Towards Higher-Order Cryptography

Raphaëlle Crubillé
joint work with:
Ugo Dal Lago

REPAS Workshop, June 2017

Why higher-order cryptogaphy ?

Pseudo-Random Generator

(base type)

$$\mathsf{Str}^n \to \mathsf{Str}^{r(n)}$$

Why higher-order cryptogaphy ?

Pseudo-Random Generator

(base type)

$\mathrm{Str}^n \to \mathrm{Str}^{r(n)}$

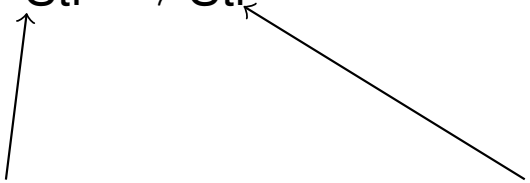
small quantity of
true randomness

Why higher-order cryptogaphy ?

Pseudo-Random Generator

(base type)

$\mathrm{Str}^n \to \mathrm{Str}^{r(n)}$



small quantity of true randomness

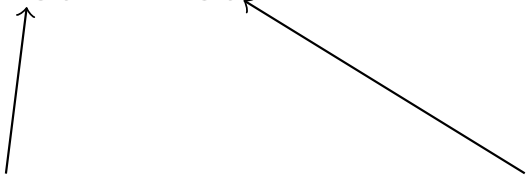
larger quantity of pseudo-randomness

Why higher-order cryptogaphy ?

Pseudo-Random Generator

(base type)

$\mathrm{Str}^n \to \mathrm{Str}^{r(n)}$: deterministic program



small quantity of true randomness

larger quantity of pseudo-randomness

Why higher-order cryptogaphy ?

Pseudo-Random Generator

(base type)

$\mathsf{Str}^n \to \mathsf{Str}^{r(n)}$

Encryption scheme

secure for passive adversary.

(KEY, ENC, DEC)

Why higher-order cryptogaphy ?

Pseudo-Random Generator

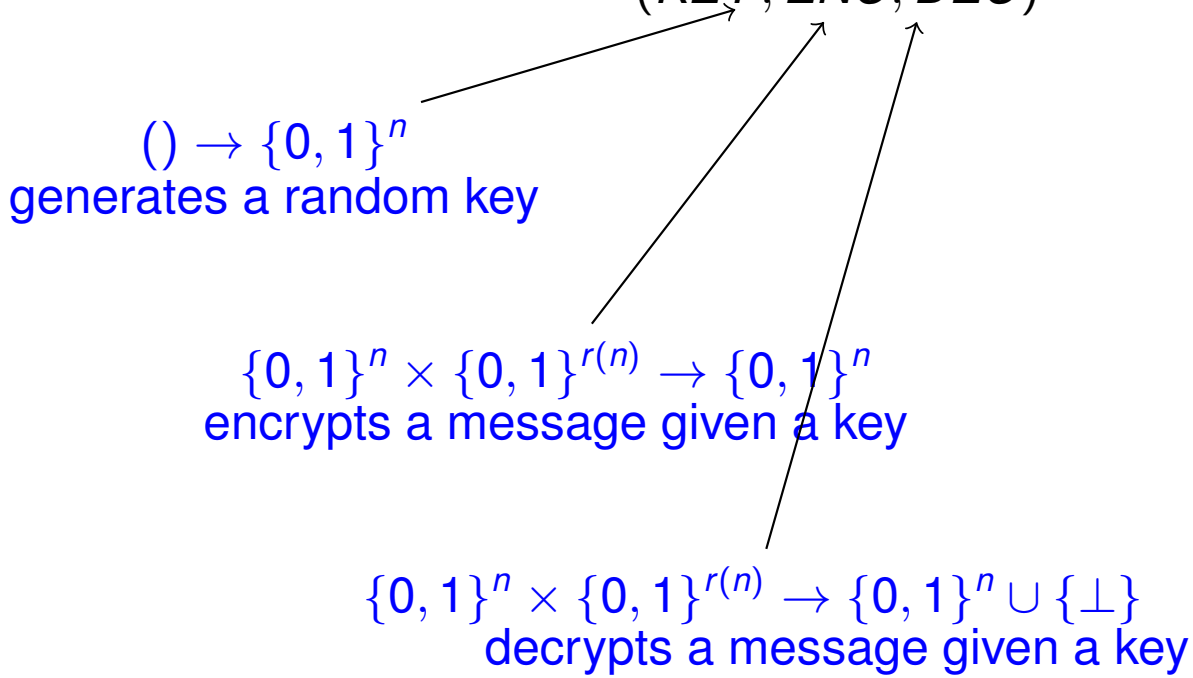
(base type)

$\mathsf{Str}^n \to \mathsf{Str}^{r(n)}$

Encryption scheme

secure for passive adversary.

(KEY, ENC, DEC)



Why higher-order cryptogaphy ?

Pseudo-Random Generator

(base type)

$\mathsf{Str}^n \to \mathsf{Str}^{r(n)}$

Encryption scheme

secure for passive adversary.

(KEY, ENC, DEC)

$() \to \{0,1\}^n$
generates a random key

probabilistic programs

$\{0,1\}^n \times \{0,1\}^{r(n)} \to \{0,1\}^n$
encrypts a message given a key

$\{0,1\}^n \times \{0,1\}^{r(n)} \to \{0,1\}^n \cup \{\bot\}$
decrypts a message given a key

Why higher-order cryptogaphy ?

Pseudo-Random Generator (base type) $\mathsf{Str}^n \to \mathsf{Str}^{r(n)}$

$\longrightarrow$ can be used to construct $\longrightarrow$

Encryption scheme secure for passive adversary. (KEY, ENC, DEC)

Why higher-order cryptogaphy ?

Pseudo-Random Generator (base type) $\mathsf{Str}^n \to \mathsf{Str}^{r(n)}$

$\longrightarrow$ can be used to construct $\longrightarrow$

Encryption scheme secure for passive adversary. (KEY, ENC, DEC)

Pseudo-Random Function (first-order) $\mathsf{Str}^n \to (\mathsf{Str}^{r(n)} \to \mathsf{Str}^{r(n)})$

Why higher-order cryptogaphy ?

Pseudo-Random Generator (base type) $\mathsf{Str}^n \to \mathsf{Str}^{r(n)}$

can be used to construct

Encryption scheme secure for passive adversary. (KEY, ENC, DEC)

Pseudo-Random Function (first-order) $\mathsf{Str}^n \to (\mathsf{Str}^{r(n)} \to \mathsf{Str}^{r(n)})$

Encryption scheme CPA-secure. (KEY, ENC, DEC)

Why higher-order cryptogaphy ?

Pseudo-Random Generator (base type) $\mathsf{Str}^n \to \mathsf{Str}^{r(n)}$ — can be used to construct → Encryption scheme secure for passive adversary. (KEY, ENC, DEC)

Pseudo-Random Function (first-order) $\mathsf{Str}^n \to (\mathsf{Str}^{r(n)} \to \mathsf{Str}^{r(n)})$ → Encryption scheme CPA-secure. (KEY, ENC, DEC)

Pseudo-Random Function (second-order) $\mathsf{Str}^n \to (\mathsf{Str}^{r(n)} \to \mathsf{Str}^{r(n)}) \to \mathsf{Str}$?

Why higher-order cryptogaphy ?

Pseudo-Random Generator (base type) $\mathsf{Str}^n \to \mathsf{Str}^{r(n)}$

—can be used to construct→ Encryption scheme secure for passive adversary. (KEY, ENC, DEC)

Goldreich-Goldwasser-Micali ↓

Pseudo-Random Function (first-order) $\mathsf{Str}^n \to (\mathsf{Str}^{r(n)} \to \mathsf{Str}^{r(n)})$

→ Encryption scheme CPA-secure. (KEY, ENC, DEC)

Pseudo-Random Function (second-order) $\mathsf{Str}^n \to (\mathsf{Str}^{r(n)} \to \mathsf{Str}^{r(n)}) \to \mathsf{Str}$

?

Some Possible Applications of Higher-Order Schemes in Security

Message Authentication code from a PRF F

Signature scheme:
(KEY, SIGN, VERIFY) with:

$$\text{VERIFY}(k, m, \text{SIGN(k,m)}) = 1$$

$\Rightarrow$ allows to sign a message.

Some Possible Applications of Higher-Order Schemes in Security

Message Authentication code from a PRF F

Signature scheme:
(KEY, SIGN, VERIFY) with:

$$\text{VERIFY}(k, m, \text{SIGN(k,m)}) = 1$$

$\Rightarrow$ allows to sign a message.

F a PRF :

$$SIGN(k, m) = F(k, m)$$

It is secure if F is secure.

Some Possible Applications of Higher-Order Schemes in Security

Message Authentication code from a PRF F

Signature scheme:
(KEY, SIGN, VERIFY) with:

$$\text{VERIFY}(k, m, \text{SIGN(k,m)}) = 1$$

$\Rightarrow$ allows to sign a message.

F a PRF :

$$SIGN(k, m) = F(k, m)$$

It is secure if F is secure.

From a PRF at order 2: Function Authentication ?

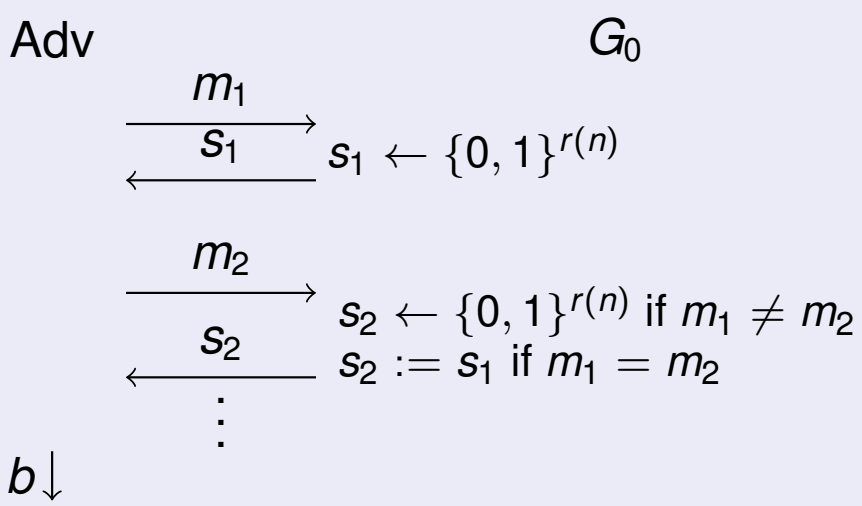
Goal: sign programs without looking at its code, but only its input/output behaviour.

