CSCI 1510

Introduction to Cryptography and Security

Course Homepage: https://cs.brown.edu/courses/csci1510/fall-2024/

- Introduce Staff
- Syllabus
- Introduction & Overview
- Q & A

Logistics

- **Lectures:** CIT 477 & Zoom (recorded)
- **Office Hour:** 3-4pm Thursdays, CIT 511 & Zoom, or by appointment
- **TA OH:** See course website (calendar)
- **EdStem** / **Gradescope** / **Course Website**
- **Prerequisites / Override:**
  CSCI 0220 & 1010
  Basic algorithms, number theory, discrete probability, complexity theory.
- **Textbook:** Katz-Lindell "Introduction to Modern Cryptography" (3rd Edition)

Class Participation

- Ask / Answer ≥ 5 technical questions throughout the semester,
  from 5 different lectures / Peihan's OH

- Bonus Points: (cap 5 points)
  If you ask a "good" question or give a "good" answer
  (thumbs up from Peihan)

- Keep track of all the questions you've asked / answered
  & bonus points you've earned (see template)
  Submit at the end of the semester.

Homeworks

- Homework 0 + 10
- Due on Fridays, 2 late days for free
- No further extension
- Lowest HW grade will be dropped.

- Collaboration / Google / ChatGPT:
  - Write up your own solution
  - Acknowledge everyone you've worked with
  - Credit all resources you've looked at

Exams

- **Midterm:** Take-Home, 10/18-25
- **Final:** Take-Home, 12/11-18
- No collaboration, no extension
- In each HW, there will be a question for you to synthesize course materials from that week into a one-page summary.

Grading

- 10% Class Participation
- 2% HW 0
- 54% HW 1-10 (best 9 out of 10)
- 14% Midterm
- 20% Final

What is Cryptography (used for)?

Study of techniques for protecting (sensitive/important) information.

Where is Cryptography used in practice?

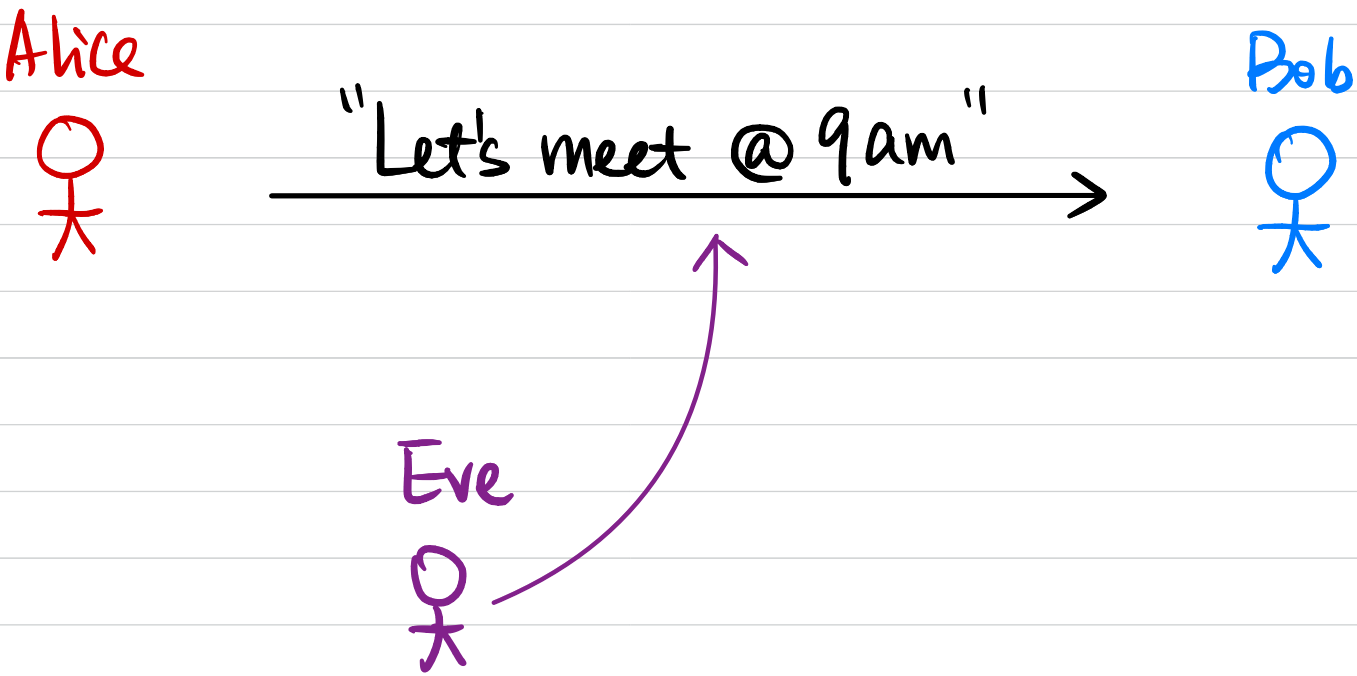
What guarantees do we want in these scenarios?

About the Course

Goal: Learn the theoretical basis of the cryptography in the real world.

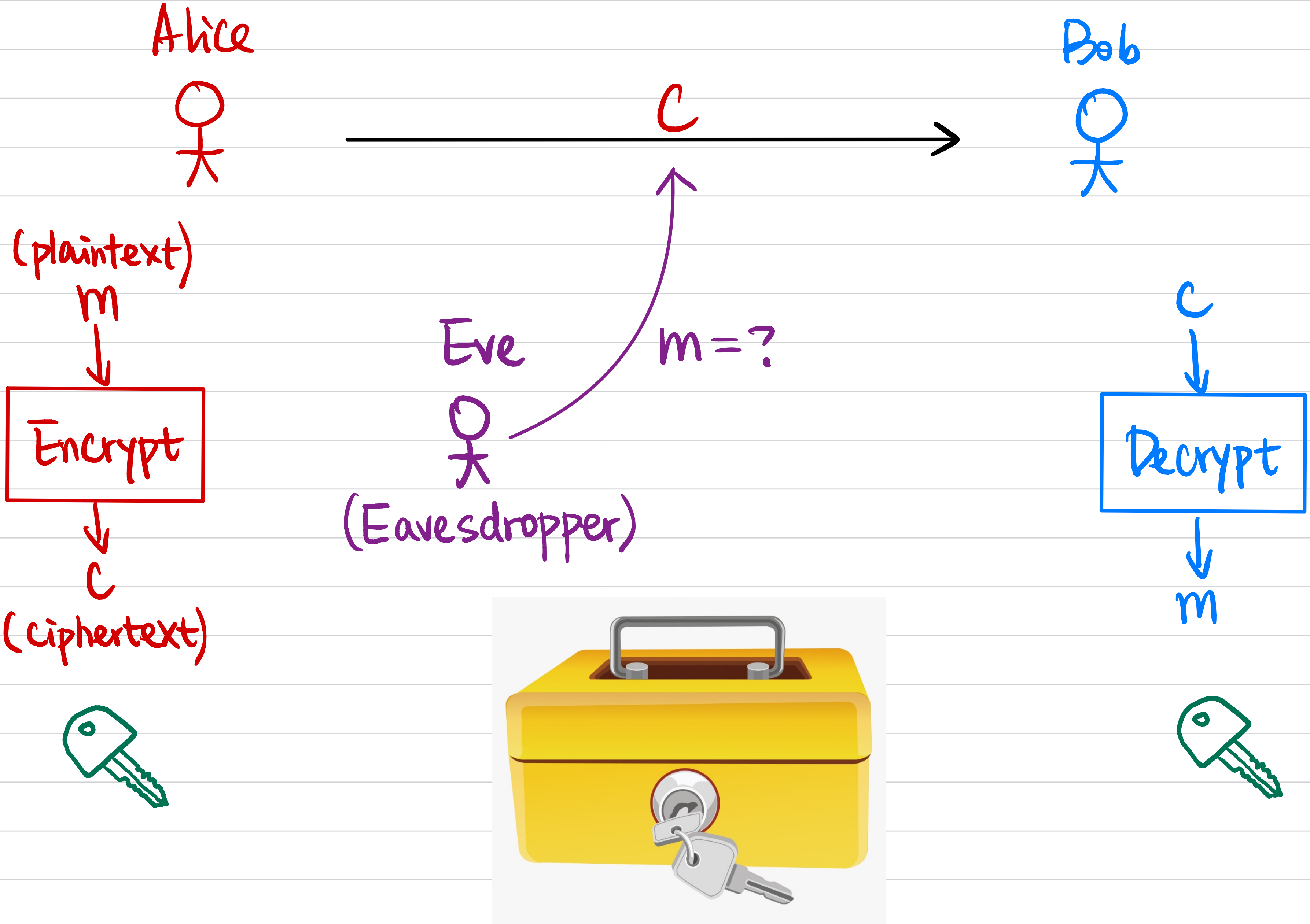
- Learn about key primitives
- Understand what security guarantees they provide
- Learn how to construct and how to prove
- Build up a "crypto mindset"
- Design & Analyze real-word cryptosystems
- Understand crypto papers & standards

