

Lecture 8

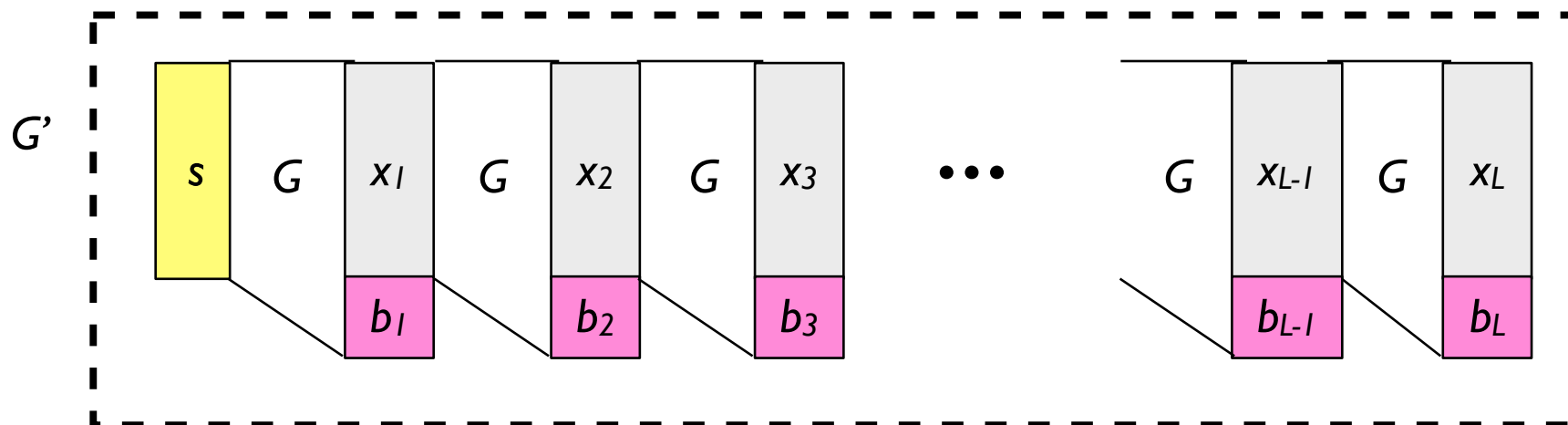
Hybrid arg: Extending a p.r.g.'s output

- **Given** $G:\{0,1\}^n \rightarrow \{0,1\}^{n+1}$ and a polynomial $L(n)$,

define $G':\{0,1\}^n \rightarrow \{0,1\}^{L(n)}$

- 1. $x_1 \leftarrow G(s)$;
- 2. $x_2 \leftarrow G([x_1])$
- ...
- Output first bits of $x_1, x_2, x_3, \dots, x_L$

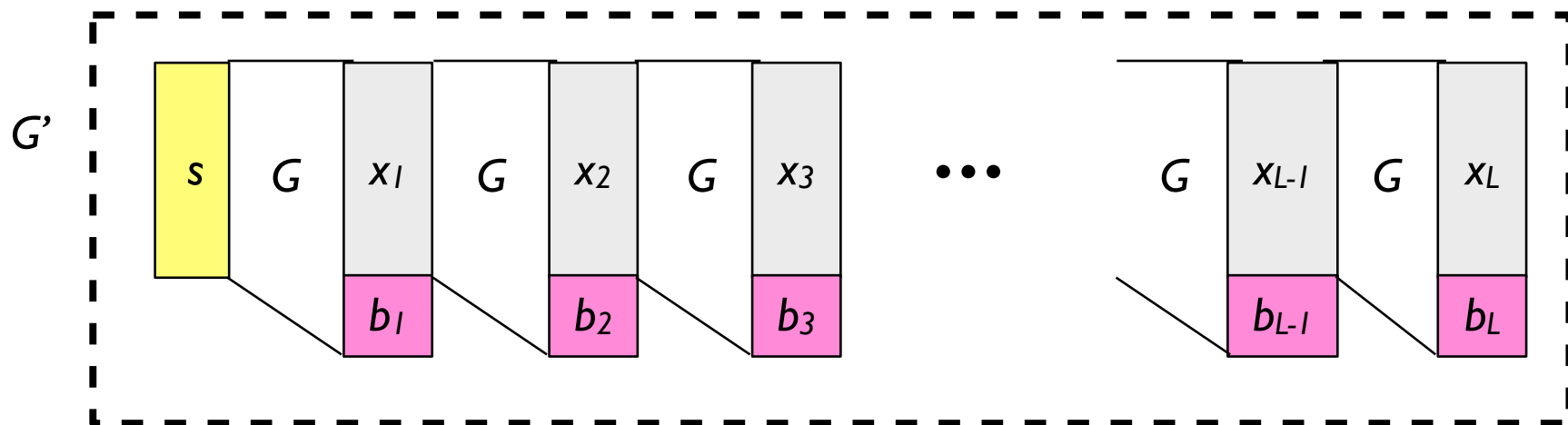
(let $[x]$ denote the last n bits of x)



