iO for TMs [JJ**M**P25]
$\mathcal{O}(f_K) \approx_c \mathcal{U}$	**Impossible**	**Same as dPRO**	**dPRO**

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to “slightly” compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$xiPRO(PRF_K(\cdot) + C(\cdot))$ $\qquad$ $xiPRO(PRF_K(x))$

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$xiPRO(PRF_K(\cdot) + C(\cdot))$ $\quad$ ~~$xiPRO(PRF_K(x))$~~

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$$xiPRO(PRF_K(\cdot) + C(\cdot)) \qquad xiPRO(PRF_K(x))$$

$$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)}) \qquad e(g_1^{x_i}, g_2^{x_j})$$

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$xiPRO(PRF_K(\cdot) + C(\cdot))$ $\quad$ ~~$xiPRO(PRF$~~$_K(x))$

$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)})$ $\quad$ $e(g_1^{x_i}, g_2^{x_j})$ $\quad$ via $(g_1^{x_i})_i$, $(g_2^{y_j})_j$

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$xiPRO(PRF_K(\cdot) + C(\cdot))$ ~~$xiPRO(PRF_K(x))$~~

$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)})$ $e(g_1^{x_i}, g_2^{x_j})$ ~~via $(g_1^{x_i})_i, (g_2^{x_j})_j$~~

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$xiPRO(PRF_K(\cdot) + C(\cdot))$ ~~$xiPRO(PRF$~~$_K(x))$

$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)})$ $e(g_1^{x_i}, g_2^{x_j})$ ~~via $(g_1^{x_i})_i, (g_2^{y_j})_j$~~

via $QFE.Enc((x_i)_i, (y_j)_j)$

and $\{sk_{i,j}\}_{i,j}$

- Hide the $g_1^{x_i}$, $g_2^{x_j}$ using wrapper of quadratic FE **[Wee'20]**

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to “slightly” compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$$xiPRO(PRF_K(\cdot) + C(\cdot)) \qquad xiPRO(PRF_K(x))$$

$$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)}) \qquad e(g_1^{x_i}, g_2^{x_j}) \qquad \text{via } (g_1^{x_i})_i, (g_2^{y_j})_j$$

via $QFE.Enc((x_i)_i, (y_j)_j)$
and $\{sk_{i,j}\}_{i,j}$

- Hide the $g_1^{x_i}, g_2^{x_j}$ using wrapper of quadratic FE **[Wee’20]**

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$xiPRO(PRF_K(\cdot) + C(\cdot))$ ~~$xiPRO(PRF_K(x))$~~

$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)})$ $e(g_1^{x_i}, g_2^{x_j})$ ~~via $(g_1^{x_i})_i, (g_2^{y_j})_j$~~

via $QFE.Enc((x_i)_i, (y_j)_j)$
and $\{sk_{i,j}\}_{i,j}$

- Hide the $g_1^{x_i}, g_2^{x_j}$ using wrapper of quadratic FE **[Wee'20]**
- Layer of amortisation of quadratic FE keys

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$xiPRO(PRF_K(\cdot) + C(\cdot))$ $\quad$ ~~$xiPRO(PRF_K(x))$~~

$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)})$ $\quad$ $e(g_1^{x_i}, g_2^{x_j})$ $\quad$ ~~via $(g_1^{x_i})_i, (g_2^{y_j})_j$~~

via $QFE.Enc((x_i)_i, (y_j)_j)$
and $\{sk_{i,j}\}_{i,j}$

- Hide the $g_1^{x_i}, g_2^{x_j}$ using wrapper of quadratic FE **[Wee'20]**
- Layer of amortisation of quadratic FE keys
- Does not go through local PRGs, LPN: "Coding Hardness"

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$xiPRO(PRF_K(\cdot) + C(\cdot))$ ~~$xiPRO(PRF_K(x))$~~

$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)})$ $e(g_1^{x_i}, g_2^{x_j})$ ~~via $(g_1^{x_i})_i, (g_2^{y_j})_j$~~

via $QFE.Enc((x_i)_i, (y_j)_j)$
and $\{sk_{i,j}\}_{i,j}$

- Hide the $g_1^{x_i}, g_2^{x_j}$ using wrapper of quadratic FE **[Wee'20]**
- Layer of amortisation of quadratic FE keys
- Does not go through local PRGs, LPN: "Coding Hardness"
- Nice way to "factor" existing iO constructions.