Secure Communication



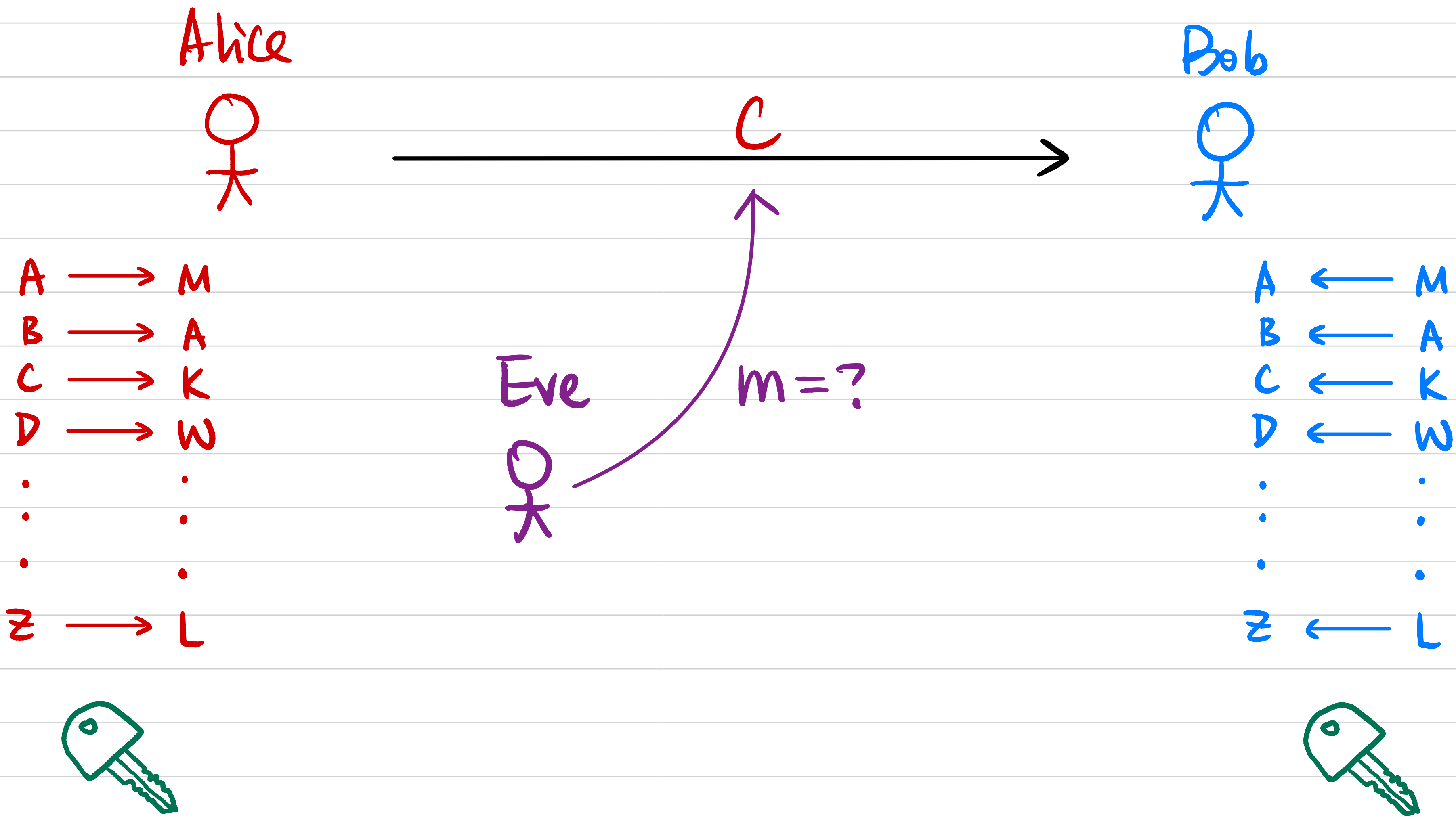
What security guarantee(s) do we want?

