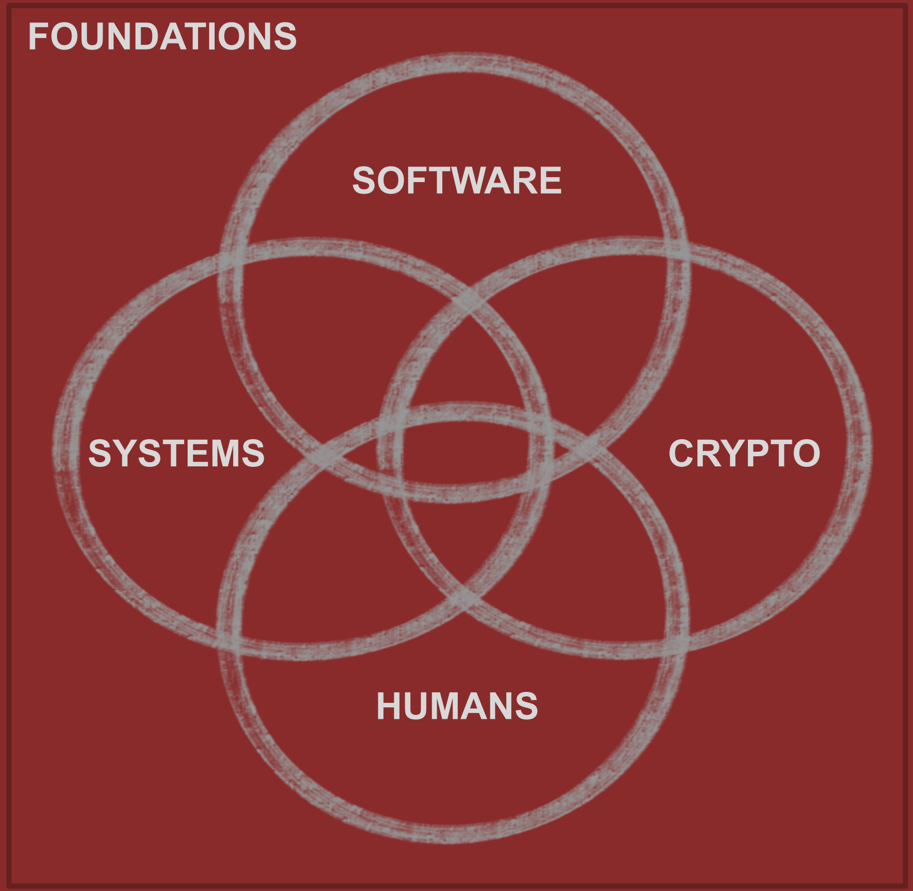
Διάλεξη #14 - Pseudorandom Functions & Permutations

Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

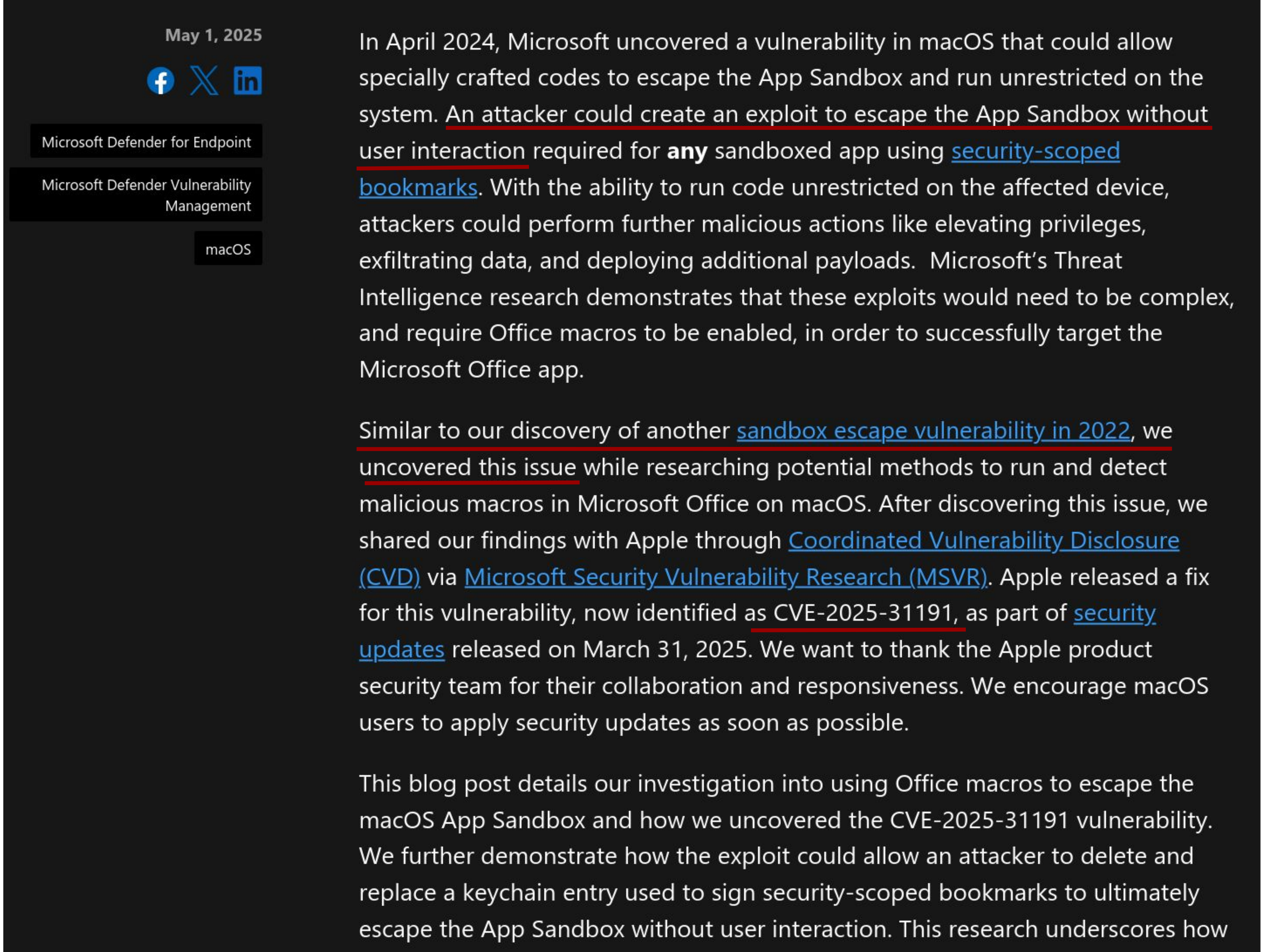
Εισαγωγή στην Ασφάλεια

Θανάσης Αυγερινός



Huge thank you to David Brumley from Carnegie Mellon University for the guidance and content input while developing this class (some slides from Dan Boneh @ Stanford!)

Security in the News



May 1, 2025

Microsoft Defender for Endpoint

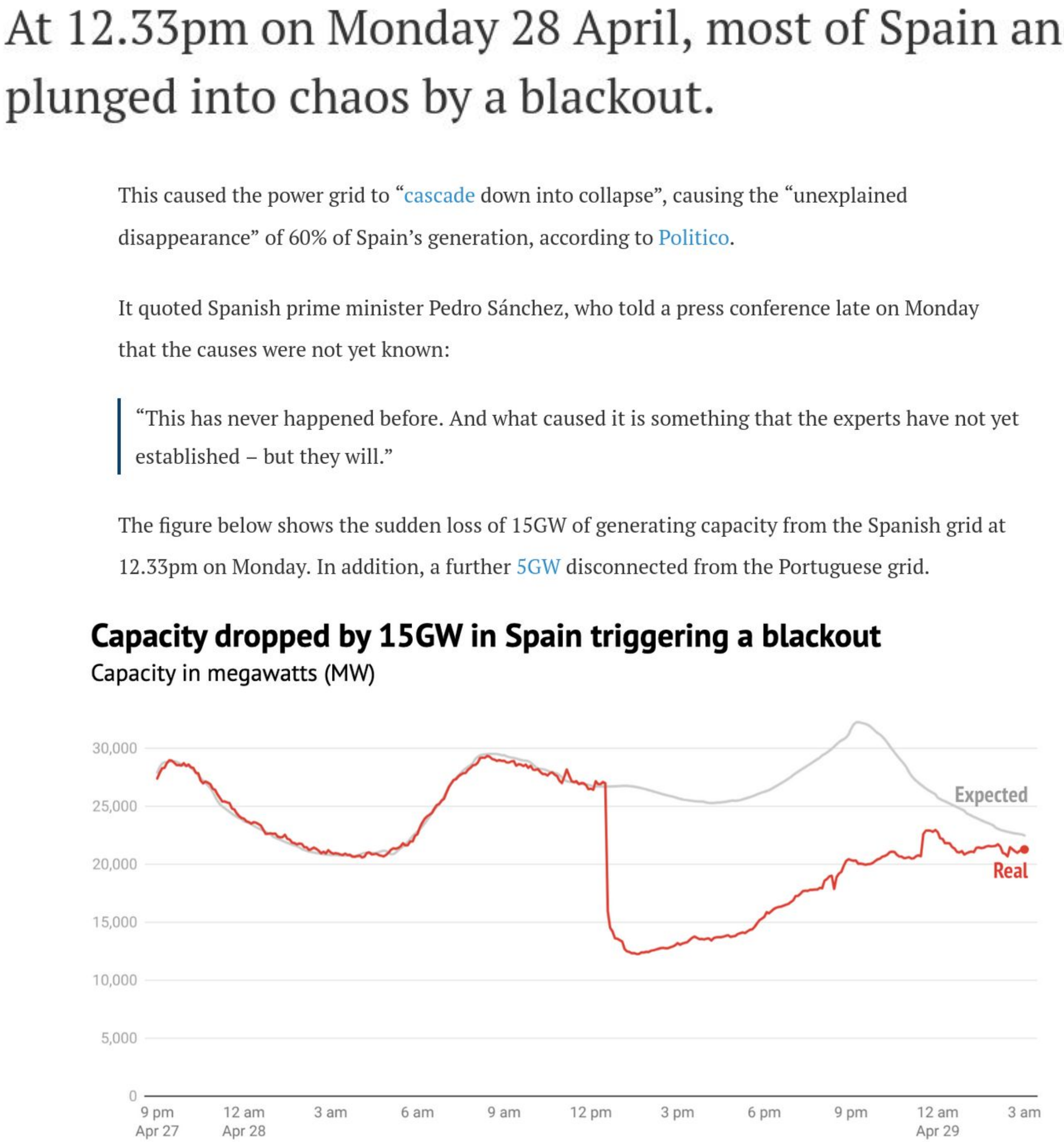
Microsoft Defender Vulnerability Management

macOS

In April 2024, Microsoft uncovered a vulnerability in macOS that could allow specially crafted codes to escape the App Sandbox and run unrestricted on the system. An attacker could create an exploit to escape the App Sandbox without user interaction required for **any** sandboxed app using security-scoped bookmarks. With the ability to run code unrestricted on the affected device, attackers could perform further malicious actions like elevating privileges, exfiltrating data, and deploying additional payloads. Microsoft's Threat Intelligence research demonstrates that these exploits would need to be complex, and require Office macros to be enabled, in order to successfully target the Microsoft Office app.

Similar to our discovery of another sandbox escape vulnerability in 2022, we uncovered this issue while researching potential methods to run and detect malicious macros in Microsoft Office on macOS. After discovering this issue, we shared our findings with Apple through Coordinated Vulnerability Disclosure (CVD) via Microsoft Security Vulnerability Research (MSVR). Apple released a fix for this vulnerability, now identified as CVE-2025-31191, as part of security updates released on March 31, 2025. We want to thank the Apple product security team for their collaboration and responsiveness. We encourage macOS users to apply security updates as soon as possible.

This blog post details our investigation into using Office macros to escape the macOS App Sandbox and how we uncovered the CVE-2025-31191 vulnerability. We further demonstrate how the exploit could allow an attacker to delete and replace a keychain entry used to sign security-scoped bookmarks to ultimately escape the App Sandbox without user interaction. This research underscores how



At 12.33pm on Monday 28 April, most of Spain and Portugal were plunged into chaos by a blackout.

This caused the power grid to “cascade down into collapse”, causing the “unexplained disappearance” of 60% of Spain’s generation, according to Politico.

It quoted Spanish prime minister Pedro Sánchez, who told a press conference late on Monday that the causes were not yet known:

> “This has never happened before. And what caused it is something that the experts have not yet established – but they will.”

The figure below shows the sudden loss of 15GW of generating capacity from the Spanish grid at 12.33pm on Monday. In addition, a further 5GW disconnected from the Portuguese grid.

Capacity dropped by 15GW in Spain triggering a blackout

Capacity in megawatts (MW)

You were hired as a consultant by the Spanish government to future proof the system against similar events in the future. What would you recommend and why?

