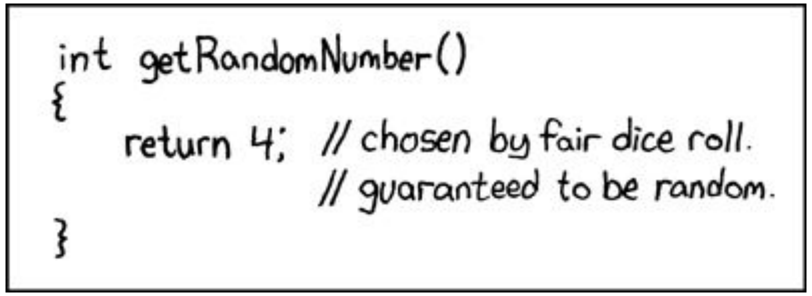
Διάλεξη #13 - On Randomness

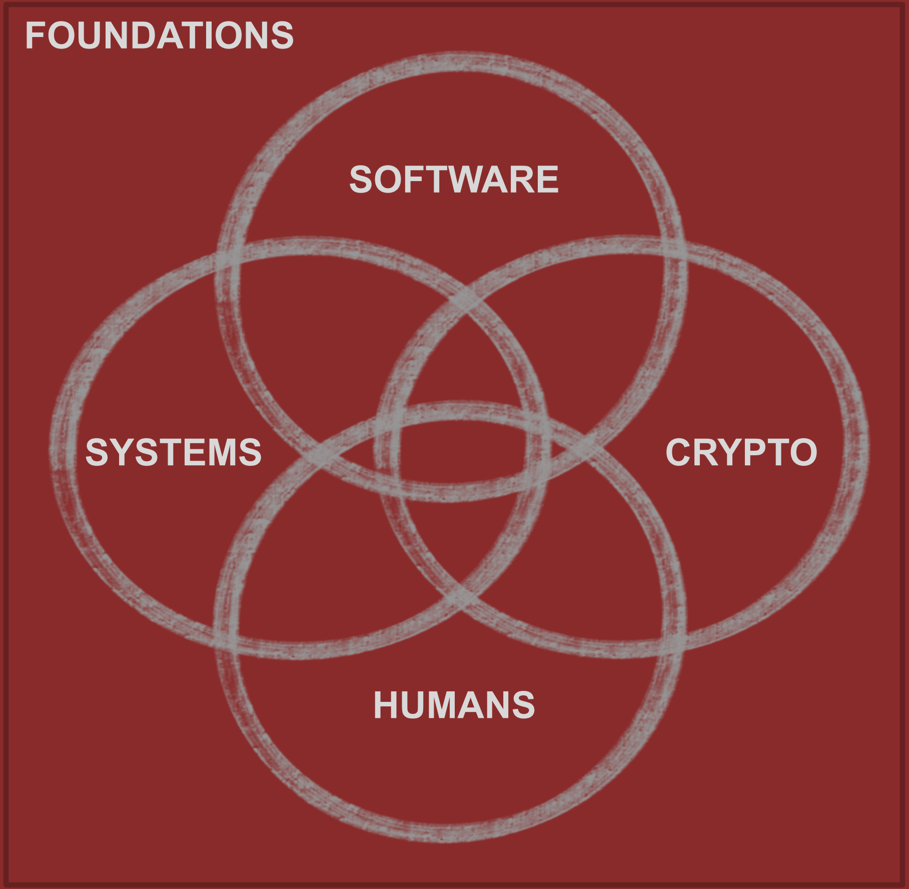
Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών

Εισαγωγή στην Ασφάλεια

Θανάσης Αυγερινός



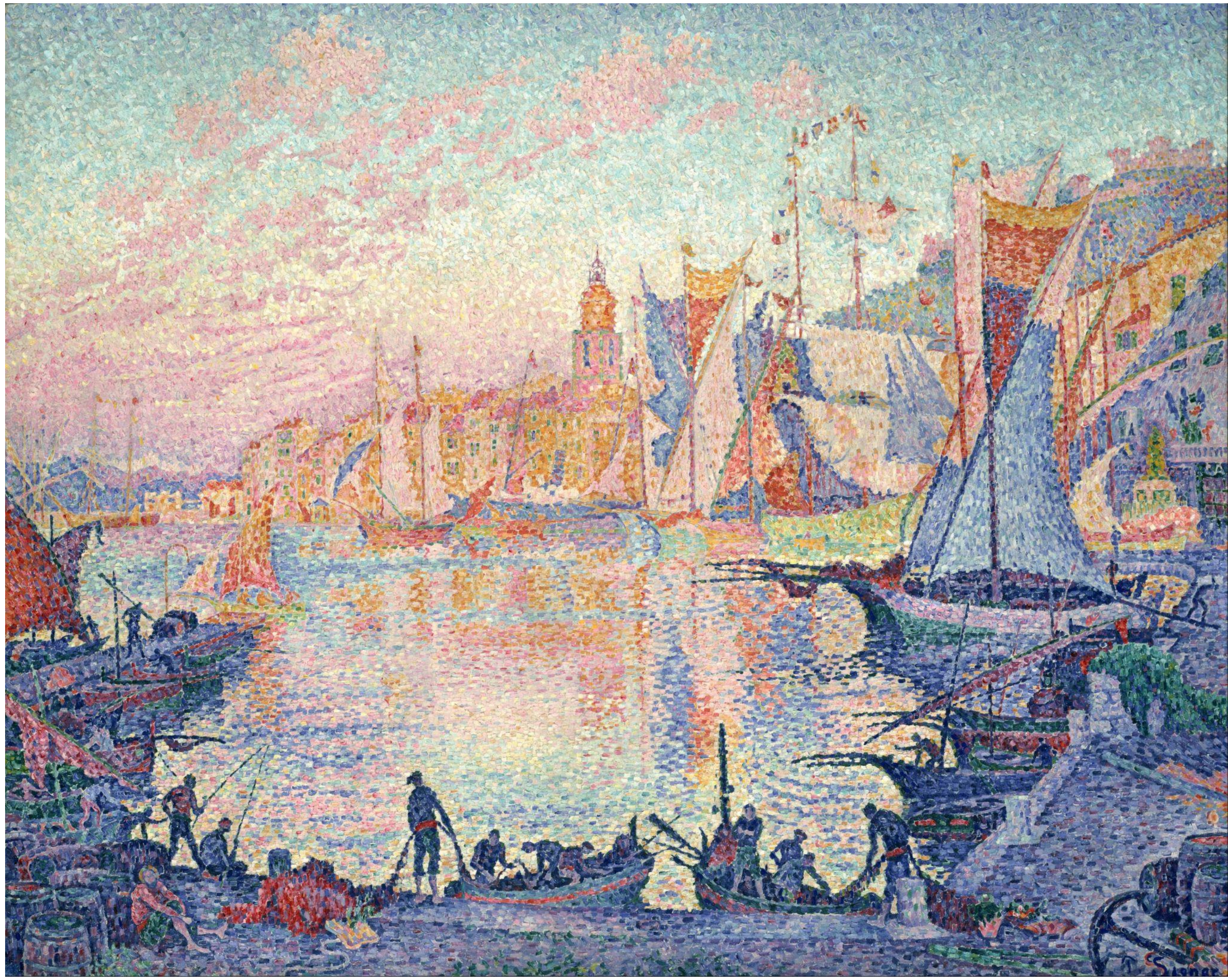
```
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
               // guaranteed to be random.
}
```



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!)