Message Secrecy

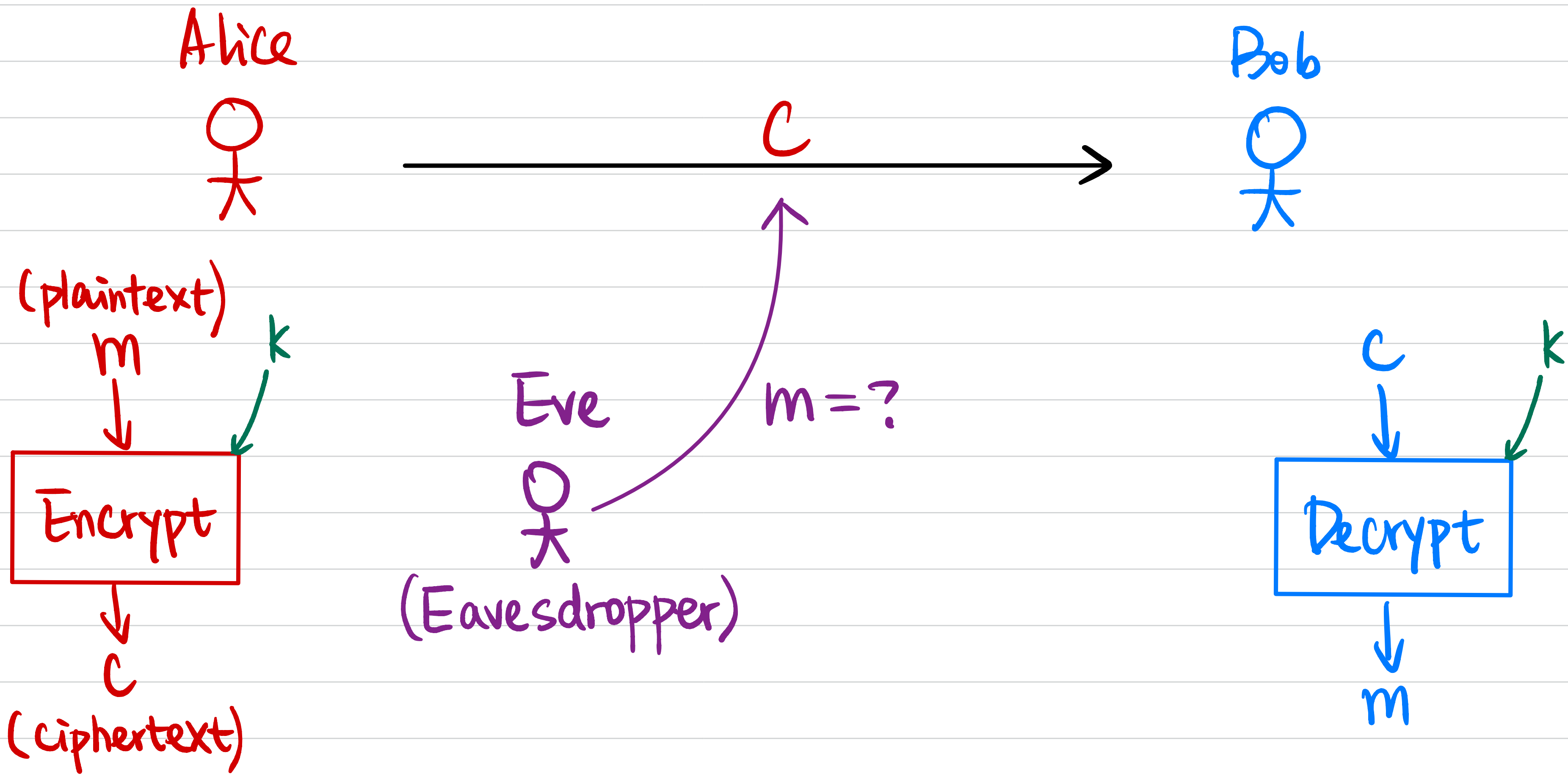


Historical Ciphers

Ex: Substitution Cipher



Modern Cryptography



How to define security ?

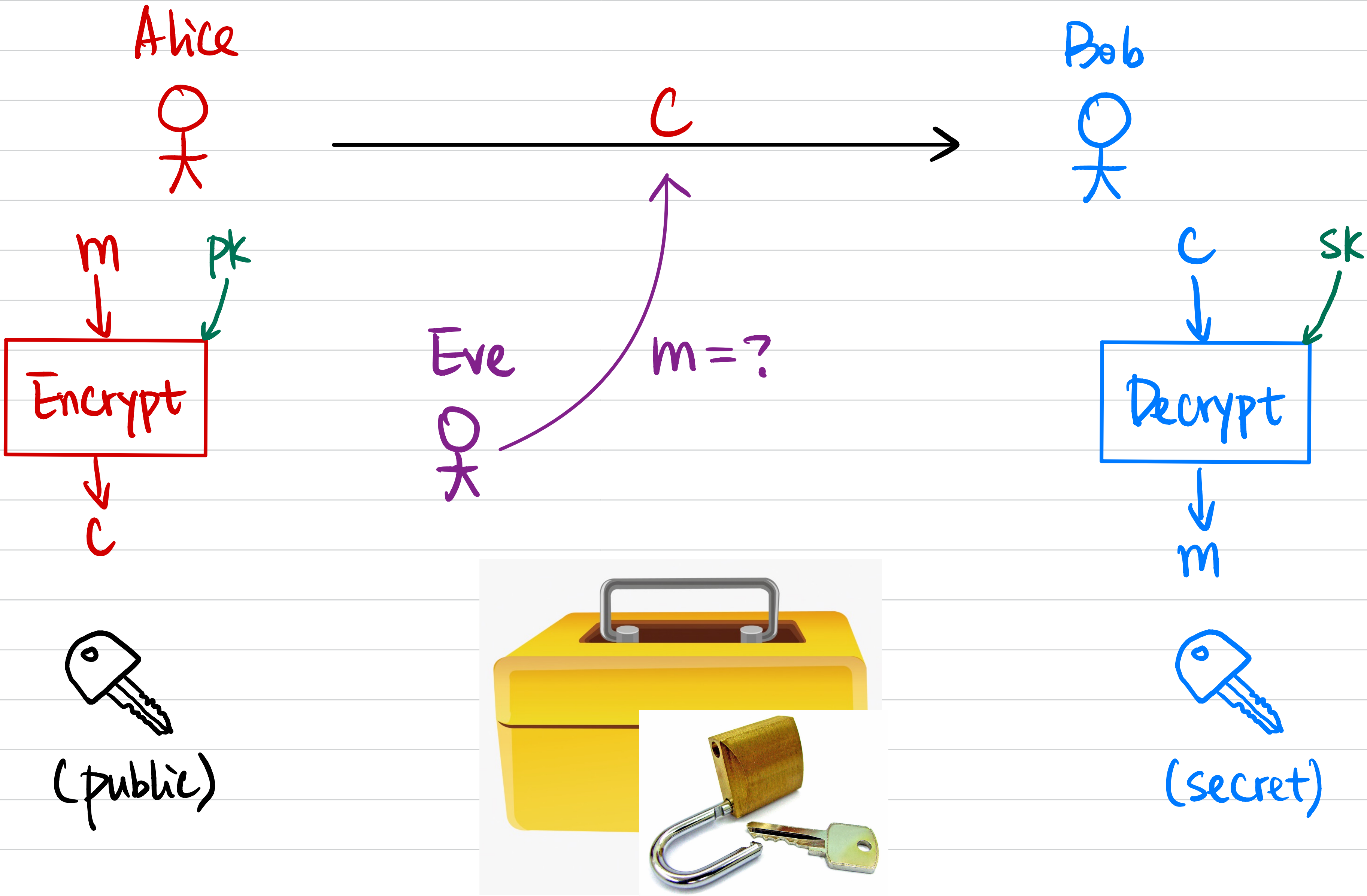
How to define security?

- It's impossible for Eve to recover k from c.

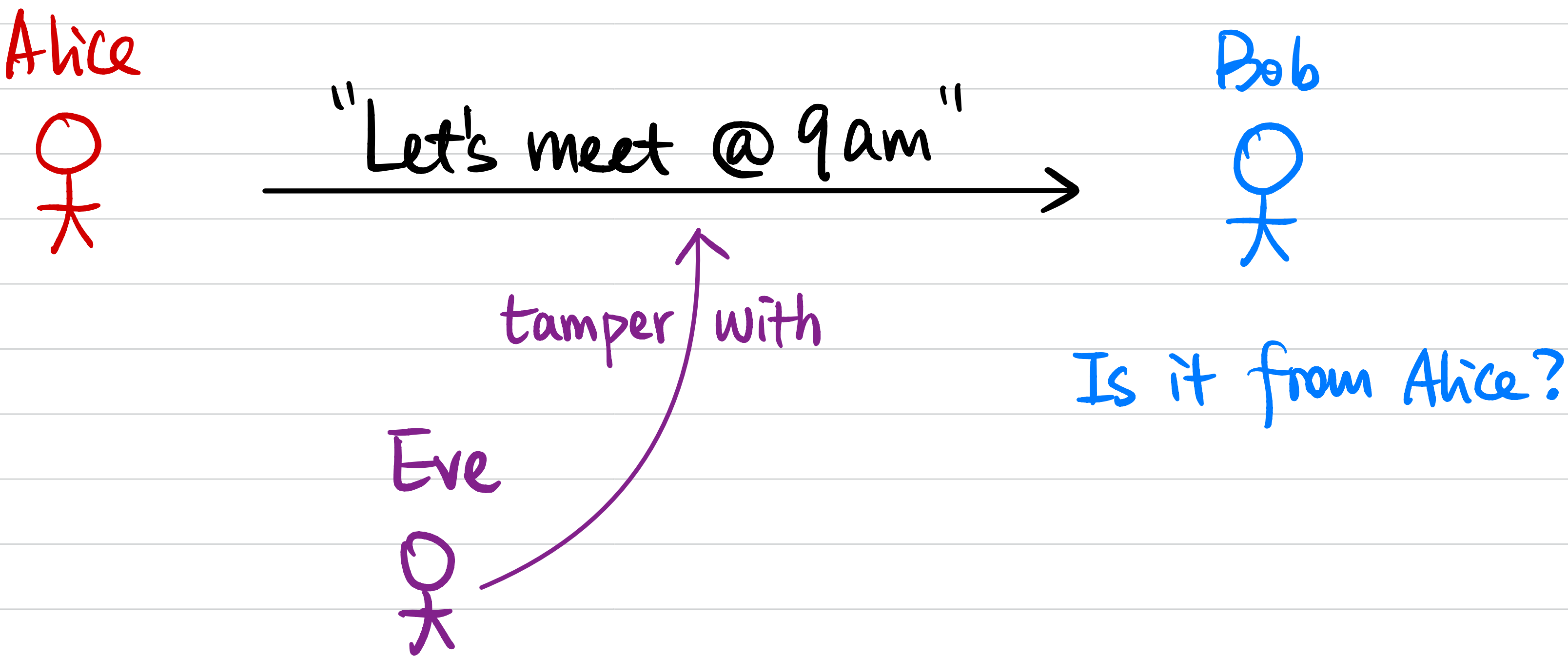
- It's impossible for Eve to recover m from c.