Hybrid arg: Extending a p.r.g.'s output

- **Theorem:** If G is a p.r.g., then G' is p.r.g.

- Given a distinguisher A for G' with advantage ϵ , we construct a distinguisher B for G with advantage ϵ/L
- Problem: If G' is insecure, how do we localize the problem to one application of G ?
- (In general: how do we argue about the security of a system made of many parts?)

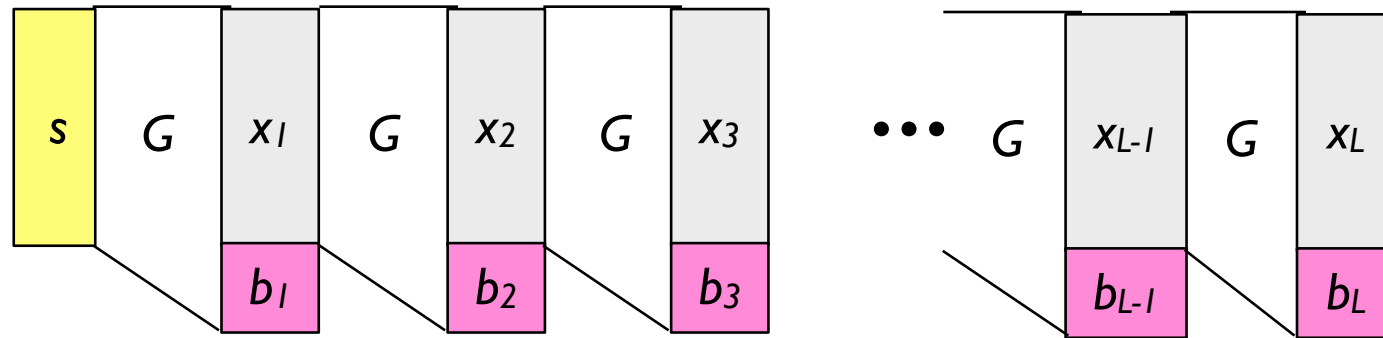


Hybrid argument

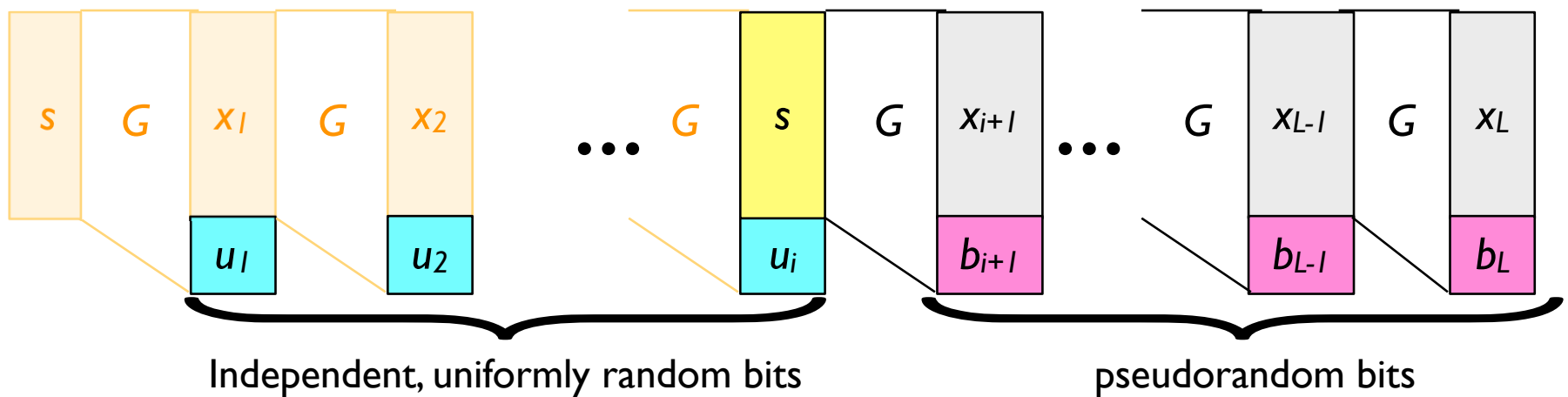
- Construct a sequence of distributions D_0, \dots, D_L
 - $D_0 = G'(s), \dots, D_L = U_L$
- Two claims:
 - Claim 1: For some index j , A can distinguish D_{j-1} from D_j