Την προηγούμενη φορά

- About cryptography
- Terminology
- Traditional ciphers
- One-time pad



Ανακοινώσεις / Διευκρινίσεις

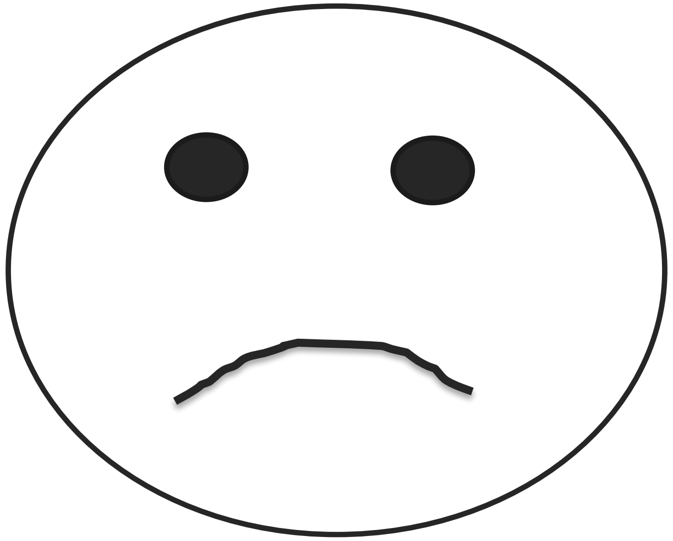
- Αναπλήρωση διάλεξης την Παρασκευή 2 Μαΐου, 11πμ - Αίθουσα Β
- Είσαι σίγουρος ότι είναι ΟΚ να επιτρέπουμε το κλειδί "0" σε OTP; Ή αντίστοιχα κλειδιά τα οποία έχουν συγκεκριμένα patterns;

Σήμερα

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

The “Bad News” Theorem

Theorem: Perfect secrecy requires |K| >= |M|



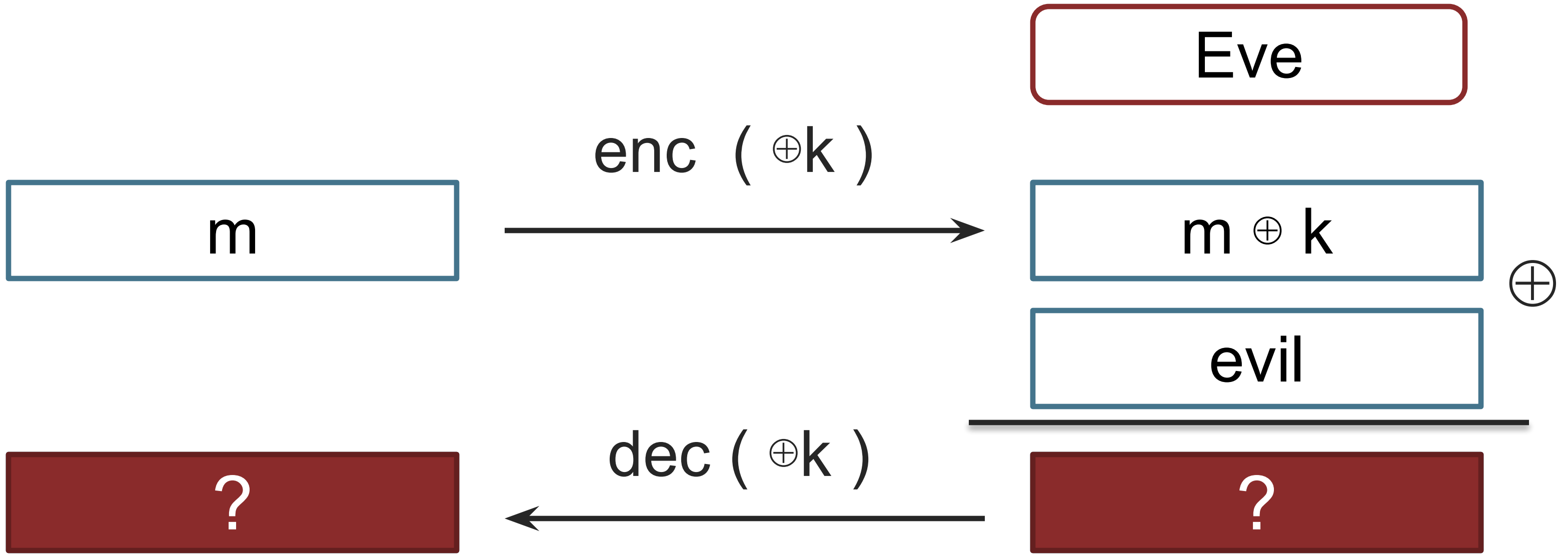
In practice, we usually shoot for *computational security*

More bad news

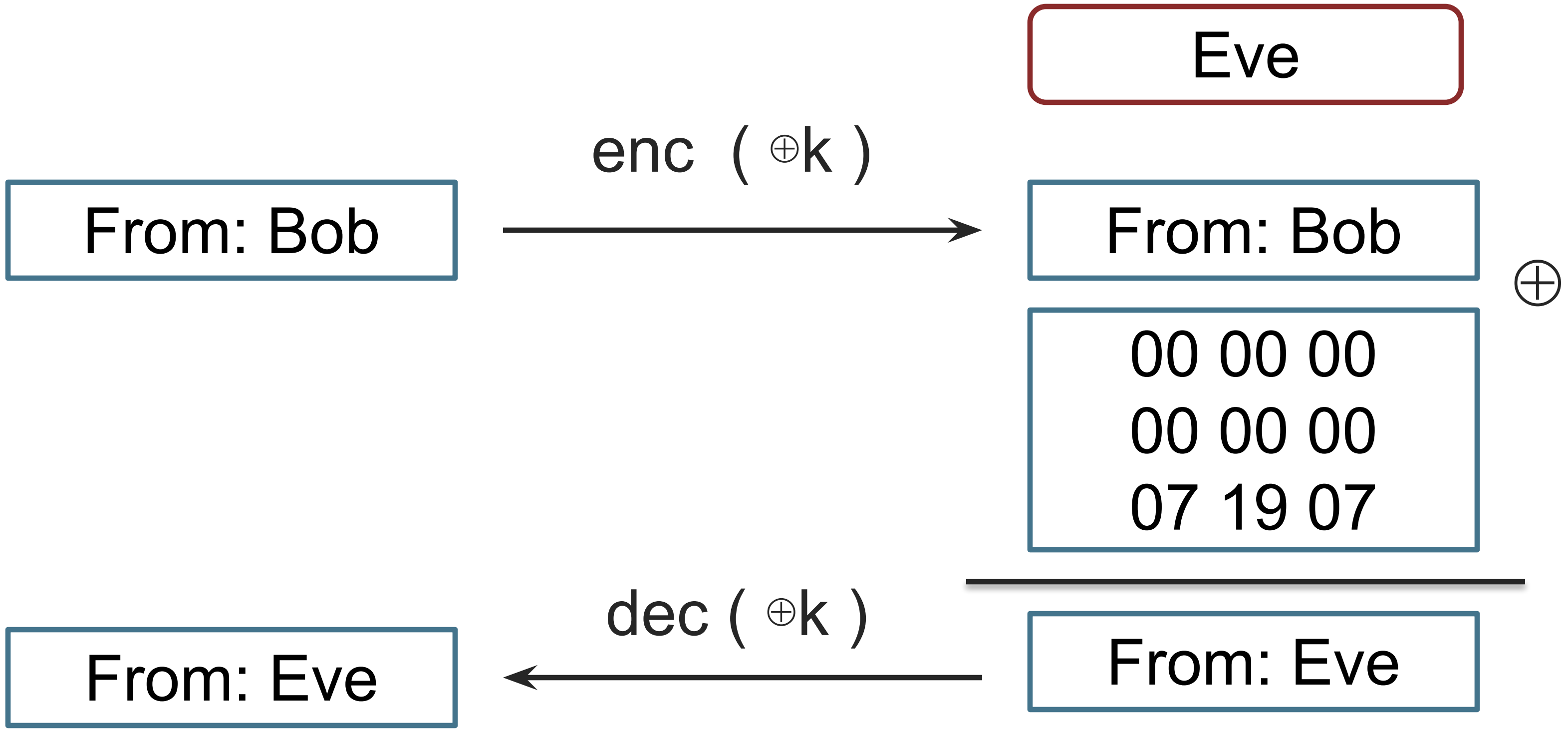
The OTP provides perfect secrecy …

… but is that enough?

