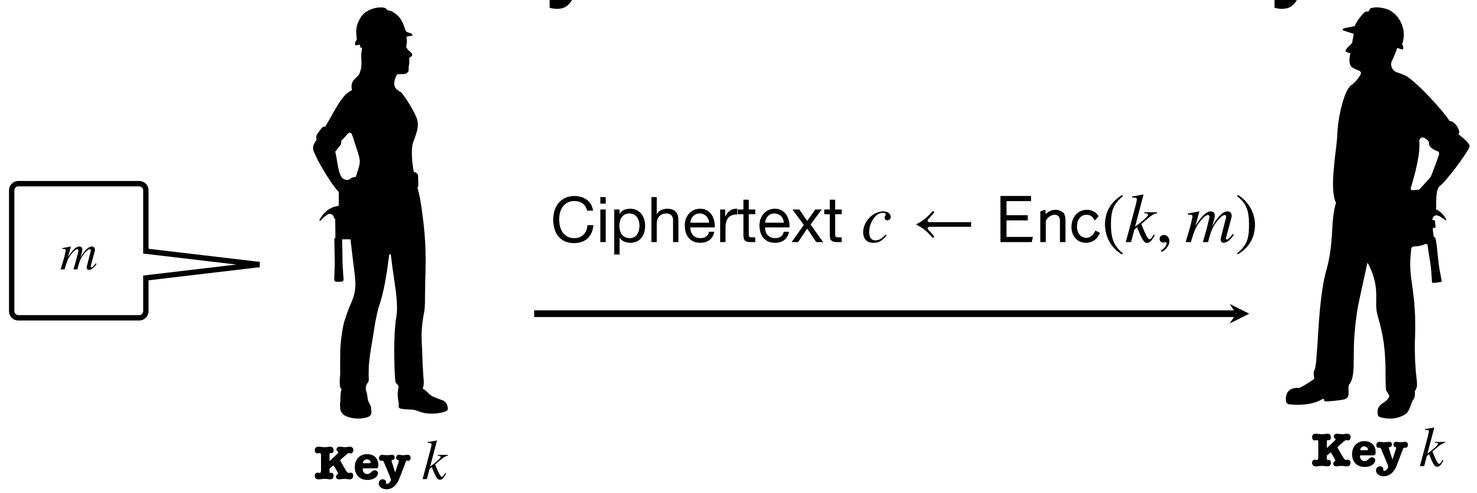
Key notion: Symmetric-Key Encryption



Three (possibly randomized) polynomial-time algorithms:

Key Generation Algorithm: $\mathrm{Gen}(1^\lambda) \rightarrow k$

Has to be randomized (why?)

Encryption Algorithm: $\mathrm{Enc}(k, m) \rightarrow c$

Decryption Algorithm: $\mathrm{Dec}(k, c) \rightarrow m$

Shannon's Perfect Secrecy Definition

$\forall m \in \mathcal{M}, \forall c \in \mathcal{C}, M$ is adversary's guess

$$\Pr[M = m \mid \mathsf{Enc}(\mathcal{K}, m) = c] = \Pr[M = m]$$

after before

✓ CT reveals no info about PT

But this def is difficult to work with:

How to prove that ciphertext reveals no info?

One-Time Pad

The One-time Pad Construction:

Gen: Choose an n-bit string k at random, i.e. $k \leftarrow \{0,1\}^n$

Enc(k, m) with $\mathcal{M} = \{0,1\}^n$: Output $c = m \oplus k$

Dec(k, c): Output $m = c \oplus k$

Perfect Secrecy has its Price

THEOREM: For any perfectly secure encryption scheme,

$$|\mathcal{K}| \geq |\mathcal{M}|$$

Computational Indistinguishability

World 0:

$k \leftarrow \mathcal{K}$

$c = \text{Enc}(k, m_0)$

World 1:

$k \leftarrow \mathcal{K}$

$c = \text{Enc}(k, m_1)$

Eve is arbitrary **PPT distinguisher**.

She needs to decide whether c came from World 0 or World 1.

For every **PPT** Eve, there exists a negligible fn ε, st for all m_0, m_1,

$$\left| \Pr\left[\text{Eve}(c) = 0 \,\middle|\, \begin{array}{c} k \leftarrow \mathcal{K} \\ c = \text{Enc}(k, m_0) \end{array}\right] - \Pr\left[\text{Eve}(c) = 1 \,\middle|\, \begin{array}{c} k \leftarrow \mathcal{K} \\ c = \text{Enc}(k, m_1) \end{array}\right] \right| = \varepsilon(n)$$

Negligible Functions

Functions that grow slower than 1/p(n) for any polynomial p.