 - Claim 2: We can use A to break the j^{th} application of G
- How do we construct distributions D_i ?

“Hybrid” Distributions D_i

Distribution D_0 : same as $G'(U_n)$



Distribution D_i : first i bits are fresh, independent random coin flips



Hybrid argument

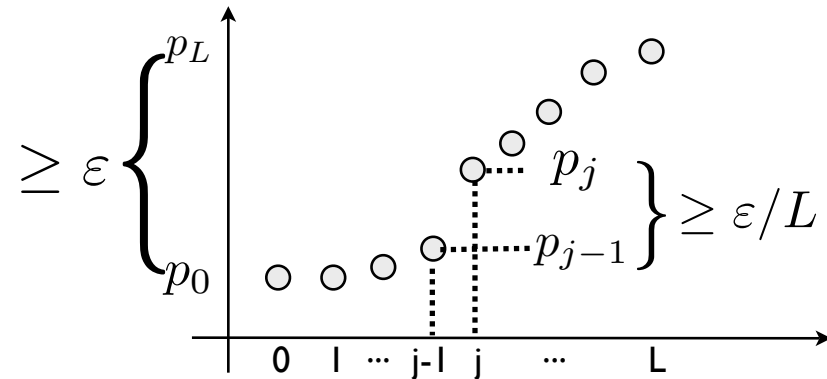
- Let $p_i = \Pr(A(D_i) = \text{“yes”})$

- Recall that A is a distinguisher for G' , so $|p_L - p_0| > \epsilon$
- Suppose that $p_L - p_0 > \epsilon$ (the case where $p_0 - p_L > \epsilon$ is similar)
- **Claim 1:** For some j between 1 and L , $p_j - p_{j-1} > \epsilon/L$

- Expand the sum:

$$p_L - p_0 = \sum_{i=1}^L (p_i - p_{i-1})$$

- One of the terms in the sum is at least ϵ/L

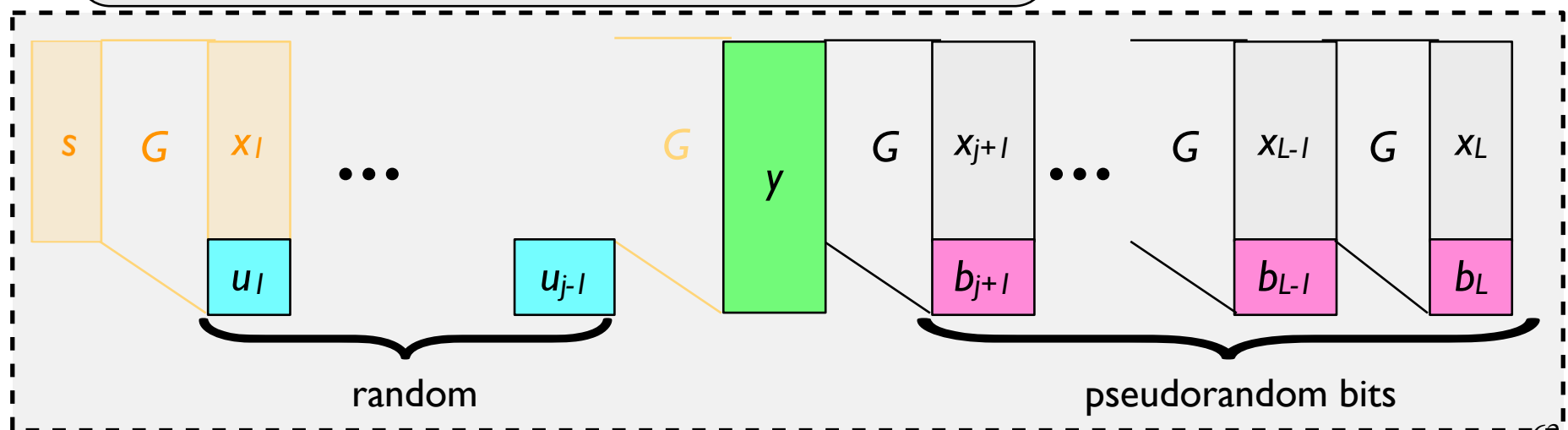


- **Claim 2:** If $p_j - p_{j-1} > \epsilon/L$, then we can construct a distinguisher B for the j^{th} invocation of G

Distinguisher B for G

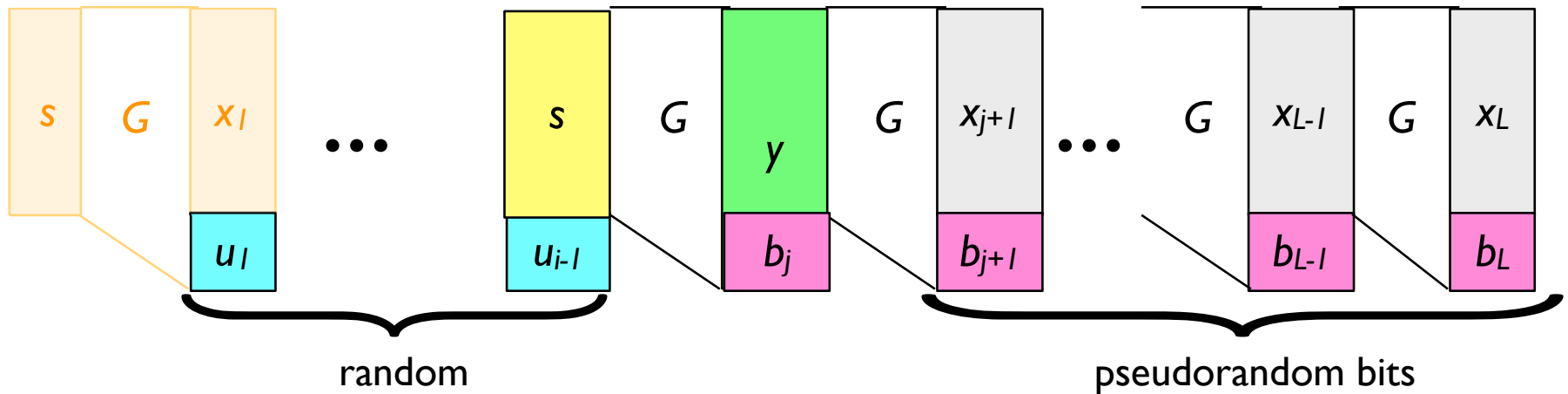
- On input y in $\{0,1\}^{n+1}$, generate an input z for A such that
 - if $y \sim G(U_n)$, then $z \sim D_{j-1}$
 - if $y \sim U_{n+1}$, then $z \sim D_j$