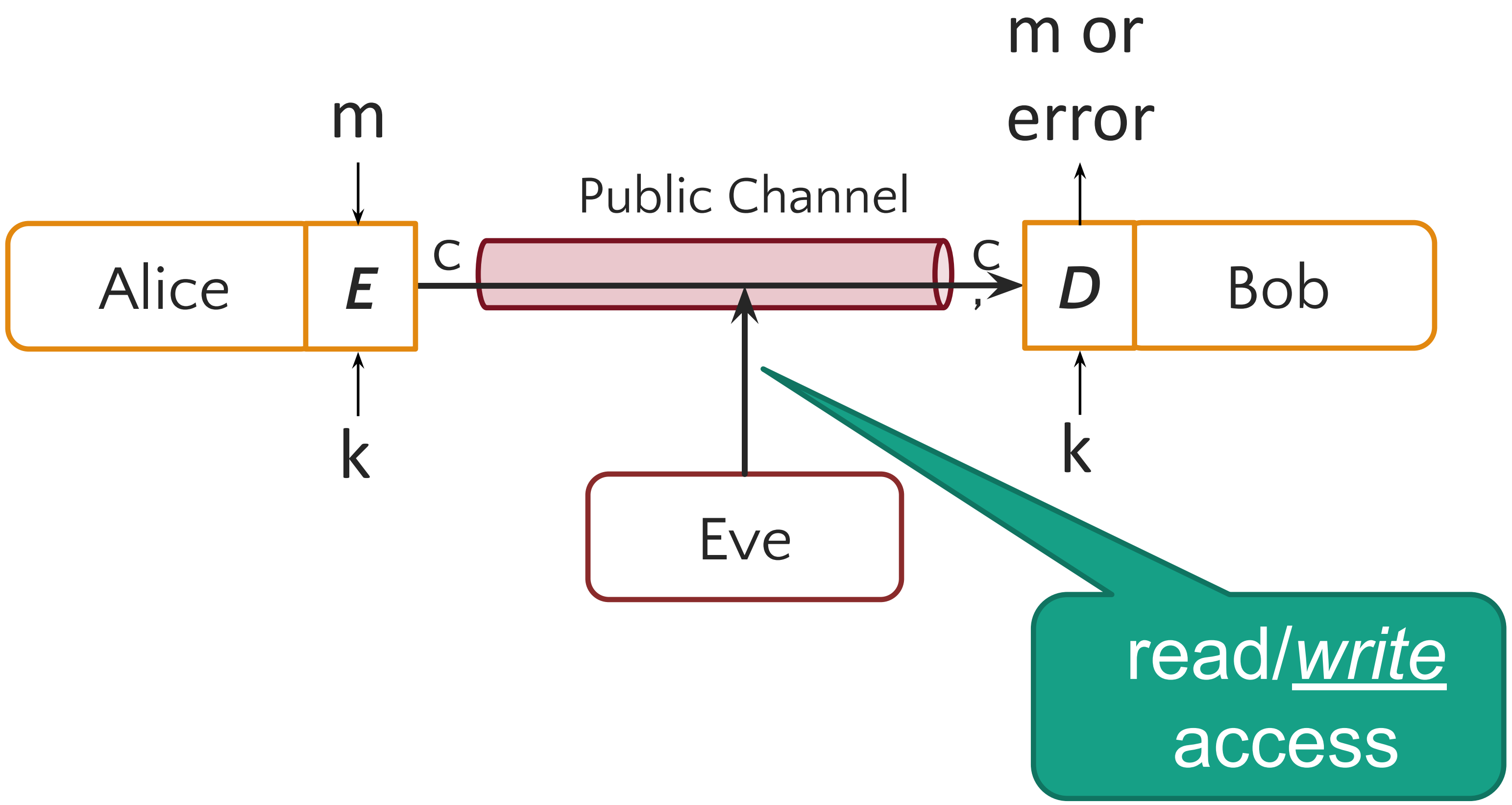
No Integrity



No Integrity



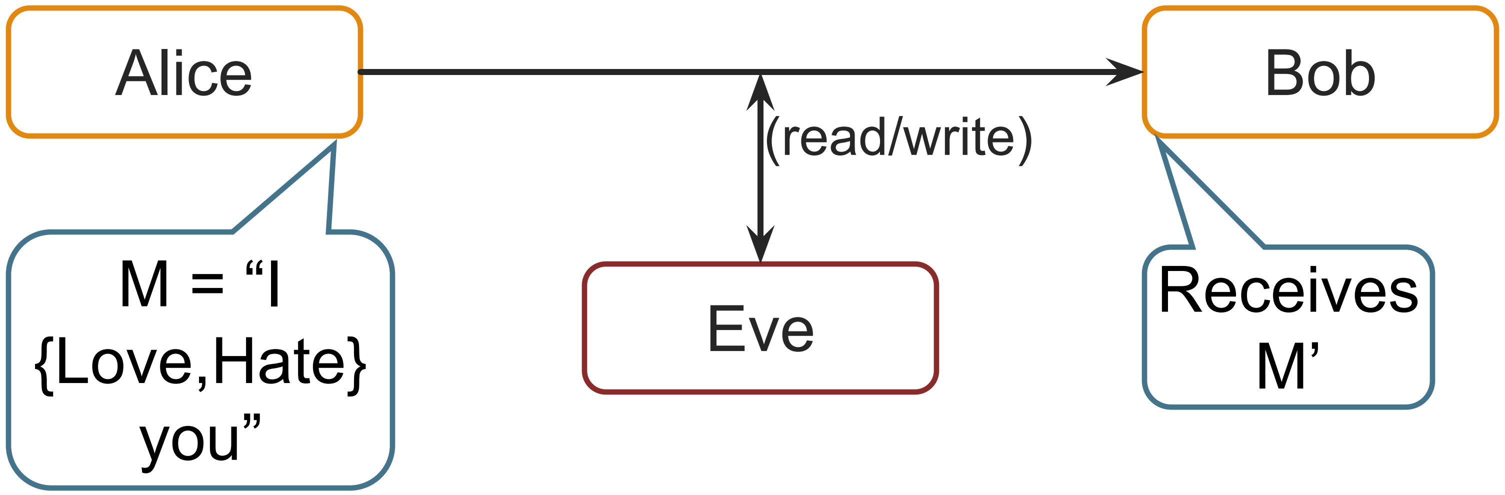
Our Goal: Secure Communication



Sub Goal 2: *Integrity*

Eve should not be able to alter *m* without detection