- It's impossible for Eve to recover any character of m from c.

Public-Key Encryption



Message Integrity



Message Integrity

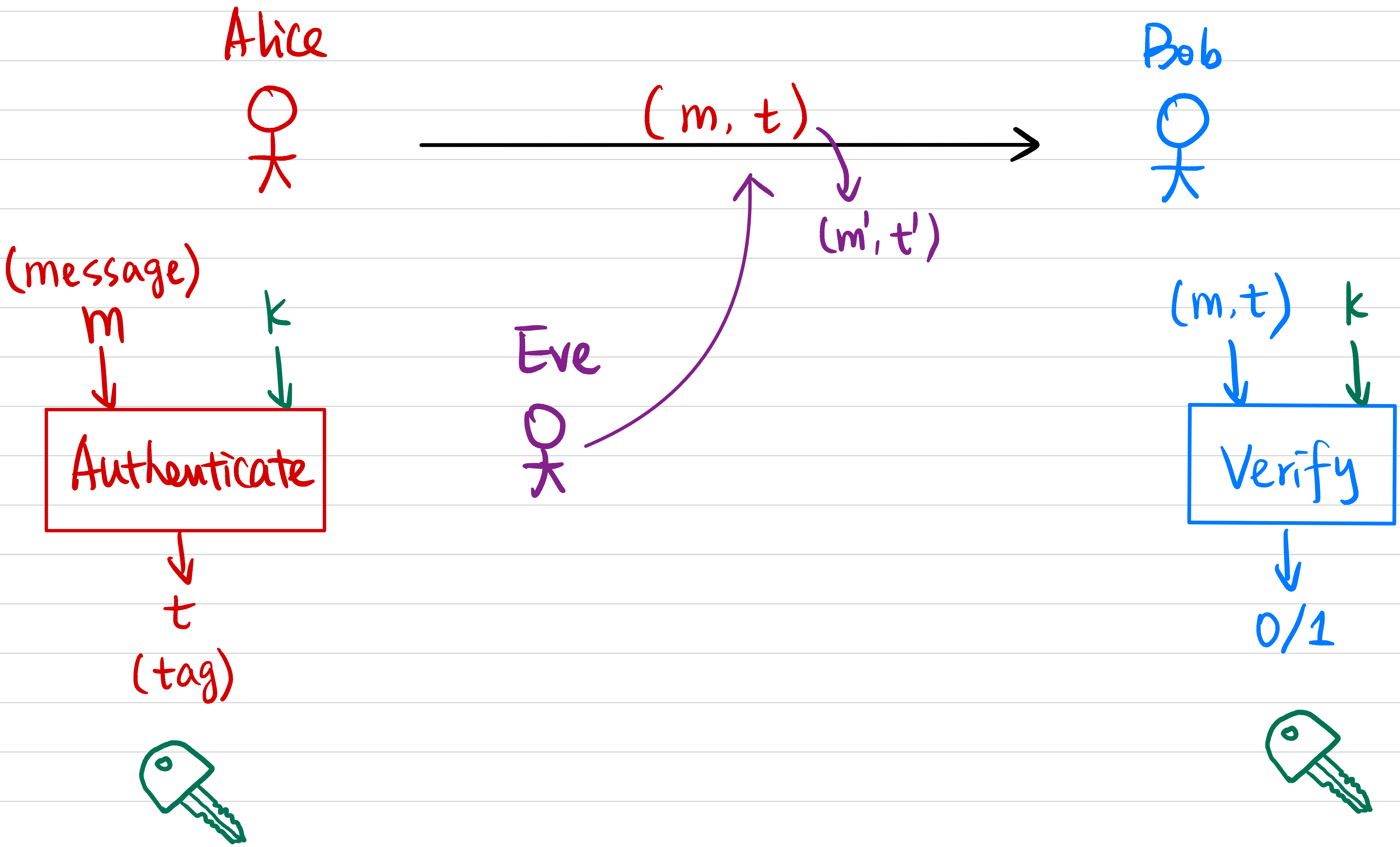
Is it from Google?

http vs. https

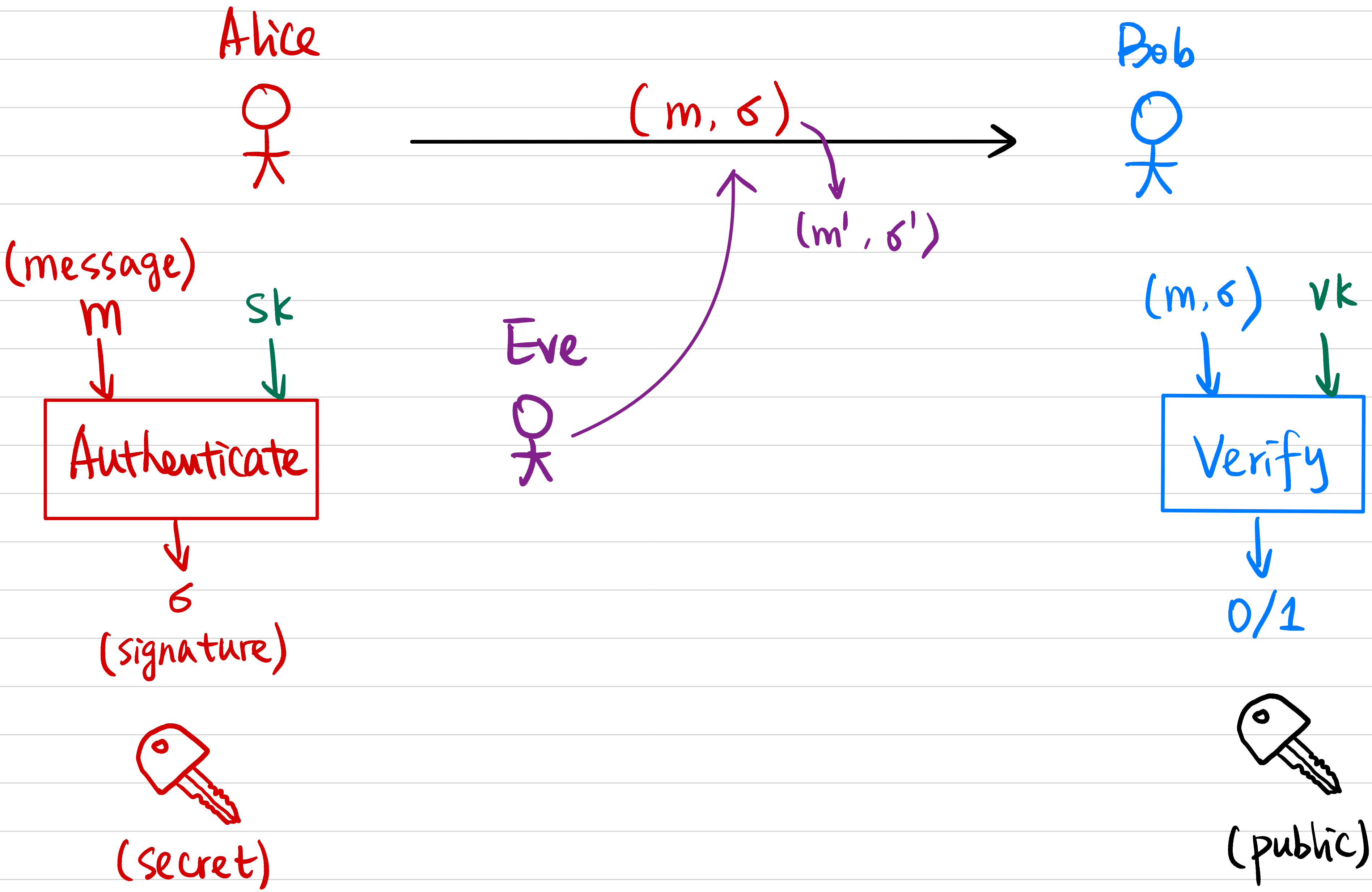
How to achieve message integrity?