Applications ?

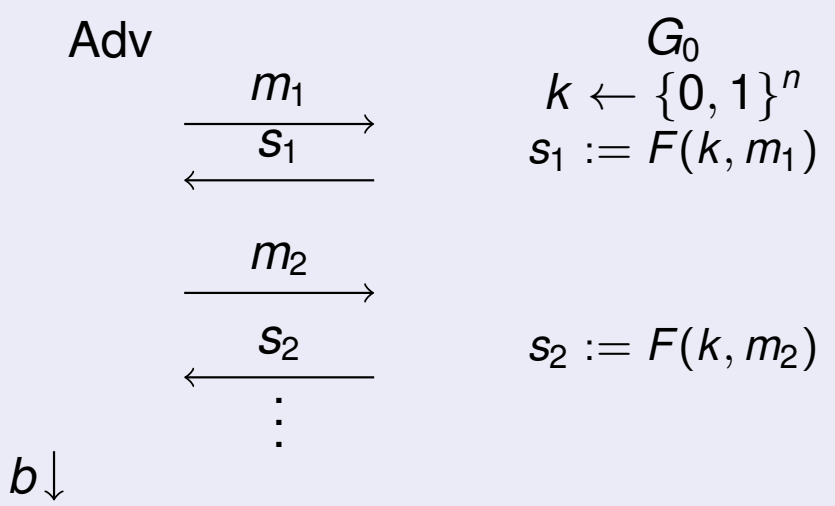
- Cloud computing
- Obfuscation

Security for order 1 PRF $F : \mathrm{Str}^n \to (\mathrm{Str}^{r(n)} \to \mathrm{Str}^{r(n)})$

Game 0(F)

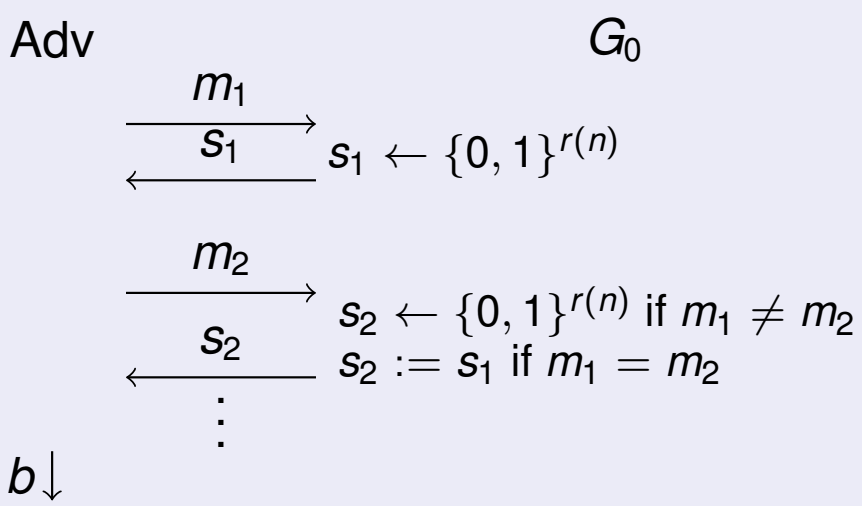


Game 1(F)

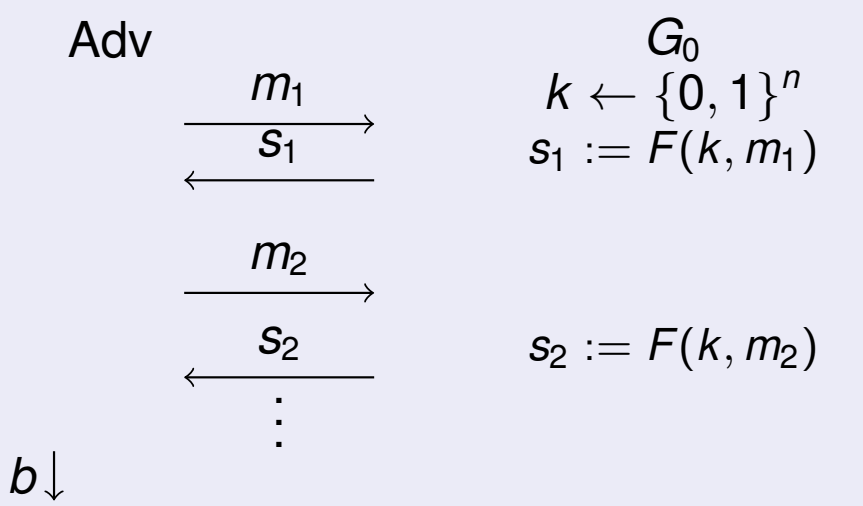


Security for order 1 PRF $F : \mathrm{Str}^n \to (\mathrm{Str}^{r(n)} \to \mathrm{Str}^{r(n)})$

Game 0(F)



Game 1(F)



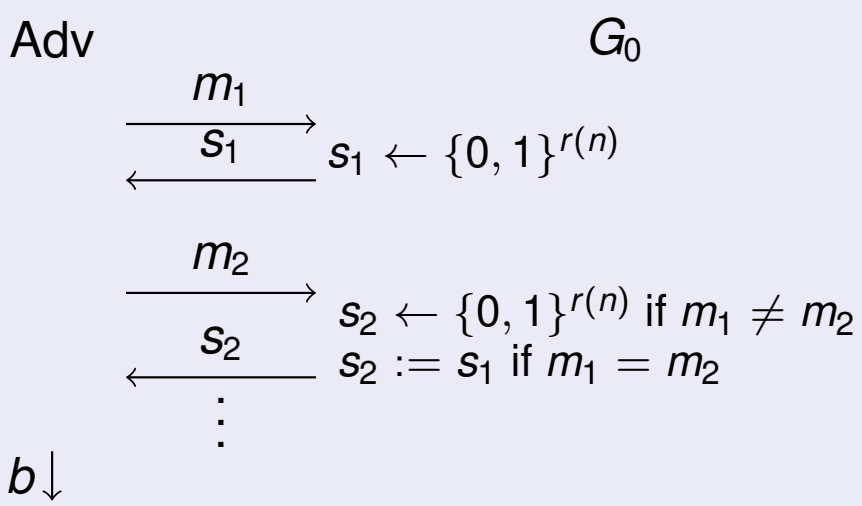
Definition (Advantage of a PRF-adversary against F)

$$\mathrm{Advantage}(Adv) = |Prob_{Game0}(b = 0) - Prob_{Game1}(b = 0)|$$

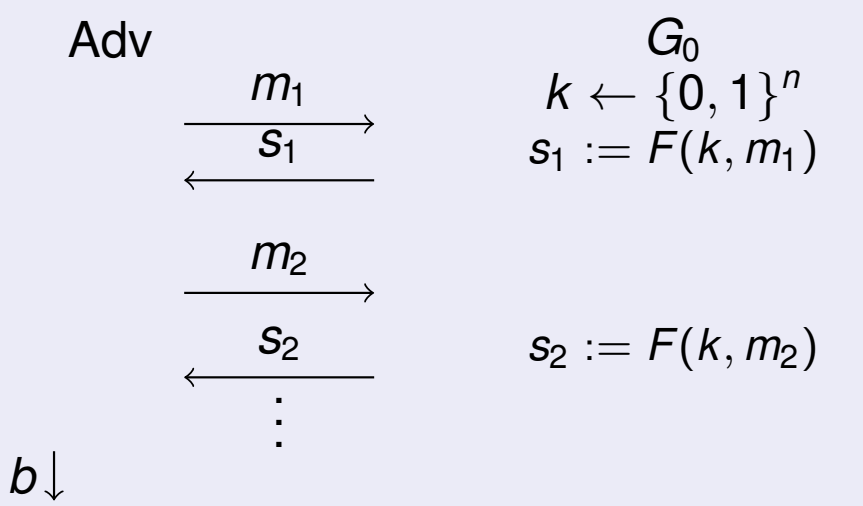
To which extent is *Adv*'s behaviour different in G_0 and G_1.

Security for order 1 PRF $F : \mathrm{Str}^n \to (\mathrm{Str}^{r(n)} \to \mathrm{Str}^{r(n)})$

Game 0(F)



Game 1(F)



Definition (Advantage of a PRF-adversary against F)

$$\mathrm{Advantage}(Adv) = |Prob_{Game0}(b = 0) - Prob_{Game1}(b = 0)|$$

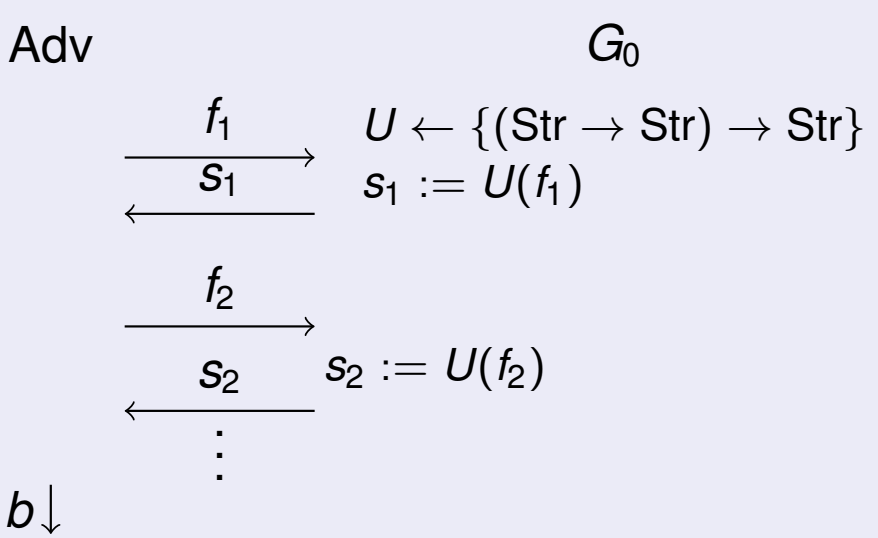
To which extent is *Adv*'s behaviour different in G_0 and G_1.