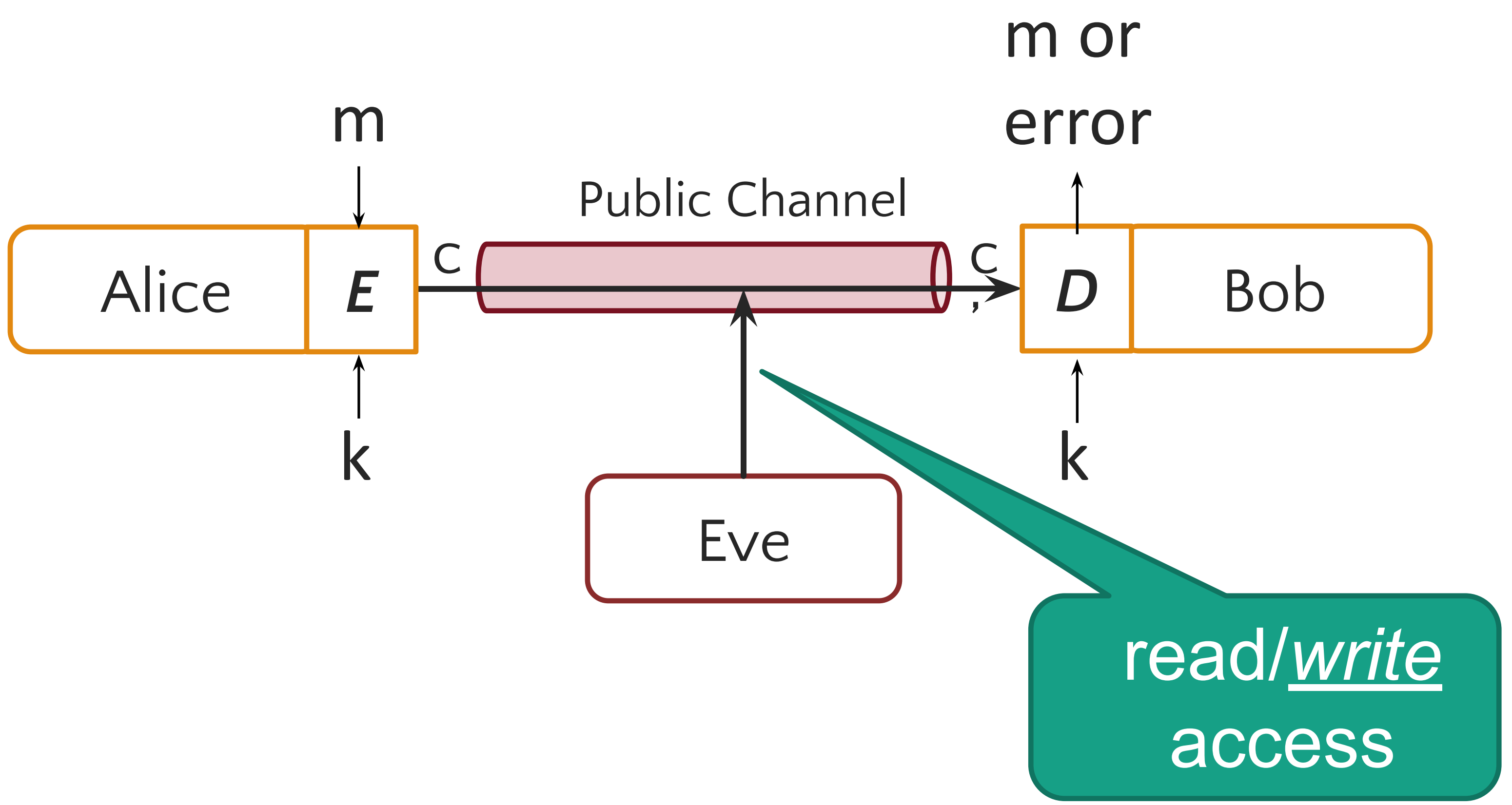
Detecting Modifications



Bob should be able to determine if M’=M

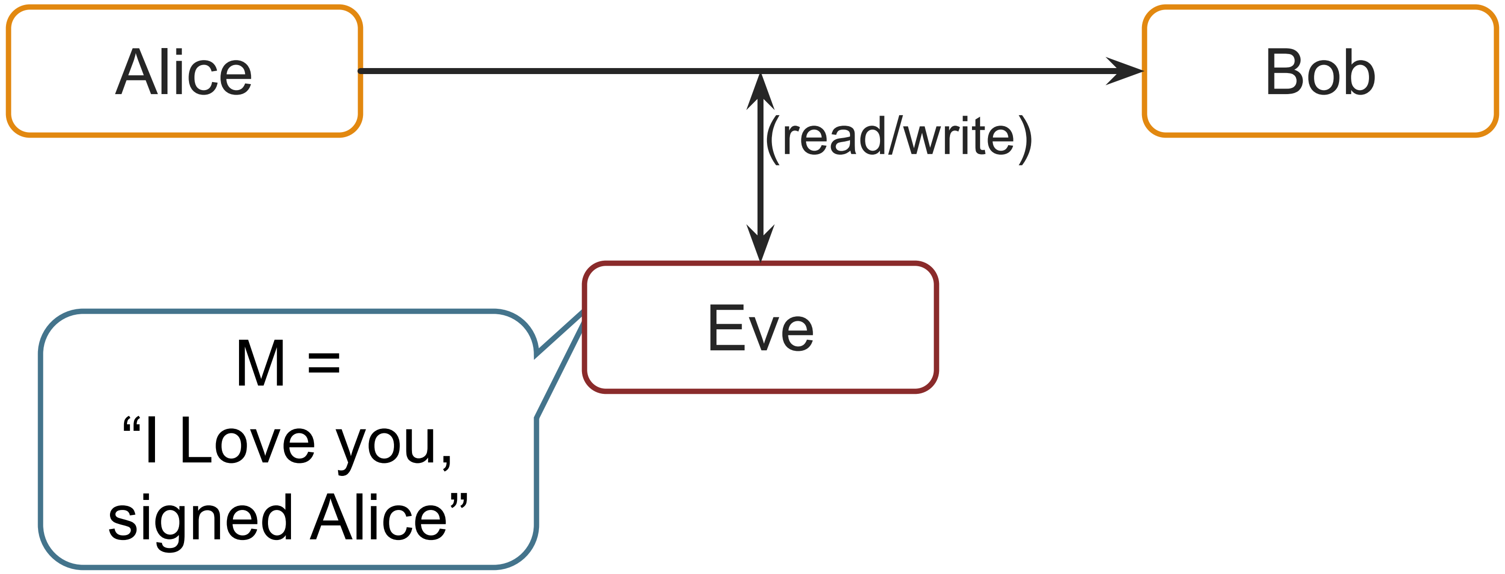
Ex: Eve should not be able to change Alice’s message without detection (even if Eve doesn’t know content of M)

