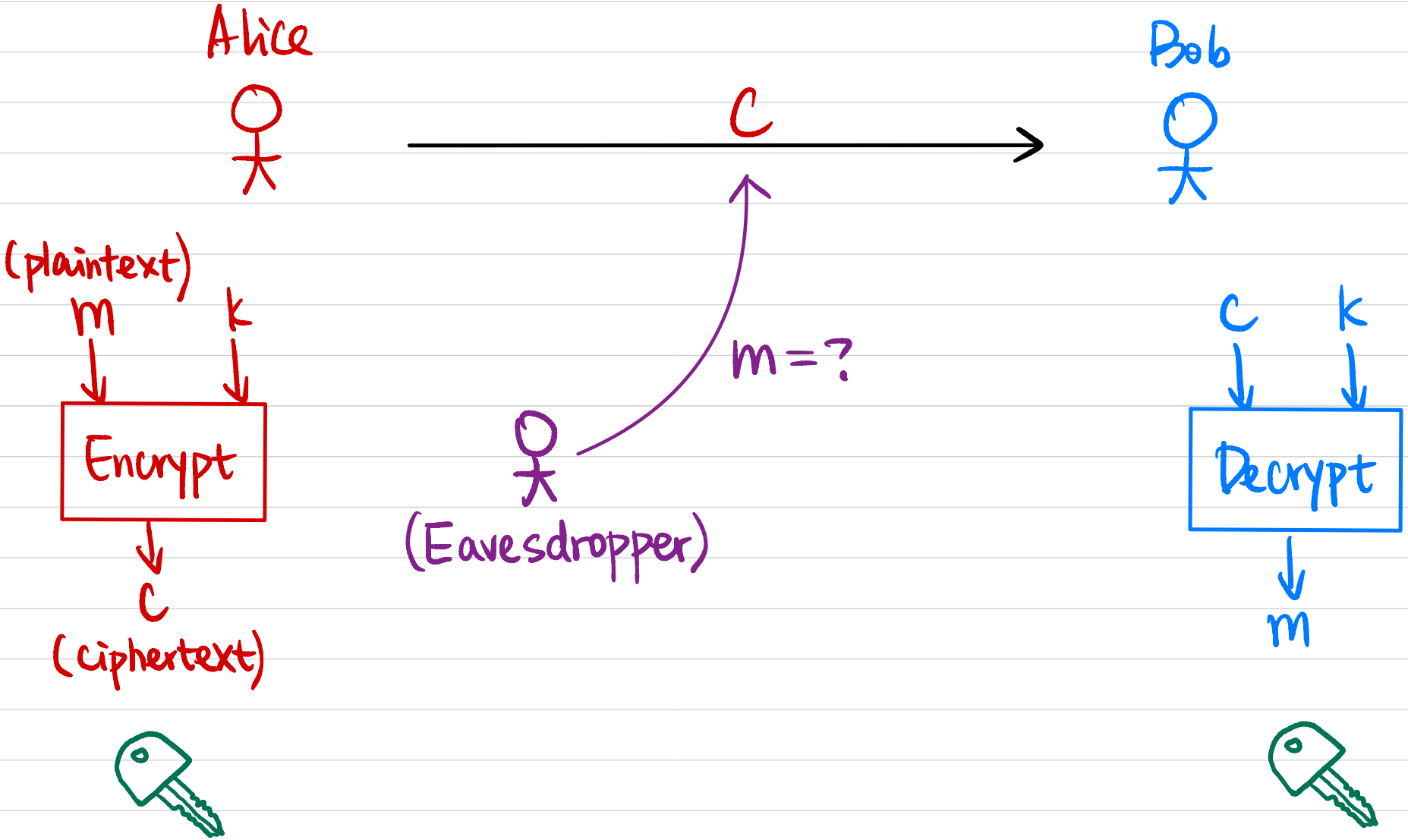
CSCI 1515 Applied Cryptography

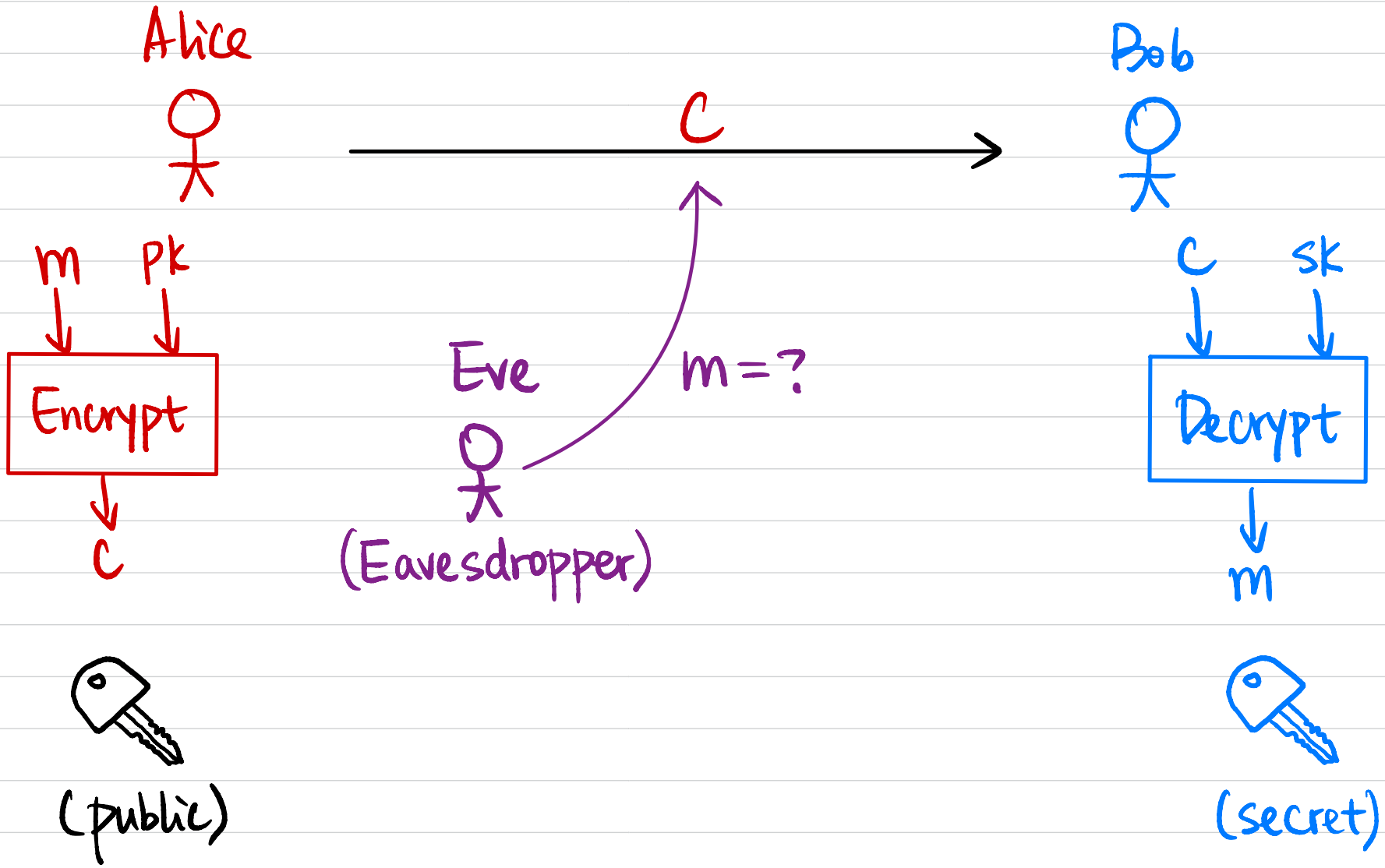
This Lecture:

- Encryption Scheme Basics
- One-Time Pad (OTP)
- Computational Assumptions
- RSA Assumption / Encryption