Definition (Security for PRF)

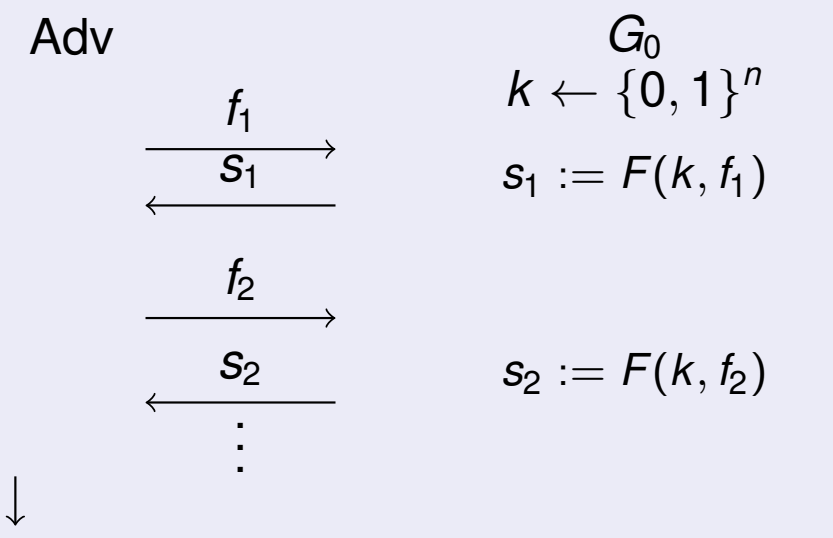
A PRF F is **secure** if $\forall Adv$ polytime, the advantage of *Adv* against F is negligible.

Security for an Order 2 PRF $F : \mathrm{Str}^n \to (\mathrm{Str}^{r(n)} \to \mathrm{Str}^{r(n)}) \to \mathrm{Str}^{r(n)}$?

Game 0(F)

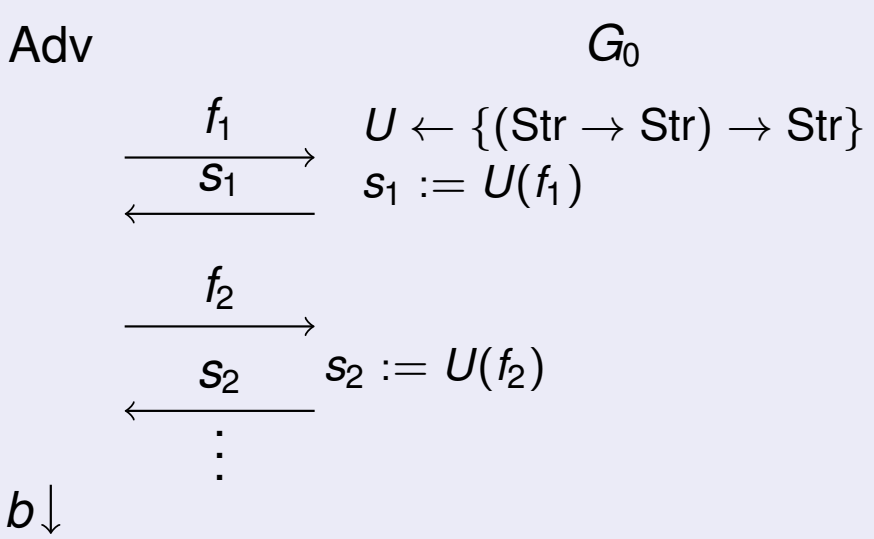


Game 1(F)

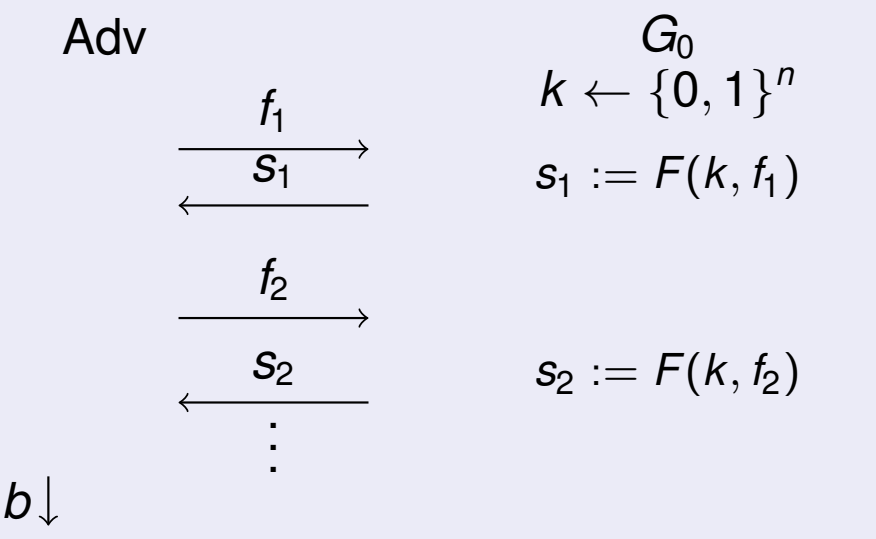


Security for an Order 2 PRF $F : \mathrm{Str}^n \to (\mathrm{Str}^{r(n)} \to \mathrm{Str}^{r(n)}) \to \mathrm{Str}^{r(n)}$?

Game 0(F)



Game 1(F)



There is no polytime computable F winning this security game for all Adv.

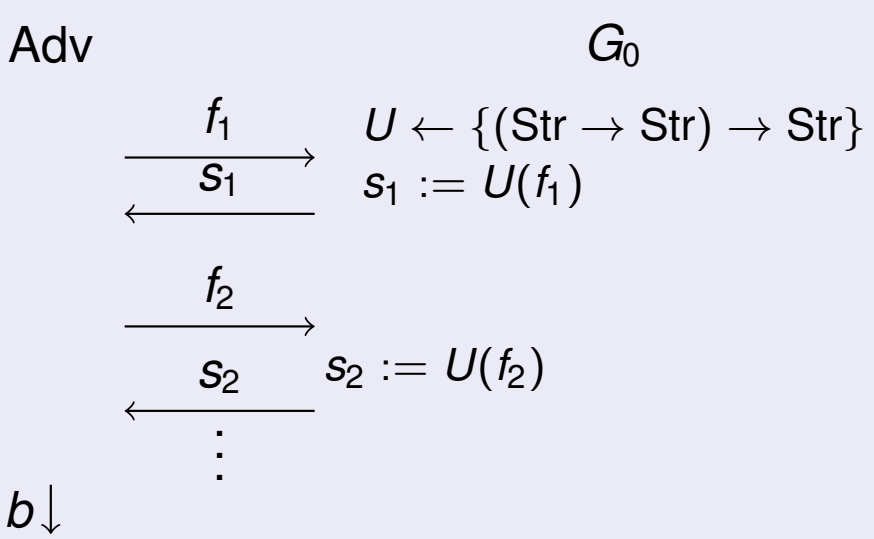
proof

Adversary: chooses a random string m.

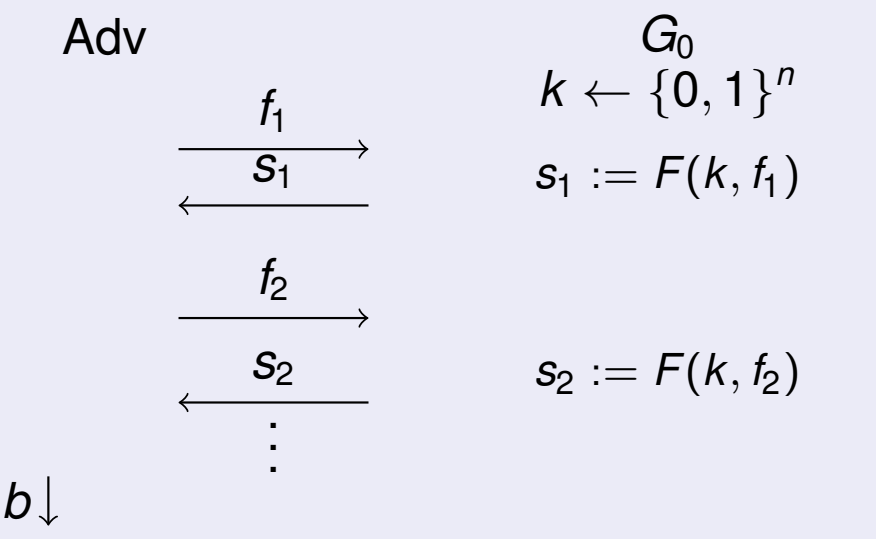
$$f_1 = 0; \qquad f_2(z) = \begin{cases} 1 \text{ if } z = m \\ 0 \text{ otherwise.} \end{cases}$$

Security for an Order 2 PRF $F : \mathrm{Str}^n \to (\mathrm{Str}^{r(n)} \to \mathrm{Str}^{r(n)}) \to \mathrm{Str}^{r(n)}$?