Definition: A function $\varepsilon : \mathbb{N} \to \mathbb{R}$ is **negligible** if
for every polynomial function p,
there exists an n_0 s.t.
for all $n > n_0$:

$$\varepsilon(n) < \frac{1}{p(n)}$$

Question: Let $\varepsilon(n) = 1/n^{\log n}$. Is ε negligible?

Pseudorandom Generators

Informally: **Deterministic** Programs that stretch a "truly random" seed into a (much) longer sequence of **"seemingly random"** bits.

Q1: How to define "seemingly random"?

Q2: Can such a G exist?

PRG Def 1: Indistinguishability

Definition [Indistinguishability]:

A deterministic polynomial-time computable function

$$G : \{0,1\}^n \to \{0,1\}^m \text{ is a } \textbf{PRG} \text{ if:}$$

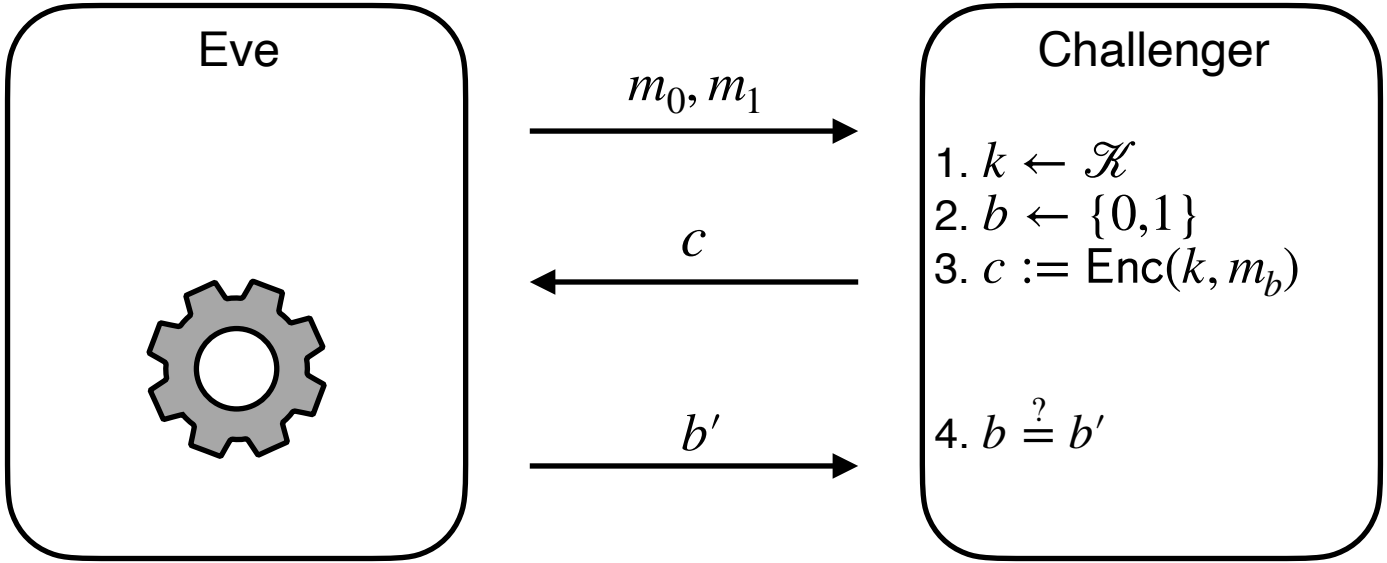
(a) It is expanding: $m > n$ and

(b) for every PPT algorithm D (called a distinguisher) if there is a negligible function ε such that:

$$\left| \Pr[D(G(U_n)) = 1] - \Pr[D(U_m) = 1] \right| = \varepsilon(n)$$

Notation: U_n (resp. U_m) denotes the random distribution on n-bit (resp. m-bit) strings; m is shorthand for $m(n)$.