Message Secrecy: Symmetric-Key Encryption



Message Secrecy: Public-Key Encryption



Syntax

Symmetric-Key Encryption (SKE) Scheme $\Pi = (Gen, Enc, Dec)$

$k \leftarrow Gen$

$c \leftarrow Enc(k, m)$ $Enc_k(m)$

$m := Dec(k, c)$ $Dec_k(c)$

Public-Key Encryption (PKE) Scheme $\Pi = (Gen, Enc, Dec)$

$(pk, sk) \leftarrow Gen$

$c \leftarrow Enc(pk, m)$ $Enc_{pk}(m)$

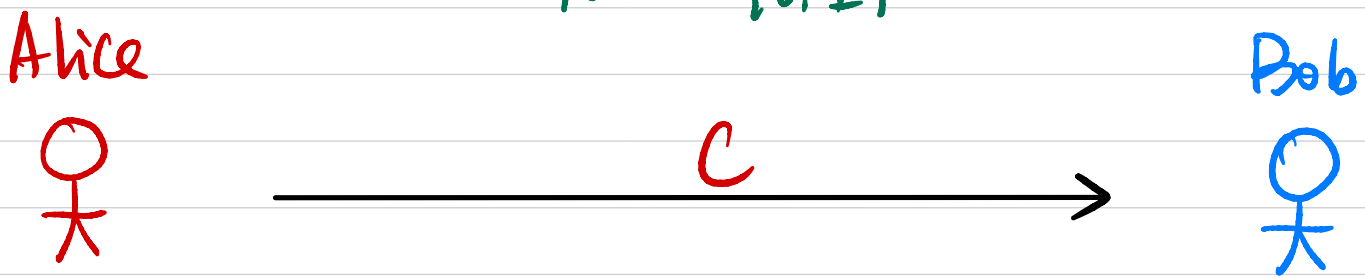
$m := Dec(sk, c)$ $Dec_{sk}(c)$

Why ever using SKE ?

- Efficiency
- PKI Infrastructure
- Quantum Attacks

One-Time Pad (OTP)

$k \xleftarrow{\$} \{0,1\}^n$



Encrypt:

Secret key $k = 0100101$

$\oplus$ plaintext $m = 1001001$

---

ciphertext $c = 1101100$

m \ k	$\oplus$ 0	1
0	0	1
1	1	0

Correctness?

Security?

Decrypt:

Secret key $k = 0100101$

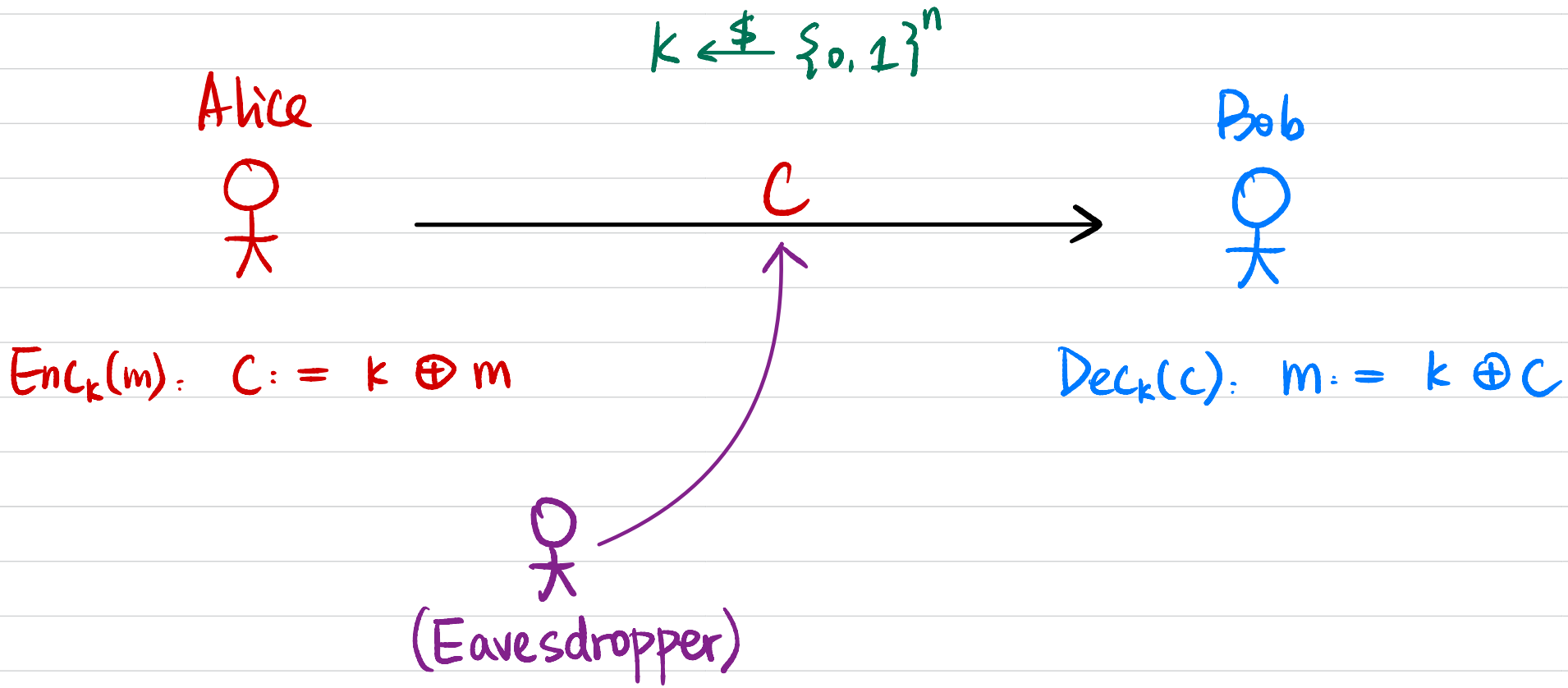
$\oplus$ ciphertext $c = 1101100$

---

plaintext $m = 1001001$

$$k \oplus (k \oplus m)$$
$$= (k \oplus k) \oplus m$$
$$= 00\cdots0 \oplus m$$
$$= m$$

One-Time Pad (OTP)



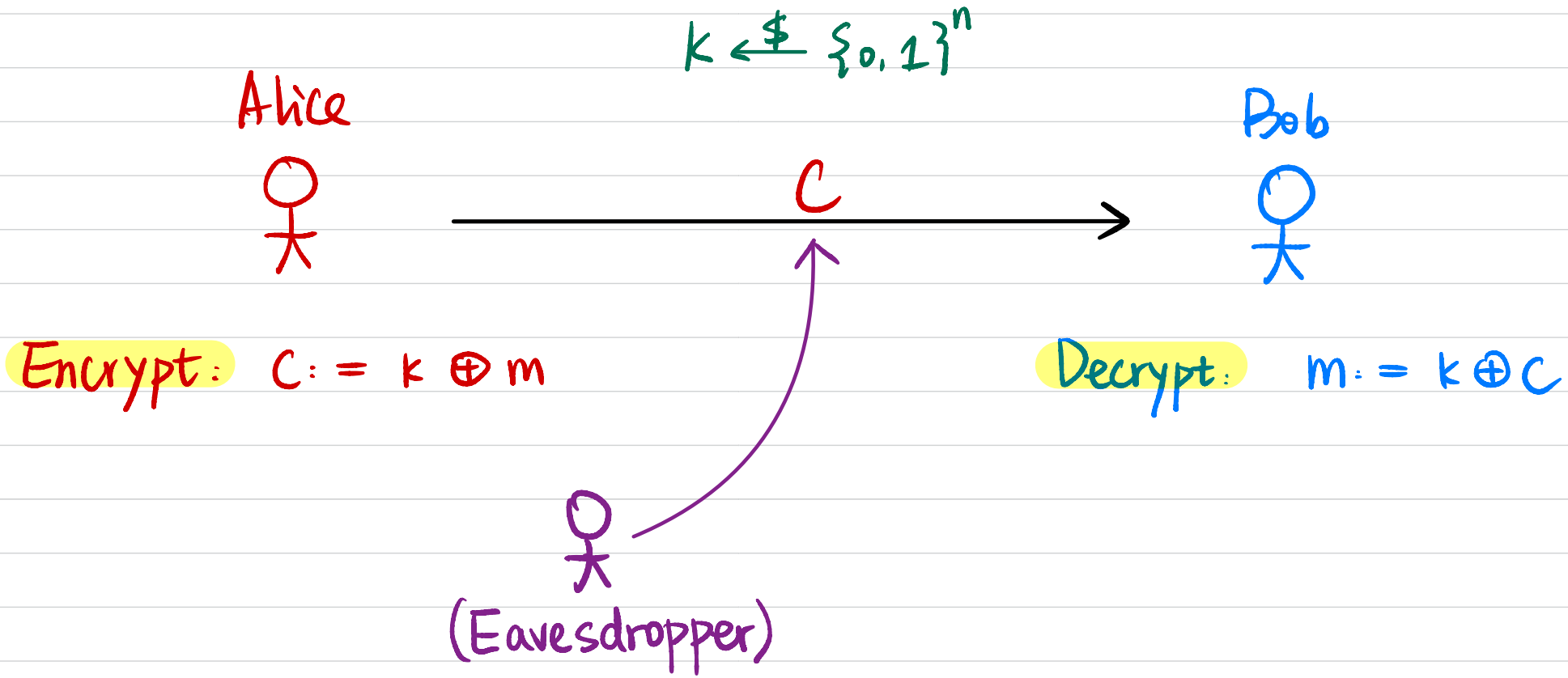
Distribution of c? $\forall m \in \{0,1\}^n$ $c \sim$ uniform over $\{0,1\}^n$