Our Goal: Secure Communication



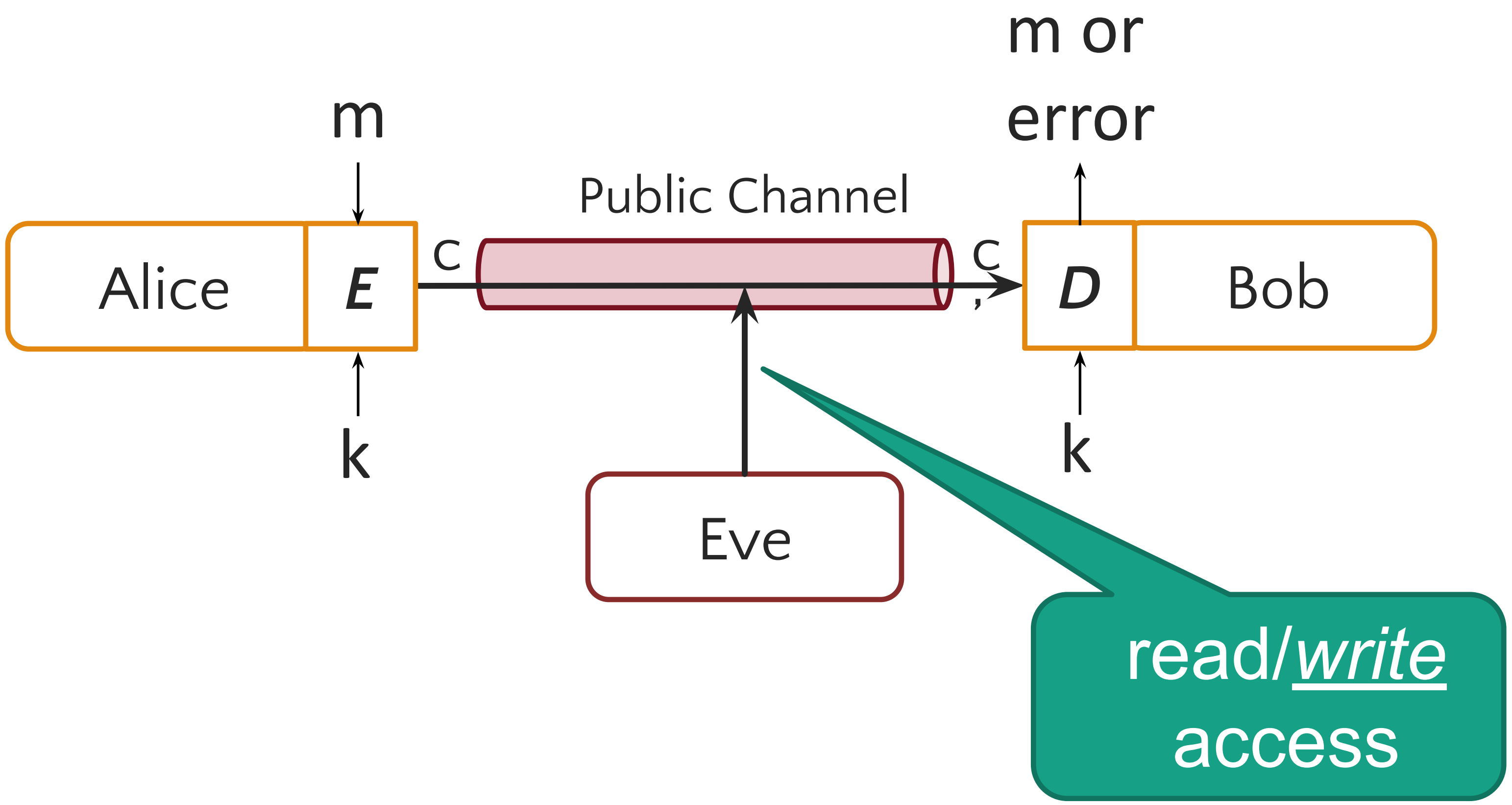
Sub Goal 3: *Authenticity*
Eve should not be able to forge messages as Alice

Detecting Message Injection



Bob should be able to determine whether M was sent by Alice

Our Goal: Secure Communication



Secure Communication means: Secrecy, Integrity, and Authenticity

Still open: the pieces we need for secure communication

Everyone shares *same* secret k

Only 1 party has a secret

	Symmetric Trust Model	Asymmetric Trust Model
Message Privacy	Private key encryption • Stream Ciphers • Block Ciphers	Asymmetric encryption (aka public-key encryption)
Message Authenticity and Integrity	Message Authentication Code (MAC)	Digital Signature Scheme

Principle 1: All algorithms are *public* (Kerckhoffs's Principle)
Principle 2: Security is determined *only* by key size
Principle 3: If you roll your own, it will be *insecure*

A Crucial Ingredient: Randomness!

Crucial Ingredient: Randomness

- Explicit usage
  - Generate secret keys
  - Generate random “nonces” for encryption (more later on)
- Less obvious usage:
  - Generate passwords for new users
  - Shuffle cards in a poker game or votes in an election
  - Choose which work items to audit for correctness

Insecure Randomness: C rand()

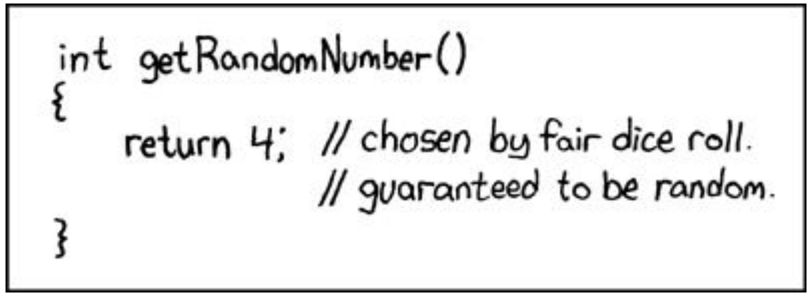
- Many languages have a built-in “random” function

```
unsigned long int next = 1;

/* srand: set seed for rand() */
void srand(unsigned int seed) {
   next = seed;
}

/* rand: return pseudo-random integer on 0..32767 */
int rand(void) {
   next = next * 1103515245 + 12345;
   return (unsigned int)(next/65536) % 32768;
}
```

What’s the problem?

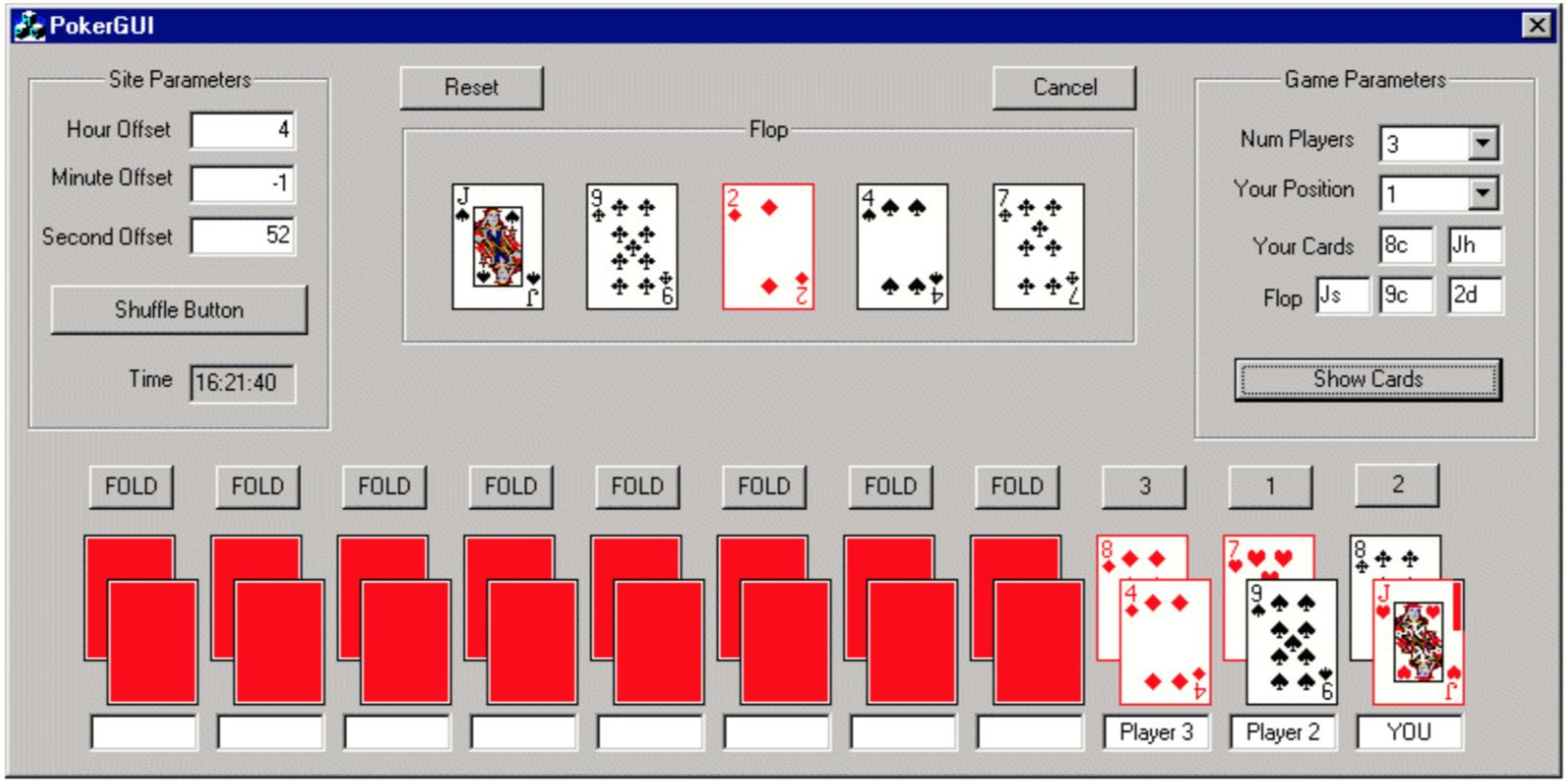
Insecure Randomness: C rand()

