CIS 5560

Cryptography
Lecture 5

Announcements

- **HW 1 out yesterday**
  - Due **Friday**, Feb 6 at 5PM on Gradescope
  - Covers PRGs, OTPs, indistinguishability
- HW0 due this Friday (Jan 30)

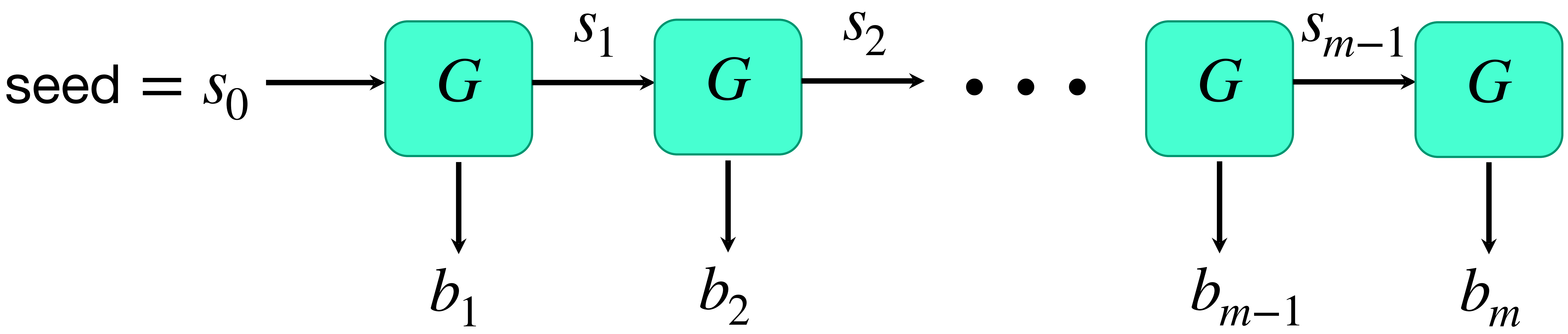
Recap of last lecture

Construction: PRG Length extension

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

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

Construction of $G'(s_0)$:



Technique: Hybrid argument

Key idea: instead of directly trying to go from first distribution to second, take small steps!

1. **Construct the steps:**
   A sequence of (polynomially-many) distributions $H_1, \ldots, H_{m-1}$ b/w the two target distributions.
2. **Show that it's easy to move between steps:**
   Argue that each pair of neighboring distributions are indistinguishable.
3. **Start moving:**
   Conclude that the target distributions are indistinguishable via contradiction:
   A. Assume the target distributions are distinguishable
   B. Must be the case that an intermediate pair of distributions is distinguishable
   C. This contradicts 2 above.

Proof that G' is a PRG

PRG Indistinguishability of G says that the following distributions are indistinguishable:

$\{G(x) \mid x \leftarrow \{0,1\}^n\}$ and $\{y \mid y \leftarrow \{0,1\}^{n+1}\}$

Our goal: show that $\{G'(x) \mid x \leftarrow \{0,1\}^n\}$ and $\{y \mid y \leftarrow \{0,1\}^m\}$ are indistinguishable

Step 1: create more (supposedly) indistinguishable distributions:

$H_0 = \{G'(x) \mid x \leftarrow \{0,1\}^n\}$

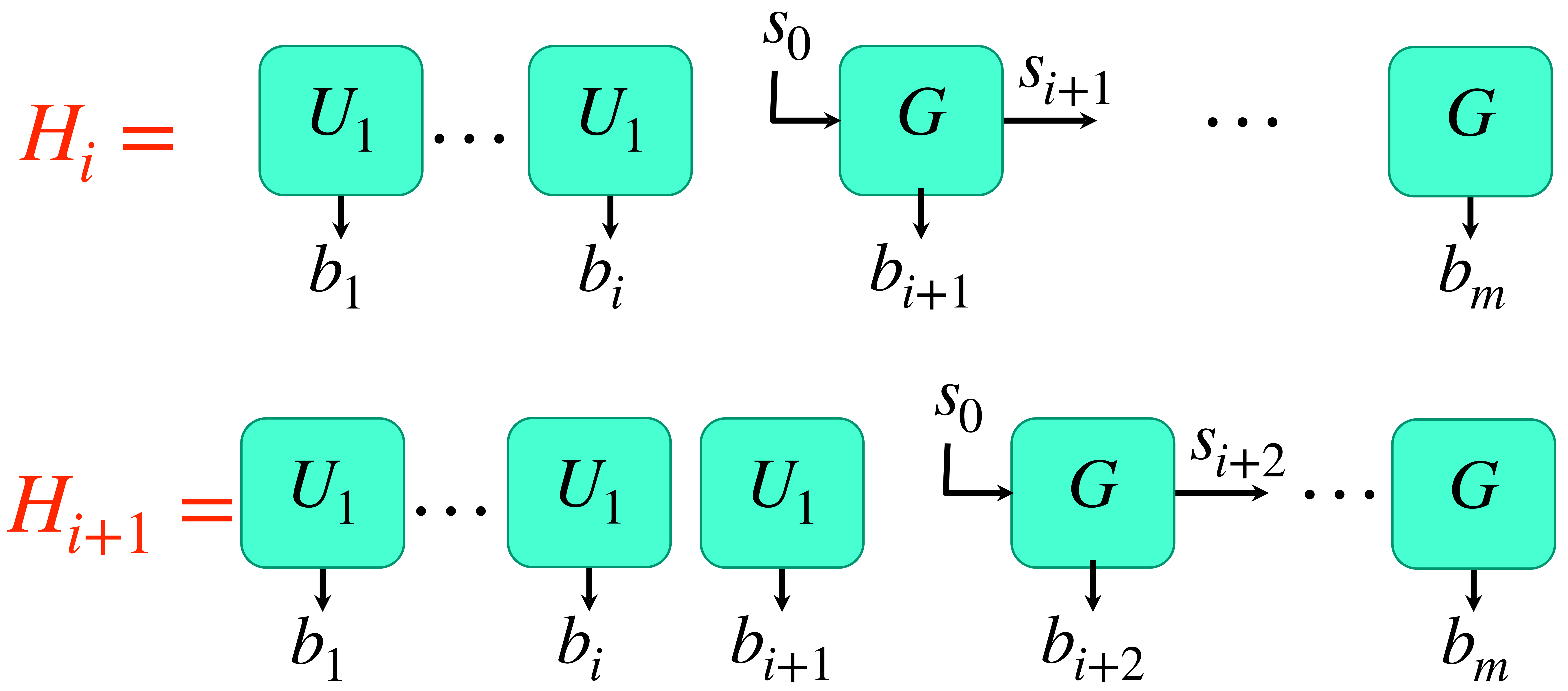
$= \{\text{running } G \ m \text{ times}\}$

$H_i = \{\text{Output } i \text{ uniform bits and run } G \ m - i \text{ times}\}$

$H_m = \{y \mid y \leftarrow \{0,1\}^m\}$

Proof that G' is a PRG

Step 2: Showing that H_i and H_{i-1} are indistinguishable:



Proof that G' is a PRG

Step 2: Showing that H_i and H_{i-1} are indistinguishable:

Proof by contradiction:

Assume they are not. That is, there exists a PPT distinguisher D against them.
Then we will construct a distinguisher D' against G as follows:

$D'(y = b \parallel s_0)$:

1. Sample i random bits $b_1, \ldots, b_i$.
2. Set $b_{i+1} := b$.
3. Run $m - i - 1$ iterations of G using s_0 as seed, and let $b_{i+2}, \ldots, b_m$ be the result.
4. Run $D(b_1, \ldots, b_m)$ and output whatever it outputs.