Ανακοινώσεις / Διευκρινίσεις

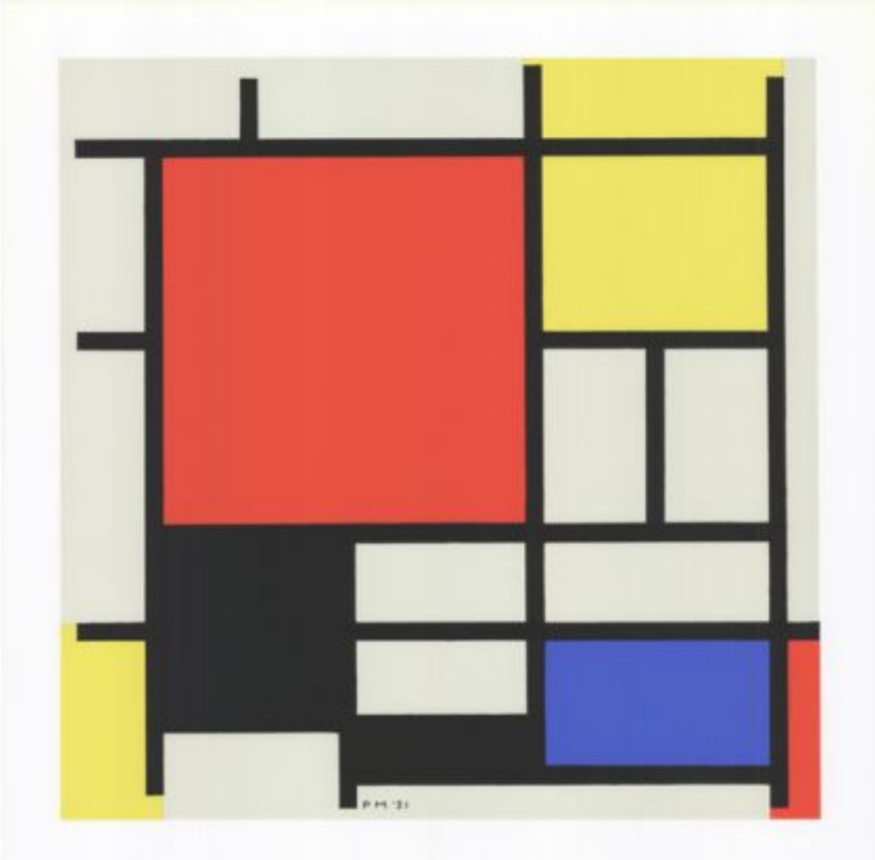
- Encryption and Encoding are not the same thing!

Την προηγούμενη φορά

- Problems with just OTP
- Randomness and Pseudorandomness
- Probability and Math Reminders
- PseudoRandom Functions (PRFs)
- PseudoRandom Permutations (PRPs)

Σήμερα

- PseudoRandom Functions (PRFs)
- PseudoRandom Permutations (PRPs)
- Block Ciphers
- Semantic Security
- Encryption Modes



Random Functions and Permutations

Thinking About Mathematical Functions

A function is just a mapping from inputs to outputs:

f_1

x	$f_1(x)$
1	4
2	13
3	12
4	1
5	7

f_2

x	$f_2(x)$
1	1
2	2
3	3
4	4
5	5

f_3

x	$f_3(x)$
1	12
2	3
3	7
4	8
5	10

. . .

~~Which function is *not* random?~~

Thinking About Mathematical Functions

A function is just a mapping from inputs to outputs:

f_1

x	$f_1(x)$
1	4
2	13
3	12
4	1
5	7

f_2

x	$f_2(x)$
1	1
2	2
3	3
4	4
5	5

f_3

x	$f_3(x)$
1	12
2	3
3	7
4	8
5	10

...

What is random is the way we *pick* a function

Participation Question

Consider all *functions* of the form F : X -> Y

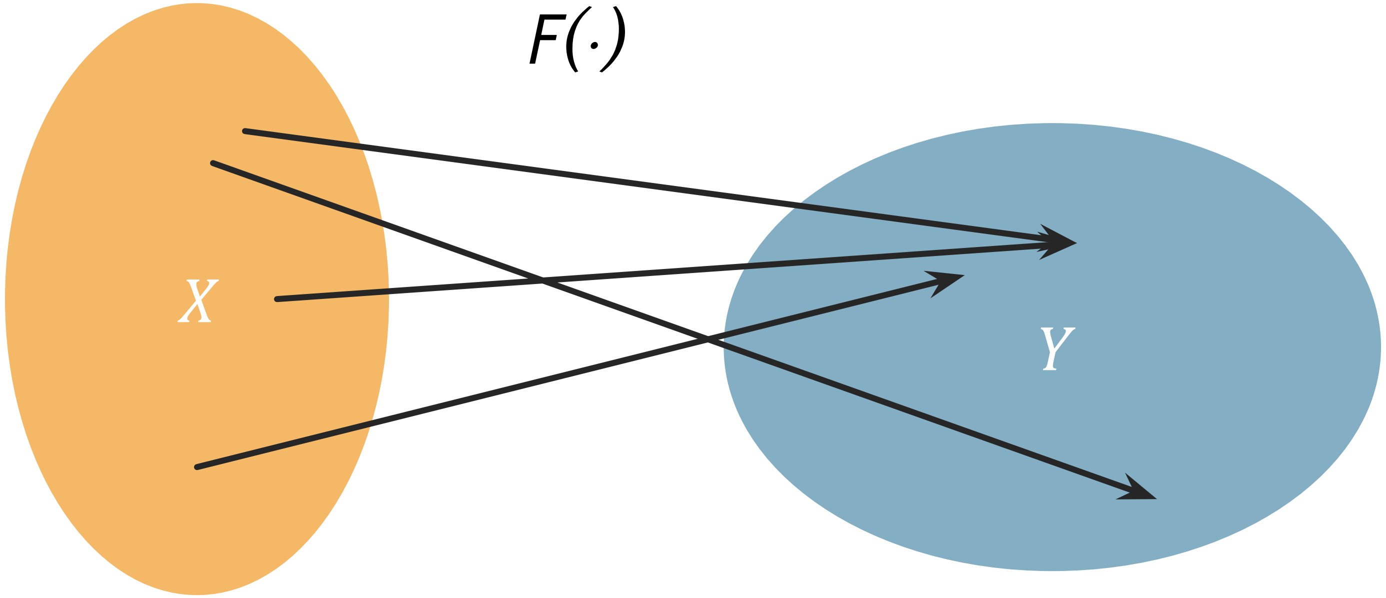
How many possible choices of F are there?

A. |X| * |Y|

B. |X|!

C. $|Y|^{|X|}$

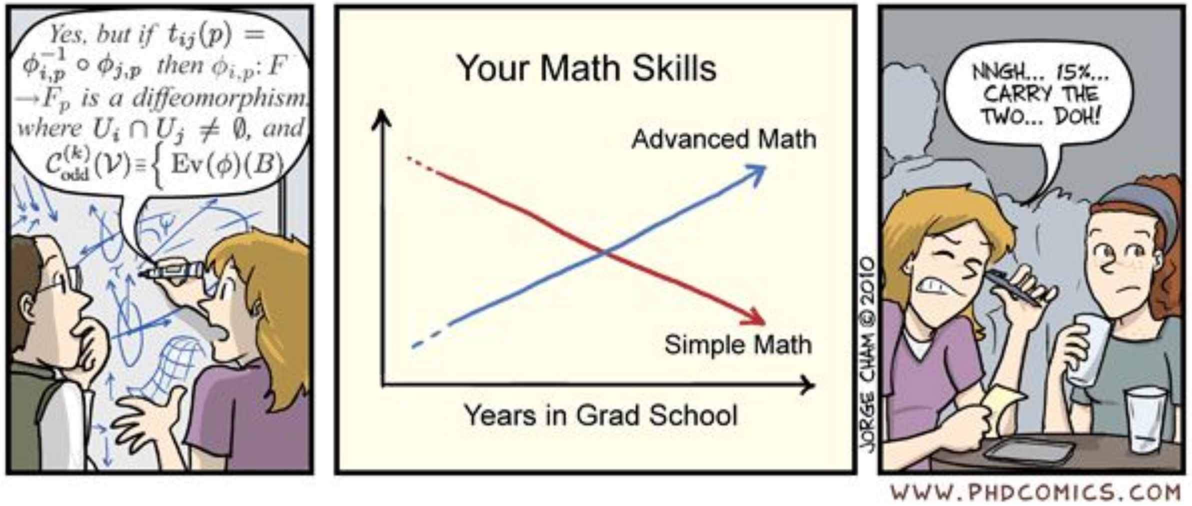
D. $|X|^{|Y|}$



Q: How many functions?

- X = {0, 1, 2} (Domain)
- Y = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} (Range)

10^3 = 1000 possible functions



Yes, but if $t_{ij}(p) = \phi^{-1}_{i,p} \circ \phi_{j,p}$ then $\phi_{i,p}: F \to F_p$ is a diffeomorphism where $U_i \cap U_j \neq \emptyset$, and $\mathcal{C}^{(k)}_{\text{odd}}(\mathcal{V}) \equiv \{ \text{Ev}(\phi)(B)$
Your Math Skills
Advanced Math
Simple Math
Years in Grad School
JORGE CHAM © 2010
NNGH... 15%... CARRY THE TWO... DOH!
WWW.PHDCOMICS.COM

Encryption with Functions

- Alice chooses f: $\{0,1\}^b \rightarrow \{0,1\}^b$ at random from *all possible functions* from $\{0,1\}^b$ to $\{0,1\}^b$
- Alice gives Bob the inverse, f^{-1}
- Given message m ∈ $\{0,1\}^b$:
  - Alice sends f(m) to Bob
  - Bob decrypts using f^{-1}

Participation Question

Is this a correct cipher?

A. Yes

B. No

C. I'm not sure

Correctness

$$\forall m \in M, k \in K : D(k, E(k,m)) = m$$

Better Encryption Scheme?

- Alice chooses f: $\{0,1\}^b$ -> $\{0,1\}^b$ at random from all possible ***permutations*** from $\{0,1\}^b$ to $\{0,1\}^b$
- Alice gives Bob the inverse, f^{-1}
- Given message $m \in \{0,1\}^b$:
  - Alice sends f(m) to Bob
  - Bob decrypts using f^{-1}

Participation Question

Is this a correct cipher?

A. Yes

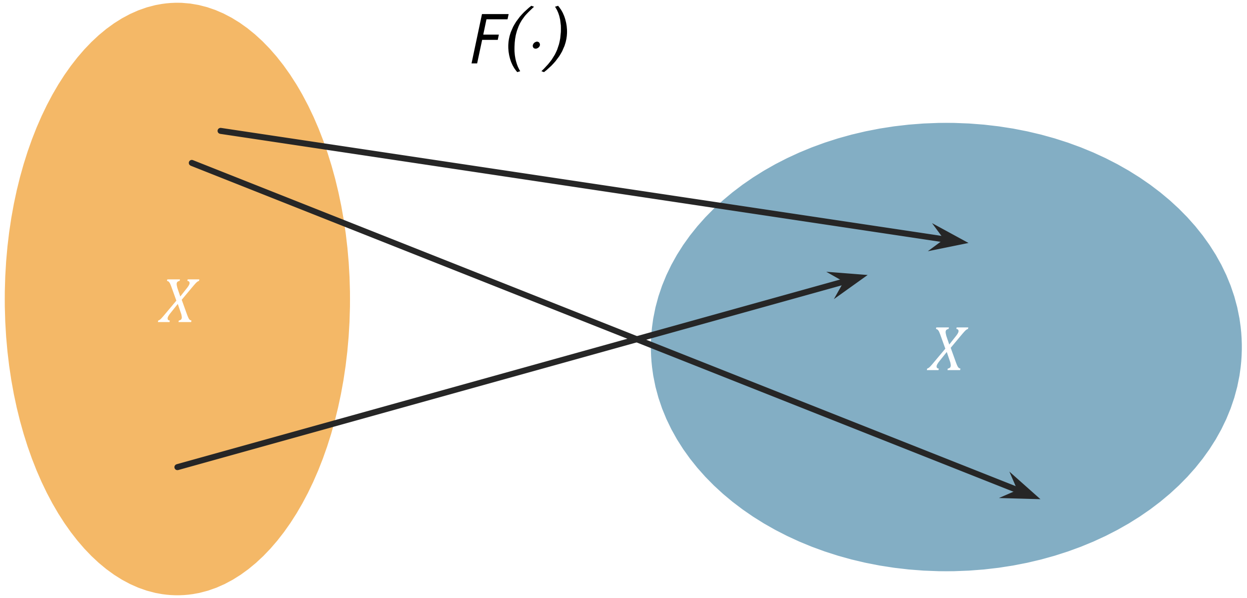
B. No

C. I'm not sure

Good cipher?