Game 0(F)



Game 1(F)



There is no polytime computable F winning this security game for all Adv.

proof

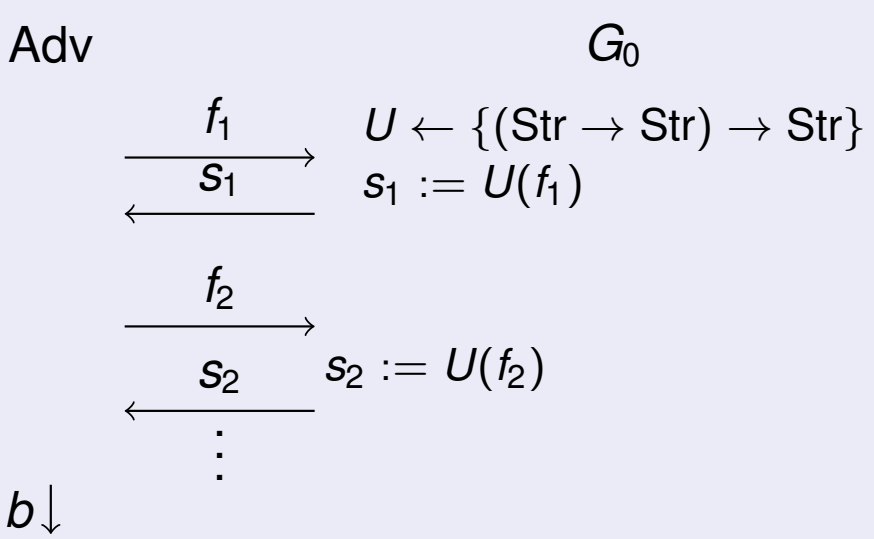
Adversary: chooses a random string m.

$$f_1 = 0; \qquad f_2(z) = \begin{cases} 1 \text{ if } z = m \\ 0 \text{ otherwise.} \end{cases}$$

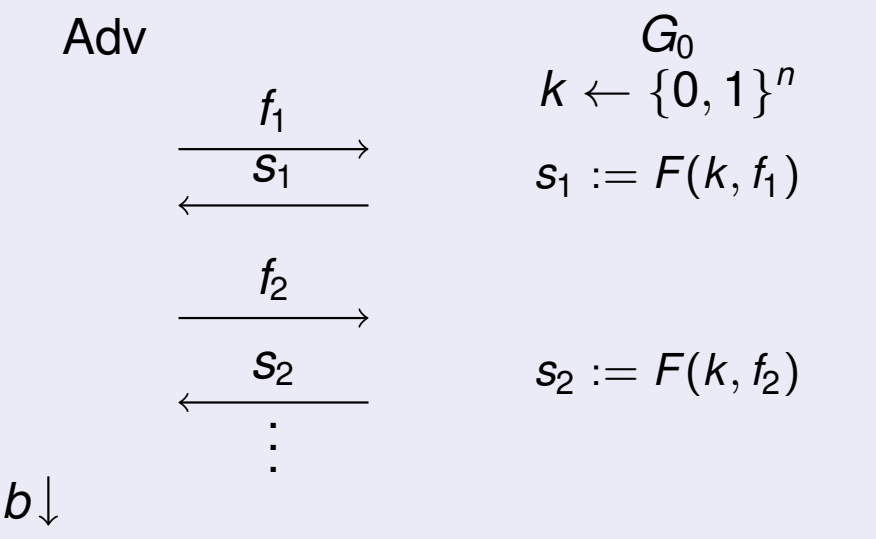
$f_1 \neq f_2 \Rightarrow U(f_1) = U(f_2)$: small probability

Security for an Order 2 PRF $F : \mathrm{Str}^n \to (\mathrm{Str}^{r(n)} \to \mathrm{Str}^{r(n)}) \to \mathrm{Str}^{r(n)}$?

Game 0(F)



Game 1(F)



There is no polytime computable F winning this security game for all Adv.

proof

Adversary: chooses a random string m.

$$f_1 = 0; \qquad f_2(z) = \begin{cases} 1 \text{ if } z = m \\ 0 \text{ otherwise.} \end{cases}$$

$f_1 \neq f_2 \Rightarrow U(f_1) = U(f_2)$: small probability

In a polynomial number of steps, the probability that F is able to distinguish between f_1 and f_2 is negligible.

$\Rightarrow F(f_1) = F(f_2)$ with high probability.

Probabilistic Game Semantics for Cryptography

Game semantics: first-order model of higher-order computations.

Requirement of the model

- probabilities $\Leftarrow$ Adversaries and programs are probabilistic
  Danos-Harmer: Probabilistic Game Semantics [LICS2000]
- strategies seen as computations (instad of denotation of a fixed language)
  $\Leftarrow$ adversaries should be *as expressive* as possible .
  Longley:Some Programming Languages Suggested by Game Models [TCS2009]
- polytime computations
  $\Leftarrow$ adversaries runtime should be *polynomial* in the security parameter.
  Hugo Ferree: Game semantics approach to higher-order complexity [JCSS2017].

Adding polytime constraints: Deterministic Case

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n \in \mathbb{N}})$
- Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathsf{Odd} \to P_G)$

Adding polytime constraints: Deterministic Case

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n\in\mathbb{N}})$
- Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathsf{Odd} \to P_G)$

Example (Strings of length $\leq p(n)$)

$\mathbf{S}[p] = (\{?\}, \{0,1\}^\star, (L_n^{\mathbf{S}[p]})_{n\in\mathbb{N}})$ with
$L_n^{\mathbf{S}[p]} = \{\epsilon, ?\} \cup \{?s \mid |s| \leq p(n)\}$

Adding polytime constraints: Deterministic Case

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n\in\mathbb{N}})$
- Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathsf{Odd} \to P_G)$

Example (Strings of length $\leq p(n)$)

$\mathbf{S}[p] = (\{?\}, \{0,1\}^\star, (L_n^{\mathbf{S}[p]})_{n\in\mathbb{N}})$ with
$L_n^{\mathbf{S}[p]} = \{\epsilon, ?\} \cup \{?s \mid |s| \leq p(n)\}$

Definitions

Polynomially Bounded Games:
G such that there exists a polynomial P with positive coefficients, such that:
$\forall n \in \mathbb{N}, \forall s \in L_G^n, |s| \leq P(n)$.

Polytime Computable Strategies:
There exists a Turing machine polytime in its first input, which on the entry (n, s) returns $f(n)(s)$.

Adding polytime constraints: Deterministic Case

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n\in\mathbb{N}})$
- Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathsf{Odd} \to P_G)$

Example (Strings of length $\leq p(n)$)

$\mathbf{S}[p] = (\{?\}, \{0,1\}^\star, (L_n^{\mathbf{S}[p]})_{n\in\mathbb{N}})$ with
$L_n^{\mathbf{S}[p]} = \{\epsilon, ?\} \cup \{?s \mid |s| \leq p(n)\}$

Definitions

Polynomially Bounded Games:
G such that there exists a polynomial P with positive coefficients, such that:
$\forall n \in \mathbb{N}, \forall s \in L_G^n, |s| \leq P(n)$.

Polytime Computable Strategies:
There exists a Turing machine polytime in its first input, which on the entry (n, s) returns $f(n)(s)$.

Definition (Bounded Exponentials)

Pol: set of polynomials p with positive integer coefficients.
$!_p G$: corresponds to $p(n)$ copies of G.

Adding polytime constraints: Deterministic Case

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n\in\mathbb{N}})$
- Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathrm{Odd} \to P_G)$

Example (Strings of length $\leq p(n)$)

$\mathbf{S}[p] = (\{?\}, \{0,1\}^\star, (L_n^{\mathbf{S}[p]})_{n\in\mathbb{N}})$ with
$L_n^{\mathbf{S}[p]} = \{\epsilon, ?\} \cup \{?s \mid |s| \leq p(n)\}$

Definitions

Polynomially Bounded Games:
G such that there exists a polynomial P with positive coefficients, such that:
$\forall n \in \mathbb{N}, \forall s \in L_G^n, |s| \leq P(n)$.

Polytime Computable Strategies:
There exists a Turing machine polytime in its first input, which on the entry (n, s) returns $f(n)(s)$.

Definition (Bounded Exponentials)

Pol: set of polynomials p with positive integer coefficients.
$!_p G$: corresponds to $p(n)$ copies of G.

Polynomially bounded games are preserved by $!_p$.

Adding polytime constraints: Deterministic Case

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n\in\mathbb{N}})$
- Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathrm{Odd} \to P_G)$

Example (Strings of length $\leq p(n)$)

$\mathbf{S}[p] = (\{?\}, \{0,1\}^\star, (L_n^{\mathbf{S}[p]})_{n\in\mathbb{N}})$ with
$L_n^{\mathbf{S}[p]} = \{\epsilon, ?\} \cup \{?s \mid |s| \leq p(n)\}$

Definitions

Polynomially Bounded Games:
G such that there exists a polynomial P with positive coefficients, such that:
$\forall n \in \mathbb{N}, \forall s \in L_G^n, |s| \leq P(n)$.

Polytime Computable Strategies:
There exists a Turing machine polytime in its first input, which on the entry (n, s) returns $f(n)(s)$.

Definition (Bounded Exponentials)

Pol: set of polynomials p with positive integer coefficients.
$!_p G$: corresponds to $p(n)$ copies of G.

Polynomially bounded games are preserved by $!_p$.

Proposition (Stability of polytime strategies)

If f, g are polytime computable strategies respectively on $G \multimap H$, and $H \multimap K$, with G, H, K bounded games, then $g \circ f$ is a polytime computable on $G \multimap K$.

Adding polytime Constraints: Probabilistic Case (1)

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n \in \mathbb{N}})$
- Probabilistic Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathsf{Odd} \to \Delta(P_G))$

Adding polytime Constraints: Probabilistic Case (1)

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n \in \mathbb{N}})$
- Probabilistic Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathsf{Odd} \to \Delta(P_G))$

Question:

Are probabilistic polytime computable strategies stable by composition ?

Adding polytime Constraints: Probabilistic Case (1)

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n \in \mathbb{N}})$
- Probabilistic Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathsf{Odd} \to \Delta(P_G))$

Question:

Are probabilistic polytime computable strategies stable by composition ?

Example

$F : \mathbf{S}[X] \multimap \mathbf{S}[P]$ a one-way function.

Adding polytime Constraints: Probabilistic Case (1)

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n\in\mathbb{N}})$
- Probabilistic Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathrm{Odd} \to \Delta(P_G))$

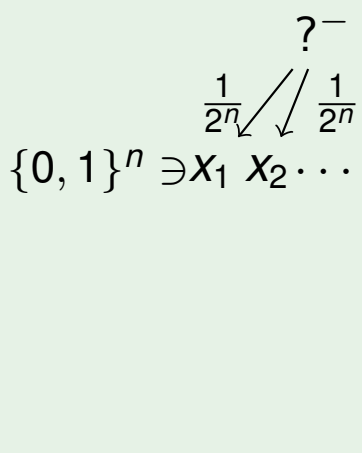
Question:

Are probabilistic polytime computable strategies stable by composition ?

