MIT 6.875/6.5620/18.425

Foundations of Cryptography
Lecture 3

Course website: ***https://mit6875.github.io/***

Lecture 2 Recap

- *Computational Indistinguishability*:
  a new definition of security for secret-key encryption.

  (new notions: p.p.t. adversaries, negligible functions,…)

- *Consequence*: Shannon's impossibility no longer applies!

- *New Notion*: Pseudorandom Generator (PRG)

- *PRG ⇒ Can encrypt* ***a single message*** *longer than the key.*

- *We saw a construction of PRG (based on subset sum). Many more later in the course.*

TODAY

How to encrypt (poly) many messages with a fixed key?

1. PRG length extension.

Theorem: If there is a PRG that stretches by one bit, there is one that stretches by poly many bits

Consequence: *Stateful* encryption of poly many messages.

2. Another new notion: Pseudorandom Functions (PRF).

Consequence: *Stateless* encryption of poly many messages.

Theorem (next lec): If there is a PRG, then there is a PRF.

New Proof Technique: Hybrid Arguments.

But first, let’s do some prep work…

Three Definitions of Pseudorandomness

Def 1 [Indistinguishability]

"No polynomial-time algorithm can distinguish between the output of a PRG on a random seed vs. a truly random string"

= "as good as" a truly random string for all practic… …oses.

ALL DEFS ARE EQUIVALENT!

Def 2 [Next-bit Unpredictability]

"No polynomial-time algorithm ca… …ct the (i+1)th bit of the output of a PRG given the first … …etter than chance"

Def 3 [Incompressi…]

"No polynomial-time algorithm can compress the output of the PRG into a shorter string"

PRG Def 1 (Recap): Indistinguishability

Definition [Indistinguishability]:

A deterministic polynomial-time computable function G: $\{0,1\}^n \rightarrow \{0,1\}^m$ is indistinguishable (or, secure against any statistical test) if:

for every PPT algorithm D (called a distinguisher) if there is a negligible function $\boldsymbol{\mu}$ *such that:*

$$|\Pr[\,\boldsymbol{D}(\boldsymbol{G}(\boldsymbol{U_n})) = \mathbf{1}\,] - \Pr[\,\boldsymbol{D}(\boldsymbol{U_m}) = \mathbf{1}\,]| = \boldsymbol{\mu}(\boldsymbol{n})$$

Notation: U_n (resp. U_m) denotes the random distribution on n-bit (resp. m-bit) strings.

PRG Def 2: Next-bit Unpredictability

Definition [Next-bit Unpredictability]:

A deterministic polynomial-time computable function G: $\{0,1\}^n \rightarrow \{0,1\}^m$ is next-bit unpredictable if:

for every PPT algorithm P (called a next-bit predictor) and every $i \in \{1, \ldots, m\}$, *if there is a negligible function* $\boldsymbol{\mu}$ *such that:*

$$\mathbf{Pr}[\, \boldsymbol{y} \leftarrow \boldsymbol{G}(\boldsymbol{U_n}) \colon \boldsymbol{P}(\boldsymbol{y_1 y_2 \ldots y_{i-1}}) = \boldsymbol{y_i}] = \frac{\mathbf{1}}{\mathbf{2}} + \boldsymbol{\mu}(\boldsymbol{n})$$

Notation: $\boldsymbol{y_1}, \boldsymbol{y_2}, \ldots \boldsymbol{y_m}$ are the bits of the m-bit string $\boldsymbol{y}$.

Def 1 and Def 2 are Equivalent

Theorem:

A PRG G is indistinguishable if and only if it is next-bit unpredictable.

Def 1 and Def 2 are Equivalent

Theorem:

A PRG G passes all (poly-time) statistical tests if and only if it passes (poly-time) next-bit tests.

NBU and Indistinguishability

- Next-bit Unpredictability (NBU): Seemingly much weaker requirement. Only says that next bit predictors, a particular type of distinguishers, cannot succeed.

- Yet, surprisingly, Next-bit Unpredictability (NBU) = Indistinguishability.

- NBU often much easier to use.

1. Indistinguishability $\Longrightarrow$ NBU

Proof: by contradiction.

Suppose for contradiction that there is a p.p.t. predictor P, a polynomial function p and an $i \in \{1, \dots, m\}$ *s.t.*

$$\Pr[\, y \leftarrow G(U_n) \colon P(y_1 y_2 \dots y_{i-1}) = y_i] \geq \frac{1}{2} + 1/p(n)$$

Then, I claim that P essentially gives us a distinguisher D!

Consider D which gets an m-bit string y and does the following:

1. Run P on the $(i-1)$-bit prefix $y_1 y_2 \dots y_{i-1}$.
2. If P returns the i-th bit y_i, then output 1 (“PRG”) else output 0 (“Random”).

If P is p.p.t. so is D.

1. Indistinguishability $\Longrightarrow$ NBU

Consider D which gets an m-bit string y and does the following:

1. Run P on the $(i-1)$-bit prefix $y_1 y_2 \dots y_{i-1}$.
2. If P returns the i-th bit y_i, then output 1 (= "PRG") else output 0 (= "Random").

We want to show: there is a polynomial p' s.t.

$$|\Pr[y \leftarrow G(U_n){:}\ D(y) = 1] - \Pr[y \leftarrow U_m{:}\ D(y) = 1]| \geq 1/p'(n)$$

1. Indistinguishability $\Longrightarrow$ NBU

Consider D which gets an m-bit string y and does the following:

1. Run P on the $(i-1)$-bit prefix $y_1 y_2 \dots y_{i-1}$.

2. If P returns the i-th bit y_i, then output 1 (= "PRG") else output 0 (= "Random").

$$\Pr[y \leftarrow G(U_n):\ D(y)\ = 1\ ]$$

$$= \Pr[y \leftarrow G(U_n):\ P(y_1 y_2 \dots y_{i-1}) = y_i]$$

(by construction of D)

$$\geq \frac{1}{2} + 1/p(n)$$ (by assumption on P)

1. Indistinguishability $\Longrightarrow$ NBU

Consider D which gets an m-bit string y and does the following:

1. Run P on the $(i-1)$-bit prefix $y_1 y_2 \ldots y_{i-1}$.
2. If P returns the i-th bit y_i, then output 1 (= “PRG”) else output 0 (= “Random”).

$$\Pr[y \leftarrow G(U_n){:}\ D(y) = 1] \geq \frac{1}{2} + 1/p(n)$$

$$\Pr[y \leftarrow U_m{:}\ D(y) = 1]$$

$$= \Pr[y \leftarrow U_m{:}\ P(y_1 y_2 \ldots y_{i-1}) = y_i]$$ (by construction of D)

$$= \frac{1}{2}$$ (since y is random)