Permutations: Definition

- f: X -> X
- A permutation:
  - Is a function that maps (->) **every** element of its domain to **one** element of its range
  - Ever element in the range is **mapped to** by exactly one element of the domain
- In math terms: f is one-to-one
  - ∀x1, x2. f(x1) = f(x2) ⇔ x1 = x2
- Colloquially, f is a shuffling of X



Participation Question

Consider all *permutations* of the form F : X –> X

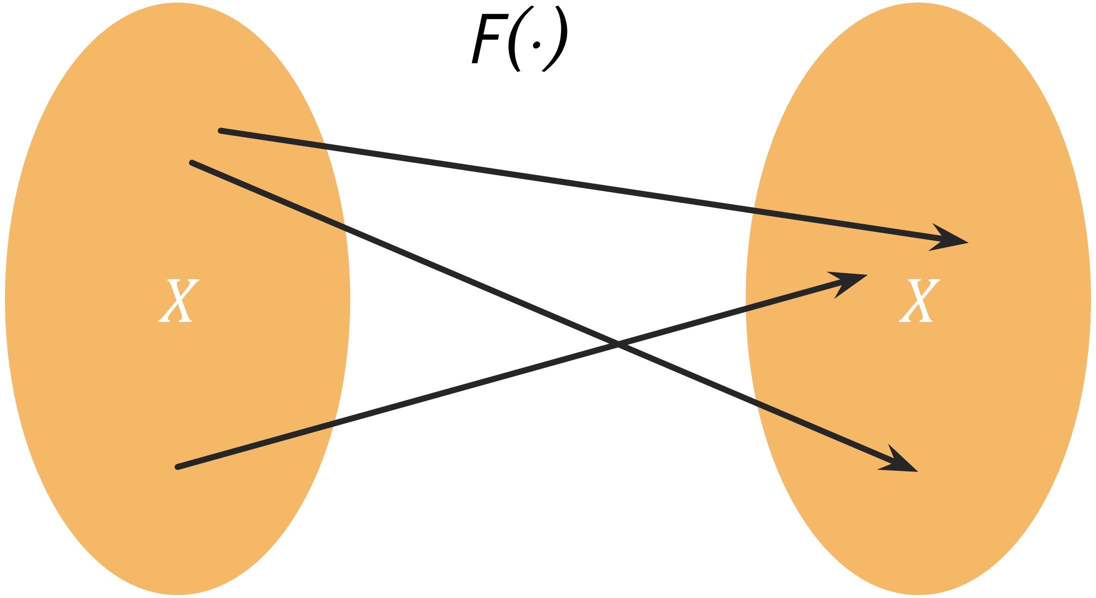
How many possible choices of F are there?

A. $2 * |X|$

B. $|X|^2$

C. $|X|! \cong (|x|/e)^{|X|}$

D. $|X|^{|X|}$



Better Encryption Scheme?

- Alice chooses f: $\{0,1\}^b$ –> $\{0,1\}^b$ at random from all possible ***permutations*** from $\{0,1\}^b$ to $\{0,1\}^b$
- Alice gives Bob the inverse, f^{-1}
- Given message $m \in \{0,1\}^b$:
  - Alice sends f(m) to Bob
  - Bob decrypts using f^{-1}

Did we bypass “bad news” theorem?

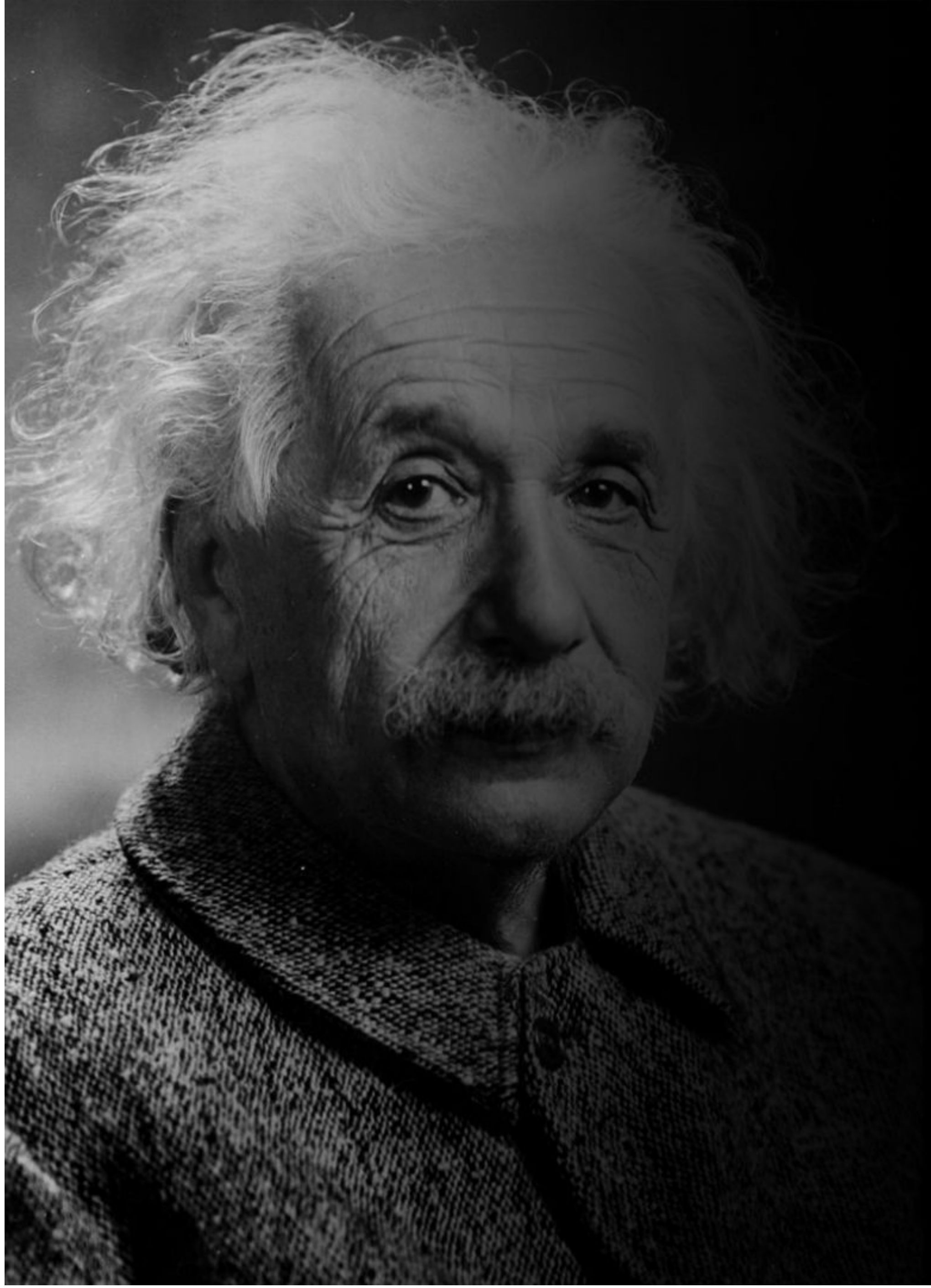
Computational security

The system can be *practically* (not perfectly) indecipherable

- Security is only preserved against efficient adversaries running in polynomial time and space, with access to randomness
- Adversaries can succeed with a very small probability (small enough that it is essentially impossible)
  - Ex: Probability of guessing a large randomly chosen value

"A scheme is secure if every Probabilistic Polynomial Time (PPT) adversary succeeds in breaking the scheme with only negligible probability"

PseudoRandom Functions and Permutations



God does not play dice with the universe.

ALBERT EINSTEIN

Pseudorandomness (overloaded term)

A **pseudorandom** sequence of numbers is one that appears to be statistically random, despite having been produced by a completely deterministic and repeatable process. Simply put, the problem is that many of the sources of randomness available to humans (such as rolling dice) rely on physical processes not readily available to computer programs.

In theoretical computer science, a distribution is **pseudorandom** against a class of adversaries if no adversary from the class can distinguish it from the uniform distribution with significant advantage. This notion of pseudorandomness is studied in computational complexity theory and has applications to cryptography.

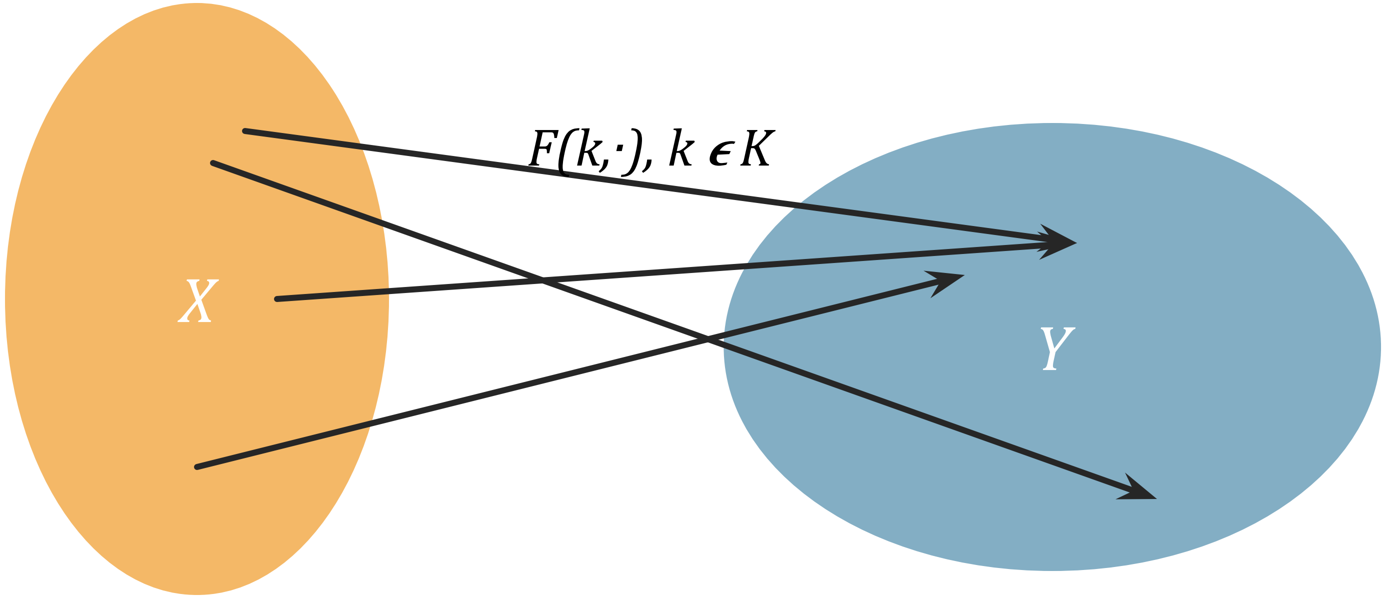
https://en.wikipedia.org/wiki/Pseudorandomness

PRFs

Pseudo Random Function (PRF) defined over (*K*, *X*, *Y*):

$$F : K \times X \to Y$$

such that there exists an "efficient" algorithm to evaluate *F(k,x)*



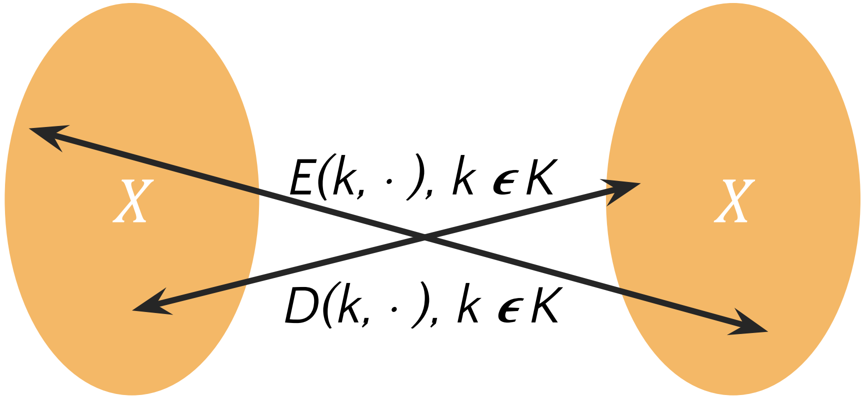
PRPs

Pseudo Random Permutation (PRP) defined over (K,X)

$$E : K \times X \to X$$

such that:

1. Exists “efficient” deterministic algorithm to evaluate *E(k,x)*
2. The function *E(k, ·)* is one-to-one
3. Exists “efficient” inversion algorithm *D(k,y)*



Let's Use Today's State-of-the-Art PRP (AES)

Question: what if we want to encrypt more than 128 bits?

PRFs and PRPs Are Still Math Functions!

- They map inputs to outputs
  - Conceptually, just a giant table
- They are not stateful!
- They are not randomized!

What if someone manages to invert our PRF/PRP function?

One Way Functions

In computer science, a **one-way function** is a function that is easy to compute on every input, but hard to invert given the image of a random input.

A function $f : \{0, 1\}^* \to \{0, 1\}^*$ is **one-way** if f can be computed by a polynomial-time algorithm, but any polynomial-time randomized algorithm F that attempts to compute a pseudo-inverse for f succeeds with negligible probability.