Now clearly, when y is pseudorandom, the bits are distributed as in H_i, while if y is random, then they are distributed as in H_{i+1}. Hence if D distinguishes, so does D'.

Since this contradicts G's indistinguishability, it must be the case that no such D exists.

Hybrid argument

B. Must be the case that an intermediate pair of distributions is distinguishable

Lemma: Let $p_0, p_1, \ldots, p_m$ be probability of outputting 0 in $H_0, H_1, \ldots, H_m$

If $p_0 - p_m$ is noticeable,

then there is an i such that $p_i - p_{i+1}$ is noticeable.

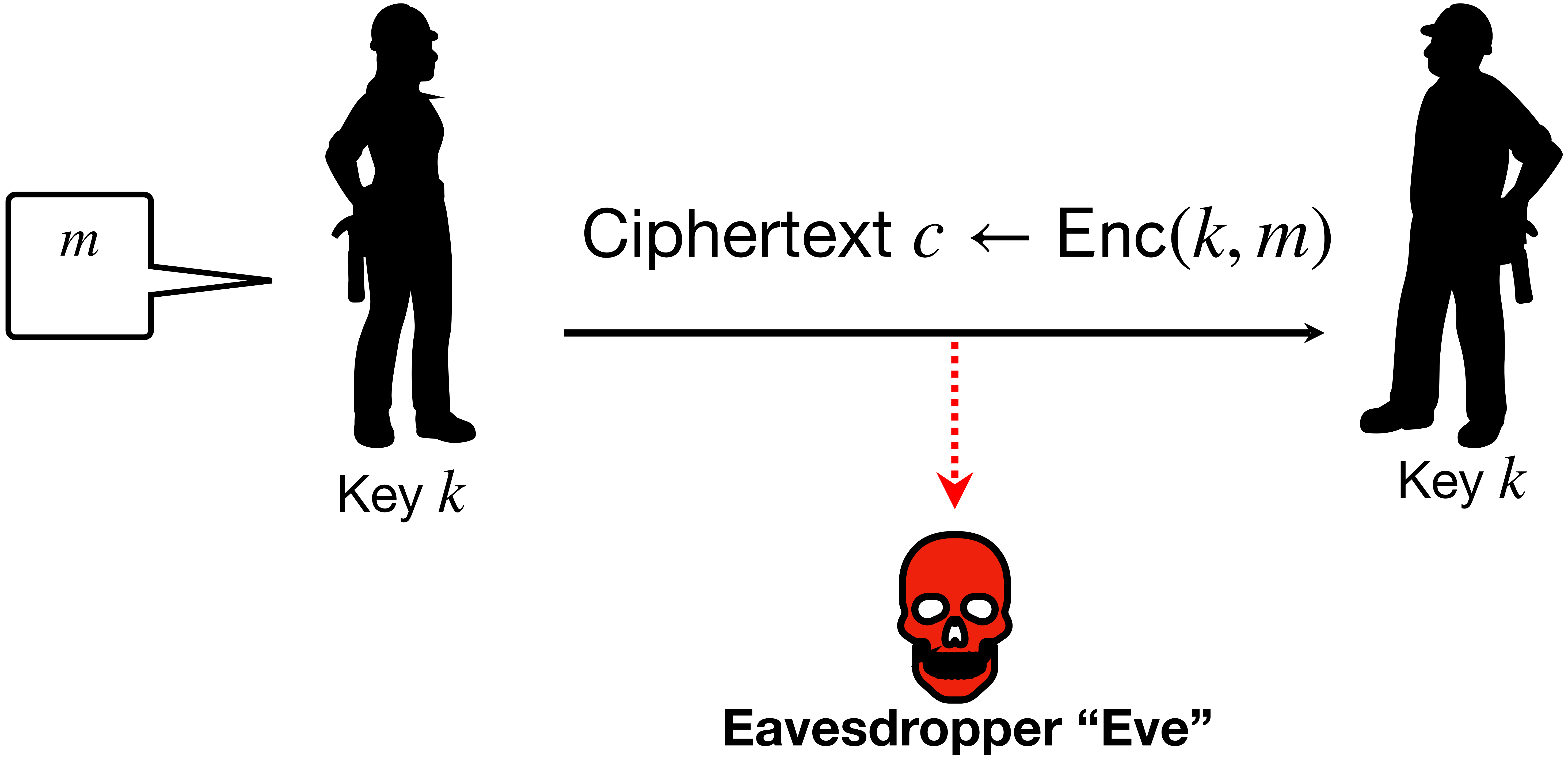
Proof:
$$\begin{aligned} 1/p(n) &\leq |p_m - p_0| \\ &= |(p_m - p_{m-1}) + (p_{m-1} - p_{m-2}) + \cdots + (p_1 - p_0)| \\ &\leq |(p_m - p_{m-1})| + |(p_{m-1} - p_{m-2})| + \cdots + |(p_1 - p_0)| \end{aligned}$$

Notice that each term in the series is the advantage of distinguishing the i-th pair.
Cannot be that all advantages are negligible, as their sum is noticeable. Hence at least one must be noticeable.

Today's Lecture

- Encryption for many messages
  - Definition
  - Attempted construction from PRGs
- PRFs
- PRPs
- Block ciphers

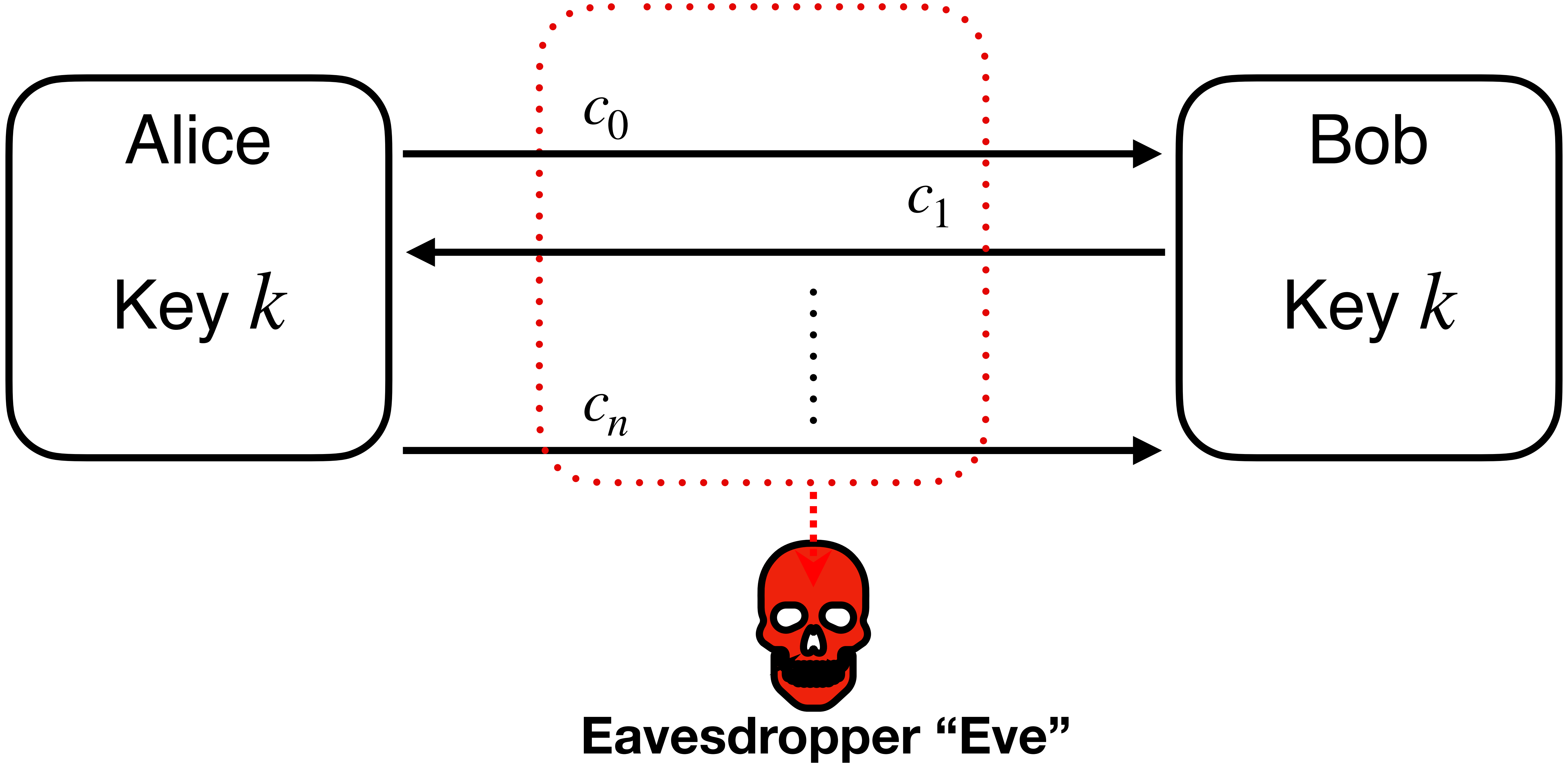
So far: Secure Communication for 1 Message