Does encryption suffice?

Message Authentication Code (MAC)



Digital Signature



Pseudorandom Number Generator

Sample $r \leftarrow \{0, 1, 2, \cdots, 9\}$

$r := \text{rand}(\text{seed})$

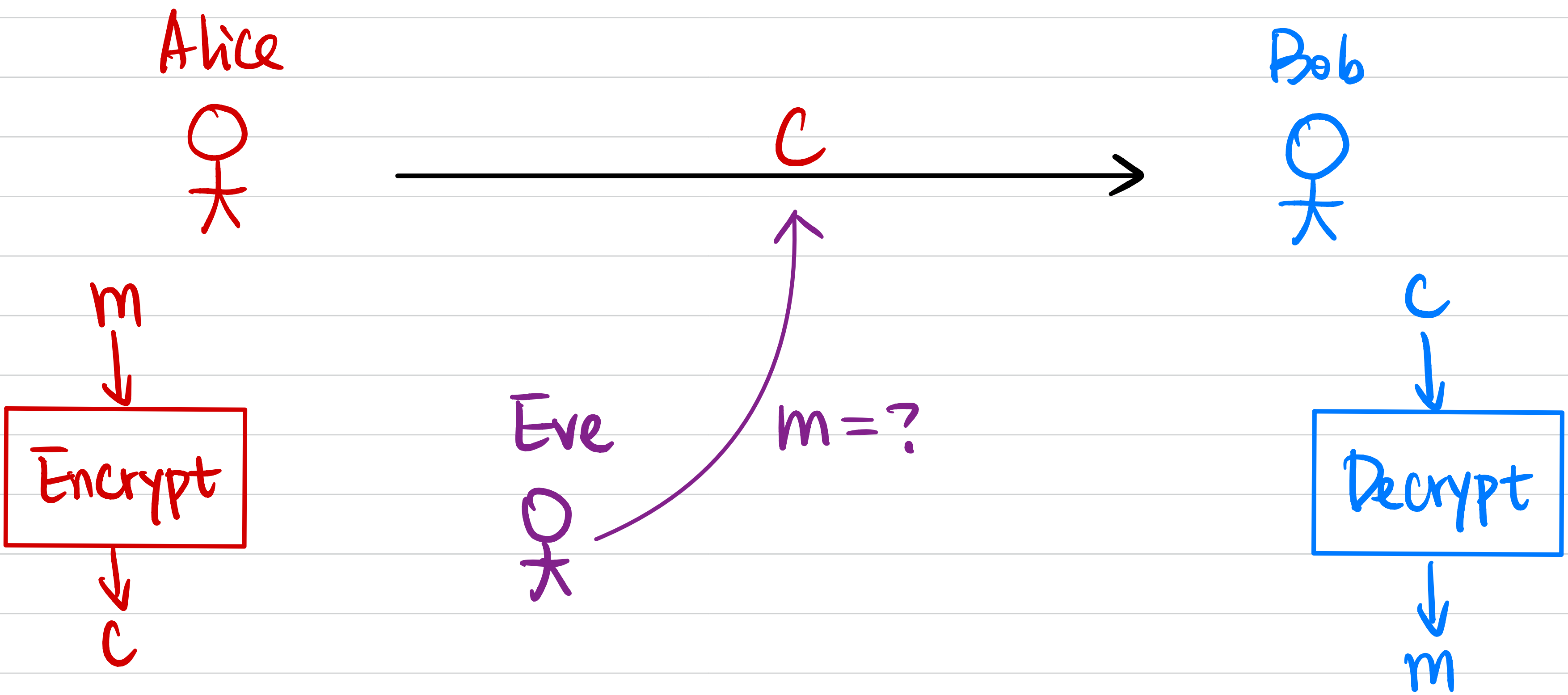
rand ↑ deterministic   seed ↑ timestamp

How to define "pseudorandomness"?