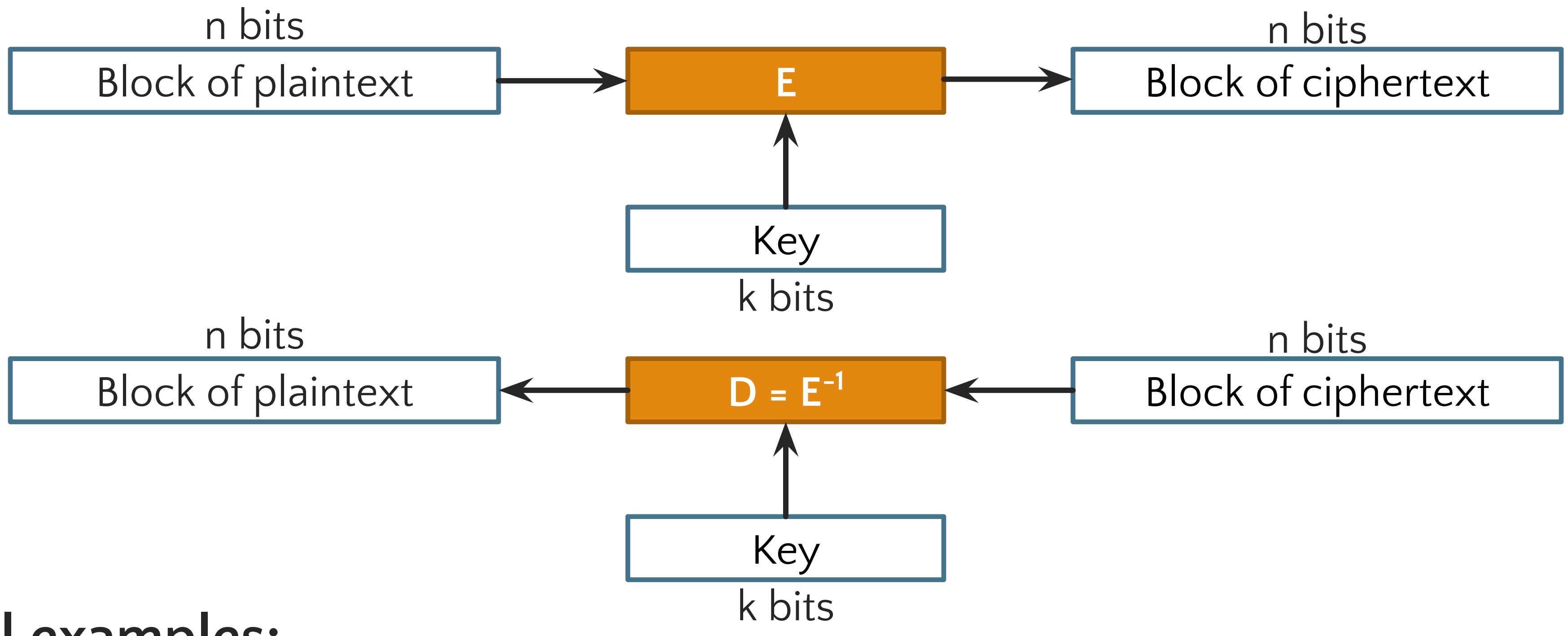
The existence of such one-way functions is still an open conjecture. Their existence would prove that the complexity classes P and NP are not equal, thus resolving the ***foremost unsolved question of theoretical computer science***.

https://en.wikipedia.org/wiki/One-way_function

Block Ciphers (aka practical PRPs)

Block Cipher ≈ PRP

Block ciphers are the crypto work horse



Canonical examples:

1. ~~DES: n = 64 bits k = 56 bits~~
2. ~~3DES: n = 64 bits k = 168 bits~~
3. AES: n = 128 bits k = 128, 192, 256 bits

History of Data Encryption Standard (DES)

- **1970s**: Horst Feistel designs Lucifer at IBM

  key = 128 bits, block = 128 bits

- **1973:** NBS asks for block cipher proposals.

  IBM submits variant of Lucifer.

- **1976:** NBS adopts DES as federal standard

  key = 56 bits, block = 64 bits

- **1997:** DES broken by exhaustive search

DES Challenge

Message	`The unkn`	`own mess`	`age is:`	`xxxxxxxx`
Ciphertext	c_1	c_2	c_3	c_4

Goal: find $k \in \{0,1\}^{56}$ s.t. $DES(k, m_i) = c_i$ for $i=1,2,3$

How expensive is it to reveal $DES^{-1}(k, c_4)$?

1976	DES adopted as federal standard		
1997	Distributed search	3 months	
1998	EFF deep crack	3 days	$250,000
1999	Distributed search	22 hours	
2006	COPACOBANA (120 FPGAs)	7 days	$10,000

⇒ 56-bit keys should not be used (128-bit key ⇒ 2^{72} days)

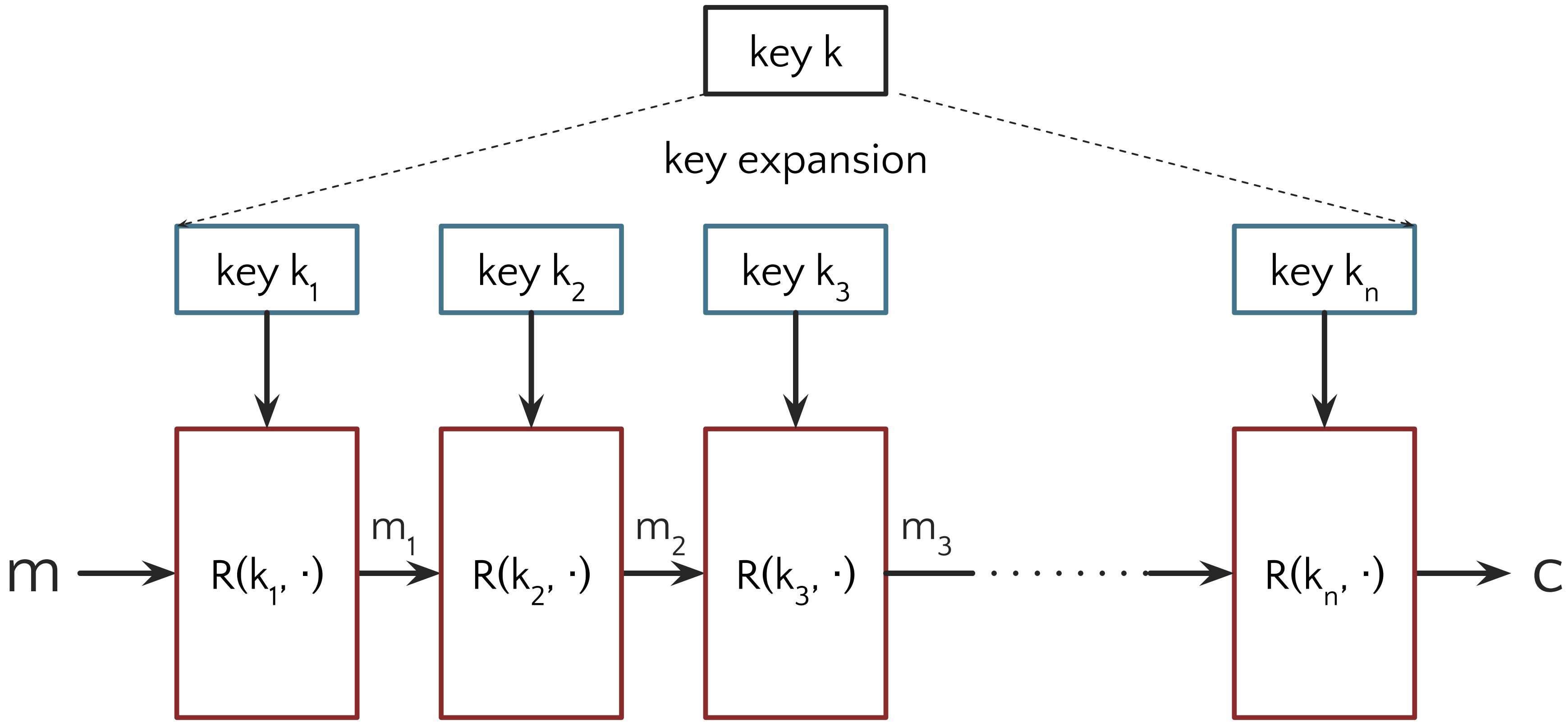
Advanced Encryption Standard (AES): The Process

- **1997**: DES broken by exhaustive search
- **1997**: NIST publishes request for proposal
- **1998**: 15 submissions
- **1999**: NIST chooses 5 finalists
- **2000**: NIST chooses Rijndael as AES
  (developed by Daemen and Rijmen at K.U. Leuven, Belgium)

Key sizes: 128, 192, 256 bits

Block size: 128 bits

Block Ciphers Built by Iteration

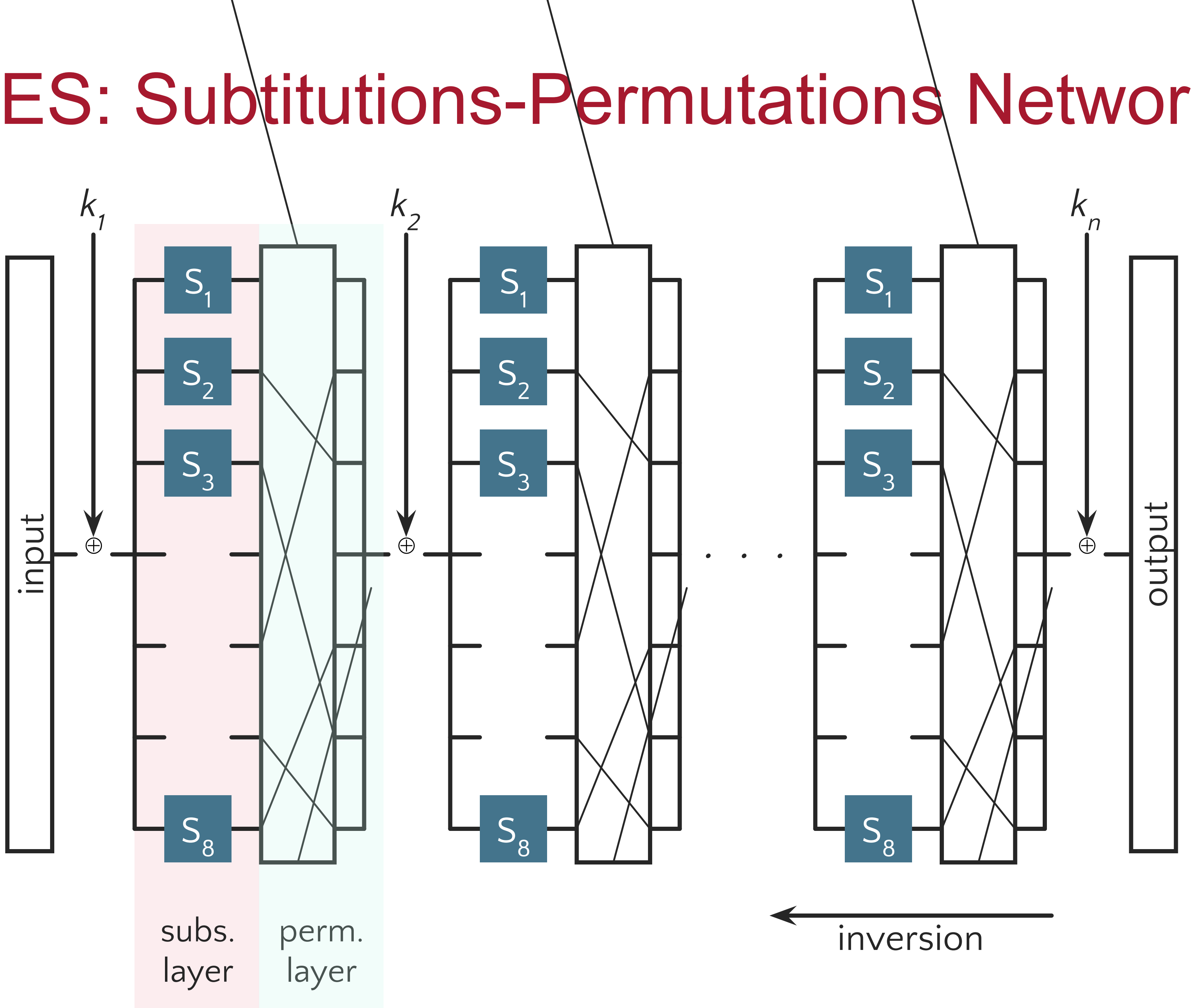


R(k, m) is called a *round function*

invoked up to n times

Ex: DES (n=16), 3DES (n=48), AES128 (n=10)

AES: Subtitutions-Permutations Network

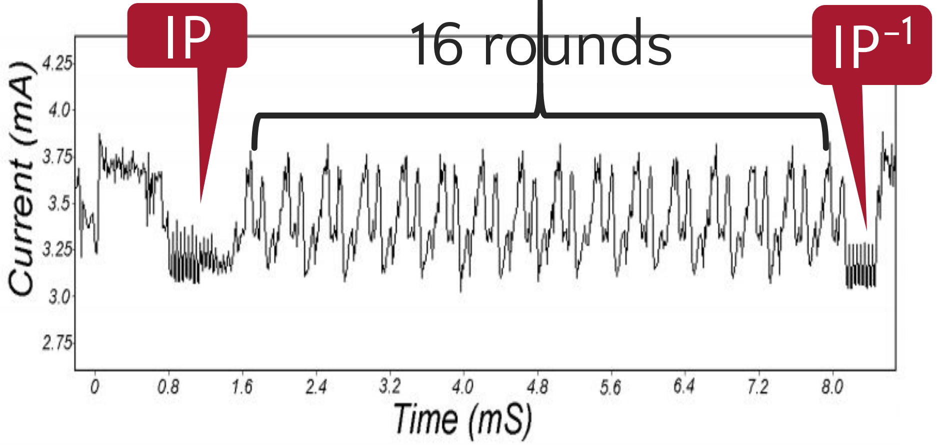


Attacks on the Implementation

1. Side channel attacks:

   – Measure **time** to do enc/dec, measure **power** for enc/dec

smartcard



[Kocher, Jaffe, Jun, 1998]

Card is doing DES

2. Fault attacks:

   – Computing errors in the last round expose the secret key k

⇒ never implement crypto primitives yourself …

Can We Encrypt Using Block Ciphers?