Alice wants to send a message m to Bob without revealing it to Eve.

SETUP: Alice and Bob meet beforehand to agree on a secret key k.

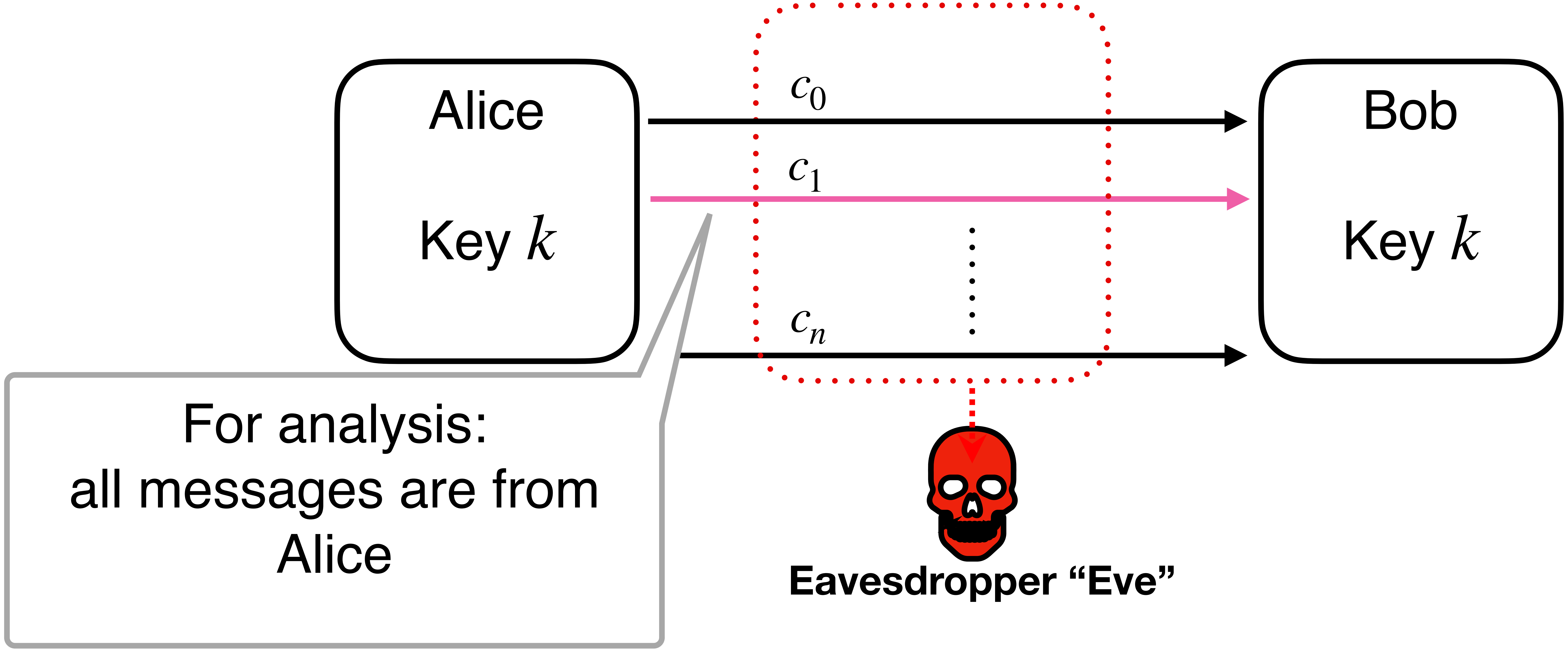
What about secure *conversations*?



Alice and Bob want to send *many* messages to each other, without revealing *any* of them to Eve.

Requirement: Must use the same key!

What about secure *conversations*?



Alice and Bob want to send *many* messages to each other,
without revealing *any* of them to Eve.
Requirement: Must use the same key!

Construction Attempt #1: Stream Ciphers

Gen$(1^\lambda) \rightarrow k$:

1. Sample an n-bit string at random.

Enc$(k, m) \rightarrow c$:

1. Expand k to an $n+1$-bit string using PRG: $s = G(k)$
2. Output $c = s \oplus m$

Dec$(k, c) \rightarrow m$:

1. Expand k to an $n+1$-bit string using PRG: $s = G(k)$
2. Output $m = s \oplus c$

Is this secure for multiple messages?

No! It becomes a two-time pad!

Multi-message Indistinguishability

- How to formalize? Can we generalize the old definition?

For every $(m_0, m_1, \ldots, m_\ell), (m'_0, m'_1, \ldots, m'_\ell)$, for every PPT adversary A

$$\left| \Pr_{k \leftarrow \mathscr{K}} \left[A \begin{pmatrix} \mathsf{Enc}(k, m_0) \\ \vdots \\ \mathsf{Enc}(k, m_\ell) \end{pmatrix} = 1 \right] - \Pr_{k \leftarrow \mathscr{K}} \left[A \begin{pmatrix} \mathsf{Enc}(k, m'_0) \\ \vdots \\ \mathsf{Enc}(k, m'_\ell) \end{pmatrix} = 1 \right] \right| = \varepsilon(\lambda)$$

- Problems:
  - Messages are fixed ahead of time; cannot depend on cipher text
  - Unwieldy when ℓ grows.