- Many languages have a built-in “random” function
- Given a few outputs, remaining values are ***predictable***!



```
int getRandomNumber()
{
    return 4;  // chosen by fair dice roll.
               // guaranteed to be random.
}
```

https://xkcd.com/221/



Hour Offset 4
Minute Offset -1
Second Offset 52
Time 16:21:40
Num Players 3
Your Position 1
Your Cards 8c Jh
Flop Js 9c 2d
Player 3
Player 2
YOU

More Details

"How We Learned to Cheat at Online Poker: A Study in Software Security"

https://www.developer.com/tech/article.php/616221/How-We-Learned-to-Cheat-at-Online-Poker-A-Study-in-Software-Security.htm

Sony PS3 vs. Randomness

27th Chaos Communication Congress

Console Hacking 2010

PS3 Epic Fail

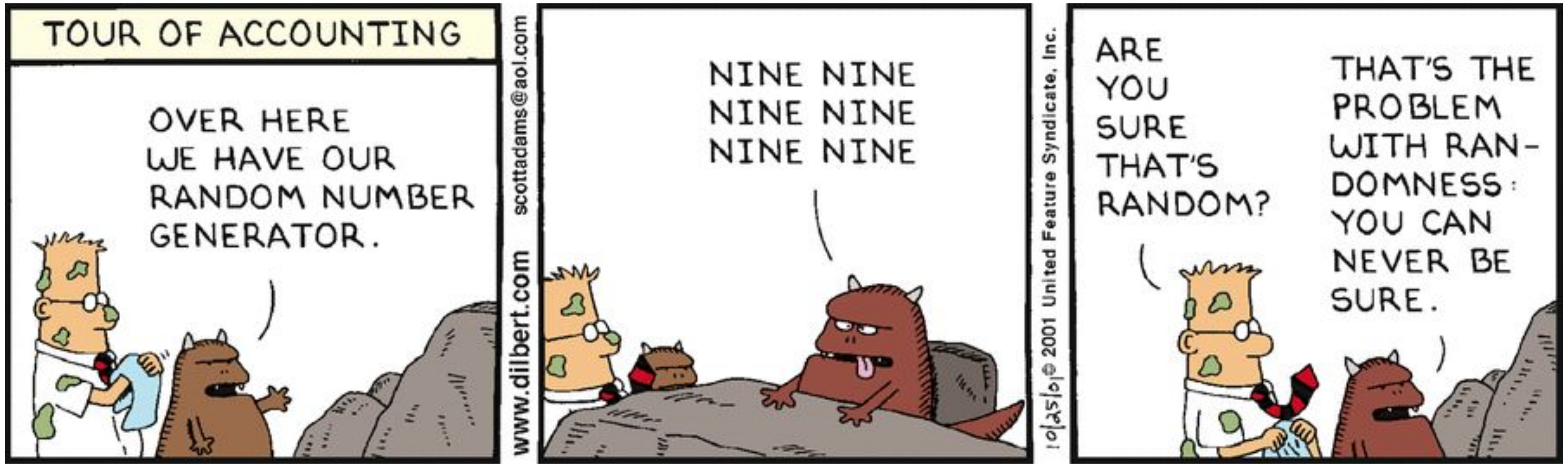
fail0verflow

bushing, marcan, segher, sven

- 2010/2011: Hackers found/released ***private root key*** for PS3
- Key used to sign software
  - Load any software on PS3 and execute as “trusted”
  - i.e., Anyone can **pretend to be** Sony
- Flaw: Used same “random” number for every ECDSA signature

More Details
https://events.ccc.de/congress/2010/Fahrplan/attachments/1780_27c3_console_hacking_2010.pdf

So… where does randomness come from?


TOUR OF ACCOUNTING
OVER HERE WE HAVE OUR RANDOM NUMBER GENERATOR.
NINE NINE NINE NINE NINE NINE
ARE YOU SURE THAT'S RANDOM?
THAT'S THE PROBLEM WITH RAN- DOMNESS: YOU CAN NEVER BE SURE.
scottadams@aol.com
www.dilbert.com
10/25/01 © 2001 United Feature Syndicate, Inc.

http://dilbert.com/strip/2001-10-25

Obtaining "True" Randomness

- Gather entropy from unpredictable events

  Ex: Linux "entropy pool" includes mouse & keyboard timing

  – Exposed via
    - /dev/random – **NEVER USE /dev/random – its API is broken and wrong**
    - /dev/urandom – beware of subtle issues with file descriptors and child processes!!!
    - `getrandom` syscall – **always use this syscall when available**
    - [Randomness in the Operating System, or How To Keep Evil Children Out Of Your Pool and Other Random Facts – Corrgan-Gibbs and Jana]()

- Physical random sources (do not use directly!)
  – RDRAND instruction
  – External devices

More fun conversations at:
https://lwn.net/Articles/889452/

Quiz Question

Which of the following is likely to consistently provide secure randomness any time you query it?

A. C’s `rand()` function

B. /dev/urandom

C. Physical random sources

D. /dev/random

Think this is an easy choice? Think again!

Couple of Reminders from Probability

Probability 101

U: finite set (e.g. U = $\{0,1\}^n$)

Probability distribution P over U is a function P: U ⟶ [0,1] s.t.

$$\sum_{x \in U} P(x) = 1$$

A ⊆ U is called an event and $Pr[A] = \sum_{x \in A} P(x) \in [0,1]$

A random variable is a function X:U⟶V .

- X takes values in U and defines a distribution on V

Independence

Definition: events A and B are independent if Pr[A and B] = Pr[A] * Pr[B]

Random variables X,Y taking values in V are independent if

$$\forall a,b \in V: \quad \Pr[\, X=a \text{ and } Y=b] = \Pr[X=a] * \Pr[Y=b]$$

Example: $U = \{0,1\}^2 = \{00, 01, 10, 11\}$ and $r \overset{\$}{\longleftarrow} U$

Define r.v. X and Y as: $X = \text{lsb}(r)$, $Y = \text{msb}(r)$

$$\Pr[\, X=0 \text{ and } Y=0 \,] = \Pr[\, r=00 \,] = 1/4 = \Pr[X=0] * \Pr[Y=0]$$

The Birthday Paradox

In a room of 23 people, the probability that you share a birthday with one other person is greater than 50%.