- Is a block cipher a secure encryption algorithm?
- Are they useful?

Semantic Security

Goldwasser and Micali, Turing Award 2012

What Is a Secure Encryption Alg.?

Attacker's abilities: obtains one ciphertext (for now)

Attempt #1: Attacker cannot recover key

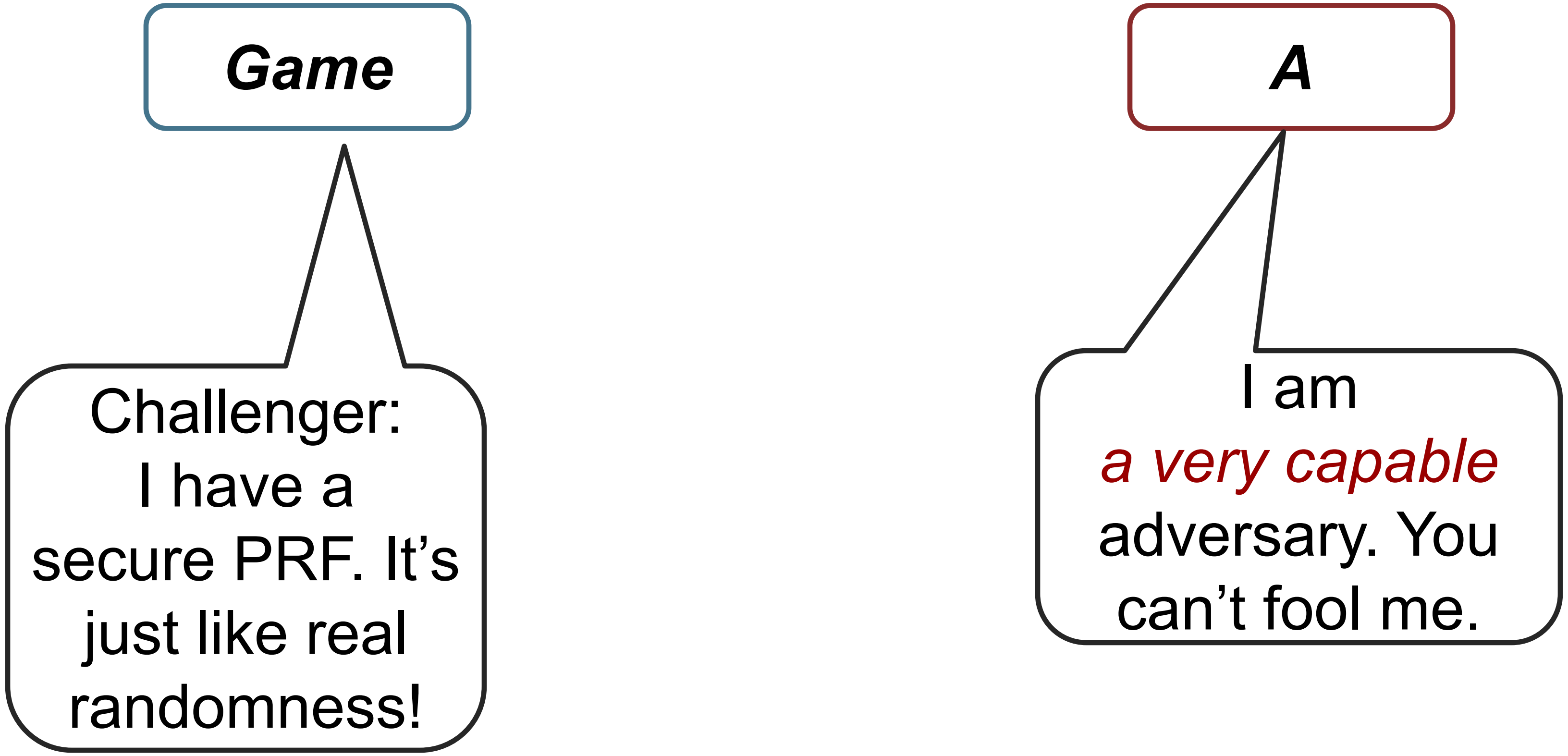
Insufficient: Consider $E(k,m) = m$

Attempt #2: Attacker cannot recover all of plaintext

Insufficient: Consider $E(k,m_0 \mid\mid m_1) = m_0 \mid\mid F(k,m_1)$

Recall Shannon's Intuition:
c (output of E) should reveal no information about *m*

Defining Security: Adversarial Indistinguishability Game



Game

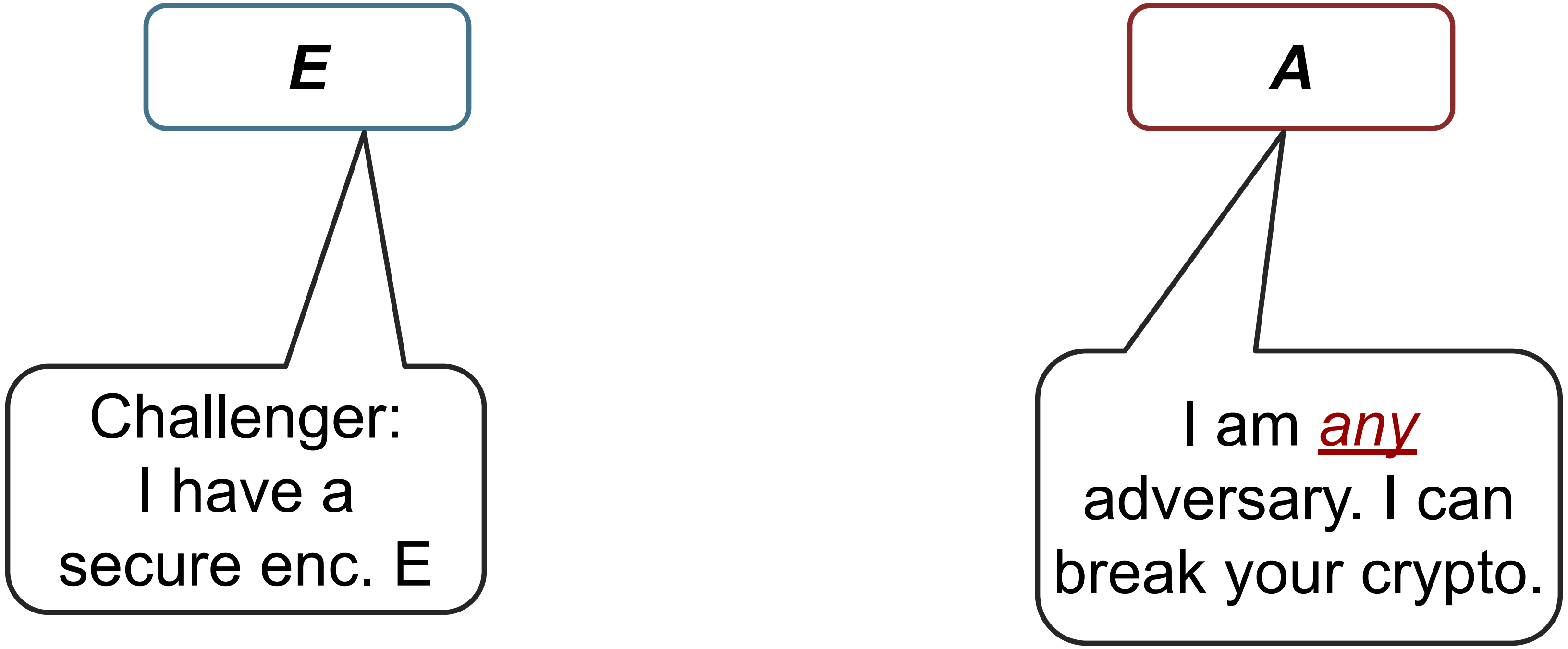
Challenger: I have a secure PRF. It’s just like real randomness!

A

I am *a very capable* adversary. You can’t fool me.

Security Games

Adversarial Indistinguishability Game



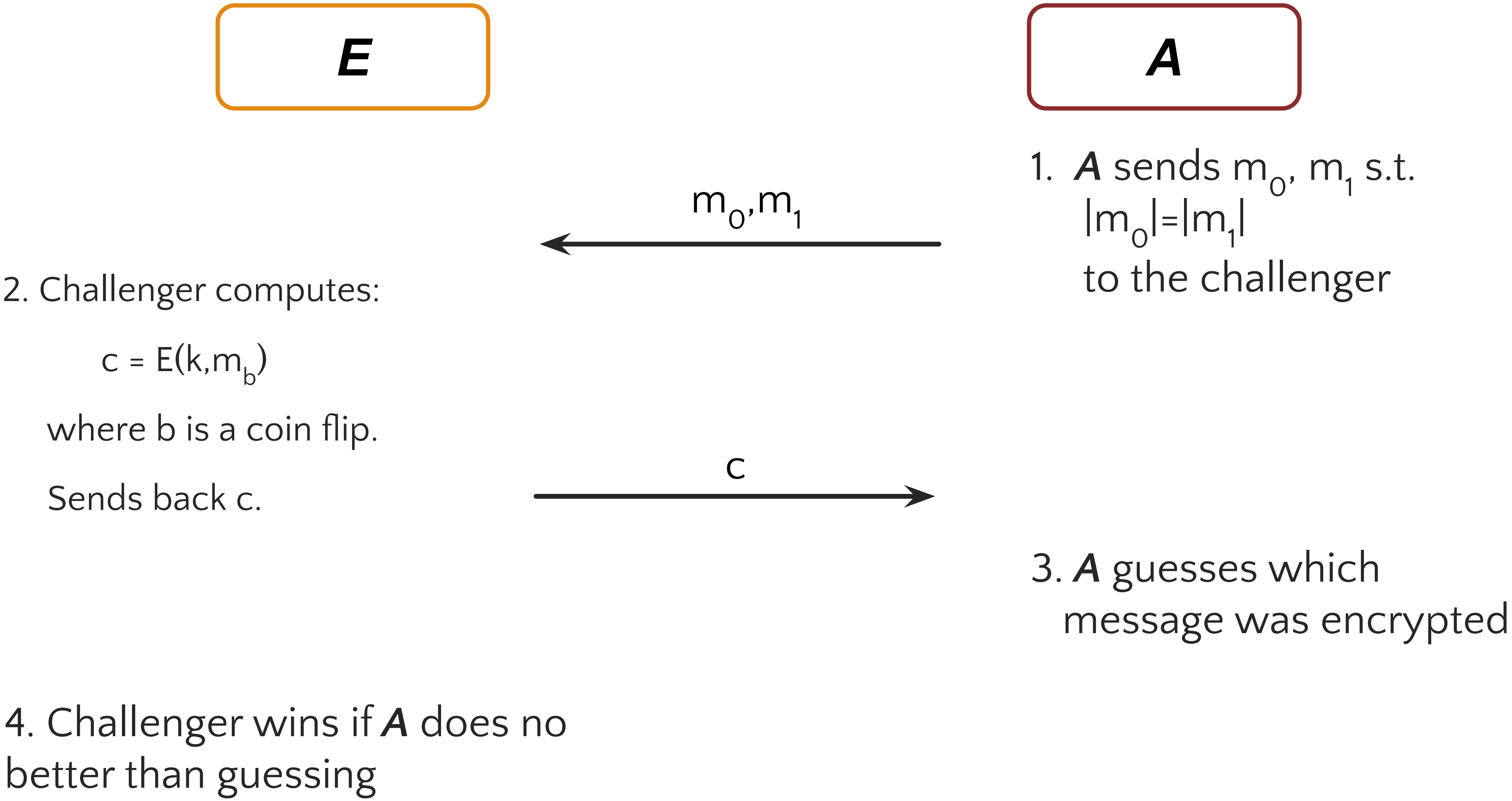
E

Challenger:
I have a
secure enc. E

A

I am *any*
adversary. I can
break your crypto.