New Style of Definition: Game-based Security

Old: Single-message Indistinguishability

For every m_0, m_1, for every PPT "distinguishing" adversary A

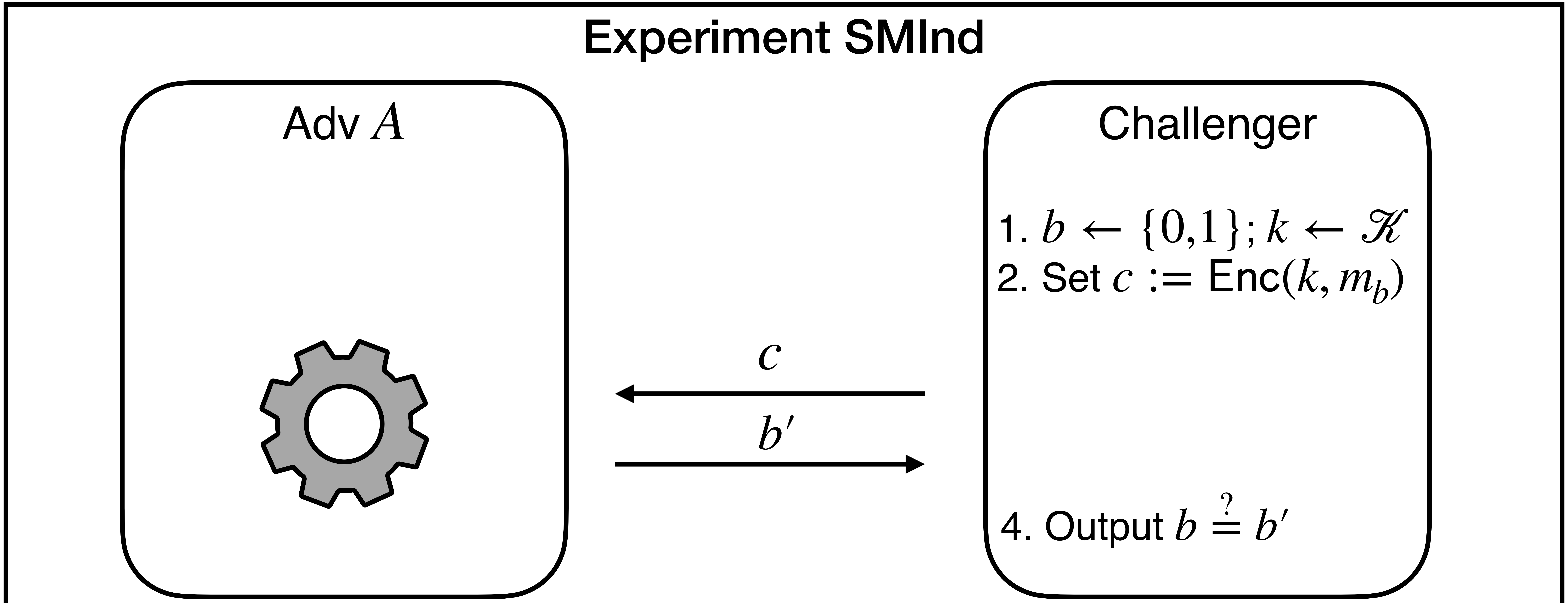
there exists a negligible function ε such that

$$\left| \Pr_{k \leftarrow \mathscr{K}} [A(\mathsf{Enc}(k, m_0)) = 1] - \Pr_{k \leftarrow \mathscr{K}} [A(\mathsf{Enc}(k, m_1)) = 1] \right| = \varepsilon(\lambda)$$

New: Single-msg Indistinguishability Game

For every m_0, m_1, for every PPT “distinguishing” adversary A

$$| \Pr[\text{SMInd} = 1] - \Pr[\text{random guess}] | = \text{negl}(\lambda)$$ “Advantage”

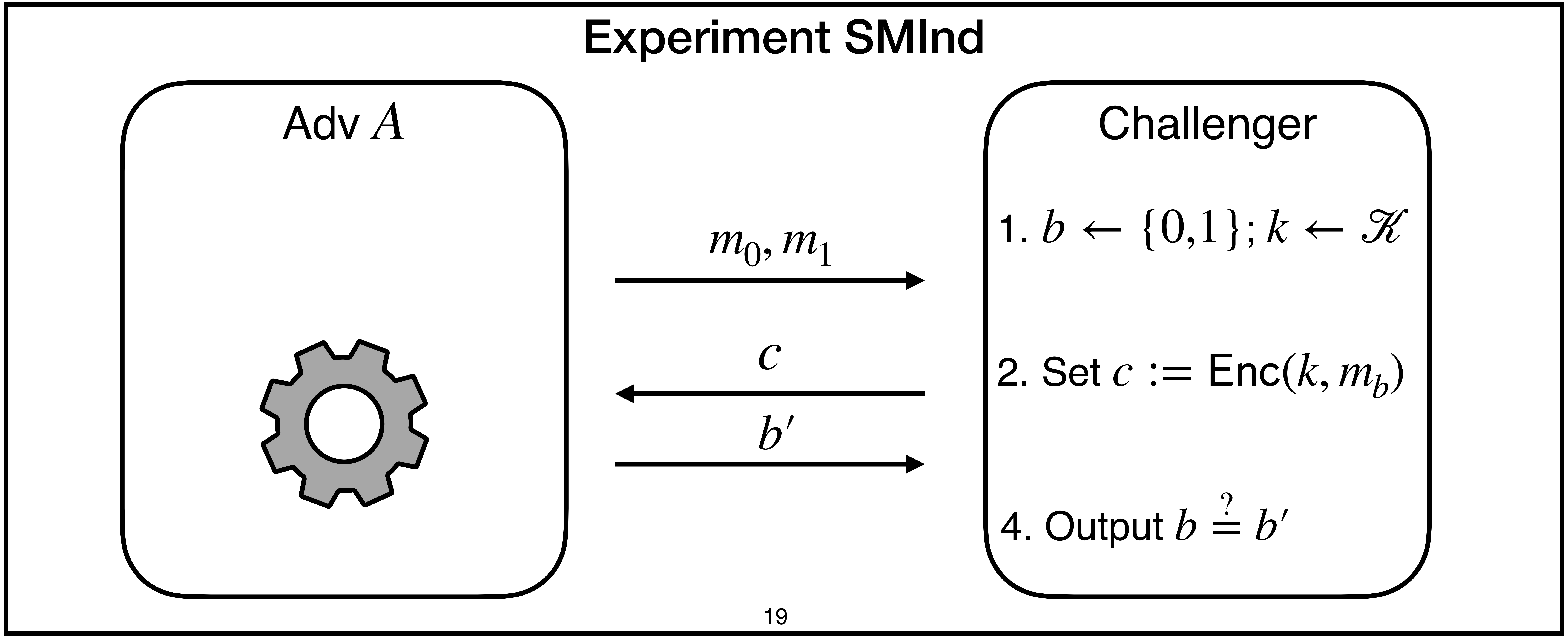


New: Single-msg Indistinguishability Game

For every PPT “distinguishing” adversary A

$$|\Pr[\text{SMInd} = 1] - \Pr[\text{random guess}]| = \text{negl}(\lambda)$$

“Advantage”



New: Single-msg Indistinguishability Game

For every PPT "distinguishing" adversary $\mathscr{A}$

$$\left| \Pr\left[b = b' \;\middle|\; \begin{array}{r} b \leftarrow \{0,1\}, k \leftarrow \mathscr{K} \\ (m_0, m_1) \leftarrow A \\ c := \mathsf{Enc}(k, m_b) \\ b' \leftarrow A(c) \end{array} \right] - \frac{1}{2} \right| = \mathsf{negl}(\lambda)$$

"Advantage"

New: Single-msg Indistinguishability Game

We will show that any scheme that satisfies one defn automatically satisfies other.

Proof sketch.

Denote by ϵ the advantage of any adversary A against the old defn.
We will show that the advantage of A in the new defn is $\epsilon/2$.