The Birthday Paradox

Let $r_1, \ldots, r_n \in U$ be indep. identically distributed random vars.

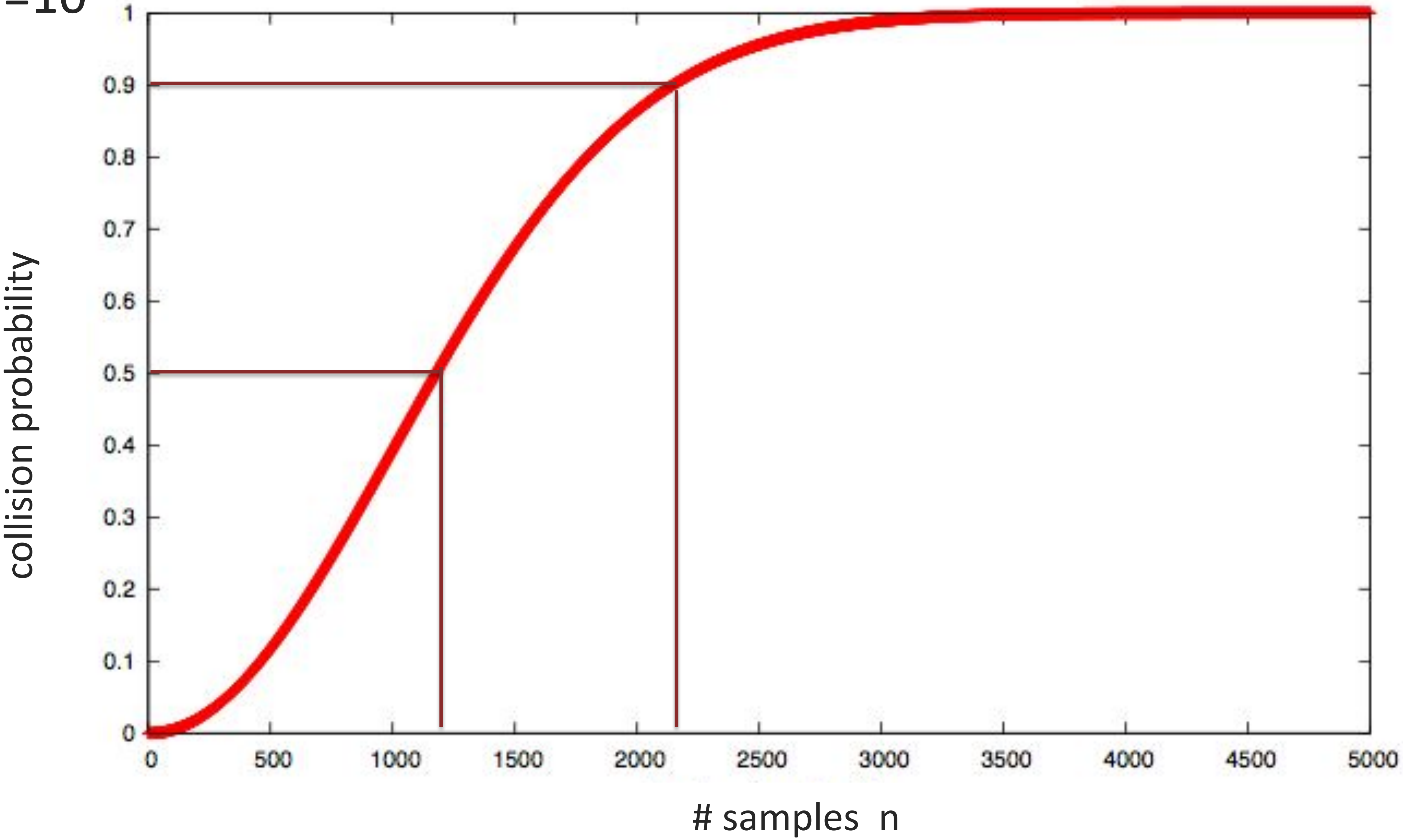
Theorem: when $n = 1.2 \times |U|^{1/2}$ then $\Pr[\ \exists\, i \neq j:\ r_i = r_j\ ] \geq ½$

Example: Let $U = \{0,1\}^{128}$

After sampling about 2^{64} random messages from U,

some two sampled messages will likely be the same

|U|=10^6



1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
collision probability
# samples n

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

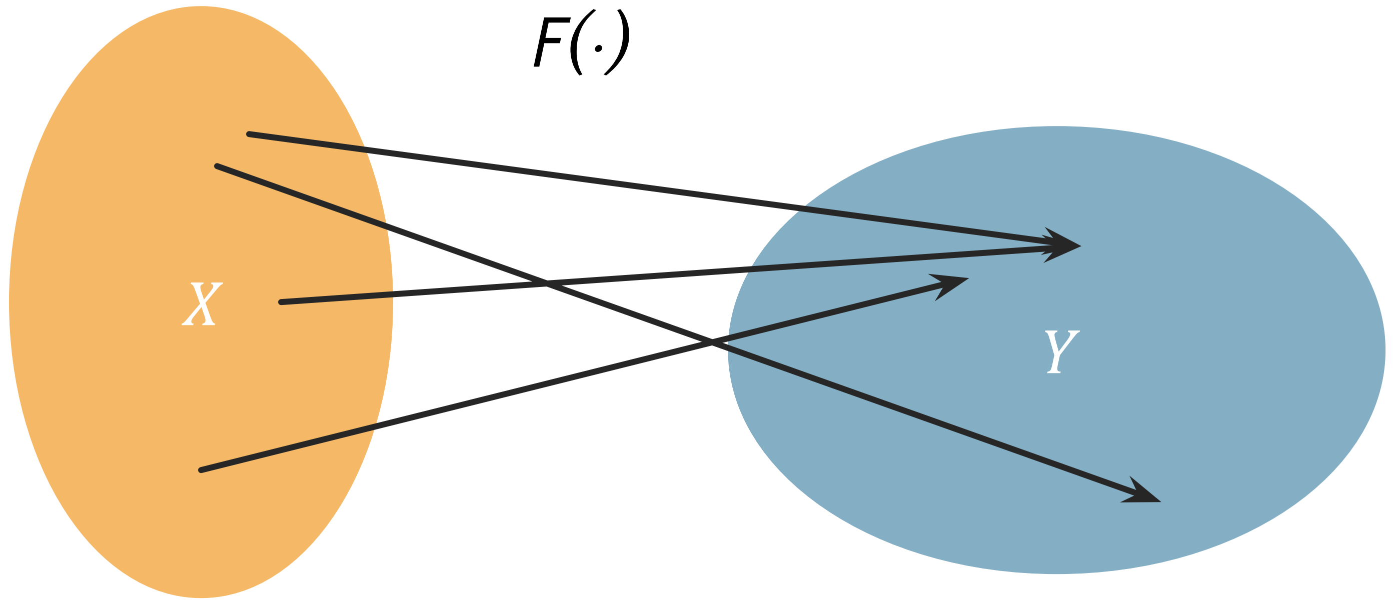
How many possible choices of F are there?