1. Indistinguishability $\Longrightarrow$ NBU

Consider D which gets an m-bit string y and does the following:

1. Run P on the $(i-1)$-bit prefix $y_1 y_2 \ldots y_{i-1}$.

2. If P returns the i-th bit y_i, then output 1 (= "PRG") else output 0 (= "Random").

$$\Pr[y \leftarrow G(U_n):\ D(y) = 1] \geq \frac{1}{2} + 1/p(n)$$

$$\Pr[y \leftarrow U_m:\ D(y) = 1] = \frac{1}{2}$$

So, $|\Pr[y \leftarrow G(U_n):\ D(y) = 1] - \Pr[y \leftarrow U_m:\ D(y) = 1]| \geq 1/p(n)$ $\blacksquare$

2. NBU $\Longrightarrow$ Indistinguishability

Proof: by contradiction (again!)

Suppose for contradiction that there is a distinguisher D, and a polynomial function p *s.t.*

$$| \Pr[y \leftarrow G(U_n)\colon\ D(y) \ = 1\ ] \\ - \Pr[y \leftarrow U_m\colon\ D(y) \ = 1\ ]\ | \geq 1/p'(n)$$

I want to construct a next bit predictor P out of D.

But how?!

2. NBU $\Longrightarrow$ Indistinguishability

Proof: **by contradiction (again!)**

Suppose for contradiction that there is a distinguisher D, and a polynomial function p *s.t.*

$$\Pr[y \leftarrow G(U_n)\colon\ D(y) \ = 1\,] \\ -\ \Pr[y \leftarrow U_m\colon\ D(y) \ = 1\,] \geq 1/p'(n) \ \ := \varepsilon$$

I want to construct a next bit predictor P out of D.

TWO STEPS:

- **STEP 1:** HYBRID ARGUMENT
- **STEP 2:** From Distinguishing to Predicting

Before we go there, a puzzle…

Lemma: Let $p_0, p_1, p_2, \ldots, p_m$ be real numbers s.t.

$$\boldsymbol{p_m - p_0 \geq \varepsilon}.$$

Then, there is an index i such that $\boldsymbol{p_i - p_{i-1} \geq \varepsilon/\mathbf{m}}$.

Proof:

$$p_m - p_0 = (p_m - p_{m-1}) + (p_{m-1} - p_{m-2}) + \cdots + (p_1 - p_0)$$

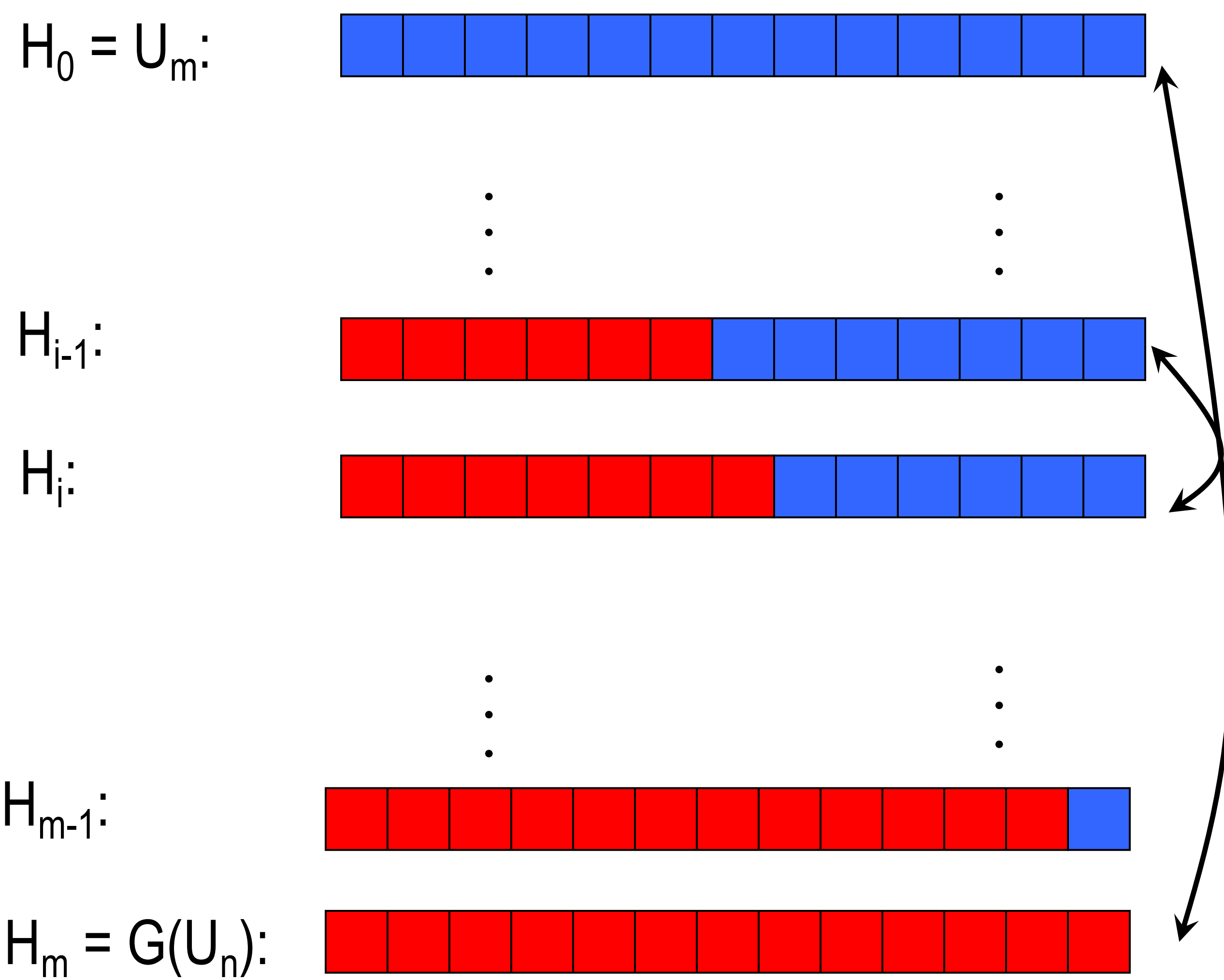
$$\geq \varepsilon$$

At least one of the m terms has to be at least ε/m (averaging). ∎

Define Hybrid Distributions:

random

pseudorandom



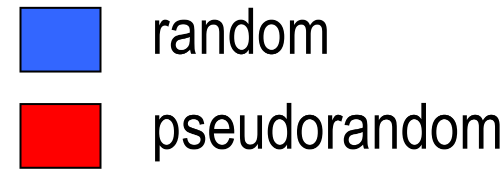
$\exists i$ such that D distinguishes between H_{i-1} and H_i with advantage $\geq \varepsilon / m$