- Algorithm $B(y)$:
 1. $u_1, \dots, u_{j-1} \leftarrow \{0,1\}$
 2. $x_j \leftarrow [y]$
 3. $b_i \leftarrow \text{last-bit}(y)$
 4. Generate b_j, \dots, b_L as in G'
 5. Return $A(u_1, \dots, u_{j-1}, b_j, \dots, b_L)$

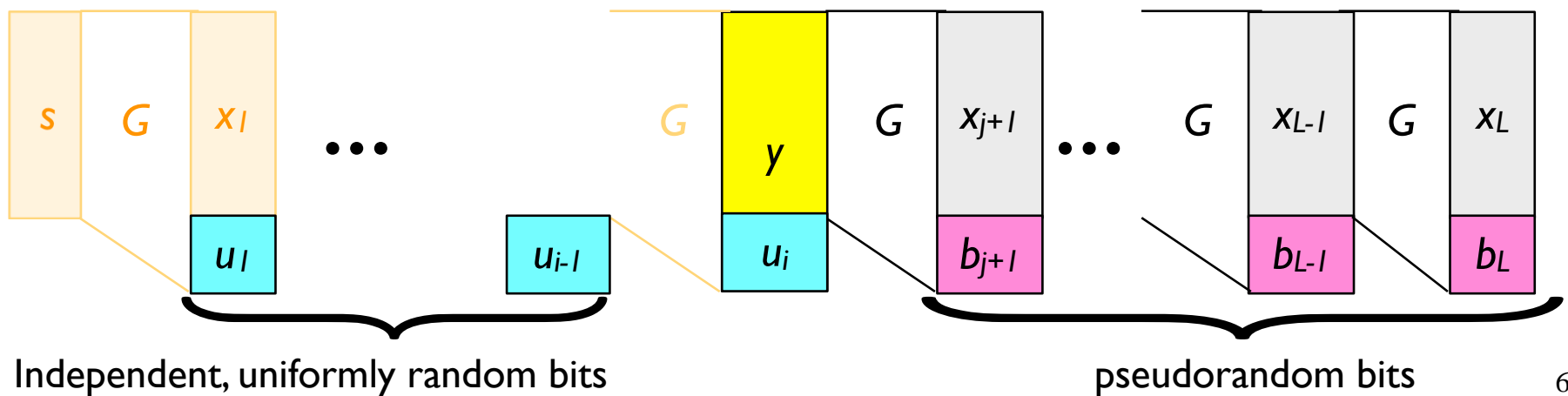


Hybrid Argument

- Claim 3: If $y \sim G(\mathbf{U}_n)$, then $z \sim D_{j-1}$



- Claim 4: If $y \sim \mathbf{U}_{n+1}$, then $z \sim D_j$



Concluding the argument

- We have constructed B so that
 - If $y \sim G(U_n)$, then B runs A on $z \sim D_{j-1}$
 - If $y \sim U_{n+1}$, then B runs A on $z \sim D_j$
- B's behavior:
 - If $y \sim G(U_n)$, then $\Pr(B(y)=1) = \Pr(A(D_{j-1})=1) = p_{j-1}$
 - If $y \sim U_{n+1}$, then $\Pr(B(y)=1) = \Pr(A(D_j)=1) = p_j$
- So...
 - $|\Pr(B(G(U_n))=1) - \Pr(B(U_{n+1})=1)| > \epsilon/L$
- $\text{run-time}(B) \approx \text{run-time}(A) + (L * \text{run-time}(G))$
- Conclusion: **Theorem: If G is a p.r.g., then G' is a p.r.g.**
 - **If** no adversary running in time $t + (L * \text{run-time}(G))$ distinguishes the output of **G** from random with advantage ϵ/L , **then** no adversary running time t distinguishes the output of **G'** from random with advantage ϵ