A. |X| * |Y|

B. |X|!

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

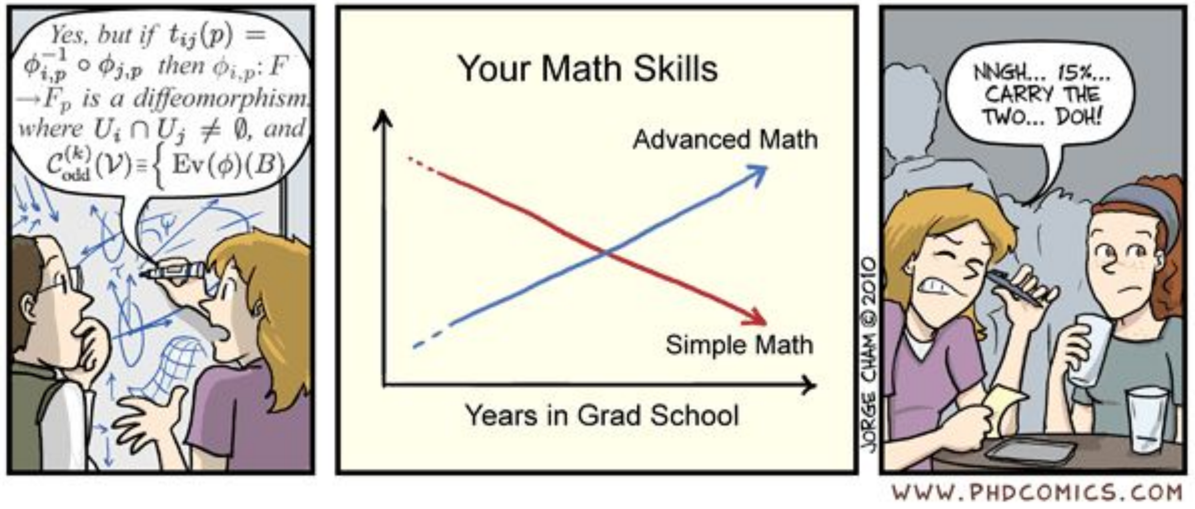
D. $|X|^{|Y|}$



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 $t_{ij}(p) = \phi_{i,p}^{-1} \circ \phi_{j,p}$ 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
NNGH... 15%... CARRY THE TWO... DOH!
JORGE CHAM © 2010
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 $m \in \{0,1\}^b$:
  - Alice sends f(m) to Bob
  - Bob decrypts using f^{-1}

Participation Question

Is this a correct cipher?

A. Yes

B. No

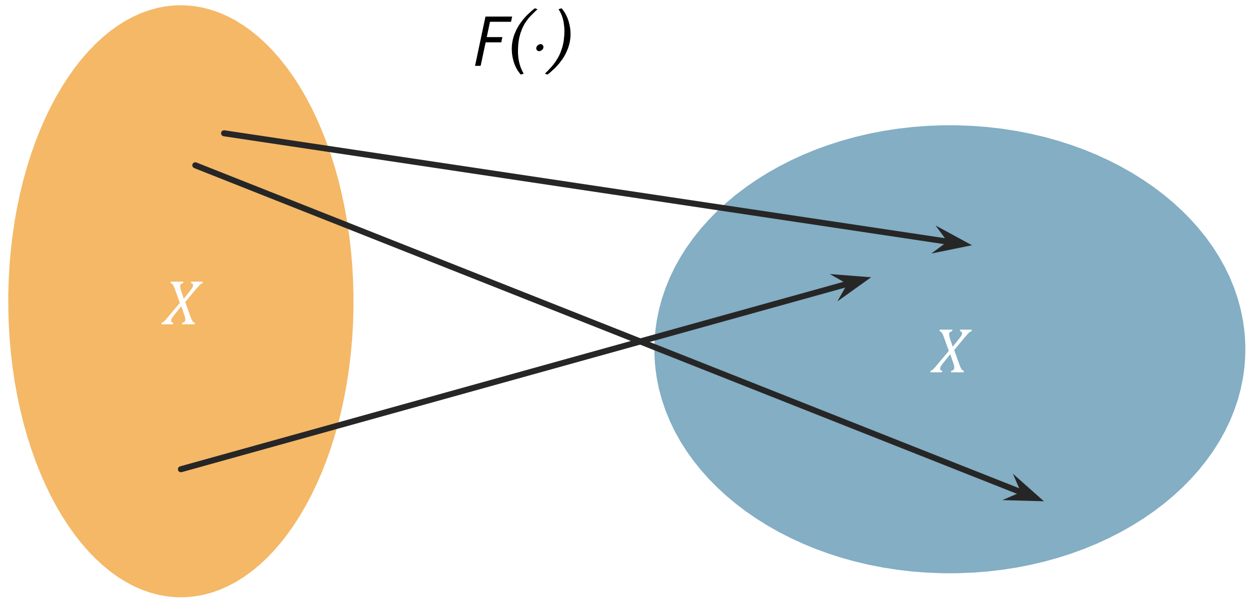
C. I'm not sure

Correctness

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

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

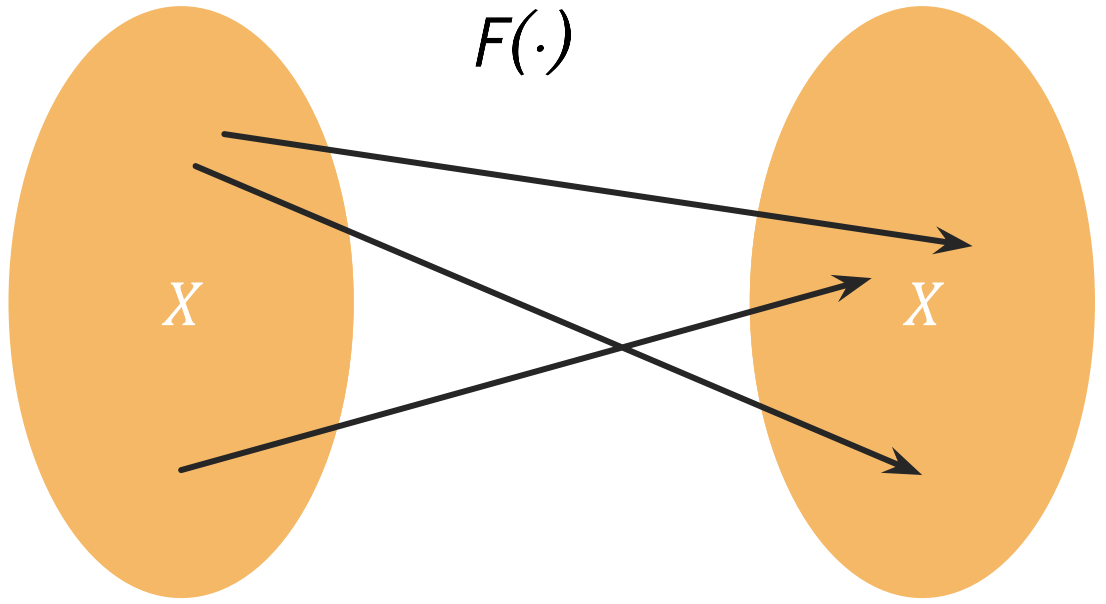
How many possible choices of F are there?

A. $2 * |X|$

B. $|X|^2$

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

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



Better Encryption Scheme?

- Alice chooses f: $\{0,1\}^b \rightarrow \{0,1\}^b$ 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?

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 ∈ $\{0,1\}^b$:
  - Alice sends f(m) to Bob
  - Bob decrypts using f^{-1}

No! Writing down f requires 2^b entries, each of which is b bits b*2^b bits in the ”key” >> Messages are only b bits. We’d be much better off just choosing a one-time pad

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"

Ευχαριστώ και καλή μέρα εύχομαι!

Keep hacking!