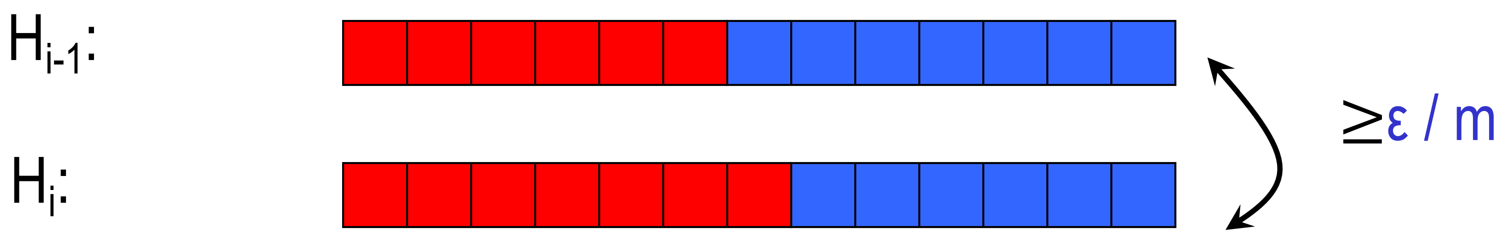
$$\Pr[D(H_i) = 1] - \Pr[D(H_{i-1}) = 1] \geq \varepsilon / m$$

D distinguishes between H_m and H_0 with advantage ε

$$\Pr[D(H_m) = 1] - \Pr[D(H_0) = 1] \geq \varepsilon$$

Hybrid Distributions:





- Let's define $p_i = \Pr[D(H_i) = 1]$.

  $p_0 = \Pr[D(U_m) = 1]$ and $p_m = \Pr[D(G(U_n)) = 1]$

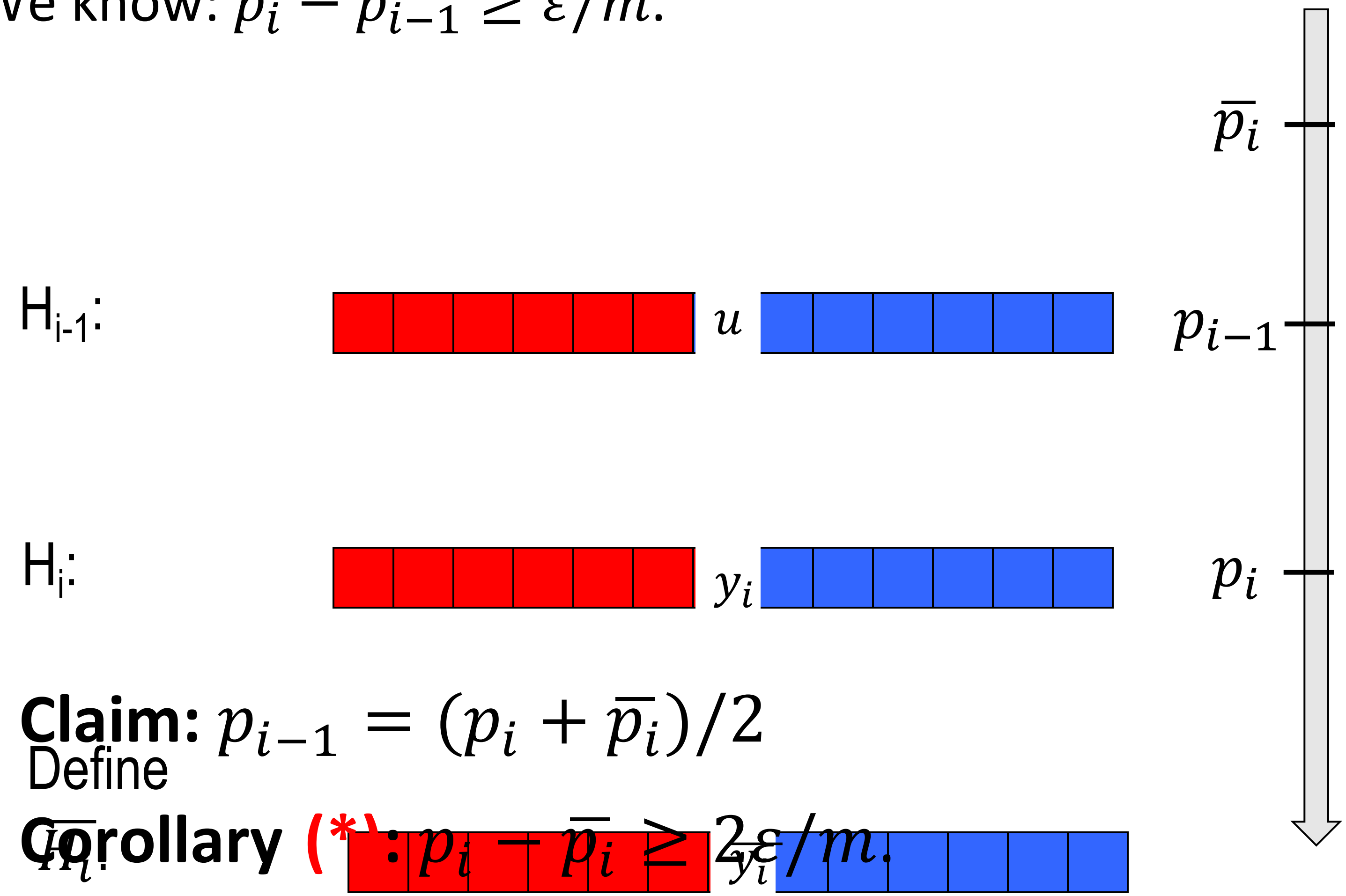
- By the **hybrid argument**, we have: $p_i - p_{i-1} \geq \varepsilon / m$.
- **Key Intuition**: D outputs 1 more often given a pseudorandom i-th bit than a random i-th bit.
- So, D gives us a "signal" as to whether a given bit is the correct i-th bit or not.

Let’s dig a bit more.

random

pseudorandom

We know: $p_i - p_{i-1} \geq \varepsilon/m$.