Let $p_0 = \Pr[A(\text{Enc}(k, m_0) = 0]$, and let $p_1 = \Pr[A(\text{Enc}(k, m_1) = 0]$. Clearly, $|p_0 - p_1| = \epsilon$

Now, A succeeds in new game when it guess correctly. i.e., its success prob is

$$\Pr[A(\text{Enc}(k, m_b) = 0 \,|\, b = 0] \Pr[b = 0] \quad + \quad \Pr[A(\text{Enc}(k, m_b) = 1 \,|\, b = 1] \Pr[b = 1].$$

But this is exactly $p_0 \cdot \frac{1}{2} + (1 - p_1) \cdot \frac{1}{2} = \frac{1 + p_0 - p_1}{2}$.

Its advantage is thus $\left| \frac{1 + p_0 - p_1}{2} - \frac{1}{2} \right| = \epsilon/2$.

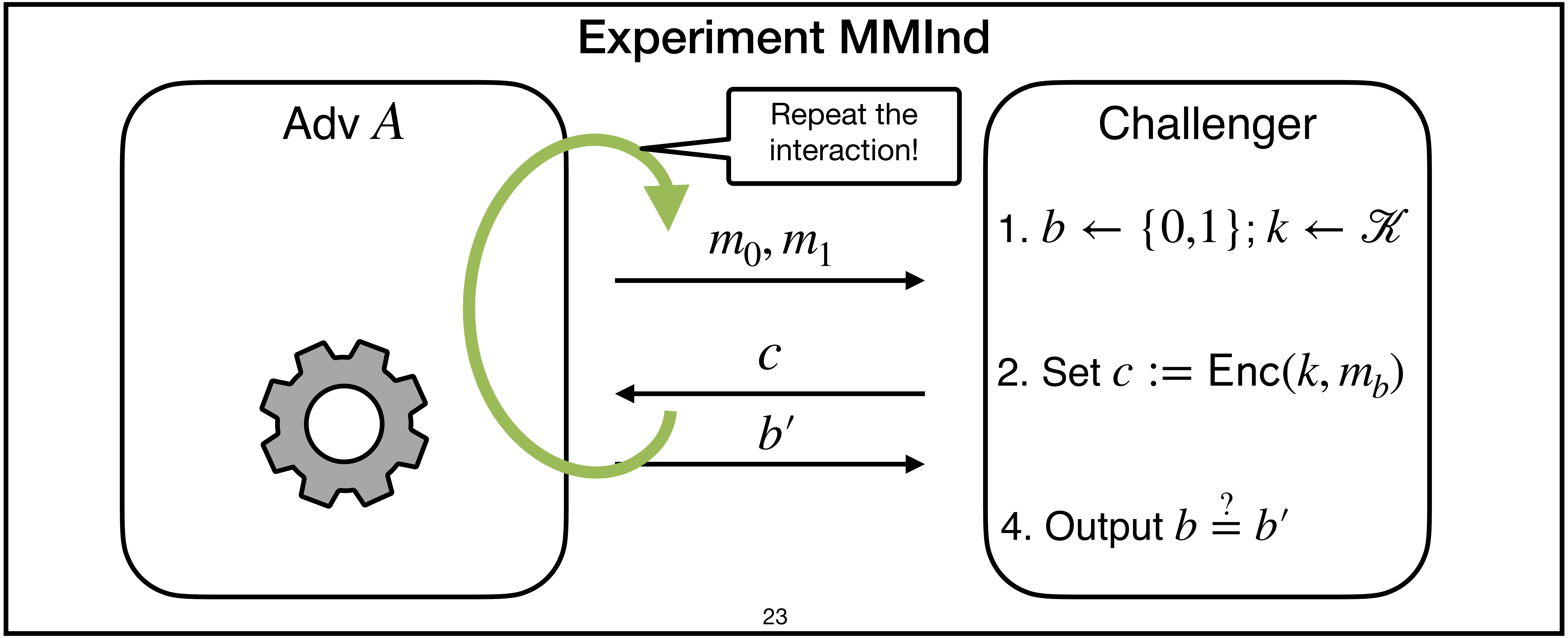
Game-based
Multi-message
Indistinguishability

New: Multi-msg Indistinguishability Game

For every PPT "distinguishing" adversary A

$$|\Pr[\mathsf{MMInd} = 1] - \Pr[\text{random guess}]| = \mathsf{negl}(\lambda)$$

"Advantage"



New: Multi-msg Indistinguishability Game

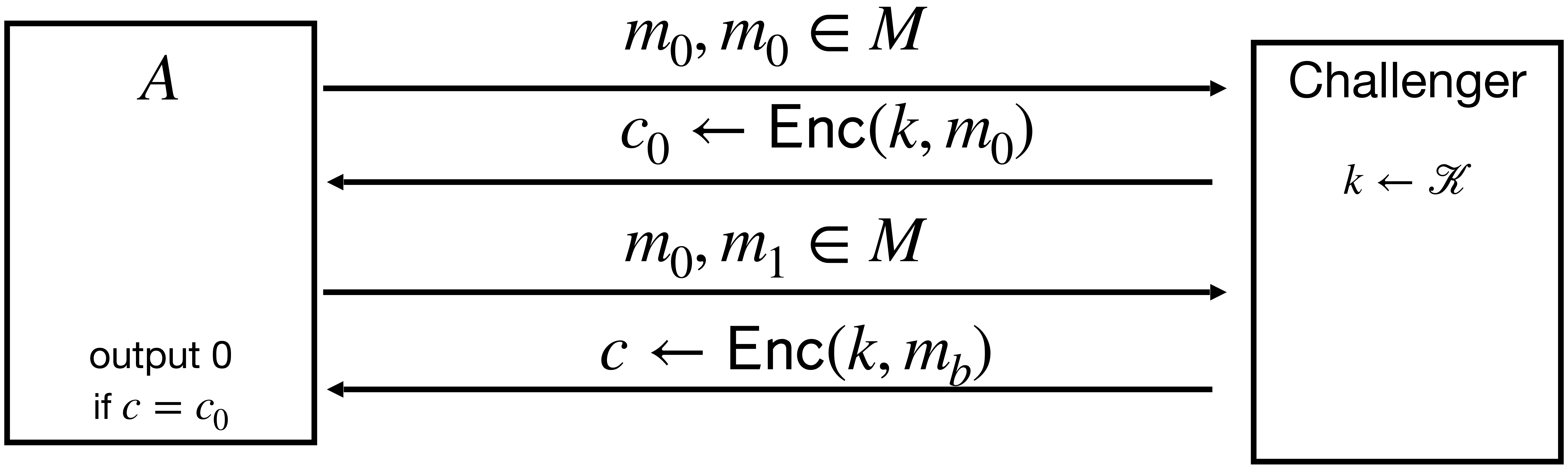
For every **PPT** A, there exists a negligible fn ε,

$$\left| \Pr\left[A(c_q) = b \;\middle|\; \begin{array}{r} k \leftarrow \mathcal{K}, b \leftarrow \{0,1\} \\ \text{For } i \text{ in } 1,\ldots,q: \\ (m_{i,0}, m_{i,1}) \leftarrow A(c_{i-1}) \\ c_i = \mathsf{Enc}(k, m_{i,b}) \end{array} \right] - \frac{1}{2} \right| < \varepsilon(n)$$

Indistinguishability under "Chosen-Plaintext Attack" IND-CPA

Stream Ciphers insecure under CPA

Problem: Enc(k, m) outputs same ciphertext for msg m.