Semantic Security Intuition



E

A

1. ***A*** sends m_0, m_1 s.t. $|m_0|=|m_1|$ to the challenger

m_0,m_1

2. Challenger computes:

   $c = E(k,m_b)$

   where b is a coin flip.

   Sends back c.

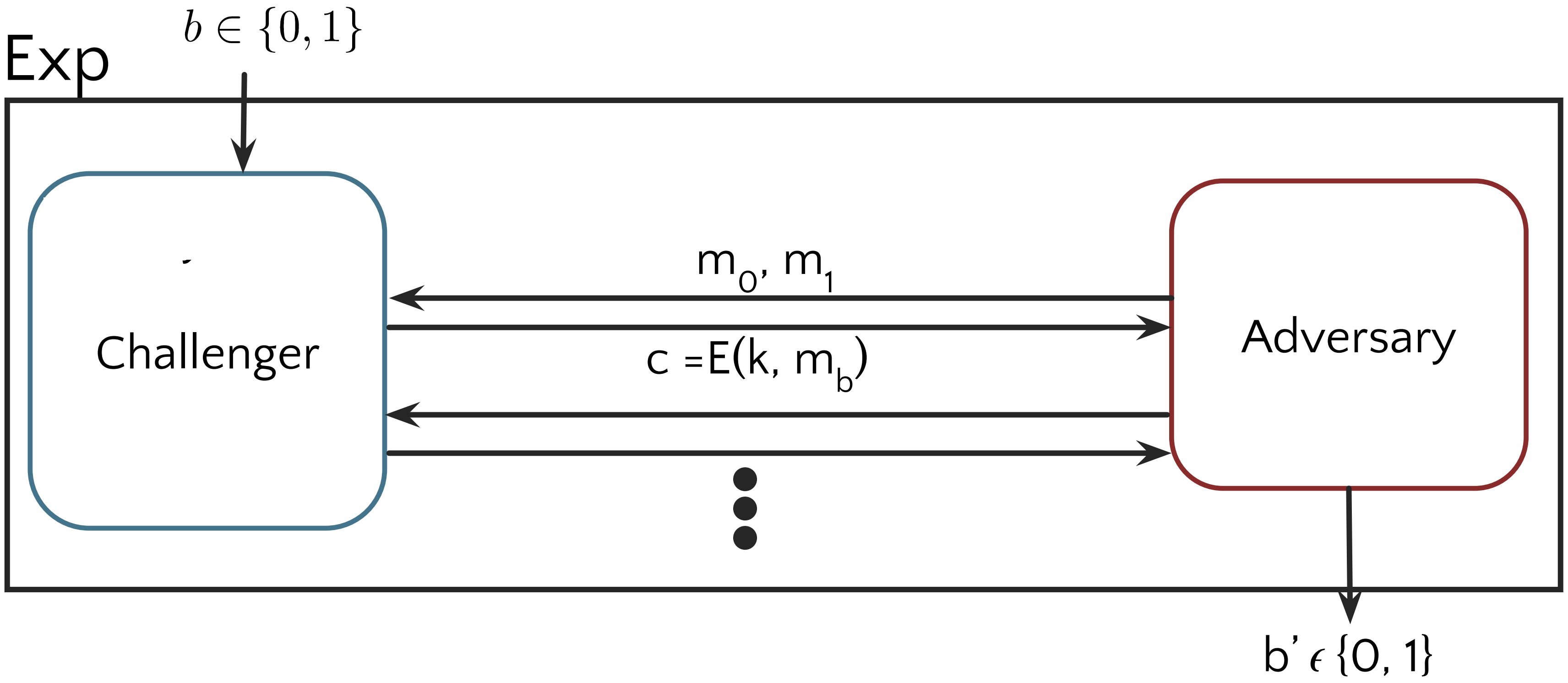
c

3. ***A*** guesses which message was encrypted

4. Challenger wins if ***A*** does no better than guessing

Semantic Security *IND-CPA* Game

For *b = 0,1* define experiment *Exp(b)* as:



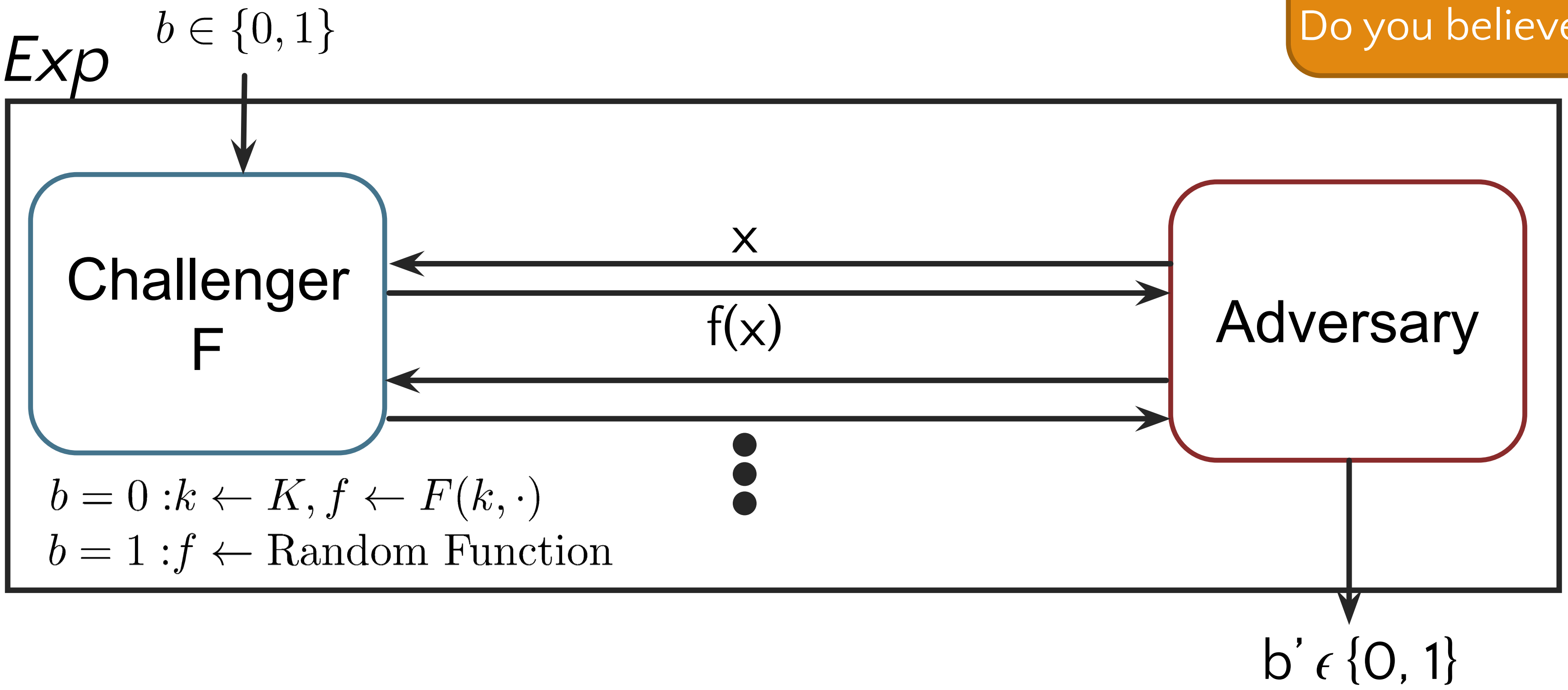
Defn: E is IND-CPA secure if for all efficient A:

$\text{Adv}_{\text{IND-CPA}}[A, E] := \Pr[\text{Exp}(1) = 1] - \Pr[\text{Exp}(0) = 1] < \varepsilon$

PRF Security Game

For b = *0,1* define experiment *Exp(b)* as:

Important
This is a definition!
Do you believe it captures PRF?

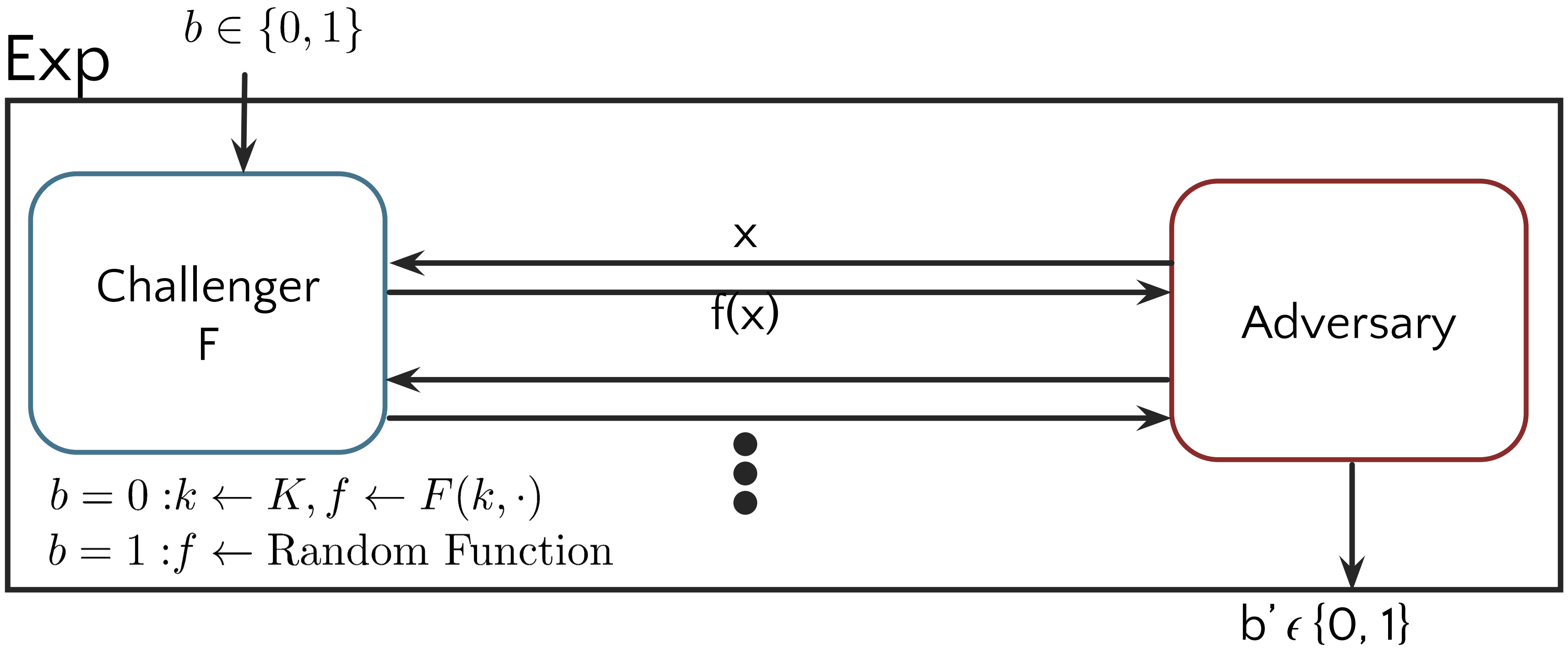

Exp
$b \in \{0,1\}$
Challenger
F
x
f(x)
Adversary
$b = 0 : k \leftarrow K, f \leftarrow F(k, \cdot)$
$b = 1 : f \leftarrow \text{Random Function}$
b' ϵ {0, 1}