Claim: $p_{i-1} = (p_i + \overline{p_i})/2$

Define

Corollary **(*)**: $p_i - \overline{p_i} \geq 2\varepsilon/m$.

$\overline{p_i} = \Pr[D(\overline{H_i}) = 1]$

So, Takeaway: D says “1” more often when fed with the “right bit” than the “wrong bit”.

u: random bit

y_i: i-th pseudorandom bit

$\overline{y_i} = 1 - y_i$

Our Predictor P

The Idea: The predictor is given the first $i-1$ pseudorandom bits (call it $y_1 y_2 \dots y_{i-1}$) and needs to guess the i-th bit.

The Predictor P works as follows:

Pick a random bit b;

Feed D with input $y_1 y_2 \dots y_{i-1} | \mathrm{b} | u_{i+1} \dots u_m$ (u's are random)

If D says "1", output b as the prediction for y_i and if D says "0", output $\bar{b}$ as the prediction for y_i

Analysis of the Predictor P

$$\Pr[x \leftarrow \{0,1\}^n; y = G(x): P(y_1 y_2 \dots y_{i-1}) = y_i]$$

$$= \Pr[D(y_1 y_2 \dots y_{i-1} b \dots) = 1 \mid b = y_i] \Pr[b = y_i] + \Pr[D(y_1 y_2 \dots y_{i-1} b \dots) = 0 \mid b \neq y_i] \Pr[b \neq y_i]$$

$$= \frac{1}{2}(\Pr[D(y_1 y_2 \dots y_{i-1} b \dots) = 1 \mid b = y_i] + \Pr[D(y_1 y_2 \dots y_{i-1} b \dots) = 0 \mid b \neq y_i])$$

$$= \frac{1}{2}(\Pr[D(y_1 y_2 \dots y_{i-1} y_i \dots) = 1] + \Pr[D(y_1 y_2 \dots y_{i-1} \overline{y_i} \dots) = 0])$$

$$= \frac{1}{2}(\Pr[D(y_1 y_2 \dots y_{i-1} y_i \dots) = 1] + 1 - \Pr[D(y_1 y_2 \dots y_{i-1} \overline{y_i} \dots) = 1])$$

$$= \frac{\mathbf{1}}{\mathbf{2}}(\mathbf{1} + (*)) \quad \geq \frac{\mathbf{1}}{\mathbf{2}} + \mathbf{1}/(\boldsymbol{m} \cdot \boldsymbol{p}(\boldsymbol{n}))$$

$\blacksquare$

Recap: NBU and Indistinguishability

- Next-bit Unpredictability (NBU): Seemingly much weaker requirement, only says that next bit predictors, a particular type of distinguishers, cannot succeed.

- Yet, surprisingly, Next-bit Unpredictability (NBU) = Indistinguishability.

- NBU often much easier to use.

- *Exercise*: Previous-bit Unpredictability (NBU) = Indistinguishability.

TODAY

How to encrypt (poly) many messages with a fixed key?

1. PRG length extension.

Theorem: If there is a PRG that stretches by one bit, there is one that stretches by poly many bits

Consequence: *Stateful* encryption of poly many messages.

2. Another new notion: Pseudorandom Functions (PRF).

Consequence: *Stateless* encryption of poly many messages.

Theorem (next lec): If there is a PRG, then there is a PRF.

New Proof Technique: Hybrid Arguments.

Length extension: One bit to Many bits

Let G: $\{0,1\}^n \rightarrow \{0,1\}^{n+1}$ be a pseudorandom generator.

Goal: use G to generate **many** pseudorandom bits.

Length extension: One bit to Many bits

Let G: $\{0,1\}^n \rightarrow \{0,1\}^{n+1}$ be a pseudorandom generator.

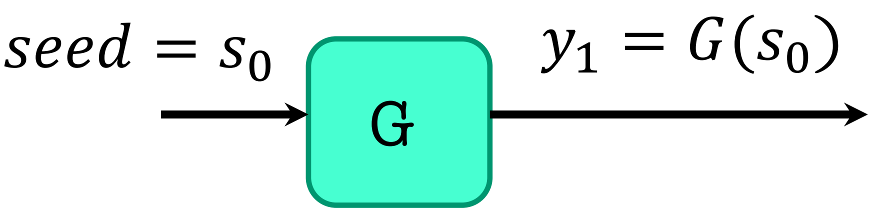
Goal: use G to generate **poly many** pseudorandom bits.

Length extension: One bit to Many bits

Let G: $\{0,1\}^n \rightarrow \{0,1\}^{n+1}$ be a pseudorandom generator.

Goal: use G to generate **poly many** pseudorandom bits.

<u>Construction of G'(s_0)</u>

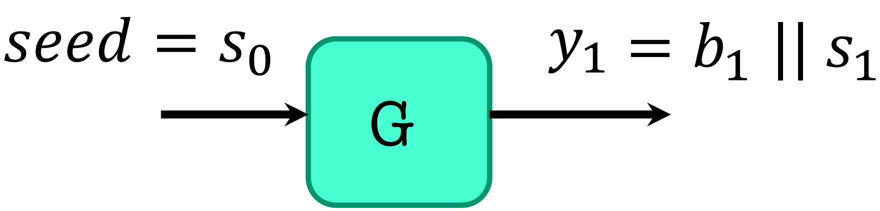


Length extension: One bit to Many bits

Let G: $\{0,1\}^n \to \{0,1\}^{n+1}$ be a pseudorandom generator.

Goal: use G to generate **poly many** pseudorandom bits.

<u>Construction of G'(s_0)</u>

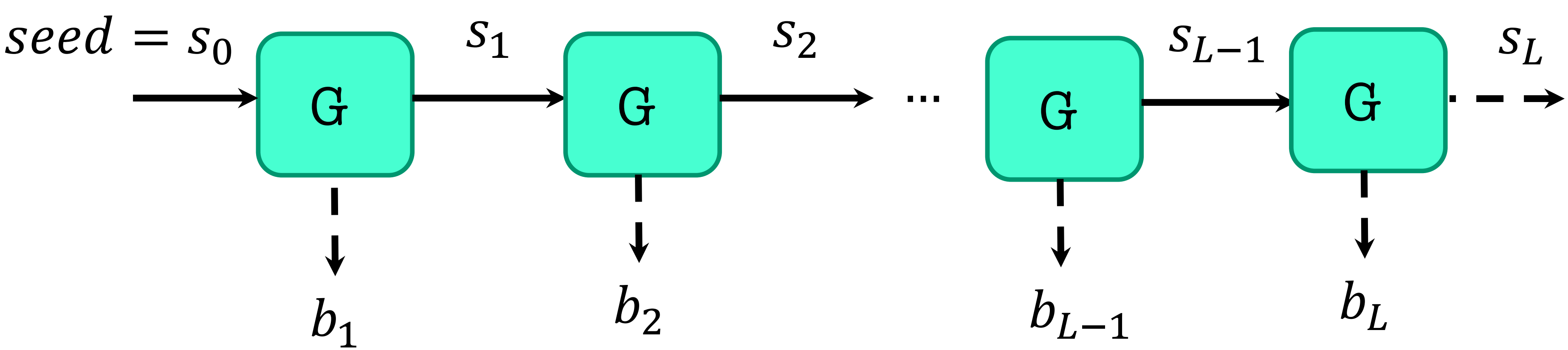


Length extension: One bit to Many bits

Let G: $\{0,1\}^n \rightarrow \{0,1\}^{n+1}$ be a pseudorandom generator.