Semantic Security



Semantic Security

For every **PPT** Eve, there exists a negligible fn ε such that

$$\Pr\left[\mathsf{Eve}(c) = b \;\middle|\; \begin{array}{r}(m_0, m_1) \leftarrow \mathsf{Eve} \\ k \leftarrow \mathscr{K} \\ b \leftarrow \{0,1\} \\ c := \mathsf{Enc}(k, m_b)\end{array}\right] < \frac{1}{2} + \varepsilon(n)$$

PRG $\Longrightarrow$ Semantically Secure Encryption

(or, How to Encrypt n+1 bits using an n-bit key)

- Gen$(1^k) \rightarrow k$:
  - Sample an n-bit string at random.
- Enc$(k, m) \rightarrow c$:
  - Expand k to an $n+1$-bit string using PRG: $s = G(k)$
  - Output $c = s \oplus m$
- Dec$(k, c) \rightarrow m$:
  - Expand k to an $n+1$-bit string using PRG: $s = G(k)$
  - Output $m = s \oplus c$

Correctness:

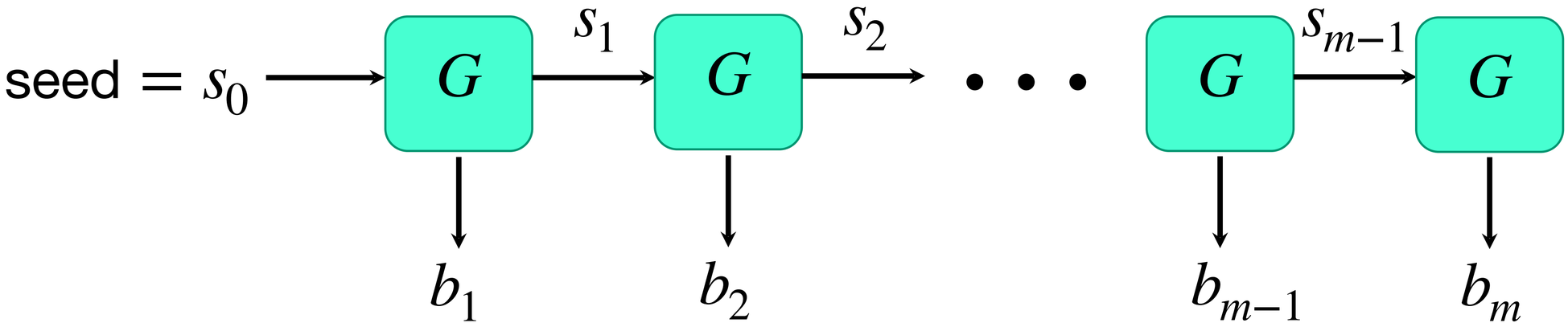
$Dec(k, c)$ outputs $G(k) \oplus c = G(k) \oplus G(k) \oplus m = m$

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!