Defn: F is a secure PRF if for all efficient *A*:

$$\text{Adv}_{\text{PRF}}[A, F, q] := |\Pr[\text{Exp}(0) = 1] - \Pr[\text{Exp}(1) = 1]| < \varepsilon$$

where A makes at most q queries

Sanity Check: Guessing

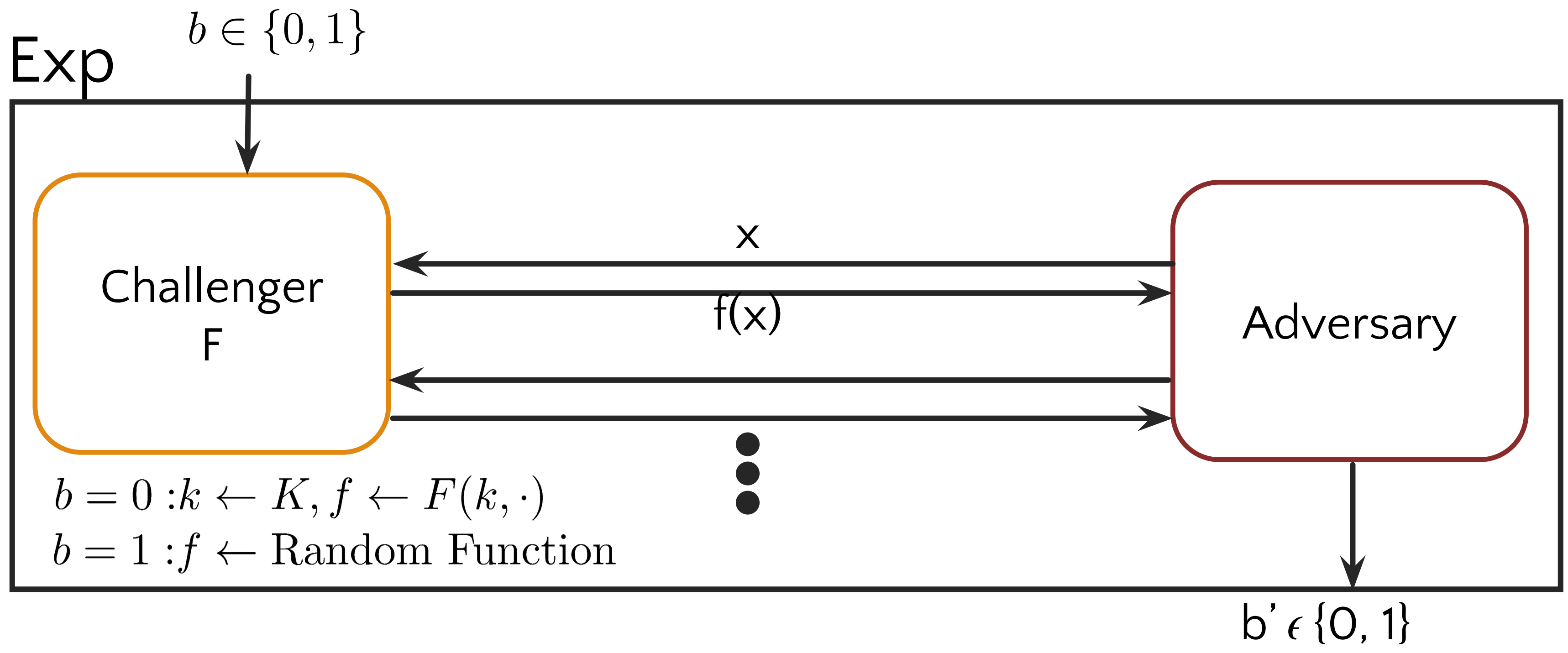

Exp
$b \in \{0,1\}$
Challenger
F
x
f(x)
Adversary
$b = 0 : k \leftarrow K, f \leftarrow F(k, \cdot)$
$b = 1 : f \leftarrow \text{Random Function}$
b' ϵ {0, 1}

$$\mathbf{Adv}_{PRF}[A, F] := |\Pr[Exp(0) = 1] - \Pr[Exp(1) = 1]| < \epsilon$$

Suppose the adversary simply flips a coin. Then
Pr[Exp(0) = 1] = 0.5 Pr[Exp(1) = 1] = 0.5

Then: $\text{Adv}_{\text{PRF}}[\boldsymbol{A},\boldsymbol{F}]$ = |.5 - .5| = 0

Example: Non-Negligible

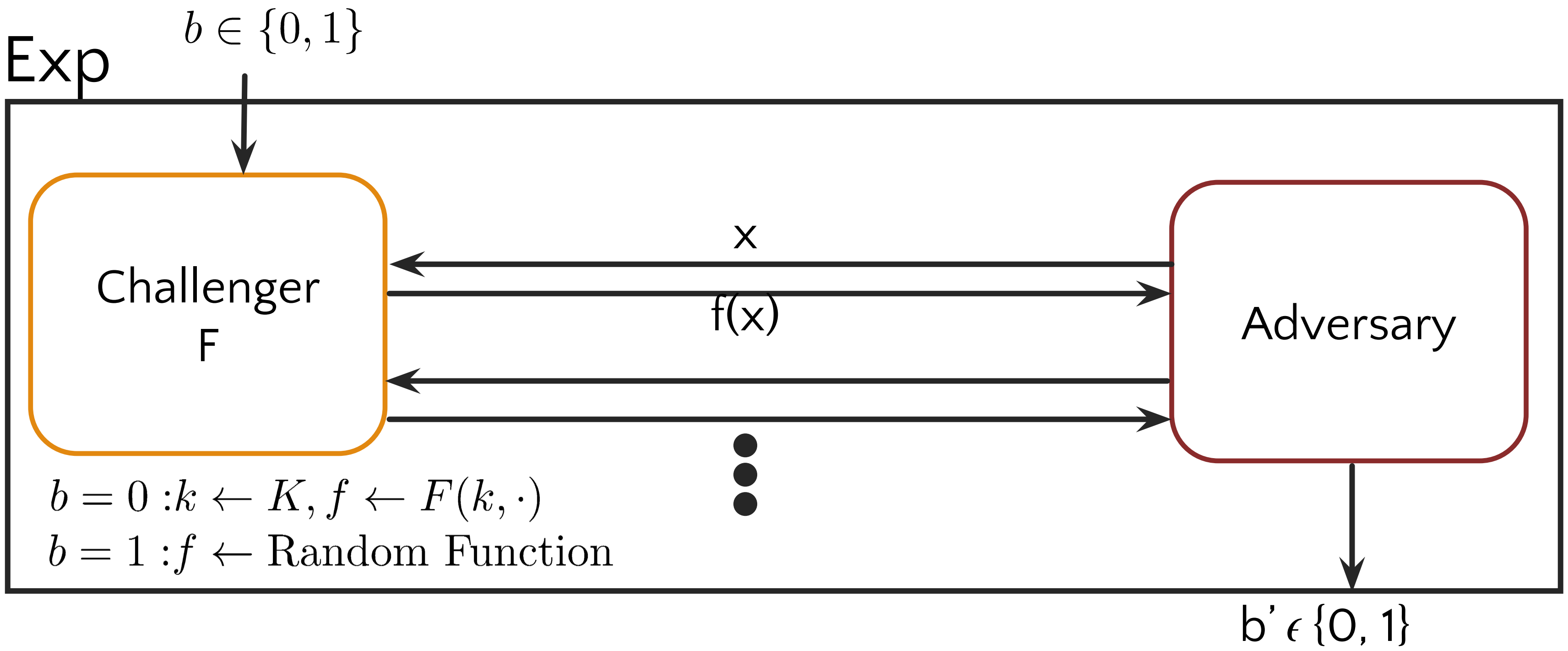

Exp
$b \in \{0,1\}$
Challenger
F
x
f(x)
Adversary
$b = 0: k \leftarrow K, f \leftarrow F(k,\cdot)$
$b = 1: f \leftarrow \text{Random Function}$
b' ϵ {0, 1}

$$\mathbf{Adv}_{PRF}[A,F] := |\Pr[Exp(0)=1] - \Pr[Exp(1)=1]| < \epsilon$$

Suppose the PRF is slightly broken, say:
Pr[Exp(0) = 1] = 0.2 Pr[Exp(1) = 1] = 0.8

Then: $\text{Adv}_{\text{PRF}}[\boldsymbol{A},\boldsymbol{F}]$ = |0.2 - 0.8| = 0.6

Example: Wrong More Than 50%


Exp
$b \in \{0,1\}$
Challenger
F
x
f(x)
Adversary
$b = 0 : k \leftarrow K, f \leftarrow F(k, \cdot)$
$b = 1 : f \leftarrow \text{Random Function}$
b’ ϵ {0, 1}

$$\mathbf{Adv}_{PRF}[A, F] := |\Pr[Exp(0) = 1] - \Pr[Exp(1) = 1]| < \epsilon$$

Suppose the adversary is almost always wrong, say:

Pr[Exp(0) = 1] = 0.8 Pr[Exp(1) = 1] = 0.2

Then: $\text{Adv}_{\text{PRF}}[\boldsymbol{A},\boldsymbol{F}]$ = |0.8 – 0.2| = 0.6

Guessing wrong > 50% of the time yields an alg. to guess right

Participation Question

Let $F : K \times X \to \{0,1\}^{128}$ be a secure PRF.

Is the following G a secure PRF?

$$G(k,x) = \begin{cases} 0^{128} & \text{if } x = 0 \\ F(k,x) & \text{otherwise} \end{cases}$$

A. No, it is easy to distinguish G from a random function

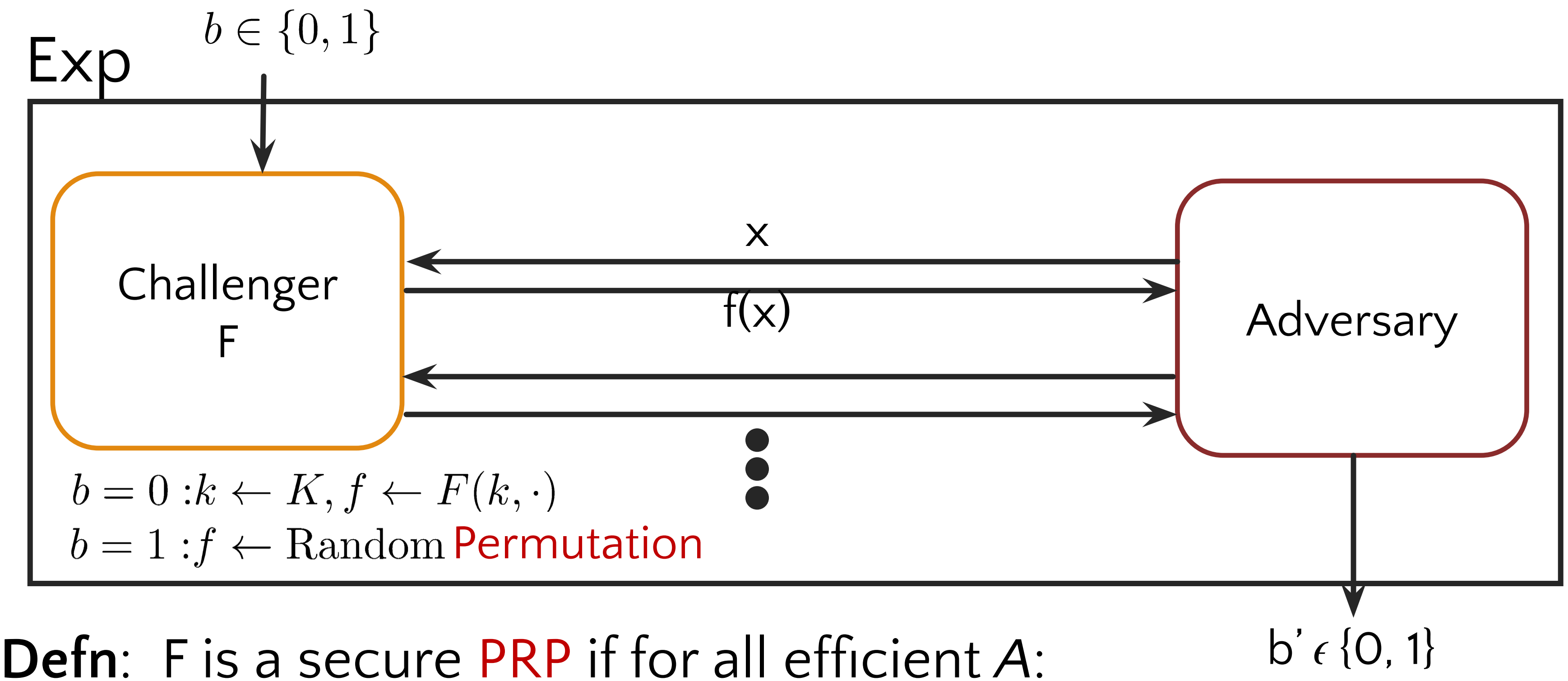
B. No, G might map more than one input to 0^{128}

C. Yes, an attack on G would also break F

D. It depends on F

PRP Security Game

For *b = 0,1* define experiment *Exp(b)* as:


Exp
$b \in \{0,1\}$
Challenger
F
x
f(x)
Adversary
$b = 0 : k \leftarrow K, f \leftarrow F(k, \cdot)$
$b = 1 : f \leftarrow$ Random Permutation
b' ϵ {0, 1}