Goal: use G to generate **poly many** pseudorandom bits.

Construction of G'(s_0) Output $b_1\ b_2\ b_3\ b_4\ b_5\ \ldots s_L$.



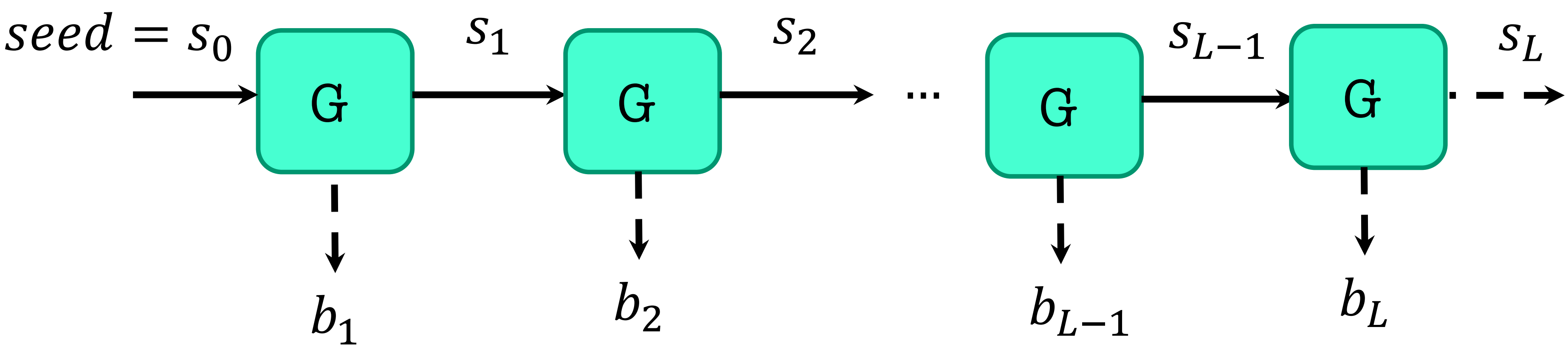
Also called a stream cipher by the practitioners.

Length extension: One bit to Many bits

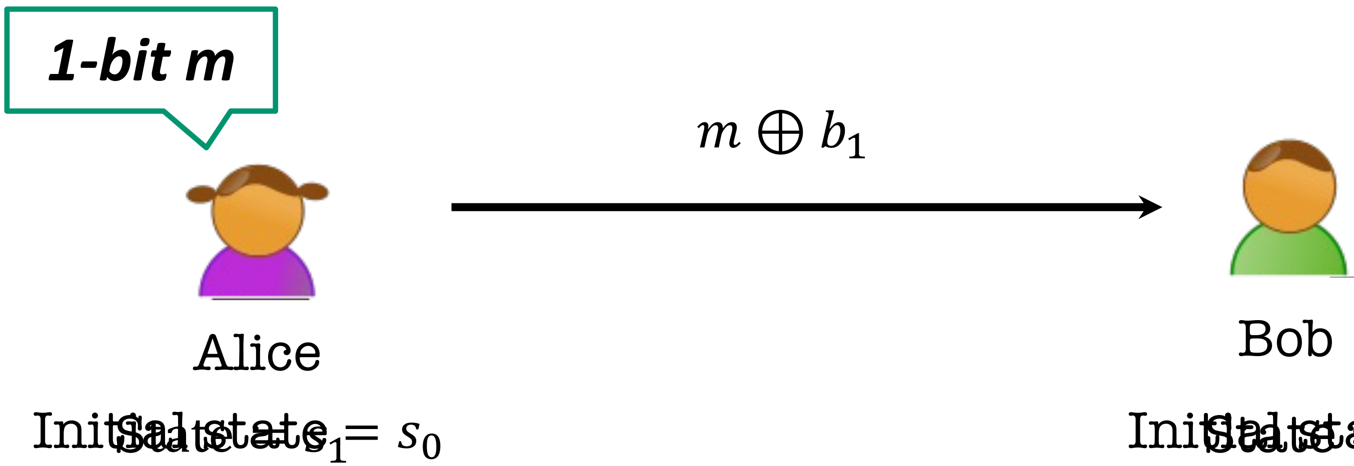
Proof of Security (exercise):

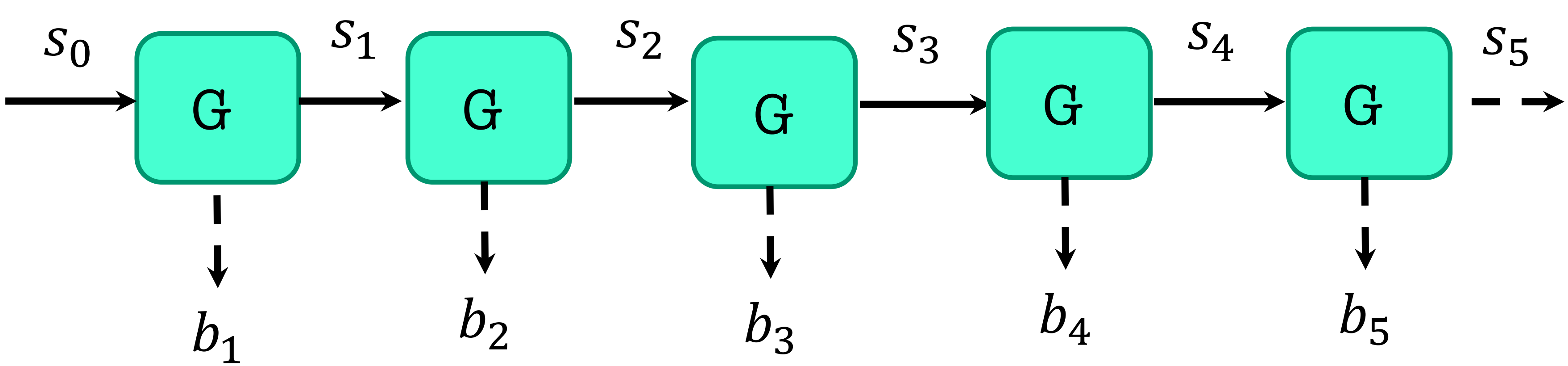
Use next-bit (or previous-bit?) unpredictability!