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$xiPRO(PRF_K(\cdot) + C(\cdot))$ ~~$xiPRO(PRF_K(x))$~~

$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)})$ $e(g_1^{x_i}, g_2^{x_j})$ ~~via $(g_1^{x_i})_i, (g_2^{y_j})_j$~~

via $QFE.Enc((x_i)_i, (y_j)_j)$
and $\{sk_{i,j}\}_{i,j}$

- Hide the $g_1^{x_i}$, $g_2^{x_j}$ using wrapper of quadratic FE **[Wee'20]**
- Layer of amortisation of quadratic FE keys
- Does not go through local PRGs, LPN: "Coding Hardness"
- Nice way to "factor" existing iO constructions.

xiO =

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

$xiPRO(PRF_K(\cdot) + C(\cdot))$ ~~$xiPRO(PRF_K(x))$~~

$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)})$ $e(g_1^{x_i}, g_2^{x_j})$ ~~via $(g_1^{x_i})_i, (g_2^{y_j})_j$~~

via $QFE.Enc((x_i)_i, (y_j)_j)$
and $\{sk_{i,j}\}_{i,j}$

- Hide the $g_1^{x_i}, g_2^{x_j}$ using wrapper of quadratic FE **[Wee'20]**
- Layer of amortisation of quadratic FE keys
- Does not go through local PRGs, LPN: "Coding Hardness"
- Nice way to "factor" existing iO constructions.

xiO =

x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to "slightly" compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

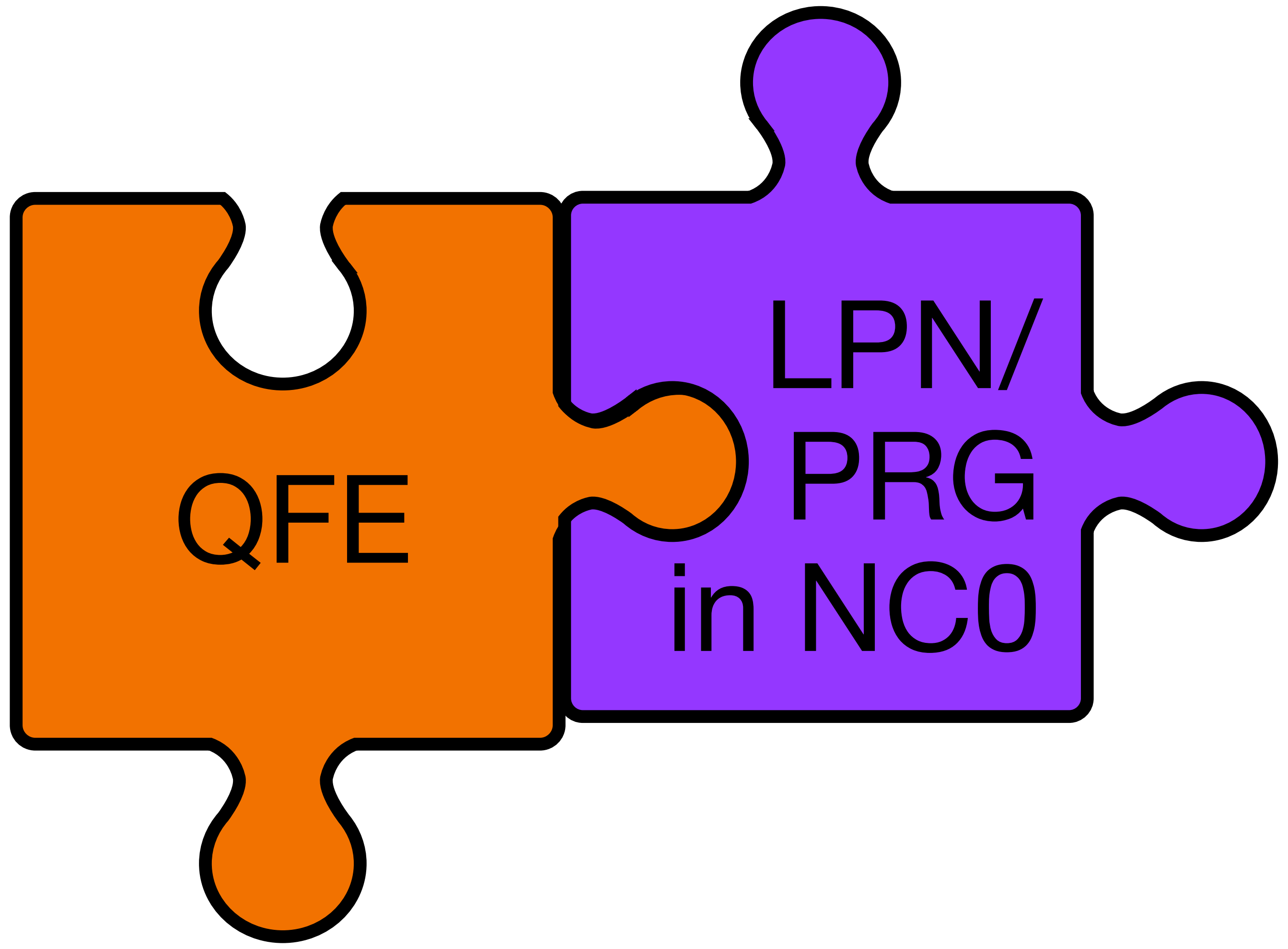
$xiPRO(PRF_K(\cdot) + C(\cdot))$ $xiPRO(PRF_K(x))$