Overview

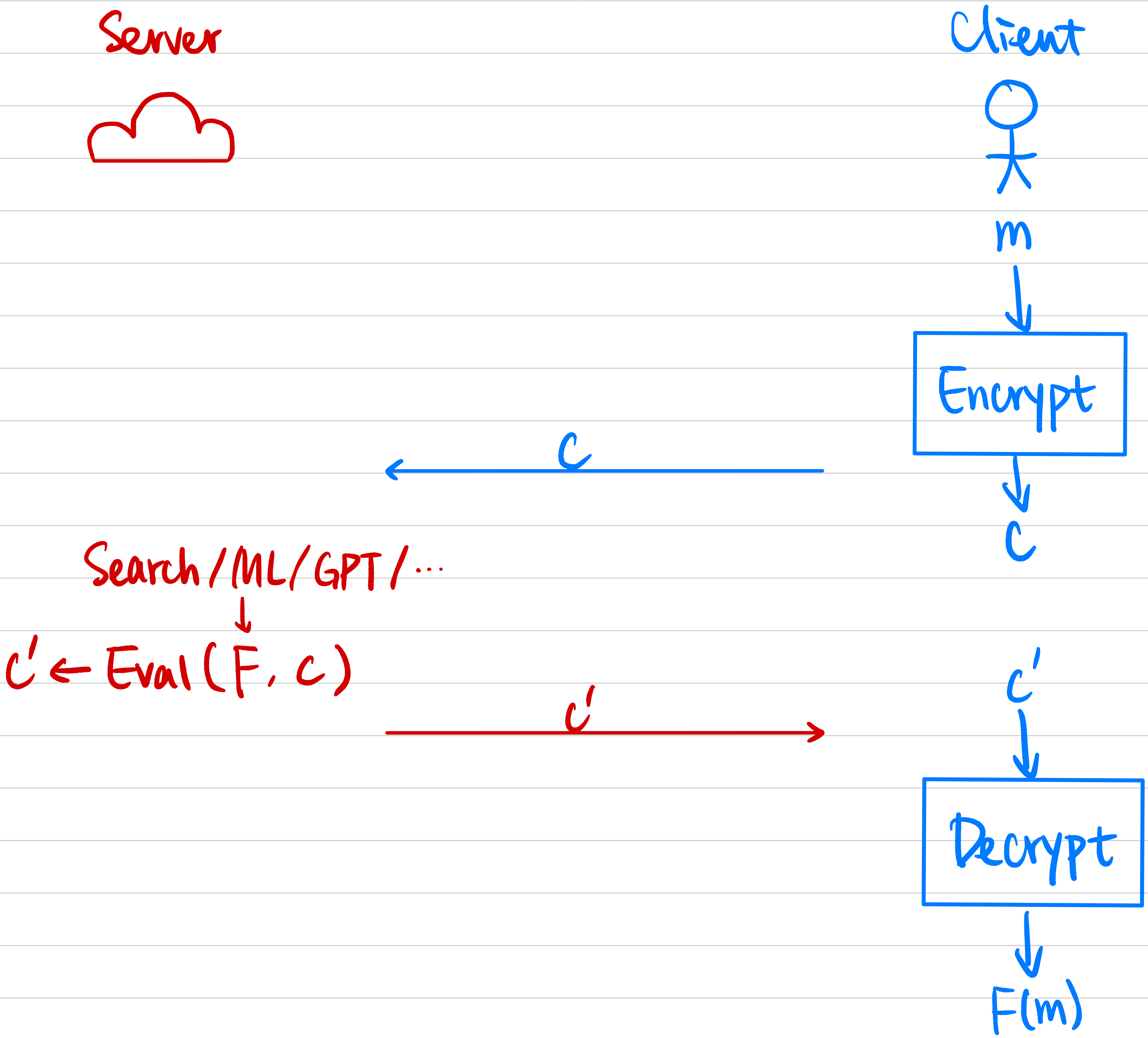
- Message Secrecy: symmetric-/public-key encryption
- Message Integrity:
  - Message Authentication Codes
  - Digital Signatures
- Key Primitives:
  - Pseudorandom Generator / Pseudorandom Function / Hash Function
  - Computational Assumptions: RSA / DLOG / Diffie-Hellman
- Encryption with Advanced Properties:
  - Fully Homomorphic Encryption (post-quantum security)
- Secure Protocols:
  - Zero-Knowledge Proofs
  - Secure Multi-Party Computation
- Program Obfuscation

Fully Homomorphic Encryption (FHE)



$$c_1 = Enc(m_1)$$
$$c_2 = Enc(m_2)$$
$$\Rightarrow \quad c' = Enc(m_1 + m_2)$$
$$c'' = Enc(m_1 \cdot m_2)$$

Example: Privacy-Preserving Query



Zero-Knowledge Proof (ZKP)

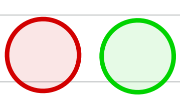
Alice

Bob

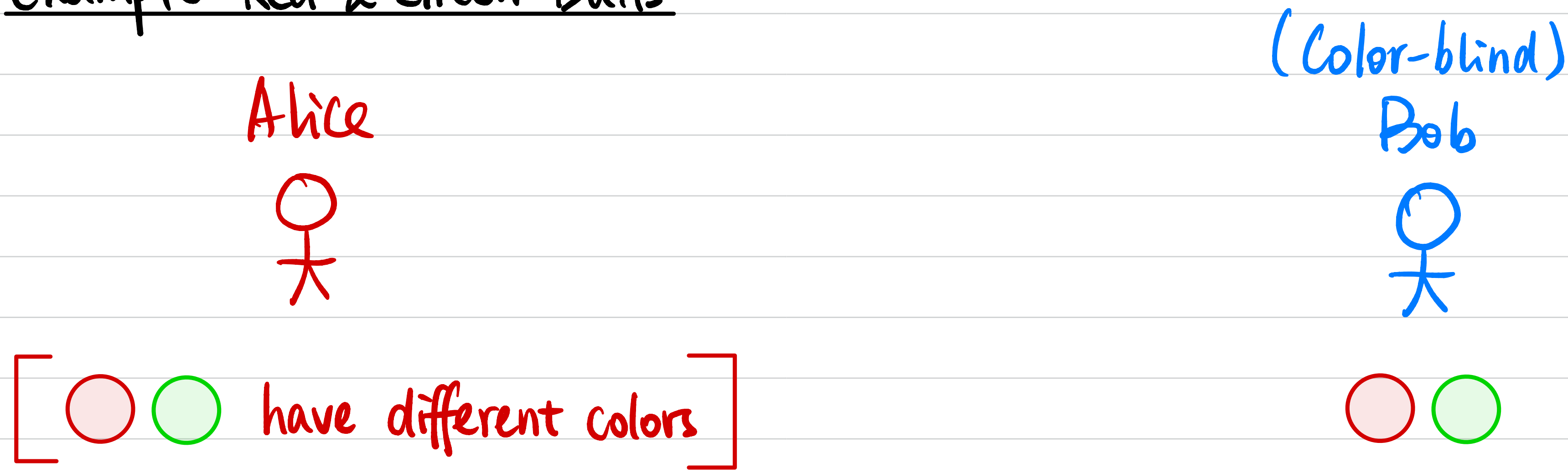
[There is a bug in your code]

[I have the secret key for this ciphertext]

[There is enough balance in my Bitcoin account]

[ have different colors]

Example: Red & Green Balls

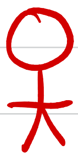


If statement is true:

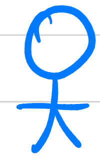
If statement is false:

Secure Multi-Party Computation (MPC)