<u>Construction of G'(s_0)</u>   Output $b_1\ b_2\ b_3\ b_4\ b_5\ \ldots s_L$.

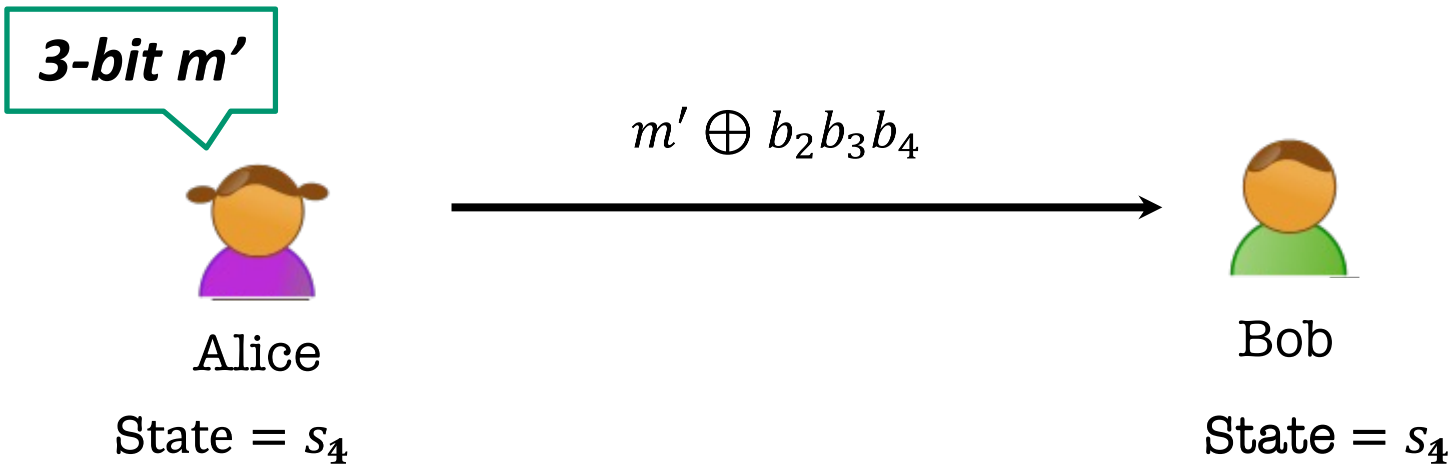


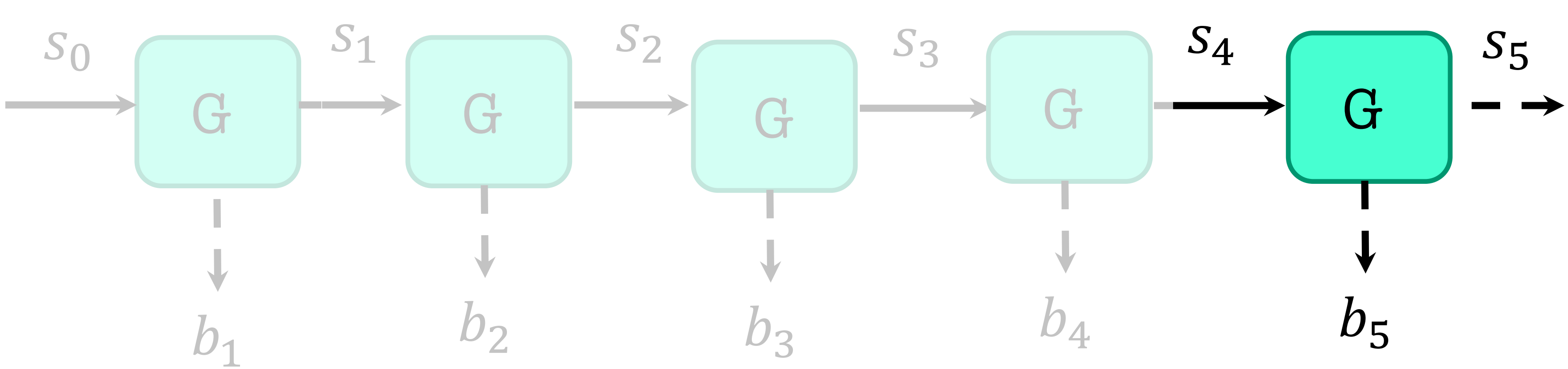
Stateful Encryption of Many Messages



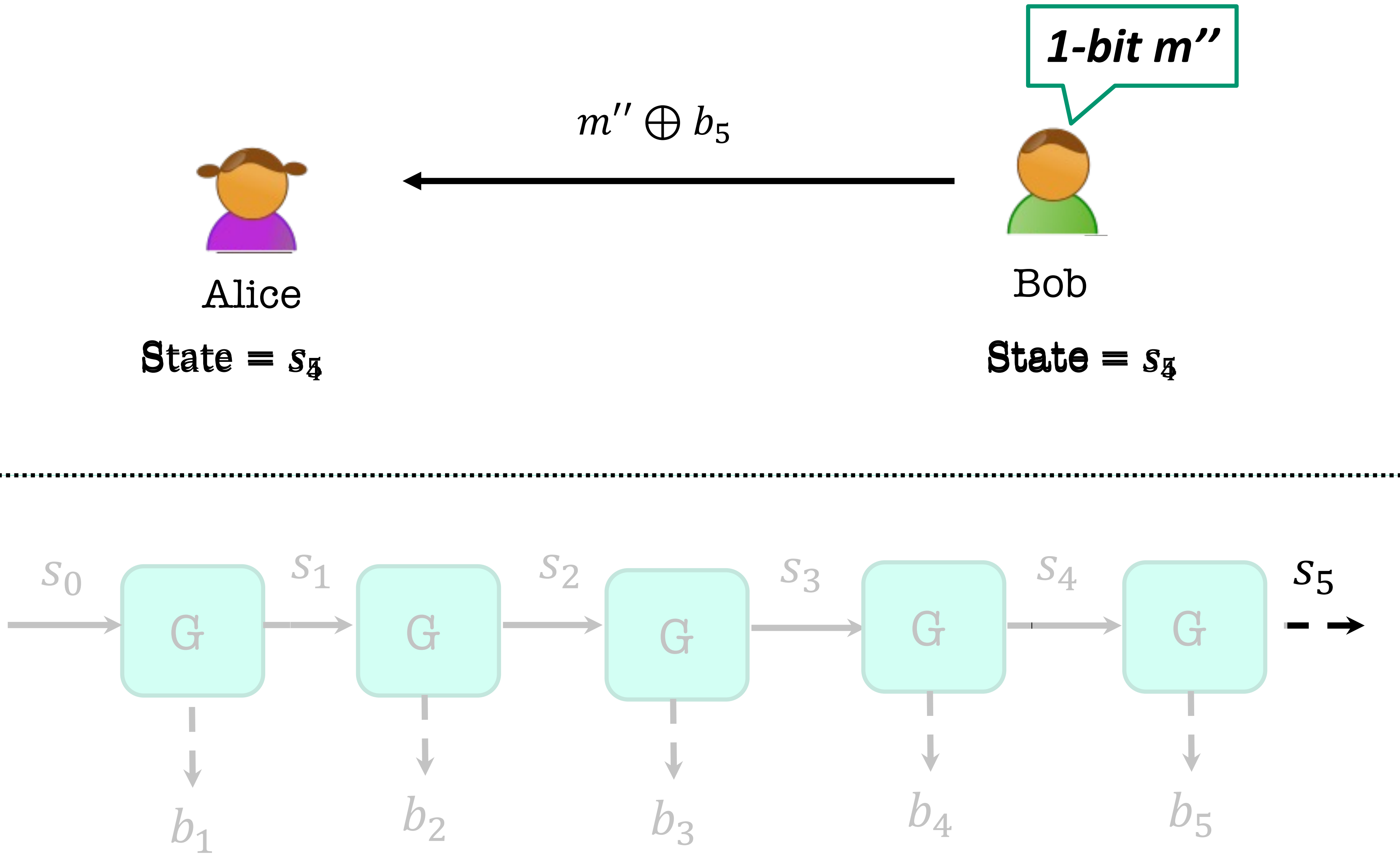


Stateful Encryption of Many Messages





Stateful Encryption of Many Messages



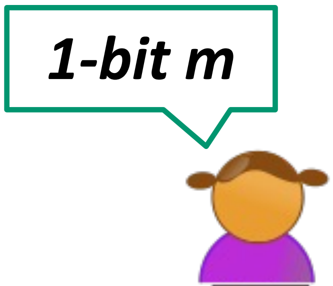
Stateful Encryption of Many Messages

- **PLUS:** Alice and Bob can keep encrypting as many bits as they wish.

- **MINUS:** Alice and Bob have to keep their states in perfect synchrony. They cannot transmit simultaneously.

IF NOT:
Correctness goes down the drain, so does security.

How to be Stateless? Here is an idea…



pick a random index, say 5

$(5, m \oplus b_5)$

Alice

Bob

Key $\mathrm{k} = s_0$