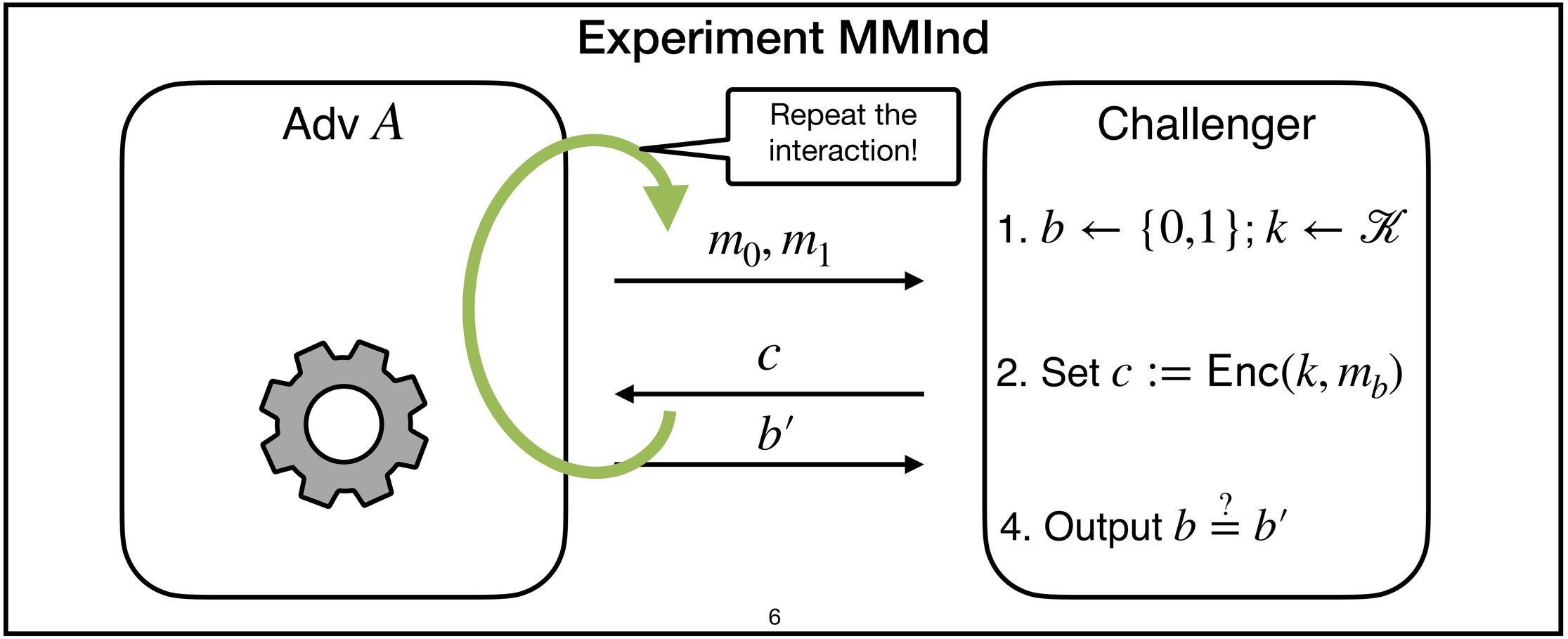
1. **Construct the steps:**
   A sequence of (polynomially-many) distributions $H_1, \ldots, H_{m-1}$ b/w the two target distributions.
2. **Show that it's easy to move between steps:**
   Argue that each pair of neighboring distributions are indistinguishable.
3. **Start moving:**
   Conclude that the 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.

New: Multi-msg Indistinguishability Game

For every PPT "distinguishing" adversary A

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

"Advantage"



New: Multi-msg Indistinguishability Game

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

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

Indistinguishability under “Chosen-Plaintext Attack” IND-CPA

Pseudorandom Functions (PRFs)

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(λ) = poly(n)
- #functions in $\mathcal{F}_\ell \leq 2^n$ (singly exponential in n)

Gen$\left(1^n\right)$: Generate a random n-bit key k.

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

PRF Security Game

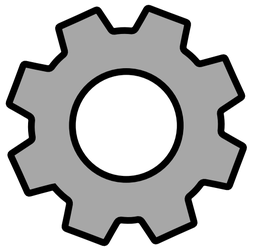
For every PPT "distinguishing" adversary A

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

"Advantage"

Experiment PRFInd

Adv A



Repeat the interaction!

x →

← y

b' →

Challenger

1. $b \leftarrow \{0,1\}$
2. If b = 0
   1. Sample $f \leftarrow \text{Fns}[X, Y]$
   2. Set $y := f(x)$
3. If b = 1
   1. Sample $k \leftarrow \mathcal{K}$
   2. Set $y := F_k(x)$
4. $b \stackrel{?}{=} b'$

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

Equivalent definition:

For every PPT "distinguishing" adversary A

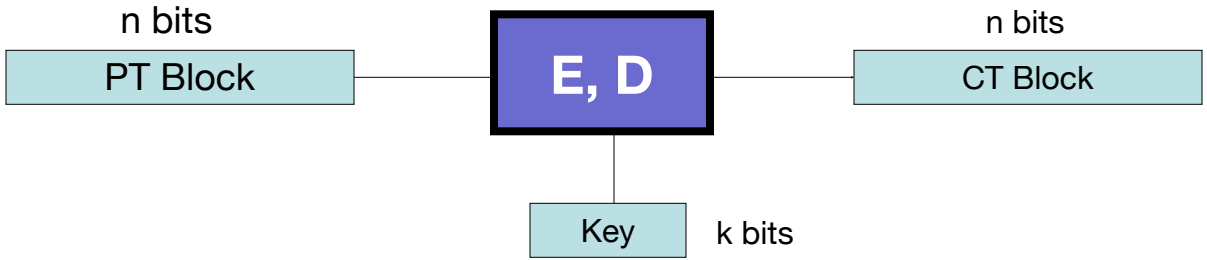
$$|\Pr[A^{F_k(\cdot)} \mid k \leftarrow \{0,1\}^n] - \Pr[A^{f(\cdot)} \mid f \leftarrow \mathsf{Funs}[X, Y]]| = \mathsf{negl}(\lambda)$$

Oracle or "opaque" access

"Advantage"

PRP/Block Cipher

A **block cipher** is a pair of efficient algs. (E, D):



Canonical examples:

1. **AES**: n=128 bits, k = 128, 192, 256 bits
2. **3DES**: n= 64 bits, k = 168 bits (historical)

Stateful encryption w/ PRFs

Gen$(1^n) \to k$:

Sample an n-bit string at random.