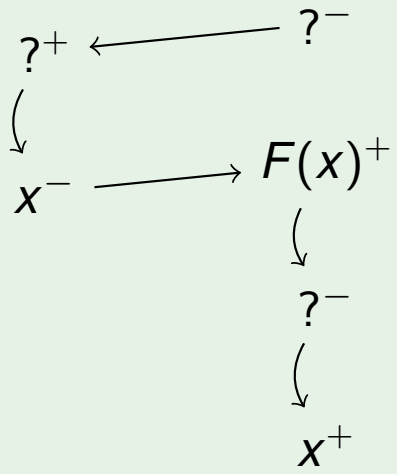
Example

$F : \mathbf{S}[X] \multimap \mathbf{S}[P]$ a one-way function.

$g : \mathbf{1} \multimap \mathbf{S}[X]$



$f : \mathbf{S}[X] \multimap \mathbf{S}[P] \oslash \mathbf{S}[X]$



Adding polytime Constraints: Probabilistic Case (1)

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n\in\mathbb{N}})$
- Probabilistic Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathrm{Odd} \to \Delta(P_G))$

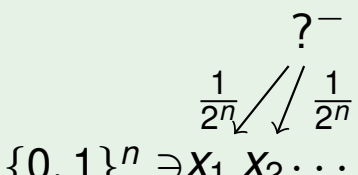
Question:

Are probabilistic polytime computable strategies stable by composition ?

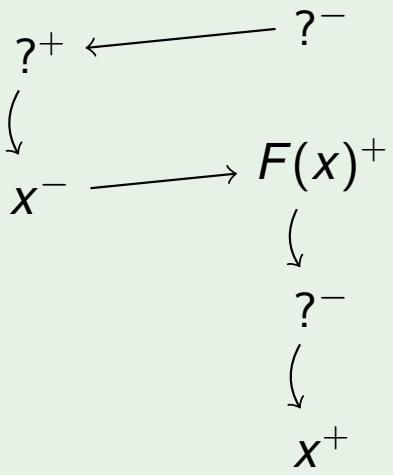
Example

$F : \mathbf{S}[X] \multimap \mathbf{S}[P]$ a one-way function.

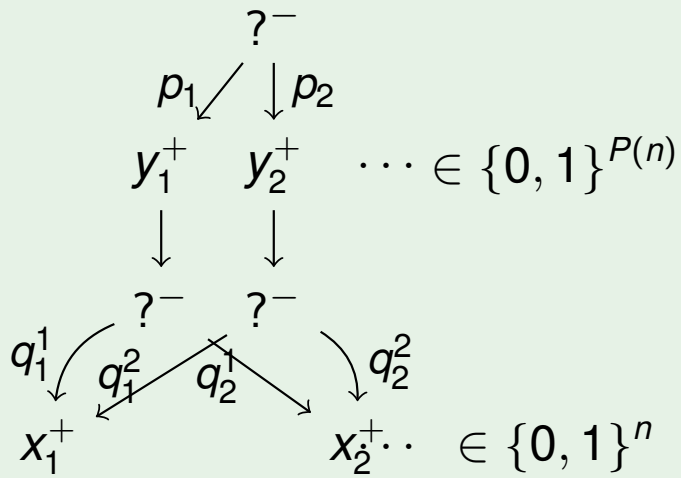
$g : \mathbf{1} \multimap \mathbf{S}[X]$



$f : \mathbf{S}[X] \multimap \mathbf{S}[P] \oslash \mathbf{S}[X]$



$f \circ g : \mathbf{1} \multimap \mathbf{S}[P] \oslash \mathbf{S}[X]$



Adding polytime Constraints: Probabilistic Case (1)

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n\in\mathbb{N}})$
- Probabilistic Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathrm{Odd} \to \Delta(P_G))$

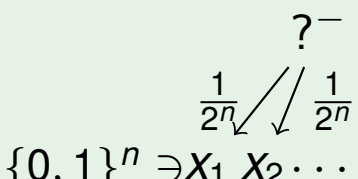
Question:

Are probabilistic polytime computable strategies stable by composition ?

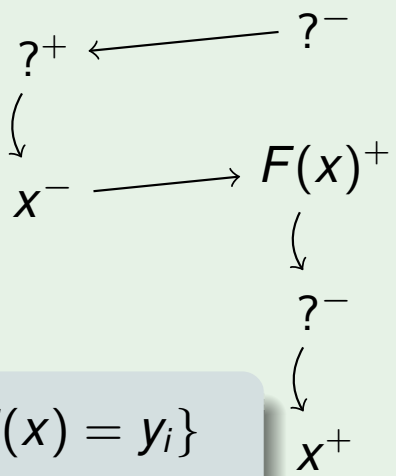
Example

$F : \mathbf{S}[X] \multimap \mathbf{S}[P]$ a one-way function.

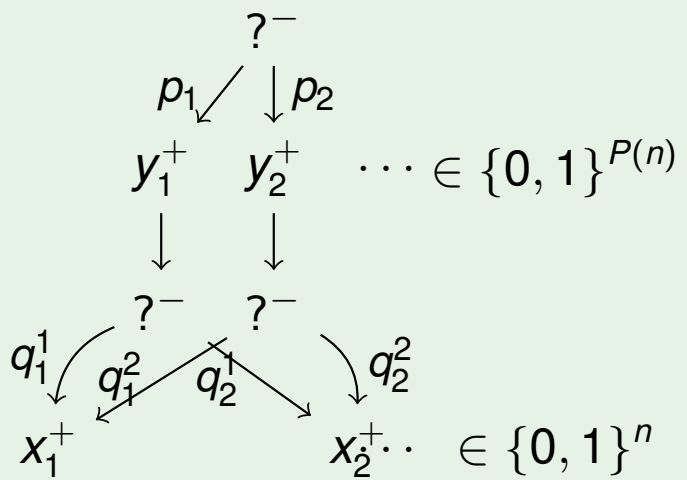
$g : \mathbf{1} \multimap \mathbf{S}[X]$



$f : \mathbf{S}[X] \multimap \mathbf{S}[P] \oslash \mathbf{S}[X]$



$f \circ g : \mathbf{1} \multimap \mathbf{S}[P] \oslash \mathbf{S}[X]$



$$p_i = \frac{1}{2^n} \cdot \#\{x \in \{0,1\}^n \mid F(x) = y_i\}$$

$$q_i^j = \begin{cases} 0 \text{ if } F(x_j) \neq y_i \\ \frac{1}{\#\{x\in\{0,1\}^n \mid F(x)=y_i\}} \text{ otherwise.} \end{cases}$$

Adding polytime Constraints: Probabilistic Case (1)

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n\in\mathbb{N}})$
- Probabilistic Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathrm{Odd} \to \Delta(P_G))$

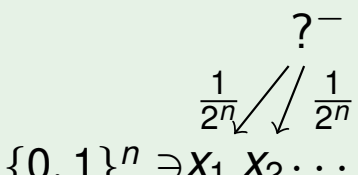
Question:

Are probabilistic polytime computable strategies stable by composition ?

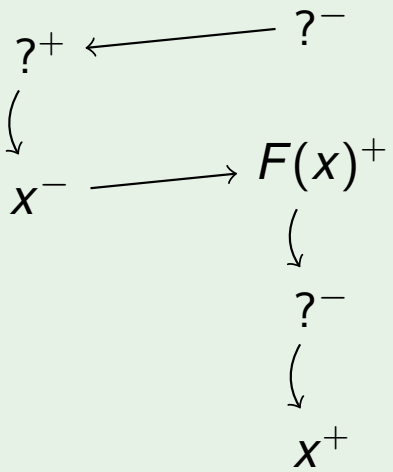
Example

$F : \mathbf{S}[X] \multimap \mathbf{S}[P]$ a one-way function.

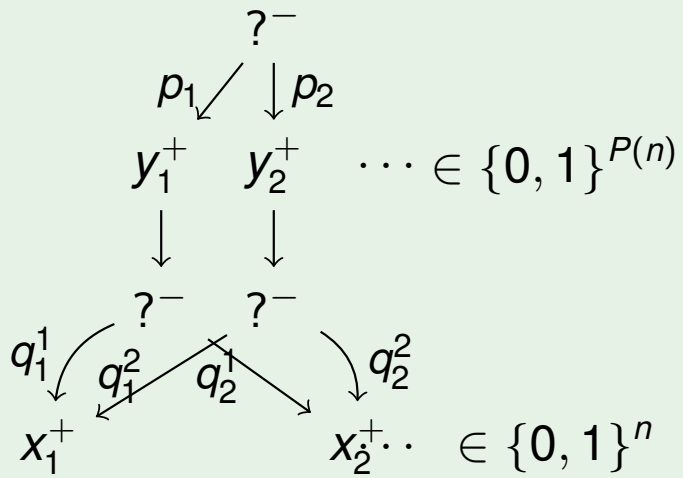
$g : \mathbf{1} \multimap \mathbf{S}[X]$



$f : \mathbf{S}[X] \multimap \mathbf{S}[P] \oslash \mathbf{S}[X]$



$f \circ g : \mathbf{1} \multimap \mathbf{S}[P] \oslash \mathbf{S}[X]$



- f, g are polytime computable functions on bounded games.

Adding polytime Constraints: Probabilistic Case (1)

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n\in\mathbb{N}})$
- Probabilistic Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathrm{Odd} \to \Delta(P_G))$

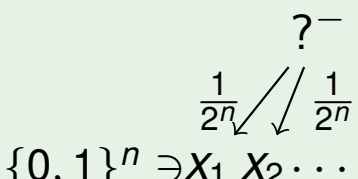
Question:

Are probabilistic polytime computable strategies stable by composition ?