$b_1 \quad b_2 \quad b_3 \quad b_4 \quad b_5 \quad \cdots \quad b_{n^{100}-1} \quad b_{n^{100}}$

Key $\mathrm{k} = s_0$

$b_1 \quad b_2 \quad b_3 \quad b_4 \quad b_5 \quad \cdots \quad b_{n^{100}-1} \quad b_{n^{100}}$

DOES THIS WORK?

Collisions! Pr[Alice's first two indices collide] $\geq 1/n^{100}$

$\Longrightarrow$ Alice is using the same one-time pad bit twice!

Here is another idea…

1-bit m

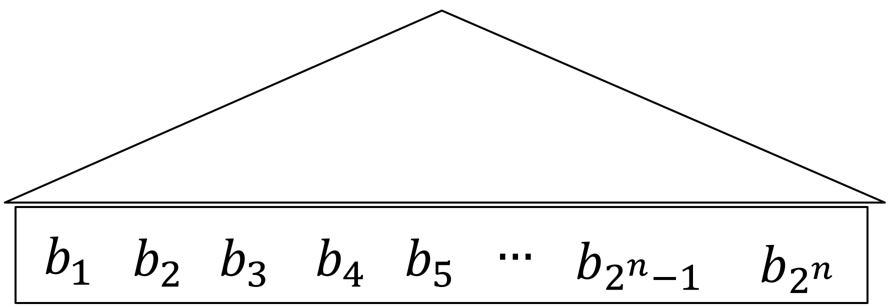
pick a random index, say 5

$(5, m \oplus b_5)$ →

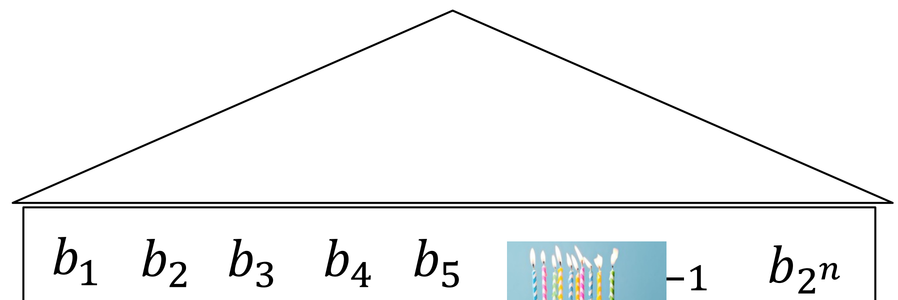
Alice

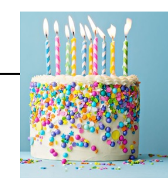
Bob

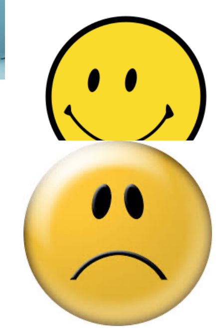
Key $\mathrm{k} = s_0$



Key $\mathrm{k} = s_0$





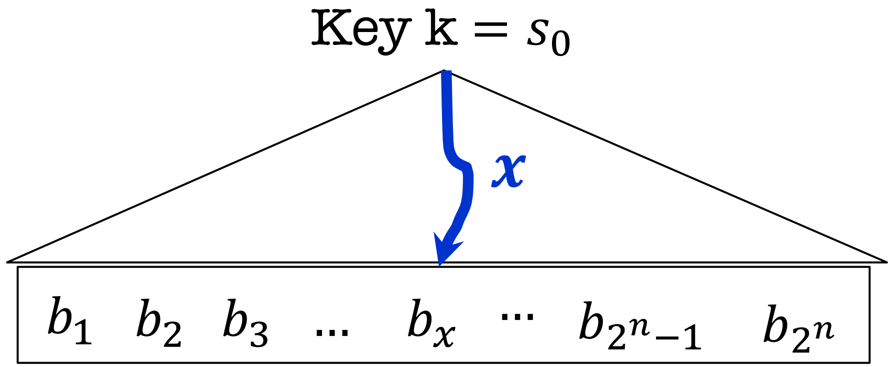


Pr[∃ collision in $t = poly(n)$ indices] $\leq t^2/2^n = \mathrm{negl}(n)$

BUT: Alice and Bob are not poly-time!