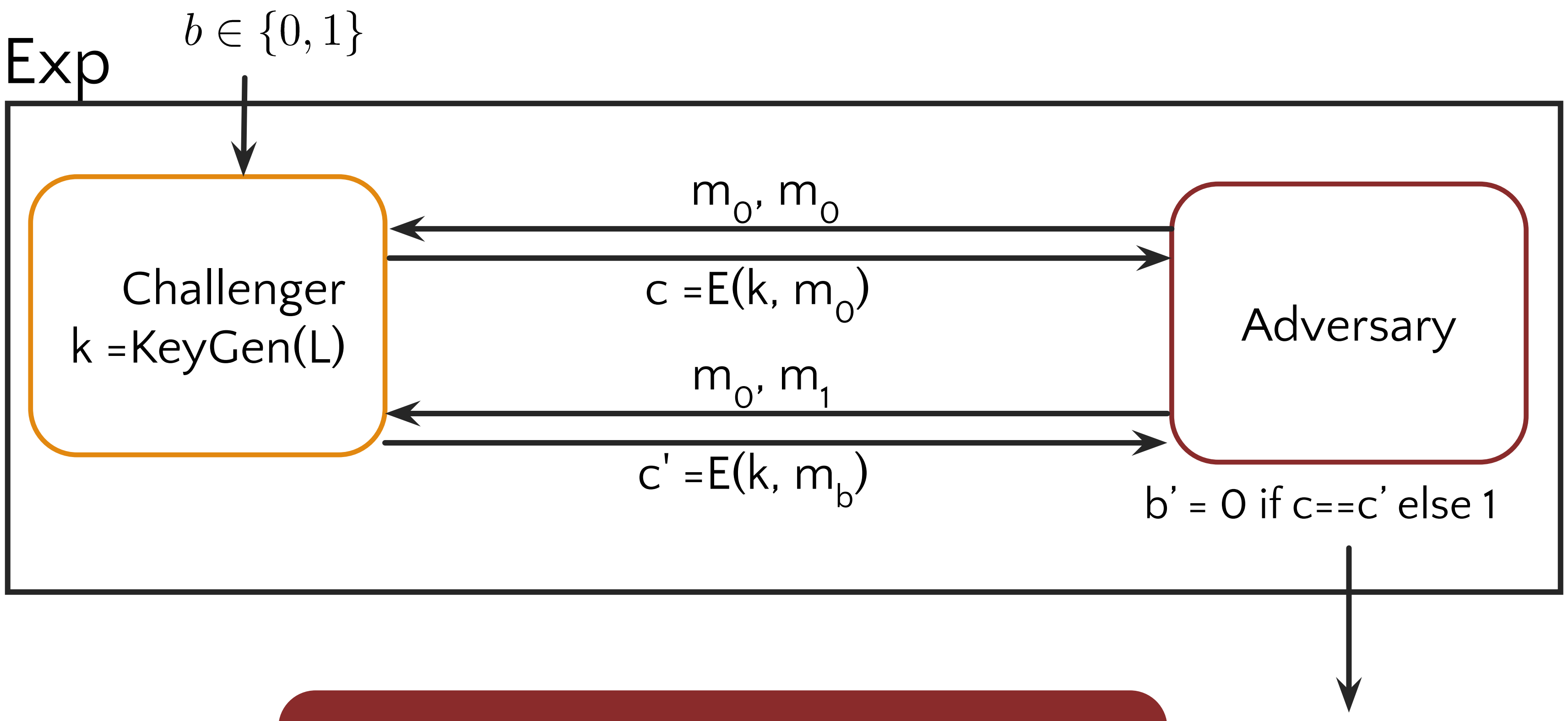
<u>Defn</u>: F is a secure PRP if for all efficient *A*:

$$\mathbf{Adv}_{\text{PRP}}\ [A, F] := |\Pr[Exp(0) = 1] - \Pr[Exp(1) = 1]| < \epsilon$$

Let’s Apply This Definition of Security

Breaking Deterministic, Stateless Encryption

For *b = 0,1* define experiment *Exp(b)* as:



Encryption must be randomized
or be stateful!

Pending Question: How do we encrypt *more* data safely??

Encryption Modes

Block Ciphers help us encrypt a single block of data securely

To encrypt multiple blocks with a single key we need to find secure *modes of operation* , i.e., ways to combine block ciphers on messages longer than a single block

Using PRPs and PRFs

Goal: build "secure" encryption from a secure PRP (e.g. AES).

First: **one-time keys**

1. Adversary's power:

   Adv sees only one ciphertext (one-time key)

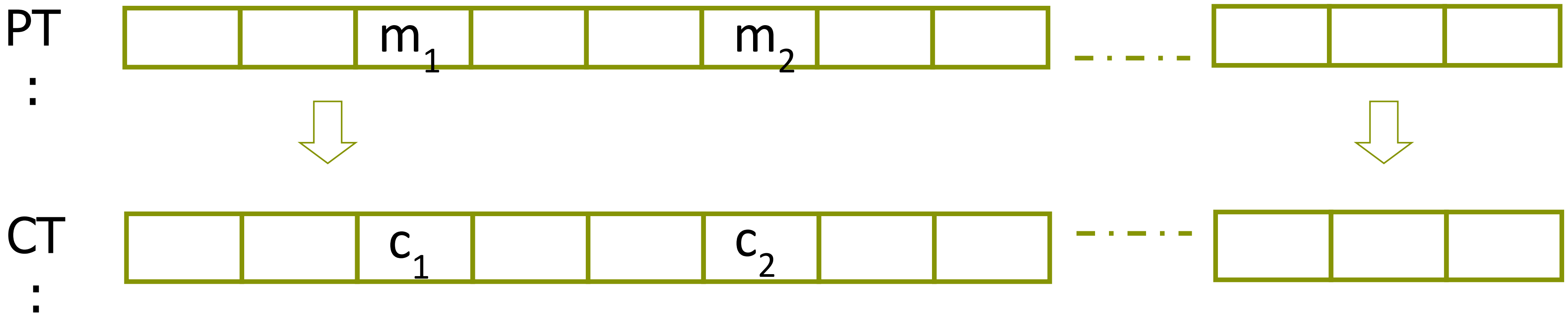
2. Adversary's goal:

   Learn info about PT from CT (semantic security)

Next up: many-time keys (a.k.a chosen-plaintext security)

ECB Mode: Insecure use of a PRP

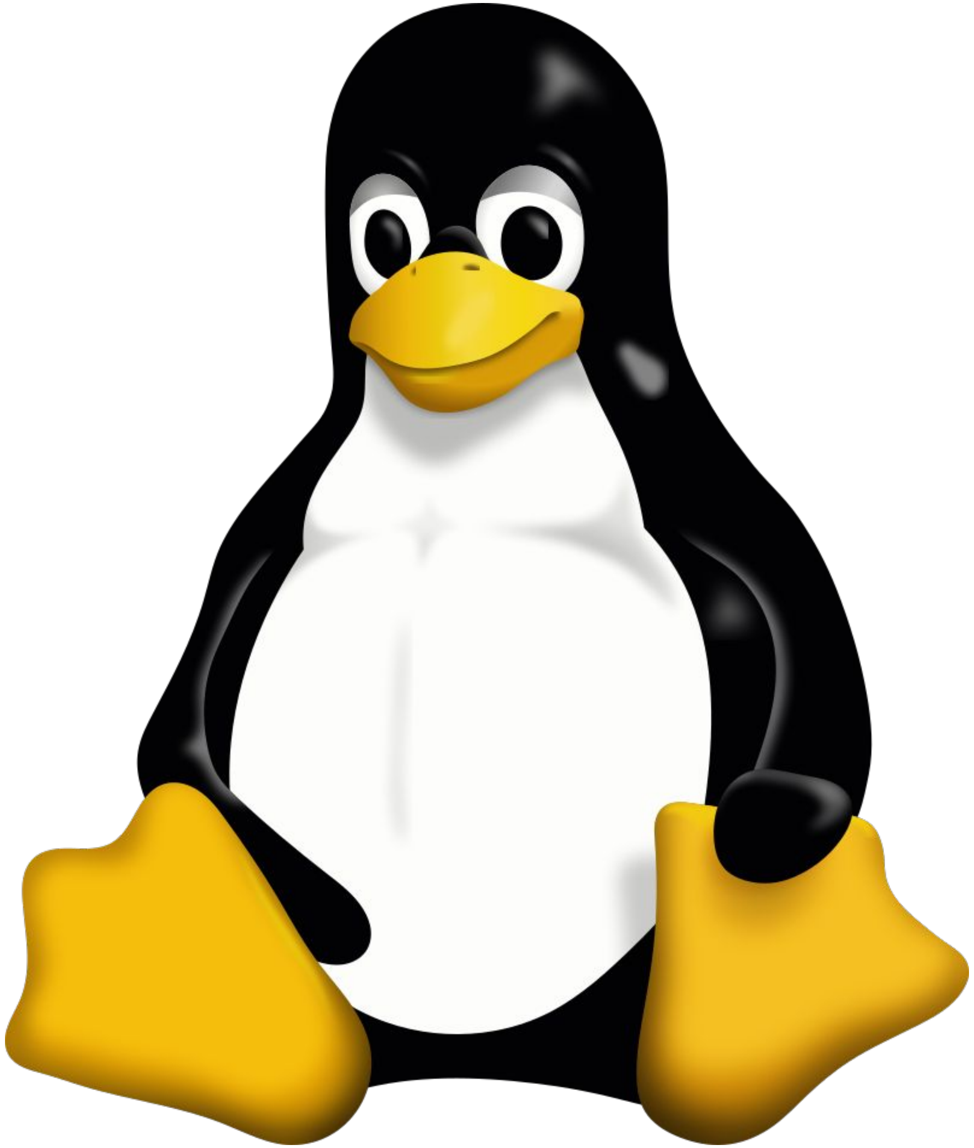
Electronic Code Book (ECB):

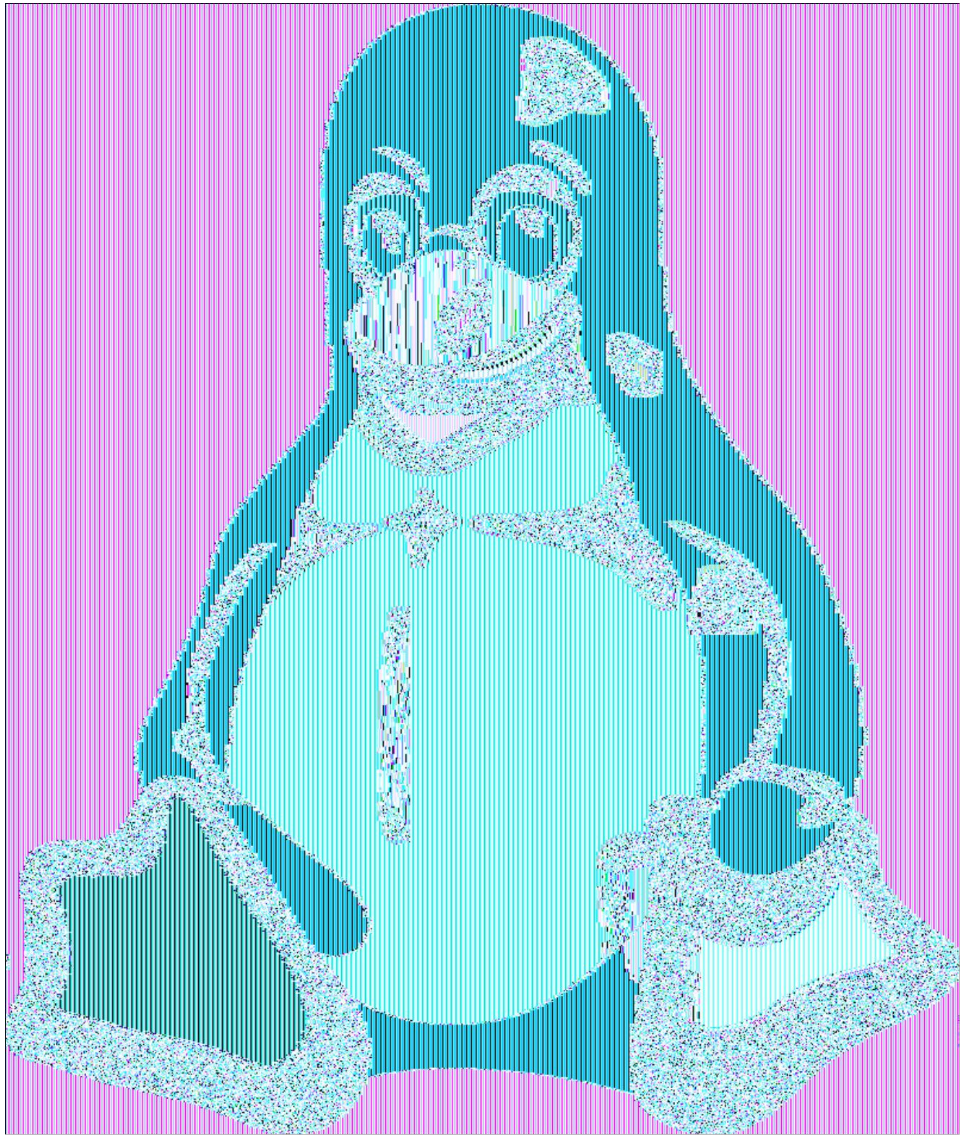


Problem:

- if $m_1=m_2$ then $c_1=c_2$

In pictures





Is there anything better?

Next Time!

Ευχαριστώ και καλή μέρα εύχομαι!

Keep hacking!