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

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

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

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

Randomized encryption w/ PRFs

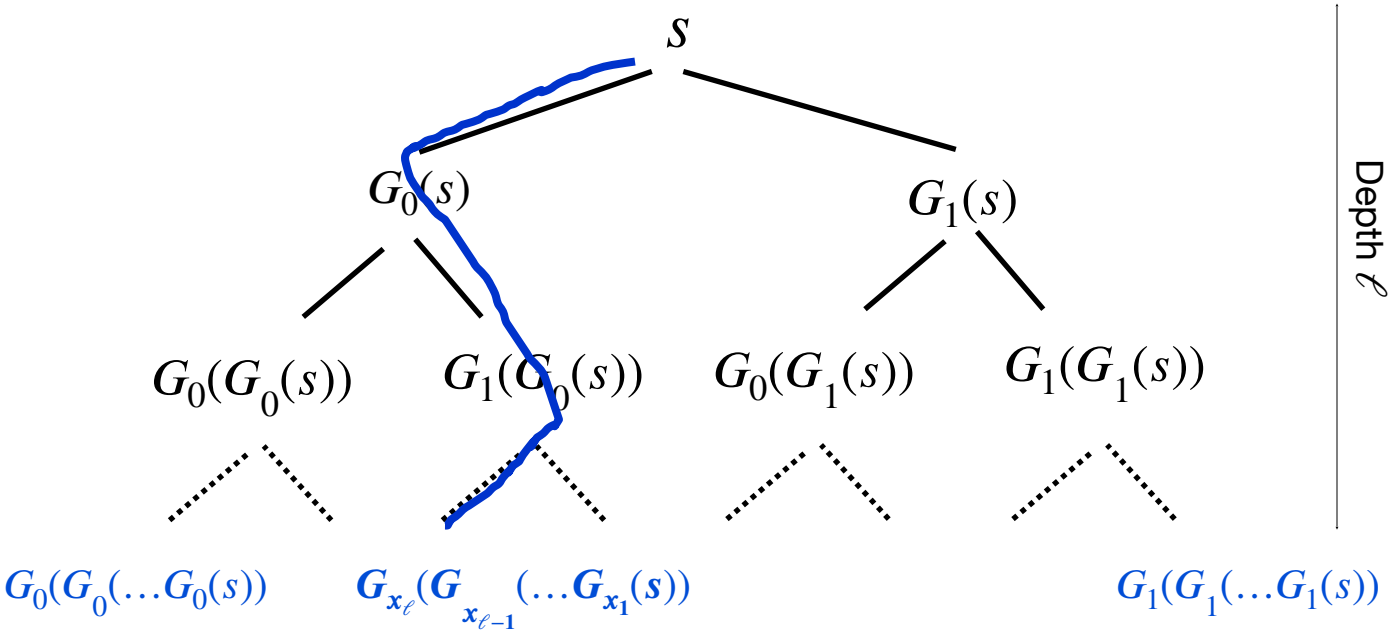
Gen(1^n): Generate a random n-bit key k that defines
$F_k : \{0,1\}^{\ell} \to \{0,1\}^m$

Enc(k, m): Pick a random r and
set the ciphertext $c := (r, y = F_k(r) \oplus m)$