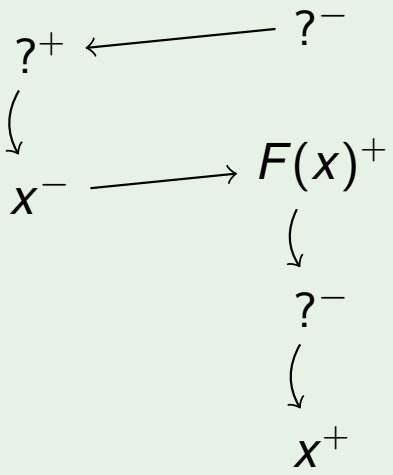
Example

$F : \mathbf{S}[X] \multimap \mathbf{S}[P]$ a one-way function.

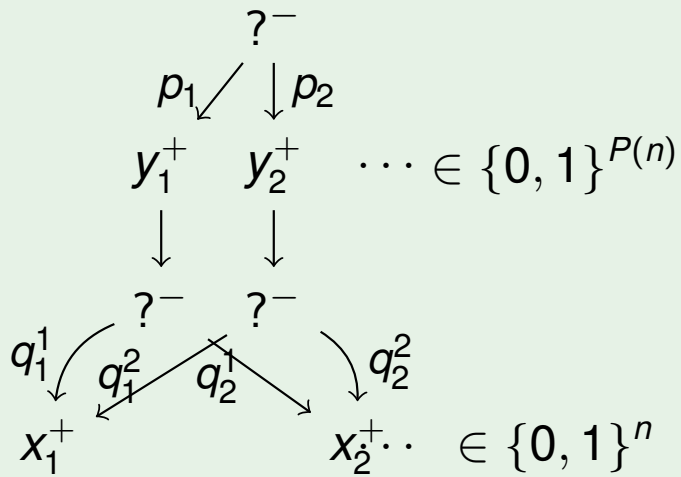
$g : \mathbf{1} \multimap \mathbf{S}[X]$



$f : \mathbf{S}[X] \multimap \mathbf{S}[P] \otimes \mathbf{S}[X]$



$f \circ g : \mathbf{1} \multimap \mathbf{S}[P] \otimes \mathbf{S}[X]$



- f, g are polytime computable functions on bounded games

Compute $f \circ g(?y_1) \Rightarrow$ find an element in $F^{-1}(y_1)$.

Adding polytime Constraints: Probabilistic Case (1)

Game parametrized by the security parameter

- Games: $G = (O_G, P_G, (L_G^n)_{n\in\mathbb{N}})$
- Probabilistic Strategies: $f : \mathbb{N} \to (L_G^n \cap \mathrm{Odd} \to \Delta(P_G))$

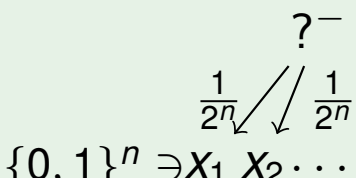
Question:

Are probabilistic polytime computable strategies stable by composition ?

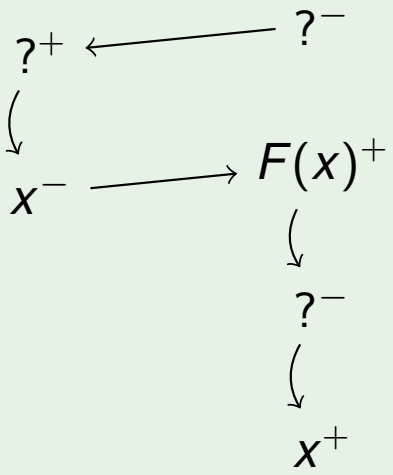
Example

$F : \mathbf{S}[X] \multimap \mathbf{S}[P]$ a one-way function.

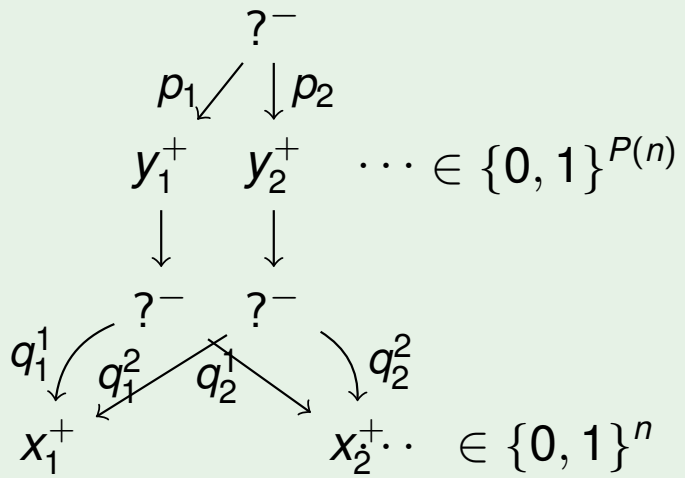
$g : \mathbf{1} \multimap \mathbf{S}[X]$



$f : \mathbf{S}[X] \multimap \mathbf{S}[P] \otimes \mathbf{S}[X]$



$f \circ g : \mathbf{1} \multimap \mathbf{S}[P] \otimes \mathbf{S}[X]$



- f, g are polytime computable functions on bounded games.
- $f \circ g : \mathbf{1} \multimap \mathbf{S}[P] \otimes \mathbf{S}[X]$ is not polytime computable.

Adding Polytime Constraints: Probabilistic Case (2)

Definition (The Category $\mathbf{CPG}^{\oplus}$)

- Objects: parametrized games
- Morphisms $G \to H$: (p, f) :
  - $p \in \mathbf{Pol} \cup \{\infty\}$,
  - f computable (deterministic) strategies on $!_p\mathbf{B} \multimap (G \multimap H)$.

Adding Polytime Constraints: Probabilistic Case (2)

Definition (The Category $\mathbf{CPG}^{\oplus}$)

- Objects: parametrized games
- Morphisms $G \to H$: (p, f) :
  - $p \in \mathbf{Pol} \cup \{\infty\}$,
  - f computable (deterministic) strategies on $!_p\mathbf{B} \multimap (G \multimap H)$.

$\Rightarrow$ Probabilistic choices: explicit call to a probabilistic oracle

Adding Polytime Constraints: Probabilistic Case (2)

Definition (The Category $\mathbf{CPG}^{\oplus}$)

- Objects: parametrized games
- Morphisms $G \to H$: (p, f) :
  - $p \in \mathbf{Pol} \cup \{\infty\}$,
  - f computable (deterministic) strategies on $!_p\mathbf{B} \multimap (G \multimap H)$.

Definition (The sub-category $\mathbf{PolyPG}^{\oplus}$)

- Objects: parametrized **bounded** games
- Morphisms $G \to H$: (p, f) with:
  - $p \in \mathbf{Pol}$
  - f **polytime** computable on $!_p\mathbf{B} \multimap (G \multimap H)$.

$\Rightarrow$ Probabilistic choices: explicit call to a probabilistic oracle

Adding Polytime Constraints: Probabilistic Case (2)

Definition (The Category $\mathbf{CPG}^{\oplus}$)

- Objects: parametrized games
- Morphisms $G \to H$: (p, f) :
  - $p \in \mathbf{Pol} \cup \{\infty\}$,
  - f computable (deterministic) strategies on $!_p\mathbf{B} \multimap (G \multimap H)$.

Definition (The sub-category $\mathbf{PolyPG}^{\oplus}$)

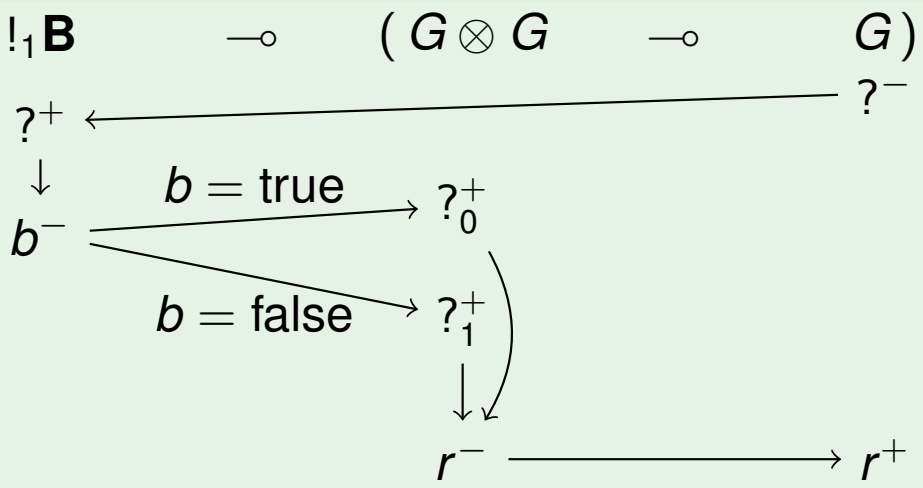
- Objects: parametrized **bounded** games
- Morphisms $G \to H$: (p, f) with:
  - $p \in \mathbf{Pol}$
  - f **polytime** computable on $!_p\mathbf{B} \multimap (G \multimap H)$.

$\Rightarrow$ Probabilistic choices: explicit call to a probabilistic oracle

Example (Fair choice between two arguments)

$s \in \mathbf{CPG}^{\oplus}(G \otimes G, G)$
$s = (1, f)$ with:

f :

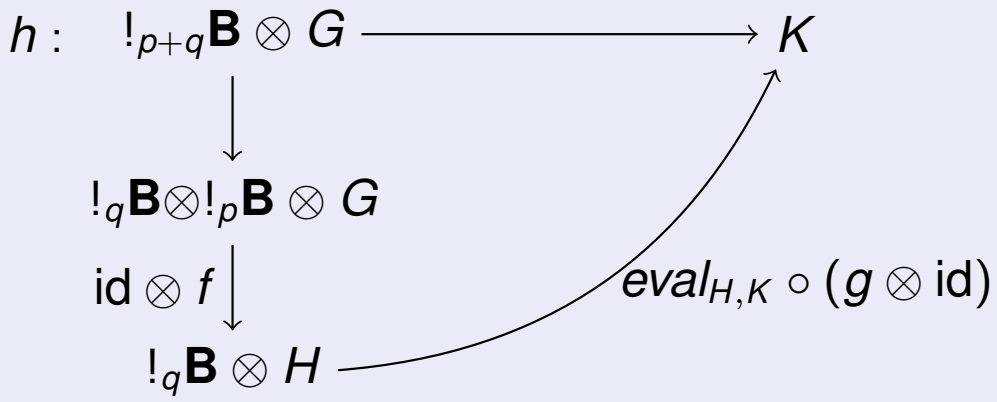


Categorical structure of $\mathbf{CPG}^{\oplus}$

Composition in $\mathbf{CPG}^{\oplus}$.