$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)})$ $e(g_1^{x_i}, g_2^{x_j})$ via $(g_1^{x_i})_i, (g_2^{y_j})_j$

via $QFE.Enc((x_i)_i, (y_j)_j)$
and $\{sk_{i,j}\}_{i,j}$

- Hide the $g_1^{x_i}, g_2^{x_j}$ using wrapper of quadratic FE **[Wee'20]**
- Layer of amortisation of quadratic FE keys
- Does not go through local PRGs, LPN: "Coding Hardness"
- Nice way to "factor" existing iO constructions.

xiO =



x-iPRO + Bilinear Maps = x-iO!

(x-$\mathcal{O}$ refers to “slightly” compressing $\mathcal{O}$ such that $\mathcal{O}(|C|) = |TT(C)|^{1-\epsilon}$)

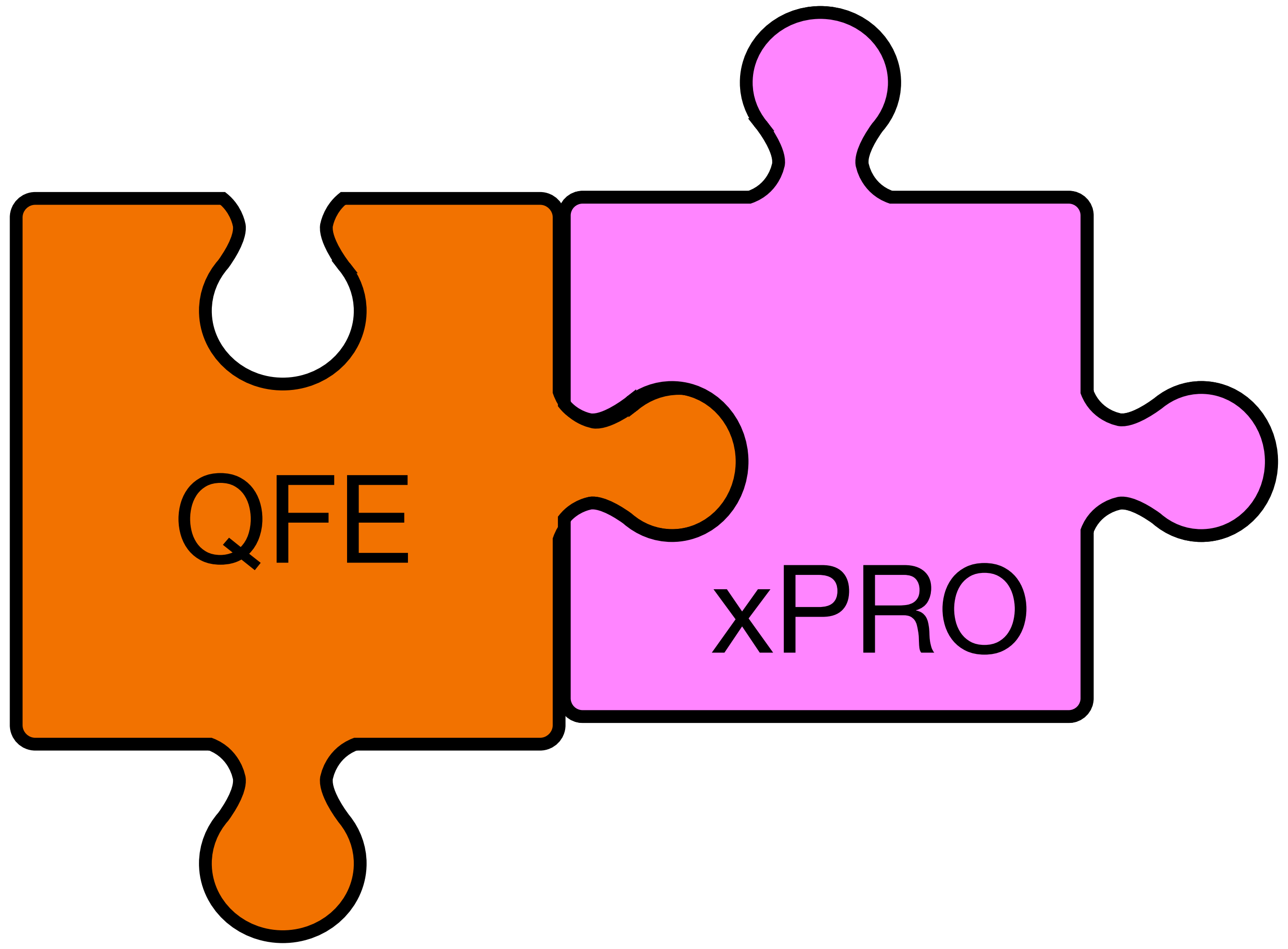
$xiPRO(PRF_K(\cdot) + C(\cdot))$ ~~$xiPRO(PRF_K(x))$~~

$xiPRO((i,j) \mapsto e(g_1^{x_i}, g_2^{x_j}) \cdot g_T^{C(i,j)})$ $e(g_1^{x_i}, g_2^{x_j})$ ~~via $(g_1^{x_i})_i, (g_2^{y_j})_j$~~

via $QFE.Enc((x_i)_i, (y_j)_j)$
and $\{sk_{i,j}\}_{i,j}$

- Hide the $g_1^{x_i}, g_2^{x_j}$ using wrapper of quadratic FE **[Wee’20]**
- Layer of amortisation of quadratic FE keys
- Does not go through local PRGs, LPN: “Coding Hardness”
- Nice way to “factor” existing iO constructions.

xiO =



Summary

Summary

- We study varying notions of **pseudorandom obfuscation**.

Summary

- We study varying notions of **pseudorandom obfuscation**.
    - Strongest notions are **impossible**.

Summary

- We study varying notions of **pseudorandom obfuscation**.
  - Strongest notions are **impossible**.
  - We show that **iPRO is sufficient** for many applications, including FHE and succinct garbling.