Dec$(k, c = (r, y))$: Output $F_k(r) \oplus y$

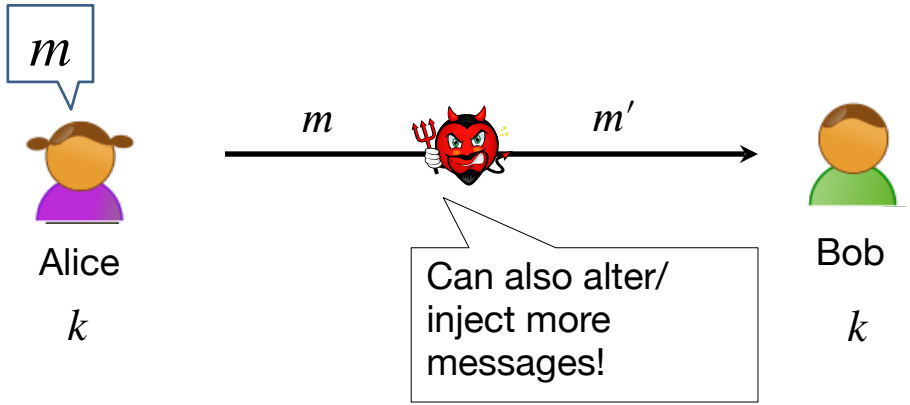
Goldreich-Goldwasser-Micali PRF

Construction: Let G(s) = $G_0(s) \,||\, G_1(s)$ where $G_0(s)$ and $G_1(s)$ are both n bits each.



Each path/leaf labeled by $x \in \{0,1\}^\ell$ corresponds to $f_s(x)$.

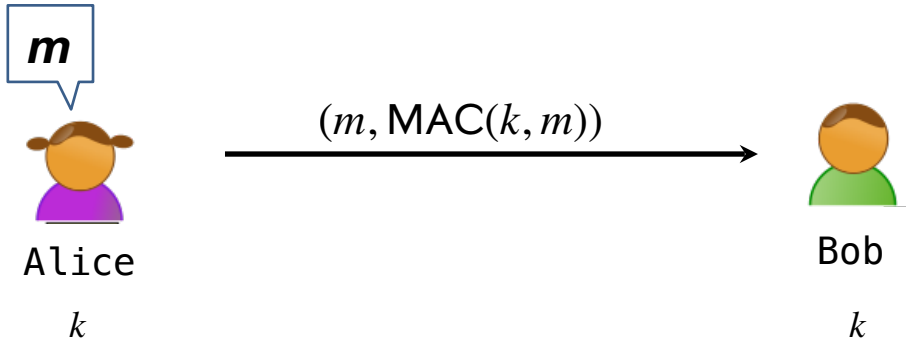
The authentication problem



This is known as a **man-in-the-middle attack.**

How can Bob check if the message is indeed from Alice?

Constructing a MAC

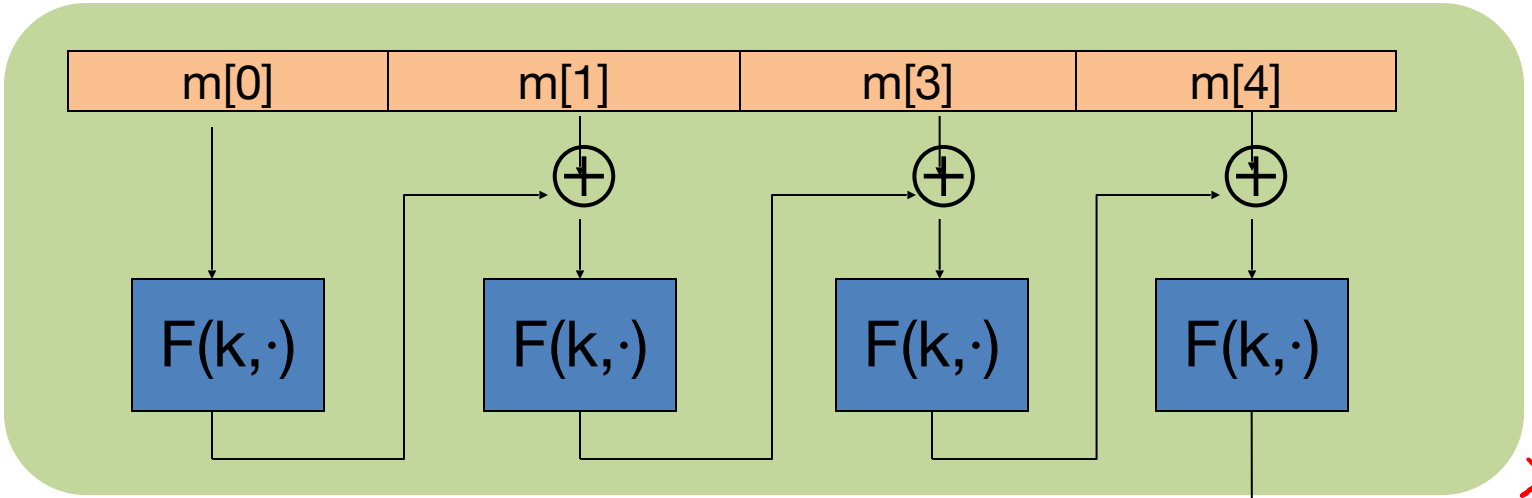


Gen(1^n): Produces a PRF key $k \leftarrow K$.
MAC$(k,\ m)$: Output $F_k(m)$.
Ver$(k,\ m,\ t)$: Accept if $F_k(m) = t$, reject otherwise.