Pseudorandom Functions

But these are ~~good~~ bad ideas…



Goal: Never compute this exponentially long string explicitly!

Instead, we want a function $\boldsymbol{f_k(x) = b_x}$, the x^{th} bit in the implicitly defined (pseudorandom) string.

Computable in time $poly(|x|) = poly(n)$.

$\boldsymbol{f_k(x_1), f_k(x_2)}, \ldots$ computationally indistinguishable from random bits, for random (or any distinct) $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots$

$|x| = n =$ length of the string x.

TODAY

How to encrypt (poly) many messages with a fixed key?

1. PRG length extension.

Theorem: If there is a PRG that stretches by one bit, there is one that stretches by poly many bits

Consequence: *Stateful* encryption of poly many messages.

2. Another new notion: Pseudorandom Functions (PRF).

Consequence: *Stateless* encryption of poly many messages.

Theorem (next lec): If there is a PRG, then there is a PRF.

New Proof Technique: Hybrid Arguments.

Pseudorandom Functions

Collection of functions $\mathcal{F}_\ell = \{f_k : \{0,1\}^\ell \to \{0,1\}^m\}_{k \in \{0,1\}^n}$

- indexed by a key k
- n: key length, ℓ: input length, m: output length.
- Independent parameters, all poly(sec-param) = poly(n)
- #functions in $\mathcal{F}_\ell \leq 2^n$ (singly exponential in n)

Gen(1^n): Generate a random n-bit key k.

Eval(k, x) is a poly-time algorithm that outputs $f_k(x)$.

Pseudorandom Functions

Collection of functions $\mathcal{F}_\ell = \{f_k : \{0,1\}^\ell \to \{0,1\}^m\}_{k \in \{0,1\}^n}$

- indexed by a key k
- n: key length, ℓ: input length, m: output length.
- Independent parameters, all poly(sec-param) = poly(n)
- #functions in $\mathcal{F}_\ell \leq 2^n$ (singly exponential in n)

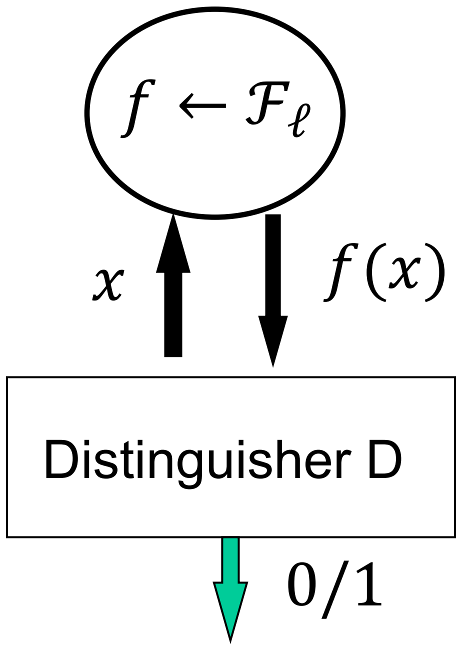
$\approx$

Collection of ALL functions $ALL_\ell = \{f : \{0,1\}^\ell \to \{0,1\}^m\}$

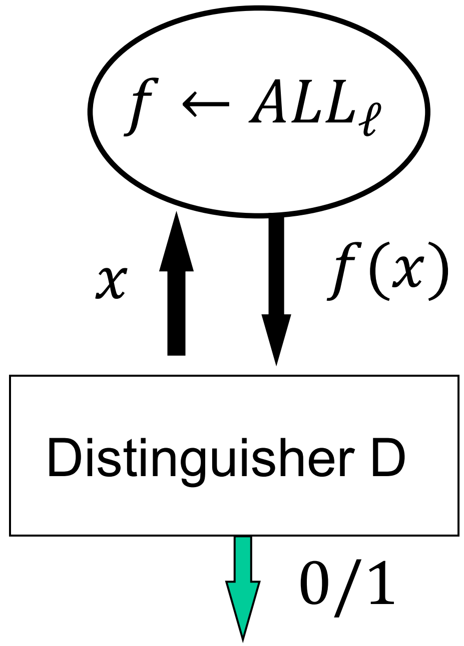
- #functions in $ALL_\ell \leq 2^{m2^\ell}$ (doubly exponential in ℓ)

Pseudorandom Functions should be "indistinguishable" from random

The pseudorandom world



The random world



For all ppt D, there is a negligible function μ s.t.

$$\left| \Pr[f \leftarrow \mathcal{F}_\ell : D^f(1^n) = 1] - \Pr[f \leftarrow ALL_\ell : D^f(1^n) = 1] \right| \leq \mu(n)$$

PRF $\Longrightarrow$ Stateless Secret-key Encryption

$Gen(1^n)$: Generate a random n-bit key k that defines

$$f_k: \{0,1\}^\ell \to \{0,1\}^m$$

(the domain size, 2^ℓ, had better be super-polynomially large in n)

$Enc(k,m)$: Pick a random x and
let the ciphertext c be the pair $(x, y = f_k(x) \oplus m)$.

$Dec(k, c = (x,y))$: Output $f_k(x) \oplus y$.

Correctness:

$Dec(k,c)$ outputs $f_k(x) \oplus y = f_k(x) \oplus f_k(x) \oplus \mathrm{m} = \mathrm{m}$.

NEXT LECTURE

How to encrypt (poly) many messages with a fixed key?

1. PRG length extension.

Theorem: If there is a PRG that stretches by one bit, there is one that stretches by poly many bits

Consequence: *Stateful* encryption of poly many messages.

2. Another new notion: Pseudorandom Functions (PRF).

Consequence: *Stateless* encryption of poly many messages.

***Theorem (next lec)*: If there is a PRG, then there is a PRF.**

New Proof Technique: Hybrid Arguments.