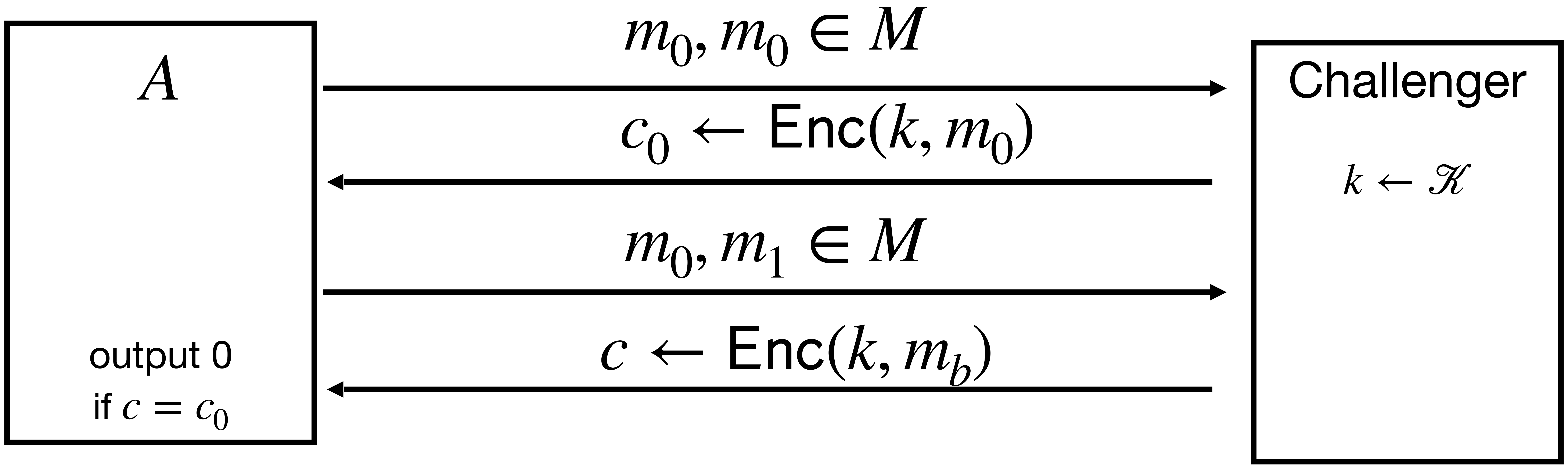
So what?

an attacker can learn that two encrypted files are the same, two encrypted packets are the same, etc.

Leads to significant attacks when message space is small

Stream Ciphers insecure under CPA

Problem: Enc(k, m) outputs same ciphertext for msg m.



If secret key is to be used multiple times

given the same plaintext message twice, encryption must produce different outputs.

Ideas for multi-message encryption

How to make encryption of same messages change?

- State? (e.g. counter of num msgs)
- Randomness?

Approach 1: Stateful encryption

$\text{Gen}(1^\lambda) \rightarrow k$:

1. Sample an n-bit string at random.

$\text{Enc}(k, m, \textbf{st}) \rightarrow c$:

1. Expand k to an $n + 1$-bit string using PRG: $s = G(k)$
2. Discard first ℓ bits of s to get s'
3. Set $\ell := \ell + 1$
4. Output $c = s' \oplus m$

$\text{Dec}(k, c) \rightarrow m$:

1. Repeat steps 1—4 of Enc
2. Output $m = s' \oplus c$

Is this secure for multiple messages?

Does this work?

Ans: Yes!

Exercise: reduce to PRG security

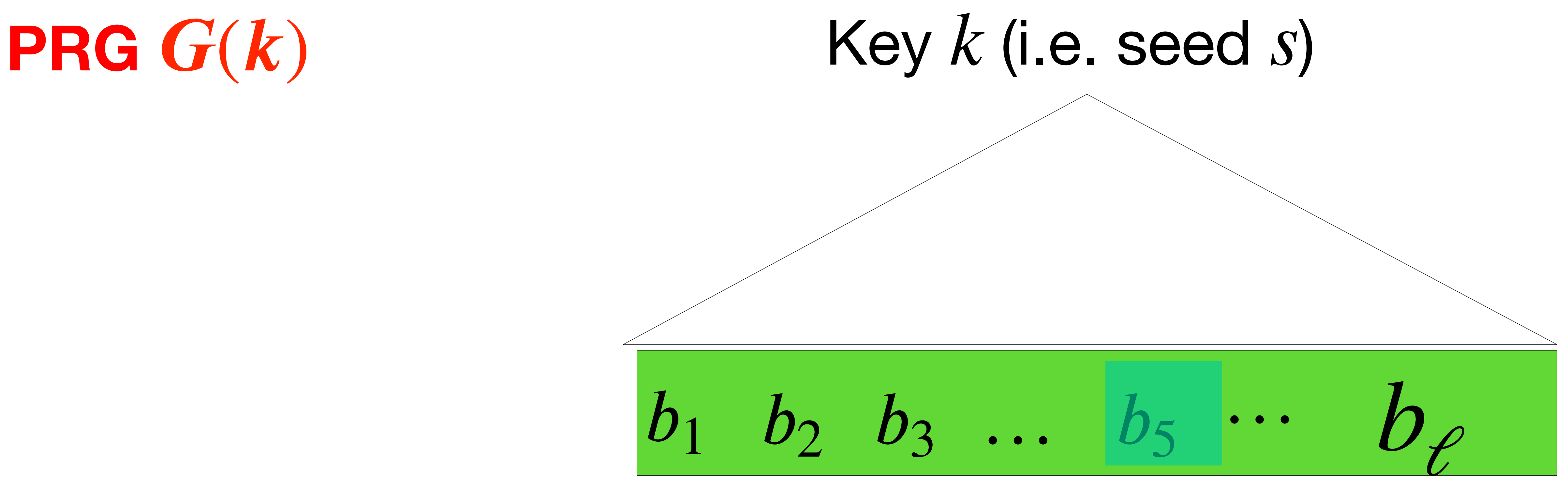
Pros:

- Relies on existing tools
- Generally fast

Cons:

- Must maintain counter of encrypted messages
- Must rerun PRG from start every time
- Sequential encryption/decryption

Problem: PRGs are sequential



- With a PRG, accessing the ℓ-th bit takes time ℓ.
- How to get efficient *random access* into output?
- That is, we want some function such that $F(\ell) = \ell$-th bit

New tool:

Pseudorandom Function

Background: Random function

- Let X be an input space, and Y be an output space.
- We will denote the set of all functions from X to Y as Fns$[X, Y]$
  - The number of such functions is $|Y|^{|X|}$.
- A random function from X to Y is a function that is sampled uniformly at random from Fns$[X, Y]$
- Important property of every random function f:
  - For each $x \in X$, $f(x)$ is uniformly and independently distributed in Y.

Stateful encryption w/ RFs

Gen$(1^n) \to k$: Sample a random function f and set $k := f$.

Enc$(k, m, \mathbf{st}) \to c$:

1. Interpret $\mathbf{st}$ as number ℓ of messages encrypted so far.
2. Output $c = f(\ell) \oplus m$

Dec$(k, c, \mathbf{st}) \to m$:

1. Interpret $\mathbf{st}$ as number ℓ of messages encrypted so far.
2. Output $m = f(\ell) \oplus c$

Does this work?

Ans: Yes!

Pros:

- Relies on existing tools
- Generally fast
- No need to run RF from start!