Security: ??

Construction: encrypted CBC-MAC

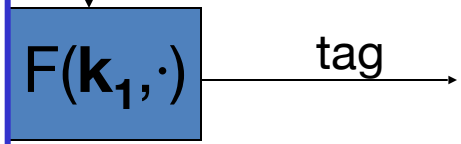
raw CBC



$X^{\leq L} = \bigcup_{i=1}^{L} X^i$

Let **F: K × X ⟶ X** be a PRP

Define new PRF **$F_{ECBC}: K^2 \times X^{\leq L} \longrightarrow X$**



Collision Resistance

Let $H : M \rightarrow T$ be a function (|M| >> |T|)

A **collision** for H is a pair $m_0, m_1 \in M$ such that:

$$H(m_0) = H(m_1) \text{ and } m_0 \neq m_1$$

A function H is **collision resistant** if for all efficient algs. A:

$$\mathbf{Adv_{CR}[A,H] = Pr[A \text{ outputs collision for } H]}$$

is negligible.

Example: SHA-256 (outputs 256 bits)

MACs from Collision Resistance

Let (MAC, V) be a MAC for short messages over (K,M,T) (e.g. AES)

Let $H : M^{\text{big}} \to M$ be a hash function

Def: $(\mathsf{MAC}^{\text{big}}, \mathsf{Ver}^{\text{big}})$ over (K, M^{big}, T) as:

$$\mathsf{MAC}^{\text{big}}(k, m) = \mathsf{MAC}(k, H(m)); \mathsf{Ver}^{\text{big}}(k, m, t) = V(k, H(m), t)$$

Thm: If MAC is a secure MAC and H is collision resistant
then $\mathsf{MAC}^{\text{big}}$ is a secure MAC.

Example: $\mathsf{MAC}(k,m) = \mathsf{AES}_{\text{2-block-cbc}}(k, \mathsf{SHA\text{-}256}(m))$ is a secure MAC.

Generic attack

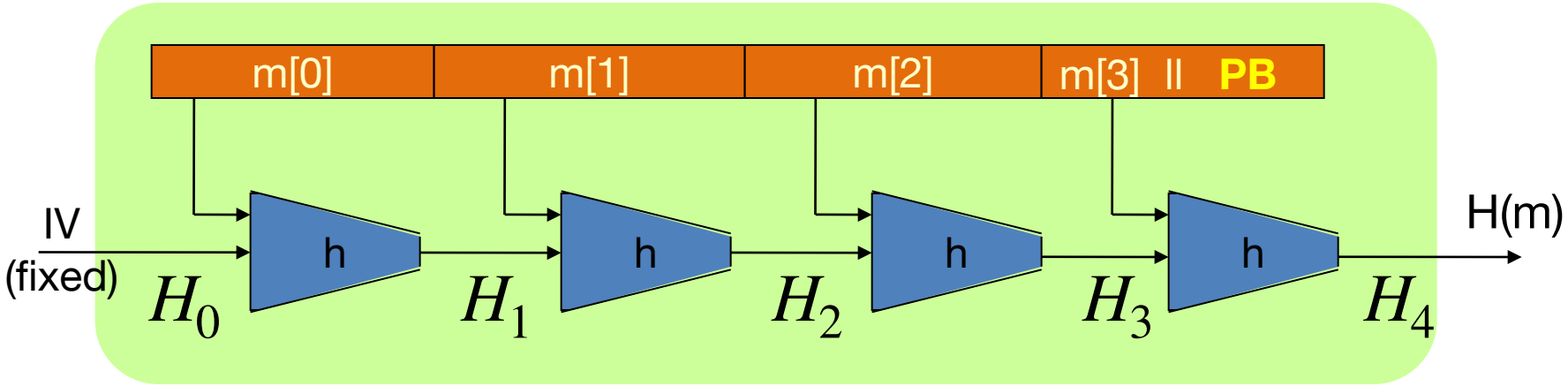
Algorithm:

1. Choose $2^{n/2}$ random messages in $\mathcal{M}$: $m_1, \ldots, m_{2^{n/2}}$ (distinct w.h.p)
2. For $i = 1,\ldots,2^{n/2}$ compute $t_i = H(m_i) \in \{0,1\}^n$
3. Look for a collision $(t_i = t_j)$. If not found, go back to step 1.