Summary

- We study varying notions of **pseudorandom obfuscation**.
    - Strongest notions are **impossible**.
    - We show that **iPRO is sufficient** for many applications, including FHE and succinct garbling.
        - Implied by iO → No impossibility!

Summary

- We study varying notions of **pseudorandom obfuscation**.
  - Strongest notions are **impossible**.
  - We show that **iPRO is sufficient** for many applications, including FHE and succinct garbling.
    - Implied by iO → No impossibility!
  - iPRO + Bilinear maps = iO

Summary

- We study varying notions of **pseudorandom obfuscation**.
  - Strongest notions are **impossible**.
  - We show that **iPRO is sufficient** for many applications, including FHE and succinct garbling.
    - Implied by iO → No impossibility!
  - iPRO + Bilinear maps = iO
    - Gives a **“modular”** approach to iO.

Summary

- We study varying notions of **pseudorandom obfuscation**.
  - Strongest notions are **impossible**.
  - We show that **iPRO is sufficient** for many applications, including FHE and succinct garbling.
    - Implied by iO → No impossibility!
  - iPRO + Bilinear maps = iO
    - Gives a **“modular”** approach to iO.
    - **Open:** Can we construct iPRO from LPN variants/PRGs in NC0? Or even LWE?

Summary

- We study varying notions of **pseudorandom obfuscation**.
  - Strongest notions are **impossible**.
  - We show that **iPRO is sufficient** for many applications, including FHE and succinct garbling.
    - Implied by iO → No impossibility!
  - iPRO + Bilinear maps = iO
    - Gives a **“modular”** approach to iO.
    - **Open:** Can we construct iPRO from LPN variants/PRGs in NC0? Or even LWE?
- (Not in talk) We give a *candidate* construction via the evasive LWE heuristic (more on this in the next talk!)

Thank you for your attention!

Bonus slides

Counterexample to PRO

Precondition

Counterexample to PRO

Precondition

aux

$x = TT(PRF_K)$	WE("x is TT of small C",0^λ)

Counterexample to PRO

Precondition

Pick a witness encryption which is instance-hiding

aux

$x = TT(PRF_K)$

WE("x is TT of small C",0^{λ})

Counterexample to PRO

Precondition

Pick a witness encryption which is instance-hiding

aux

$x = TT(PRF_K)$	WE(“x is TT of small C”,0^λ)
$\approx$	
u	WE(“u is TT of small C”,0^λ)

Counterexample to PRO

Precondition

Pick a witness encryption which is instance-hiding

aux

$x = TT(PRF_K)$	WE("x is TT of small C",0^λ)

$\approx$

u	WE("u is TT ~~of~~ small C",0^λ)

Counterexample to PRO

Precondition

Pick a witness encryption which is instance-hiding

aux

$x = TT(PRF_K)$	WE(“x is TT of small C”,0^λ)
$\approx$	
u	WE(“u is TT of small C”,0^λ)
$\approx$	
u	WE(“u’ is TT of small C”,0^λ)

Counterexample to PRO

Precondition

Pick a witness encryption which is instance-hiding

aux

$x = TT(PRF_K)$	WE(“x is TT of small C”,0^λ)
$\approx$	
u	WE(“u is TT ~~of~~ small C”,0^λ)
$\approx$	
u	WE(“u’ is TT of small C”,0^λ)
$\approx$	
u	WE(“x is TT of small C”,0^λ)

Counterexample to PRO

Postcondition

Back to main body

Counterexample to PRO

Postcondition

$$x = TT(PRF_K) = TT(PRO(PRF_K))$$

Back to main body

Counterexample to PRO

Postcondition

$$x = TT(PRF_K) = TT(PRO(PRF_K))$$

$$PRO(PRF_K)$$

Back to main body

Counterexample to PRO

Postcondition

$x = TT(PRF_K) = TT(PRO(PRF_K))$

aux

$PRO(PRF_K)$

WE(“x is TT of small C”,0^λ)