Cons:

- Must maintain counter of encrypted messages
- **How to store a random function?**

Problem: Random Functions can't be stored efficiently

A random function is a random mapping from X to Y.

Simplest representation: function table

What is the size of an arbitrary mapping?

$$|X|\log|Y|$$

For each x, $|Y|$ possible choices;
each choice has $\log|Y|$ bits representation

Problem: Random Functions can't be stored efficiently

For encryption, $|X|\log|Y|$ is too large!

Let's see why:

In our case, $|Y|$ is message length, e.g. $Y = \{0,1\}$, $|Y| = 2$.
if we encrypt, e.g., $|X| = 2^{20}$ 1-bit messages, our key is now 2^{20} bits, i.e. same as OTP!

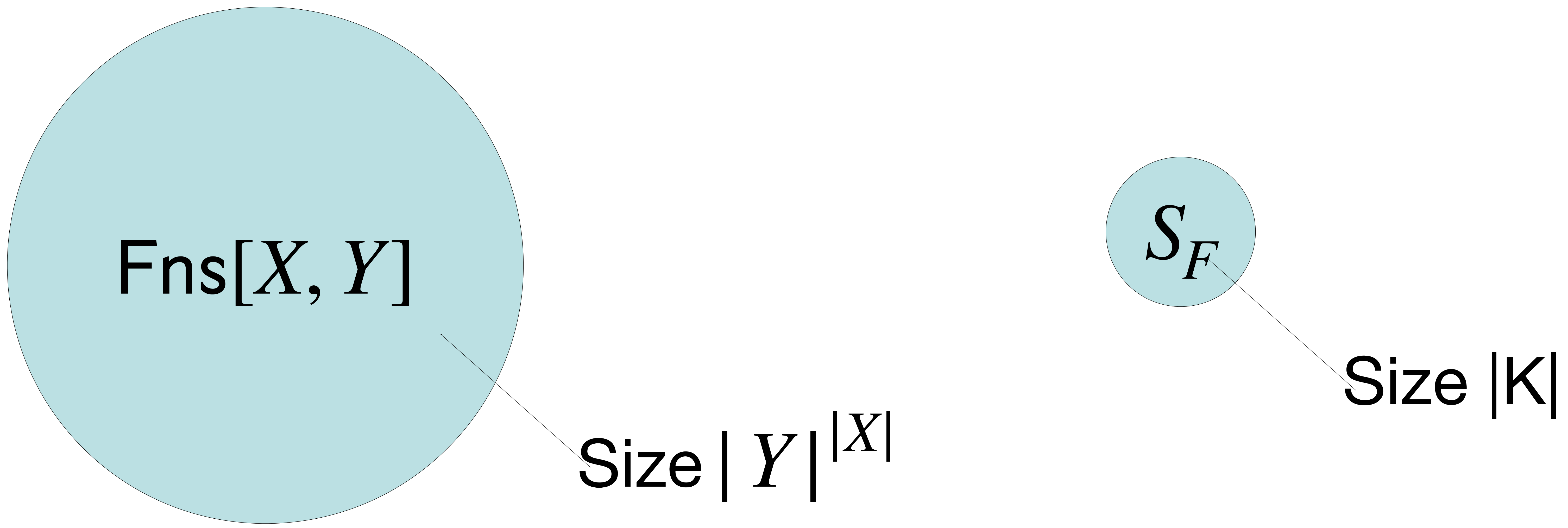
Also, $|Y|^{|X|}$ should be large (otherwise brute force possible: try all possible functions).

Solution: *Pseudo*random functions

- Replace a real random function with a function that *looks* random
- $S_F = \{F(k, \cdot) \mid k \in \mathcal{K}\} \subset \text{Fns}[X, Y]$
- Intuition: a PRF is **secure** if