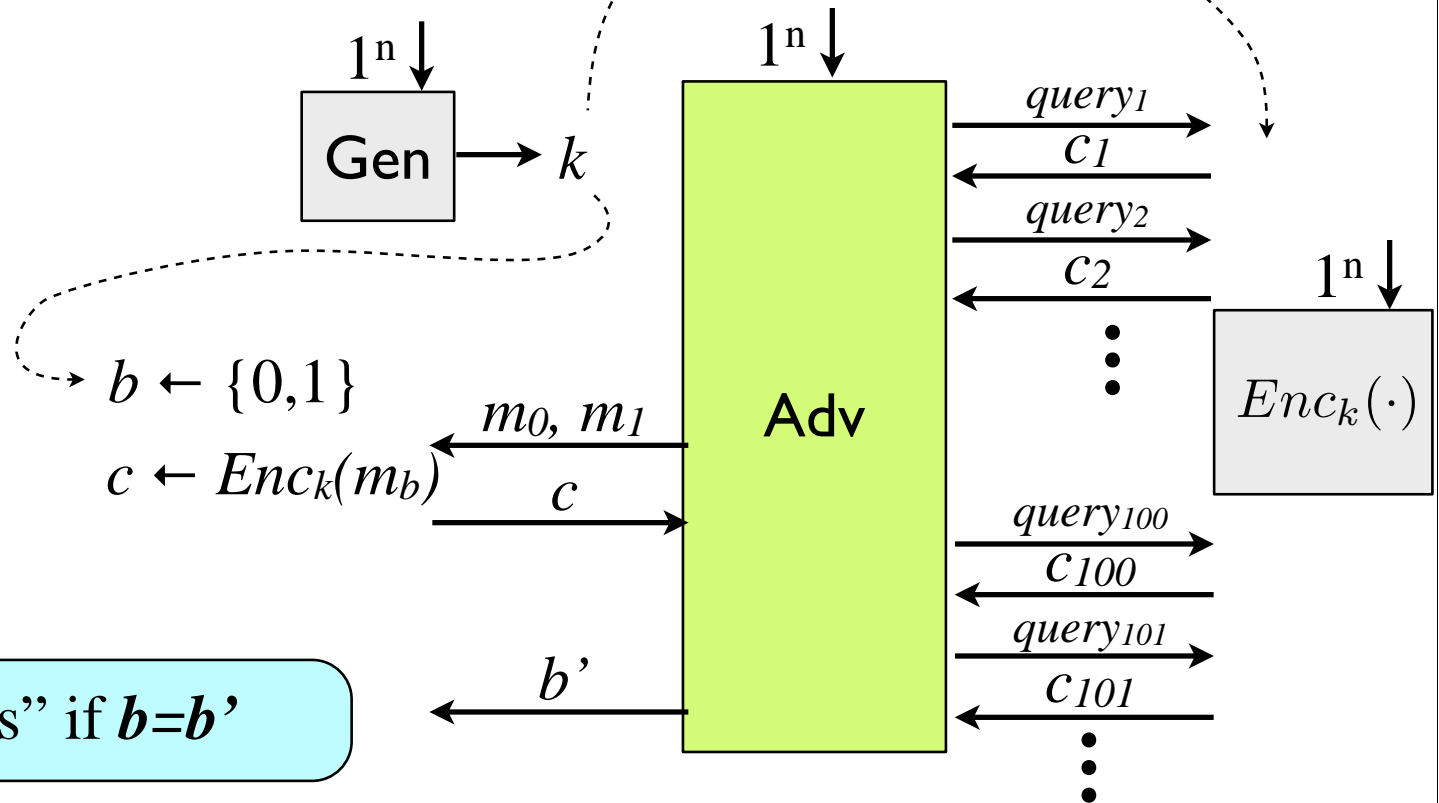
One last issue

- How does B know the “magic” index j ?
- Three answers:
 - We just have to show that B exists, so it suffices to show that j exists
 - Try all values of j (requires sampling to estimate the values p_i)
 - Choose j at random!