Back to main body

Counterexample to PRO

Postcondition

$x = TT(PRF_K) = TT(PRO(PRF_K))$

aux

$PRO(PRF_K)$

WE(“x is TT of small C”,0^λ)

u'

WE(“x is TT of small C”,0^λ)

Back to main body

Counterexample to PRO

Postcondition

$x = TT(PRF_K) = TT(PRO(PRF_K))$

aux

$w =$ $PRO(PRF_K)$

WE("x is TT of small C",0^λ)

u'

WE("x is TT of small C",0^λ)

Back to main body

Counterexample to PRO

Postcondition

$x = TT(PRF_K) = TT(PRO(PRF_K))$

aux

$w =$ $PRO(PRF_K)$

WE(“x is TT of small C”,0^λ)

$WE.Dec$

0^λ

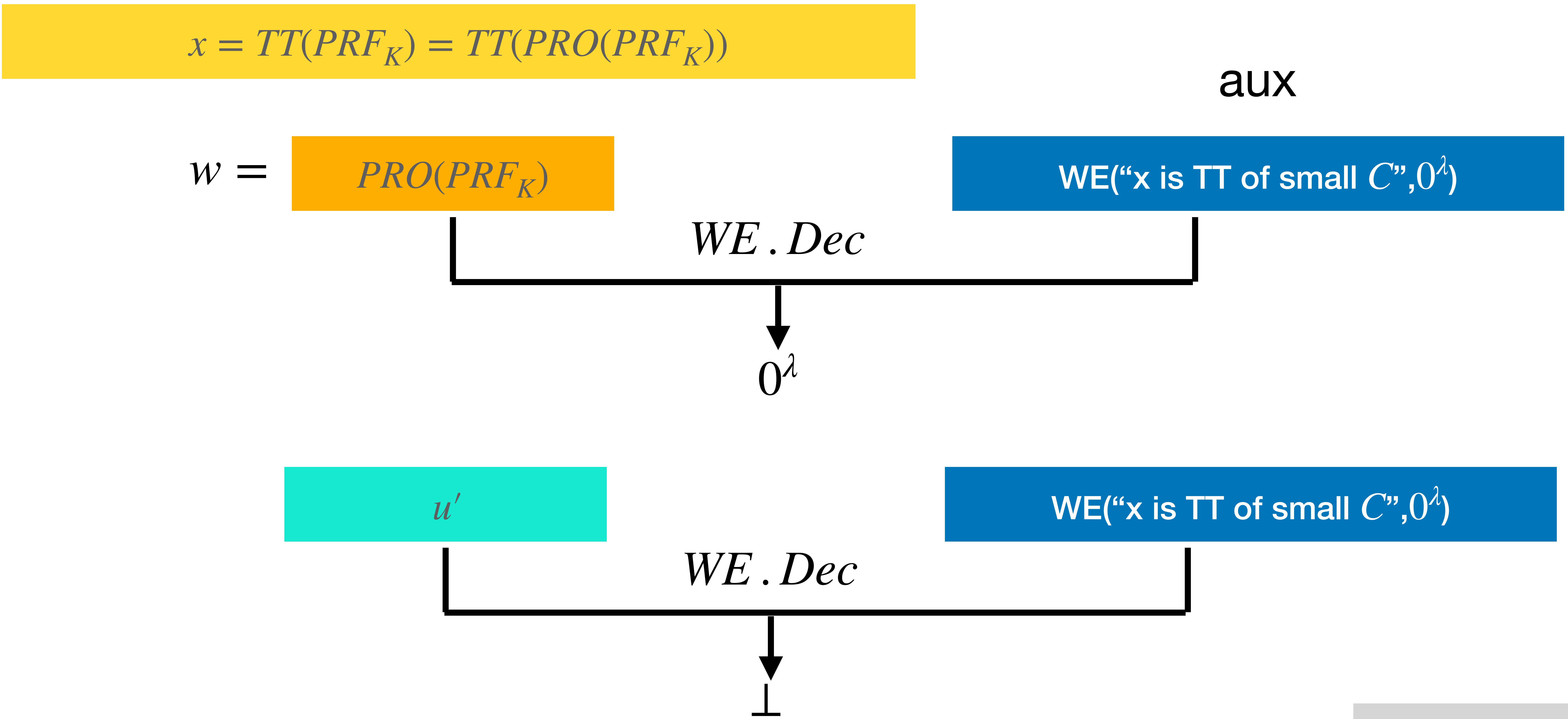
u'