Expected number of iteration ≈ 2

Running time: **$O(2^{n/2})$** (space $O(2^{n/2})$)

The Merkle-Damgard iterated construction



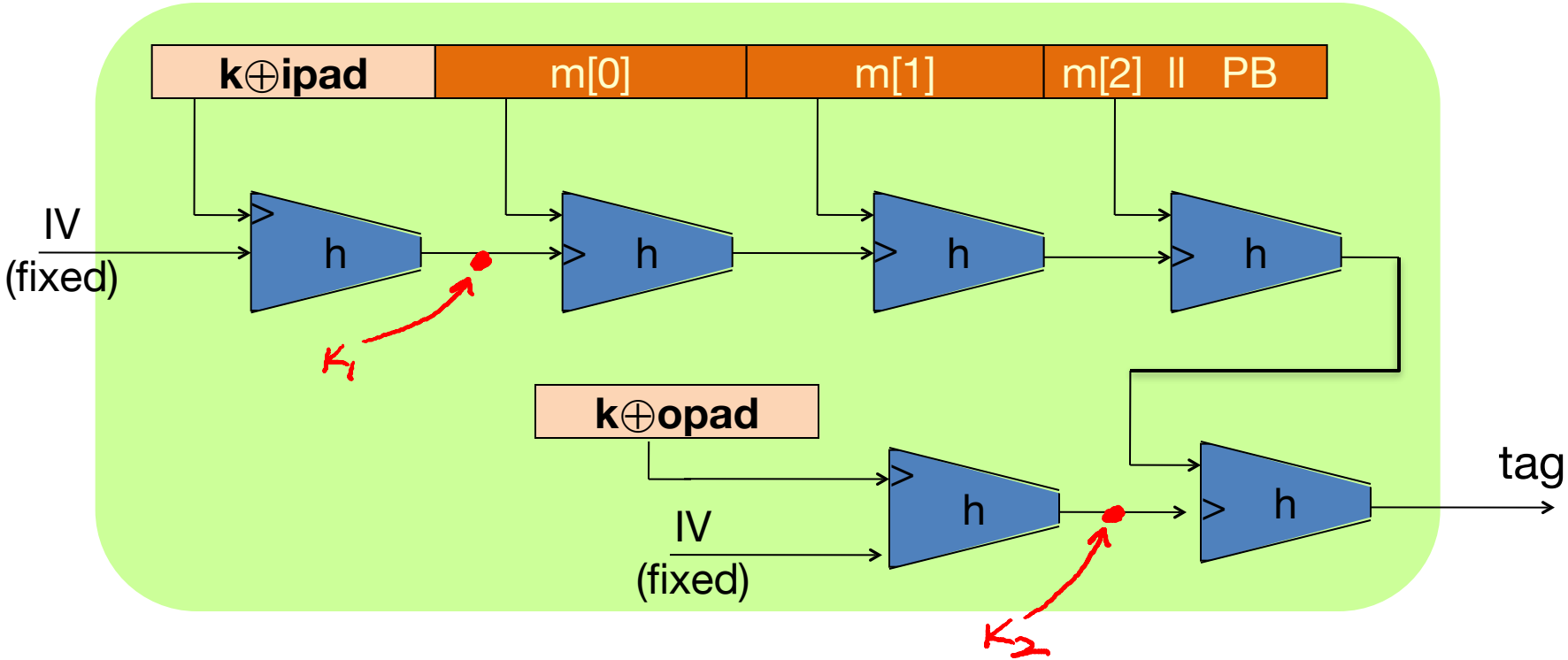
Given $h : T \times X \rightarrow T$ (compression function)

we obtain $H : X_{\leq L} \rightarrow T.$ H_i - chaining variables

PB: padding block 1000…0 || msg len (64 bits)

If no space for PB add another block

HMAC in pictures



Similar to the NMAC PRF.

main difference: the two keys k_1, k_2 are dependent