Chosen-plaintext attacks

- Capture situations where adversary has partial control over plaintext
 - Strongest notion of “passive” security (no tampering with channel)
 - Control over plaintext formalized via encryption “oracle”

CPA indistinguishability experiment



Adv "wins" if $b = b'$

(Gen, Enc, Dec) has (t, ϵ) -indistinguishable encryptions under chosen-plaintext attacks (IND-CPA) if, for all **Adv** running in time at most t ,

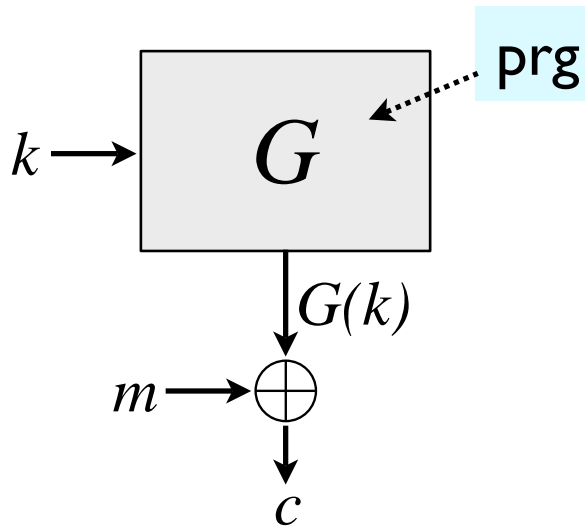
$$\Pr(\mathbf{Adv} \text{ wins}) \leq \frac{1}{2} + \epsilon$$

Properties of CPA security

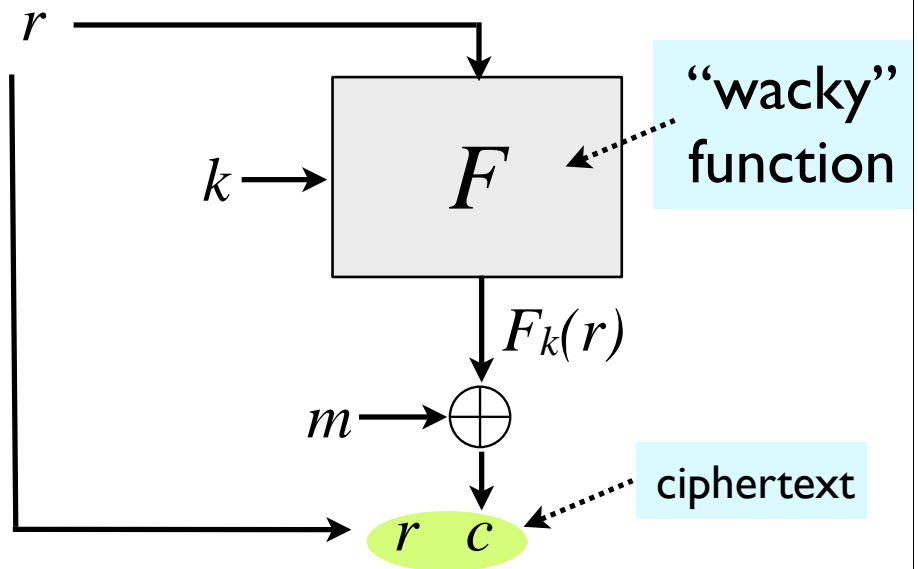
- **Proposition:** Stateless CPA-secure encryption must be randomized
 - Suppose that $\text{Enc}_k(m)$ is a deterministic function of k and m
 - Adv can use oracle to generate ciphertexts c_0, c_1 for m_0, m_1
 - Compare c to c_0, c_1 to learn b
- **Proposition 3.22+:** IND-CPA-security for a single-fixed length message implies security for multiple, arbitrary-length messages.
 - Proof via hybrid argument
 - In book
 - Note: suffices to prove security for multiple fixed-length messages
- So how can we construct such schemes?

A first attempt

PRG as stream cipher:



Throw in some fresh random stuff?



- What property is needed from F ?
- All the values $F_k(r)$ should look "random"
 - Aha! We have developed a language for capturing this...