WE(“x is TT of small C”,0^λ)

Back to main body

Counterexample to PRO

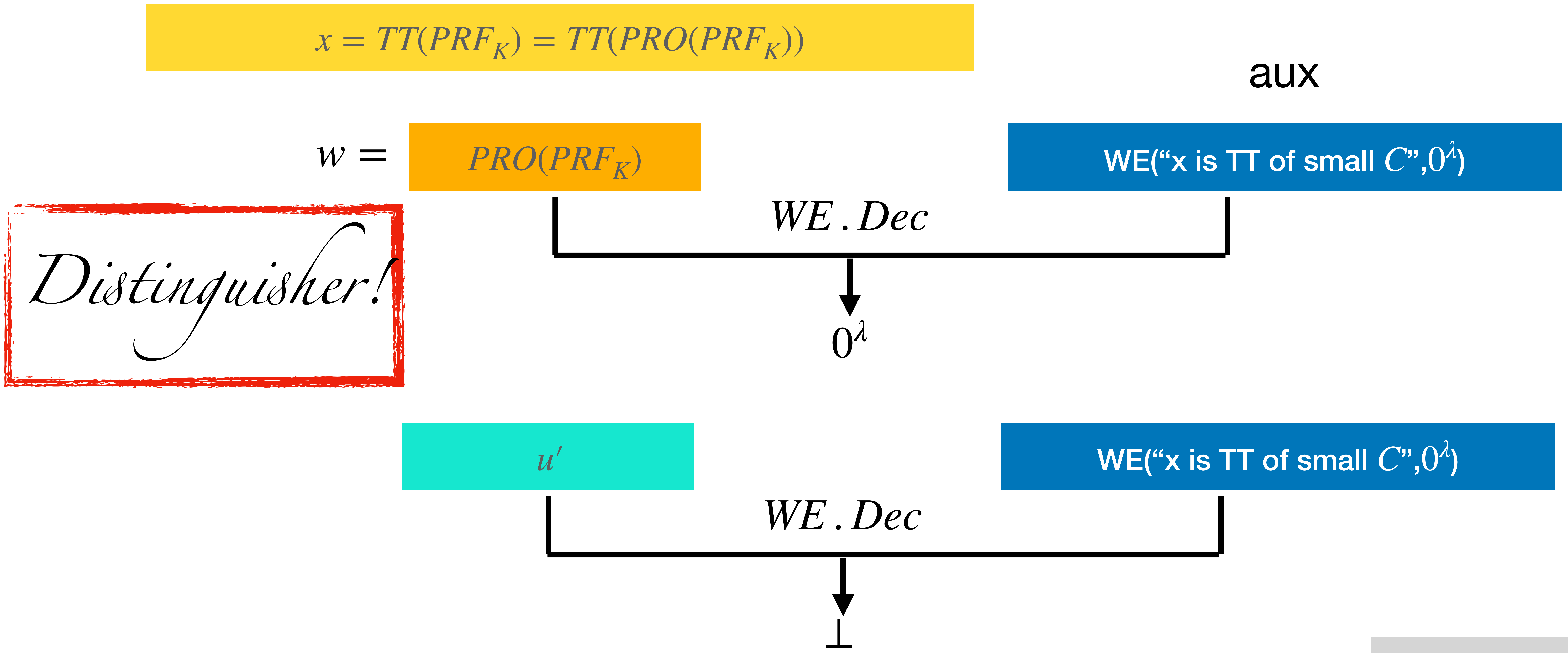
Postcondition



Back to main body

Counterexample to PRO

Postcondition



Back to main body