$s = (p, f) \in \mathbf{CPG}^{\oplus}(G, H)$, and $t = (q, g) \in \mathbf{CPG}^{\oplus}(H, K)$. Then

$$t \circ s = (p + q, \mathrm{curr}_{\mathbf{PG}}(h))$$

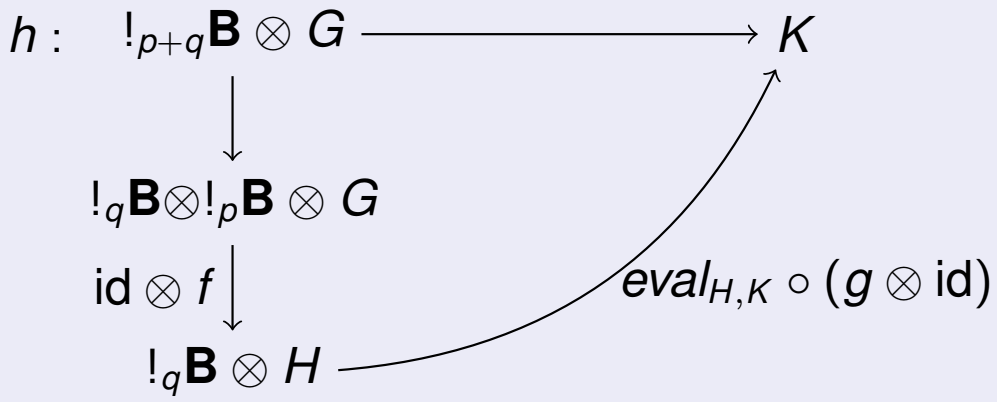


Categorical structure of $\mathbf{CPG}^{\oplus}$

Composition in $\mathbf{CPG}^{\oplus}$.

$s = (p, f) \in \mathbf{CPG}^{\oplus}(G, H)$, and $t = (q, g) \in \mathbf{CPG}^{\oplus}(H, K)$. Then

$$t \circ s = (p + q, \mathrm{curr}_{\mathbf{PG}}(h))$$



Theorem

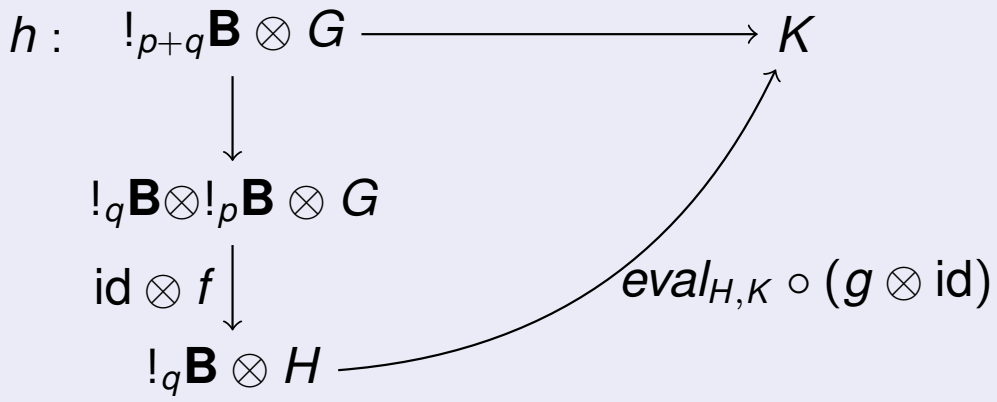
The category $\mathbf{CPG}^{\oplus}$ *is a linear category:* $\otimes, \multimap, !, \ldots$.

Categorical structure of $\mathbf{CPG}^{\oplus}$

Composition in $\mathbf{CPG}^{\oplus}$.

$s = (p, f) \in \mathbf{CPG}^{\oplus}(G, H)$, and $t = (q, g) \in \mathbf{CPG}^{\oplus}(H, K)$. Then

$$t \circ s = (p + q, \mathrm{curr}_{\mathbf{PG}}(h))$$



Theorem

The category $\mathbf{CPG}^{\oplus}$ *is a linear category:* $\otimes, \multimap, !, \ldots$.

Theorem

$(\mathbf{PolyPG}^{\oplus}, (!_p)_{p \in \mathbf{Pol}})$ *is a Bounded Exponential Situation (in Breuvart-Pagani sense).*

Computationnal Distance in $\mathbf{CPG}^{\oplus}$.

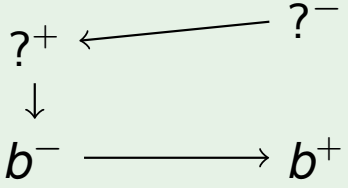
Definition (Observable probability)

If $f : \mathbf{1} \to \mathbf{B}$ is a morphism in $\mathbf{CPG}^{\oplus}$. Then for b a boolean:

$$\mathrm{Prob}(f)(b) : n \in \mathbb{N} \mapsto \sum_{m} \sum_{\substack{a=b_0 b_1 \dots b_m \\ f(n)(a)=b}} \frac{1}{2^m} \qquad \in [0,1]$$

Example (Fair probabilistic choice on booleans)

$f :$ $!_1\mathbf{B} \multimap \mathbf{B}$



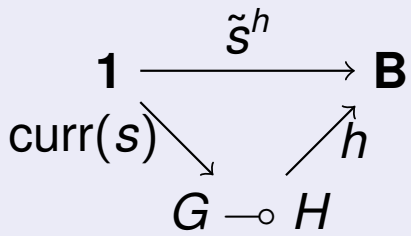
We take $s = (1, f) \in \mathbf{CPG}^{\oplus}(\mathbf{1}, \mathbf{B})$.

$$\mathrm{Prob}(s)(\mathrm{true})(n) = \frac{1}{2} \qquad \mathrm{Prob}(s)(\mathrm{false})(n) = \frac{1}{2}$$

Definition (Separation induced by a morphisms in $\mathbf{PolyPG}^{\oplus}$)

s, t morphisms in $\mathbf{CPG}^{\oplus}(G, H)$. For any $h \in \mathbf{PolyPG}^{\oplus}((G \multimap H), \mathbf{B})$

$$\delta^h(s,t) = n \mapsto |\mathrm{Prob}(\tilde{f}^h)(b) - \mathrm{Prob}(\tilde{g}^h)(b)|.$$



Computationnal indistinguishability

Definition (Equivalence to a negligible factor)

$a, b : \mathbb{N} \to [0,1]$. $a \equiv b$ if $|a - b|$ is a negligible function, i.e.:

$$\forall p \in \mathbf{Pol}, \exists N \in \mathbb{N}, \forall n \geq N, |a(n) - b(n)| \leq \frac{1}{p(n)}$$

It is an equivalence relation.

Definition (Computationnal Indistinguishability)

$s, t \in \mathbf{CPG}^{\oplus}(G, H)$. Then $s \approx t$ if

$$\forall h \in \mathbf{PolyPG}^{\oplus}(G \multimap H, \mathbf{B}),\ \delta^h(s,t) \equiv 0.$$

Example

$f :$ $!_X\mathbf{B}$ $\multimap$ $\mathbf{B}$ — $?^+$ receives $?^-$; then b_1^-, …, $?^+$, b_n^-; $\forall i, b_i = \text{true} \to \text{false}^+$; $\exists j, b_j = \text{false} \to \text{true}^+$

$g :$ $!_0\mathbf{B}$ $\multimap$ $\mathbf{B}$ — $?^-$ then true

(X, f) and $(0, g)$ are in $\mathbf{CPG}^{\oplus}(\mathbf{1}, \mathbf{B})$.

$$(X, f) \approx (0, g).$$

Crypto-Situation

Definition (Crypto-Situation)

$\mathscr{C} = (\text{SCHEME}, \text{ADV}, e)$:

- SCHEME, ADV: games;
- $e = (p, q, s^{\mathscr{C}})$, with $p, q \in \mathbf{Pol}$, and $s^{\mathscr{C}} \in \mathbf{CPG}^{\oplus}(!_p\text{SCHEME} \otimes !_q\text{ADV}, \mathbf{B})$.

Example (Pseudo Random Generator)

$\text{SCHEME}^{PRG} = \mathbf{S}[X] \multimap \mathbf{S}[p]$, $\text{ADV}^{\text{PRG}} = \mathbf{S}[p] \multimap \mathbf{B}$, $p = q = 1$.
Definition of $s^{\mathscr{C}}$:

Game $0(F)$:

Adv G_0

$k \leftarrow \{0,1\}^n$

$x := F(k)$

$b \downarrow \longleftarrow x$

Game $1(F)$:

Adv G_1

$k \leftarrow \{0,1\}^{p(n)}$

$x := k$

$b \downarrow \longleftarrow x$

$$s_{G_0} : !_1 S \otimes !_1 \text{ADV} \xrightarrow{\text{rand}^p \otimes id} \mathbf{S}[X] \otimes S \otimes \text{ADV} \downarrow \mathbf{B}$$

$$s_{G_1} : !_1 S \otimes !_1 \text{ADV} \to \mathbf{1} \otimes \text{ADV} \xrightarrow{\text{rand}^p \otimes id} \mathbf{S}[p] \otimes \text{ADV} \downarrow \mathbf{B}$$

Security for a Crypto Situation

Definition (Advantage for an adversary $\mathscr{A}$ against a scheme $\mathscr{S}$)

$\mathscr{S} : \mathbf{1} \to \mathrm{SCHEME}$, $\mathscr{A} : \mathbf{1} \to \mathrm{ADV}$.

$$\mathrm{Advantage}(\mathscr{A} \,||_{\mathscr{S}}\, \mathscr{C}) = \sup_{b \text{ boolean}} |\mathrm{Prob}(\mathrm{interact}^{\mathscr{C},\mathscr{S},\mathscr{A}})(b) - \frac{1}{2}|$$

where:

$$\mathrm{interact}^{\mathscr{C},\mathscr{S},\mathscr{A}} : \quad \mathbf{1} \longrightarrow !_p\mathbf{1} \otimes !_q\mathbf{1} \xrightarrow{!_p\mathscr{S} \otimes !_q\mathscr{A}} !_p\mathrm{SCHEME} \otimes !_q\mathrm{ADV} \xrightarrow{s^{\mathscr{C}}} \mathbf{B}$$

Definition (Security for $\mathscr{S}$ in the crypto-Situation $\mathscr{C}$)

For any $\mathscr{A} \in \mathbf{PolyPG}^{\oplus}(\mathbf{1}, \mathrm{ADV})$, the function $\mathrm{Advantage}(\mathscr{A} \,||_{\mathscr{C}}\, \mathscr{S})$ is a negligible function of n.