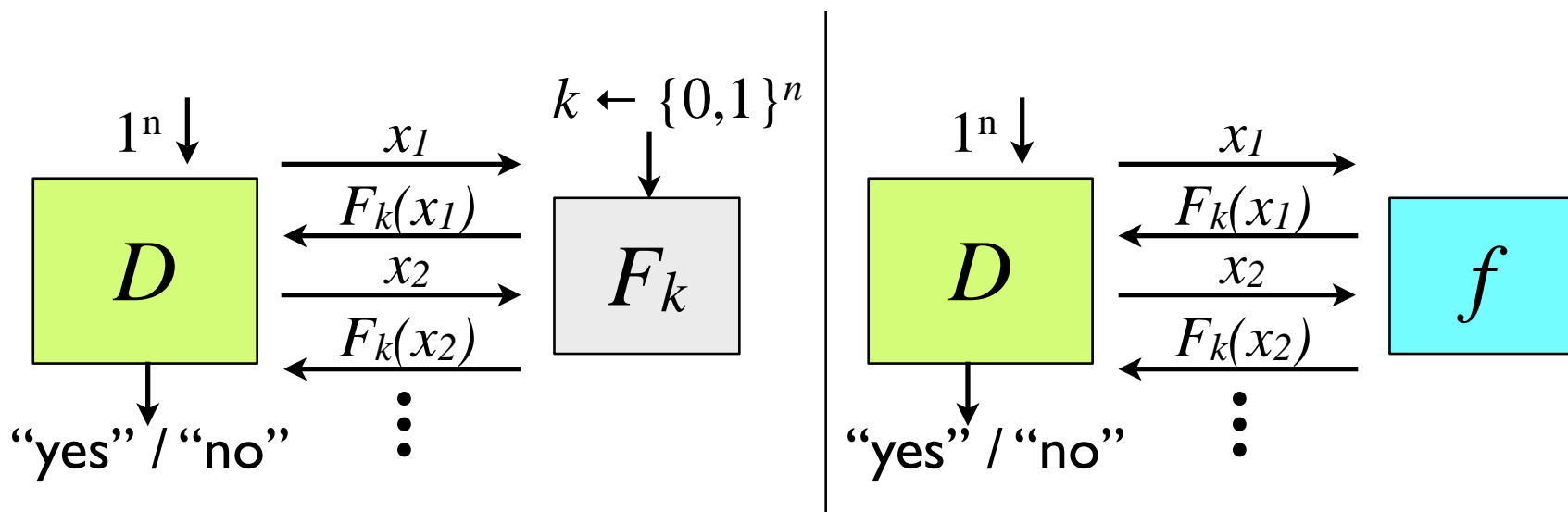
Pseudorandom Functions

- Recall how we approached PRG's:
 - Clear objective: uniformly random bits
 - “pseudorandom” = “PPT adversary with access to either random or pseudorandom”
- Here the object is a function F
 - For simplicity, suppose $|r| = |k| = |F_k(r)| = n$ bits
 - For fixed k , $F_k: \{0,1\}^n \rightarrow \{0,1\}^n$
 - F_k should look random to someone who doesn't know k
- So... what is a truly random function f ?
 - Look-up table with uniformly random entries
- What does “access to f ” mean?
 - How many bits required to describe f ? ($n2^n$)
 - Cannot “give” f to adversary...
 - Oracle language is helpful

$f(000\dots 0)$
$f(000\dots 1)$
...
$f(111\dots 1)$

Pseudorandom functions

- $F: \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$
- (For simplicity) For every k in $\{0,1\}^n$, $F_k: \{0,1\}^n \rightarrow \{0,1\}^n$



oracle notation

- F is a (length-preserving) *pseudorandom function* if for all PPT D ,

$$\left| \Pr(D^{F_k(\cdot)}(1^n) = \text{yes}) - \Pr(D^{f(\cdot)}(1^n) = \text{yes}) \right| \leq \text{negl}(n)$$
 where $k \leftarrow \{0,1\}^n$ and f is a uniformly random function