  a random function in $\text{Fns}[X, Y]$ is indistinguishable from

  a random function in S_F

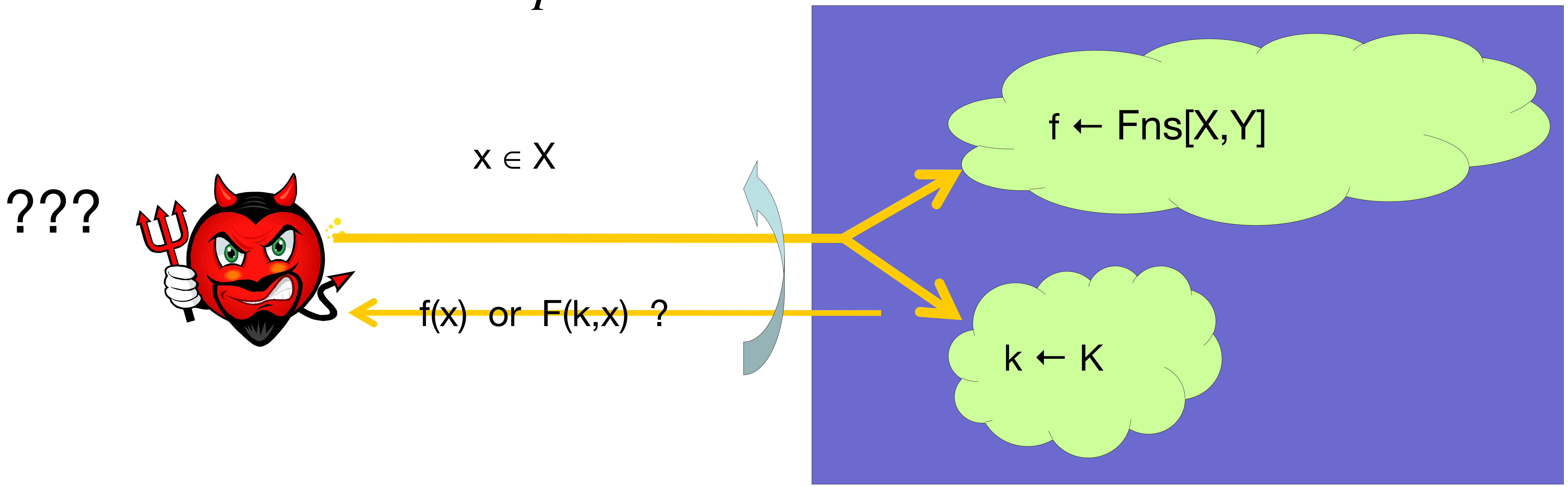


Secure PRFs

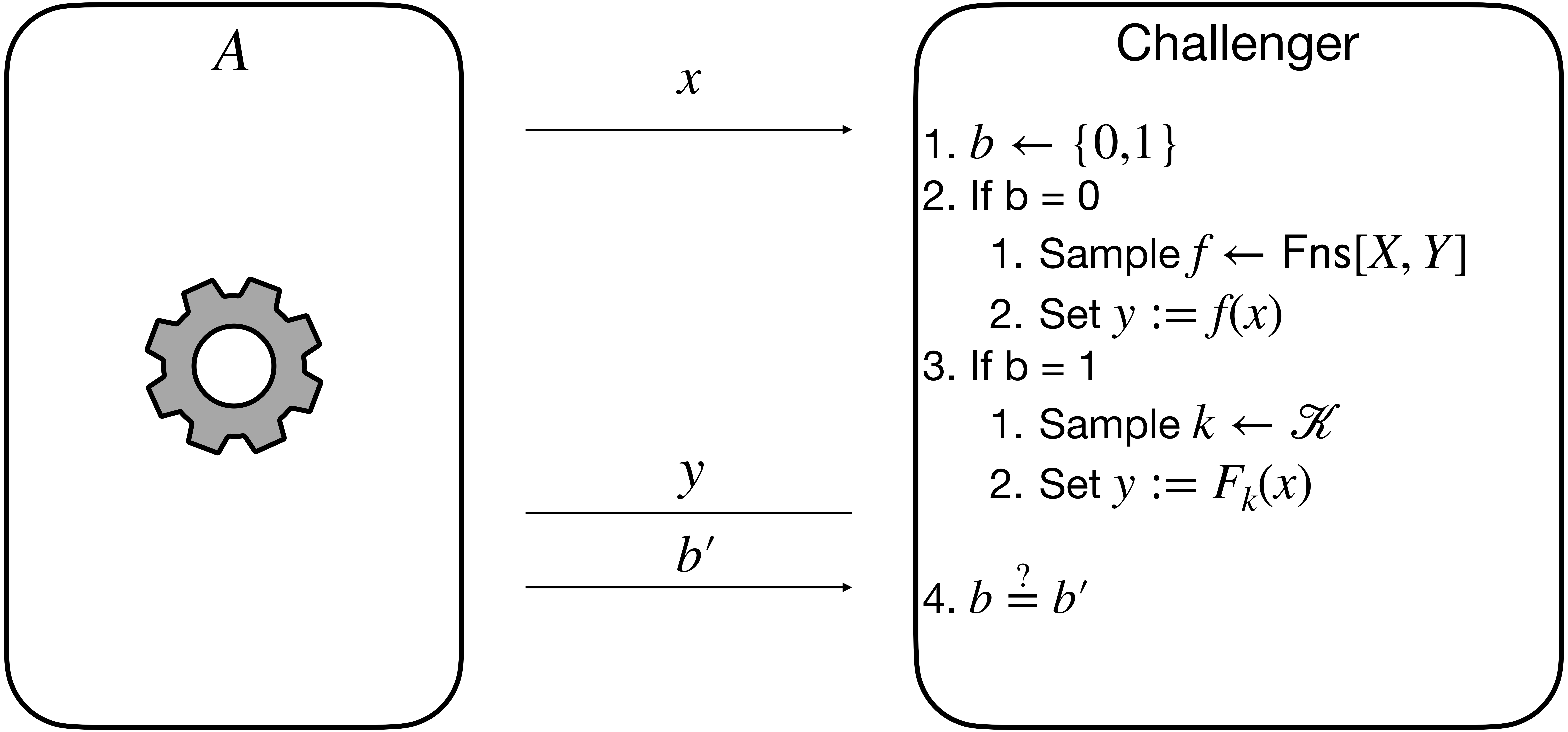
- Replace a real random function with a function that *looks* random
- $S_F = \{F(k, \cdot\,) \,|\, k \in \mathcal{K}\} \subset \text{Fns}[X, Y]$
- Intuition: a PRF is **secure** if

  a random function in $\text{Fns}[X, Y]$ is indistinguishable from

  a random function in S_F



PRF Security



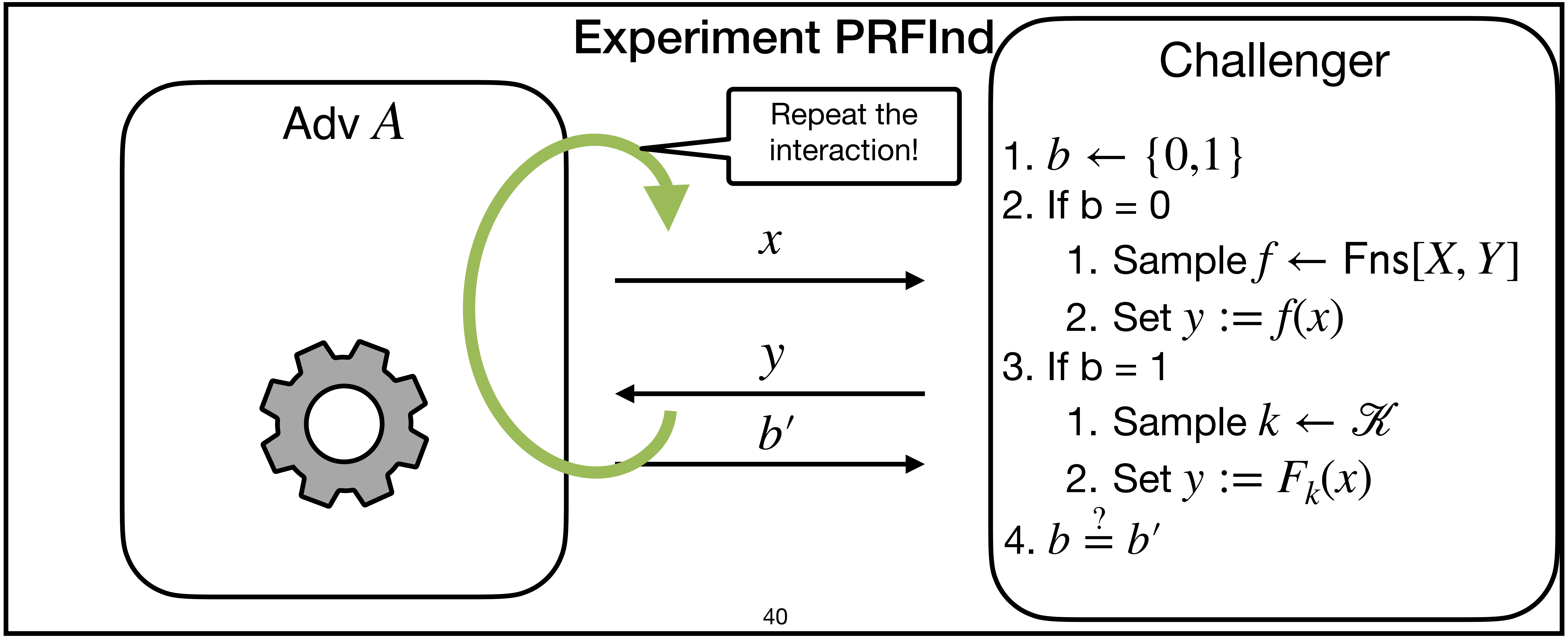
$$\Pr[b = b'] = 1/2 + \mathsf{negl}(n)$$

PRF Security Game

For every PPT "distinguishing" adversary A

$$|\Pr[\text{PRFInd} = 1] - \Pr[\text{random guess}]| = \text{negl}(\lambda)$$

"Advantage"



An example

- Let $K = X = \{0,1\}^n$.
- Consider the PRF: $\mathbf{F(k, x) = k \oplus x}$ defined over (K, X, X)
- Let's show that F is insecure:
- Adversary $\mathscr{A}$: (1) choose arbitrary $\mathbf{x_0 \neq x_1} \in \mathbf{X}$
- (2) query for $\mathbf{y_0 = f(x_0)}$ and $\mathbf{y_1 = f(x_1)}$
- (3) output `0' if $\mathbf{y_0 \oplus y_1 = x_0 \oplus x_1}$ , else `1'

$\Pr[\mathrm{EXP}(0) = 0] = 1$ $\Pr[\mathrm{EXP}(1) = 0] = 1/2^n$

$\Longrightarrow$ $\mathrm{Adv}_{\mathrm{PRF}}[\mathscr{A},F] = 1 - (1/2n)$ (not negligible)

PRFs → multi-message encryption

Ideas for multi-message encryption

- State? (e.g. counter of num msgs)
- Randomness?