Alice



Bob

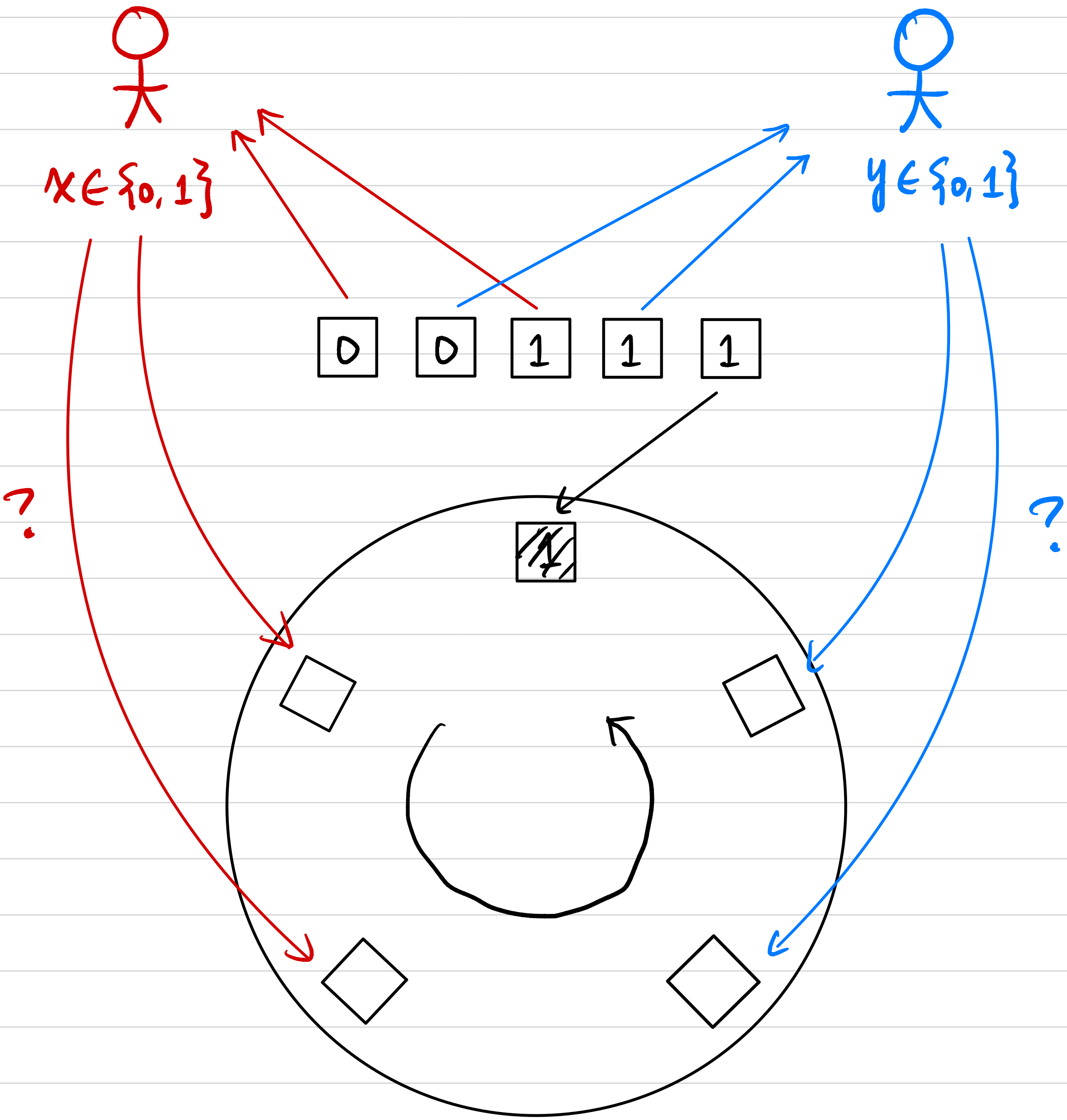


Second date?

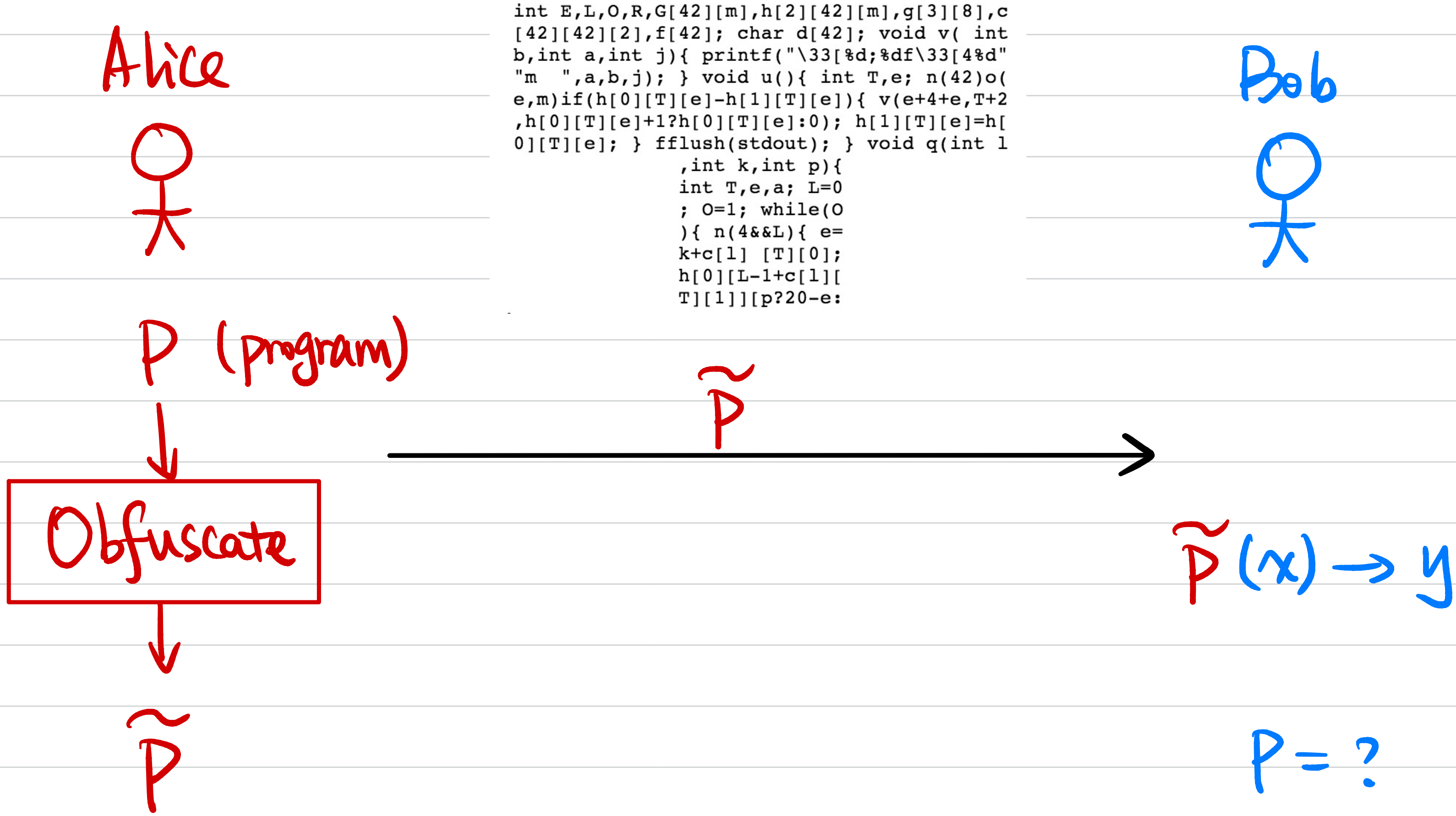
Who is richer?

Mutual friends?

Example: Private Dating



Program Obfuscation



Q & A

- Crypto background?
- Readings before/after lecture?
- CSCI 1040 (The Basics of Cryptographic Systems) "Crypto for poets"

  MATH 1580 (Cryptography) Why is it correct?

  CSCI 1510 Why is it secure?

  CSCI 1515 (Applied Cryptography) How to use it?

Encryt-then-MAC

$Gen(1^\lambda)$:

$k_1 \leftarrow Gen_1(1^\lambda)$

$k_2 \leftarrow Gen_2(1^\lambda)$

Output $k = (k_1, k_2)$

$Enc_k(m)$:

$c_1 \leftarrow Enc_1(k_1, m)$

$t_2 \leftarrow Mac_2(k_2, c_1)$

output $c = (c_1, t_2)$

$Dec_k(c)$: $c = (c_1, t_2)$

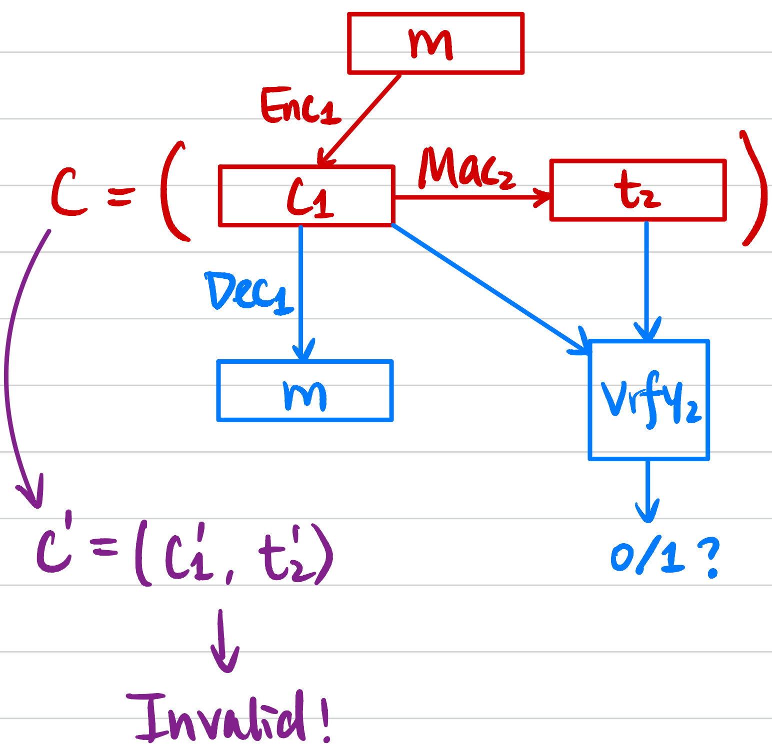
$m := Dec_1(k_1, c_1)$

$b := Vrfy_2(k_2, (c_1, t_2))$

If $b = 1$, output m

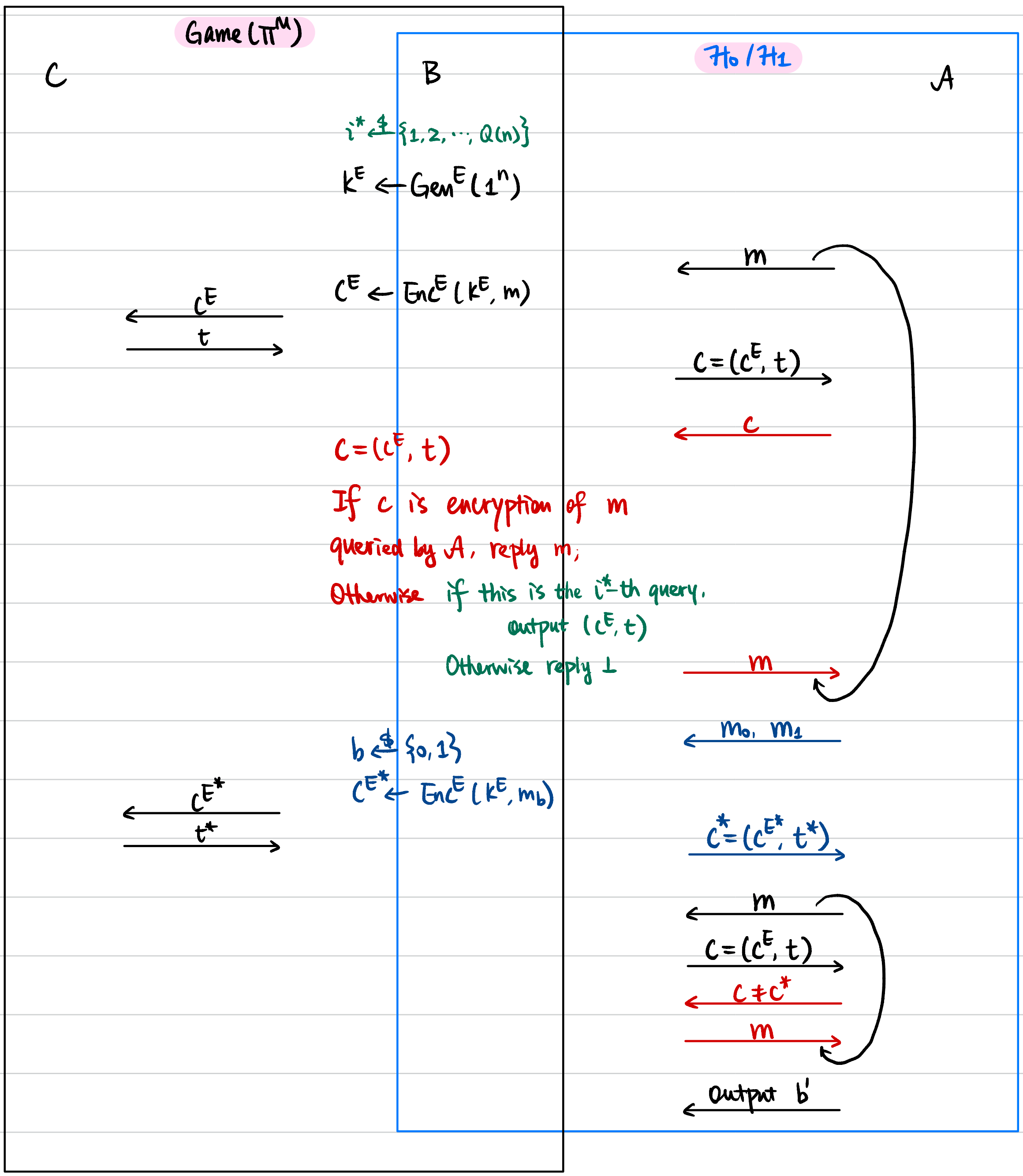
Otherwise output $\perp$

CCA-secure & Unforgeable



Lemma 1 $\forall$ PPT $\mathcal{A}$, $|\Pr[\mathcal{A} \text{ outputs } 1 \text{ in } \mathcal{H}_0] - \Pr[\mathcal{A} \text{ outputs } 1 \text{ in } \mathcal{H}_1]| \le \text{negl}(n)$.

Proof Assume not, then $\exists$ PPT $\mathcal{A}$ that distinguishes $\mathcal{H}_0$ & $\mathcal{H}_1$ with non-negligible probability $\varepsilon(n)$.



It must be the case that $\mathcal{A}$ queries for decryption of a new, valid ciphertext with probability at least $\varepsilon(n)$.

We construct a PPT B to break the strong security of Π^M.

$Q(n) := \max \# \text{ of queries by } \mathcal{A}$.

$\Pr[B \text{ outputs a valid new pair } (c^E, t)]$

$\ge \varepsilon(n) \cdot \frac{1}{Q(n)}$ → non-negligible

2023 IEEE Symposium on Security and Privacy (SP)

Weak Fiat-Shamir Attacks on Modern Proof Systems

Year: 2023, Pages: 199-216
DOI Bookmark: 10.1109/SP46215.2023.10179408

Authors

Quang Dao, Carnegie Mellon University
Jim Miller, Trail of Bits
Opal Wright, Trail of Bits
Paul Grubbs, University of Michigan

DOWNLOAD PDF SHARE ARTICLE GENERATE CITATION

Abstract

A flurry of excitement amongst researchers and practitioners has produced modern proof systems built using novel technical ideas and seeing rapid deployment, especially in cryptocurrencies. Most of these modern proof systems use the Fiat-Shamir (F-S) transformation, a seminal method of removing interaction from a protocol with a public-coin verifier. Some prior work has shown that incorrectly applying F-S (i.e., using the so-called "weak" F-S transformation) can lead to breaks of classic protocols like Schnorr's discrete log proof; however, little is known about the risks of applying F-S incorrectly for modern proof systems seeing deployment today.In this paper, we fill this knowledge gap via a broad theoretical and practical study of F-S in implementations of modern proof systems. We perform a survey of open-source implementations and find 30 weak F-S implementations affecting 12 different proof systems. For four of these—Bulletproofs, Plonk, Spartan, and Wesolowski's VDF—we develop novel knowledge soundness attacks accompanied by rigorous proofs of their efficacy. We perform case studies of applications that use vulnerable implementations, and demonstrate that a weak F-S vulnerability could have led to the creation of unlimited currency in a private smart contract platform. Finally, we discuss possible mitigations and takeaways for academics and practitioners.