Lemma (Security seen using CI)

$\mathscr{S}$ is secure w.r.t. $\mathscr{C}$ if for every $\mathscr{A}$,

$$\mathit{interact}^{\mathscr{C},\mathscr{S},\mathscr{A}} \approx \mathit{choice}$$

Security of a scheme $\Leftrightarrow$

The adversary cannot do better that a random guess.

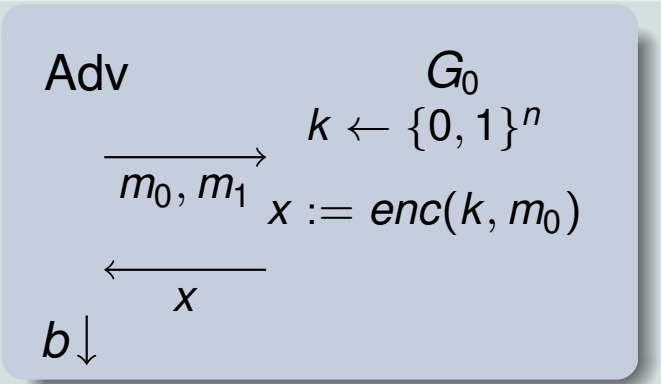
From a PRG to a Encryption Scheme for Passive Adversaries

Example (*EAV*: security situation for Encryption Scheme against P.A.)

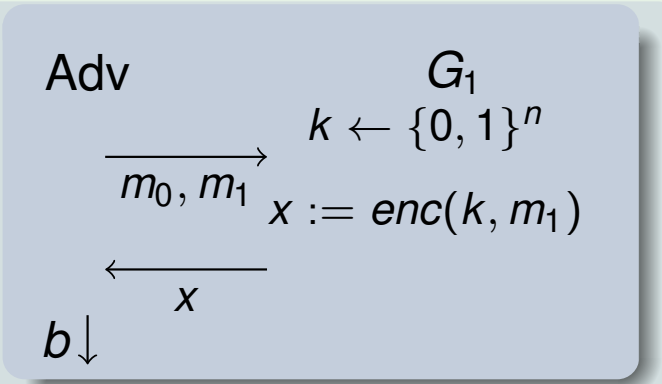
$\mathrm{SCHEME}^{\mathrm{encr}} = !_\infty \mathrm{GEN}^p \otimes !_\infty \mathrm{ENC}^p \otimes !_\infty \mathrm{DEC}^p$ where:

$$\mathrm{GEN}^p = \mathbf{S}[X]$$
$$\mathrm{ENC}^p = (\mathbf{S}[X] \otimes \mathbf{S}[p]) \multimap \mathbf{S}[p]$$
$$\mathrm{DEC}^p = (\mathbf{S}[X] \otimes \mathbf{S}[p]) \multimap \mathbf{1} \oplus \mathbf{S}[p]$$

Game 0 :



Game 1 :



Definition

From PRG to EAV $\mathscr{S}$ a PRG-scheme. $E[\mathscr{S}]$ is the EAV scheme defined by:

$$\mathrm{GEN}^p() = \mathrm{rand}$$
$$\mathrm{ENC}^p(k, m) = \mathscr{S}(k) \text{ xor } m$$
$$\mathrm{DEC}^p(k, m) = \mathscr{S}(k) \text{ xor } m$$

Security Proof: informally

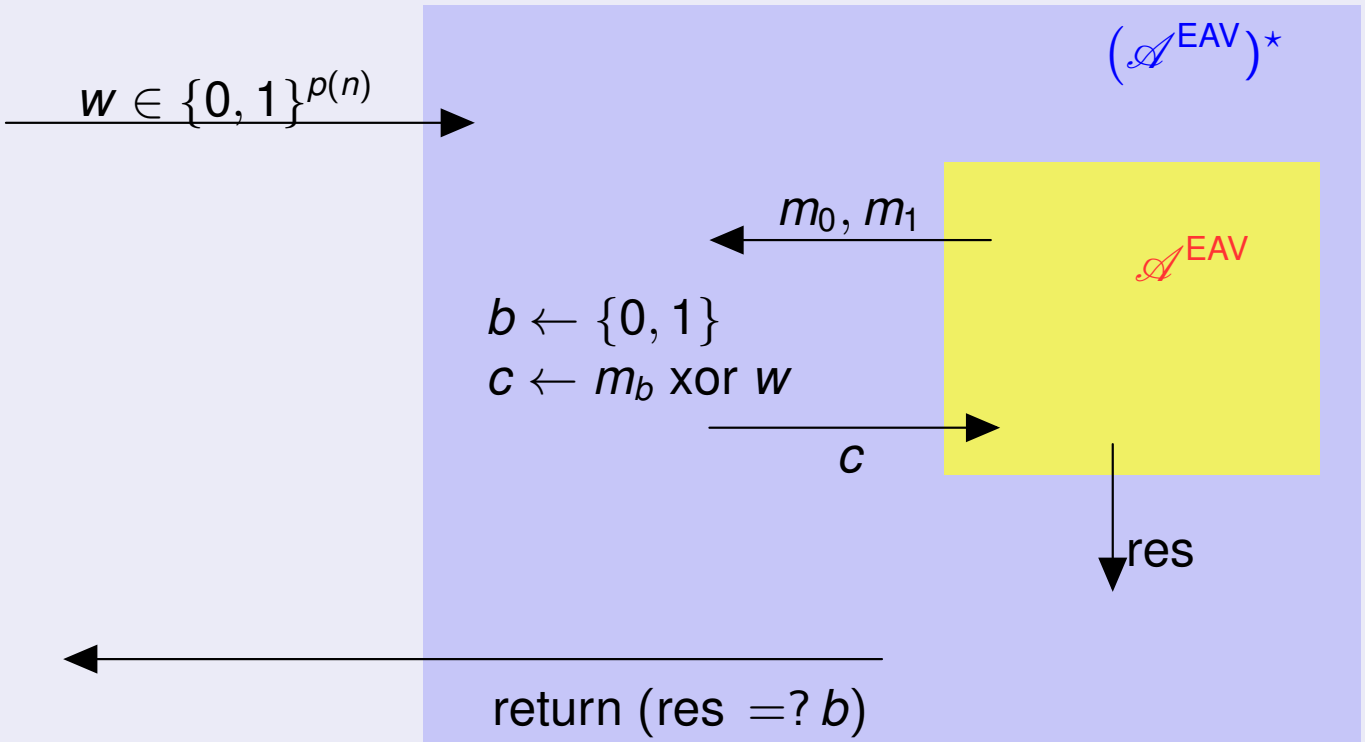
Goal

Given $\mathscr{A}^{\mathsf{EAV}}$ against EAV, construct $(\mathscr{A}^{\mathsf{EAV}})^\star$ against PRG, with:

$$\mathsf{Advantage}(\mathscr{A}^{\mathsf{EAV}} ||_{\mathsf{EAV}} E[\mathscr{S}]) \text{ not negligible}$$

$$\Rightarrow \mathsf{Advantage}((\mathscr{A}^{\mathsf{EAV}})^\star ||_{\mathsf{PRG}} \mathscr{S}) \text{ non negligible.}$$

Construction of a PRG-adversary $(\mathscr{A}^{\mathsf{EAV}})^\star$ from a EAV-adversary $\mathscr{A}^{\mathsf{EAV}}$.



Security Proof in $\mathbf{CPG}^{\oplus}$

Lemma

$$\mathit{interact}_{PRG}(\mathscr{S}, (\mathscr{A}^{EAV})^{\star}) = \bigoplus \left[\mathit{not} \circ \mathit{interact}_{PRG0}(\mathscr{S}, (\mathscr{A}^{EAV})^{\star}), \mathit{interact}_{PRG1}(\mathscr{S}, (\mathscr{A}^{EAV})^{\star}) \right]$$

Lemma

$$\mathit{interact}_{PRG0}(\mathscr{S}, (\mathscr{A}^{EAV})^{\star}) = \mathit{interact}_{EAV}(\mathit{OTP}, \mathscr{A}^{EAV})$$
$$\mathit{interact}_{PRG1}(\mathscr{S}, (\mathscr{A}^{EAV})^{\star}) = \mathit{interact}_{EAV}(E[\mathscr{S}], \mathscr{A}^{EAV})$$

Proposition

$$\begin{cases} \mathit{interact}_{EAV}(\mathit{OTP}, \mathscr{A}^{EAV}) \approx \mathit{choice} \\ \mathit{interact}_{PRG}(\mathscr{S}, (\mathscr{A}^{EAV})^{\star}) \approx \mathit{choice} \end{cases} \Rightarrow \mathit{interact}_{EAV}(E[\mathscr{S}], \mathscr{A}^{EAV}) \approx \mathit{choice}$$

Theorem

If $\mathscr{S}$ is secure for PRG, then $E[\mathscr{S}]$ is EAV secure.

Conclusion

Related Work

- Hugo Ferée: Higher-order complexity in game semantics
- Canetti: Universal Compositionnality.
  A notion of security for protocol preserved by compositionnality
  Security and Composition of Multi-Party Cryptographics Protocolls (1999).

Future Works

- formalize more well-known security results
- try to define sounds higher-order crypto situations
- Computationnal indistiguishability:
  - See CI as a congruence on a suitable λ-calculus.
  - Formalize a notion of distance on strategies/morphisms corresponding to some kind of context distance, with $\approx$ as kernel.