$\forall m_0, m_1 \in \{0,1\}^n$ $Enc_k(m_0) \equiv Enc_k(m_1)$

↑ same distribution

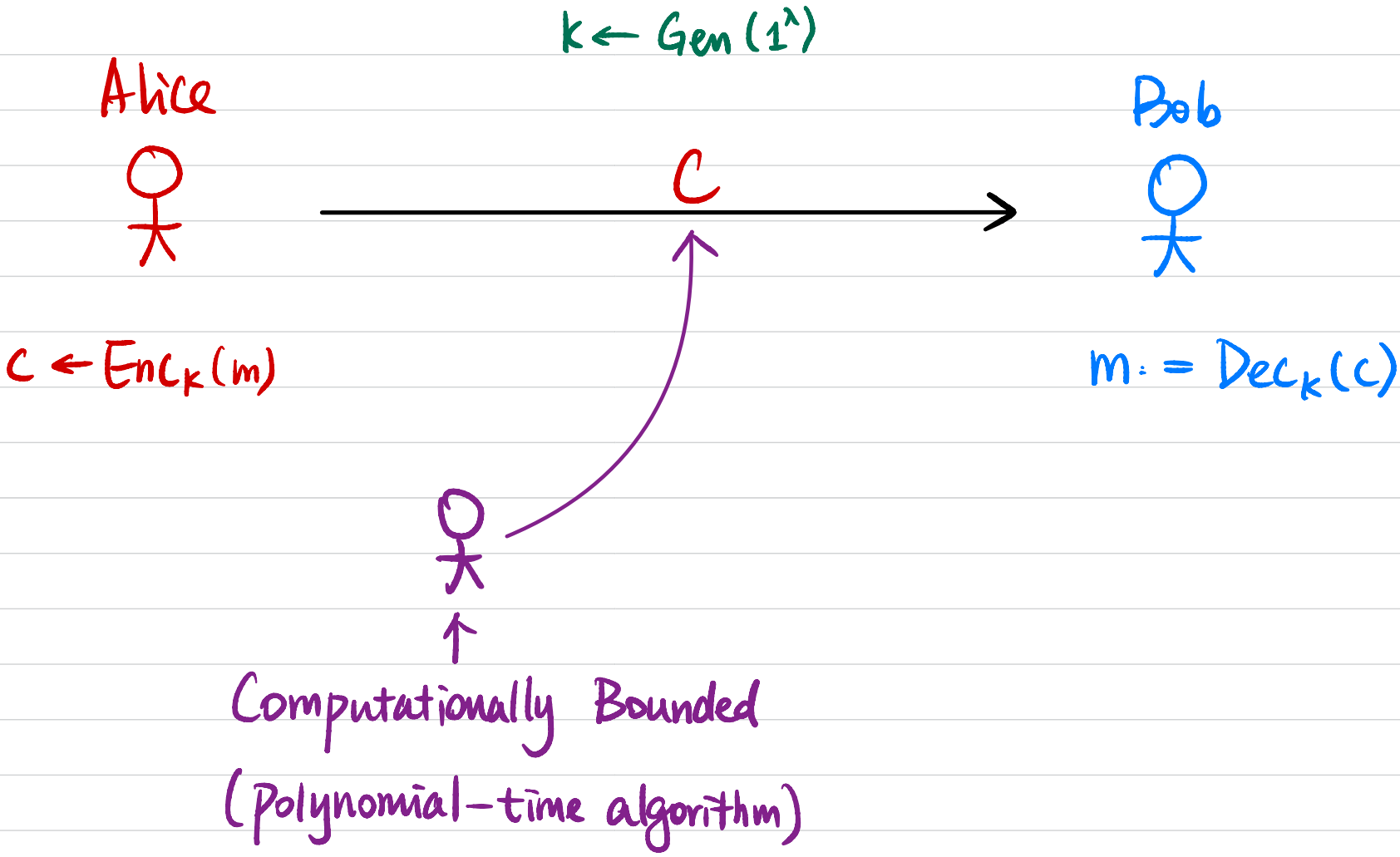
Can we re-use k? No! $\left.\begin{matrix} Enc_k(m_1) = k \oplus m_1 \\ Enc_k(m_2) = k \oplus m_2 \end{matrix}\right\} \oplus \Rightarrow m_1 \oplus m_2$

Shannon's Theorem



(Informal) For perfect (information-theoretic) security, $n \geq |m|$

Computational Security



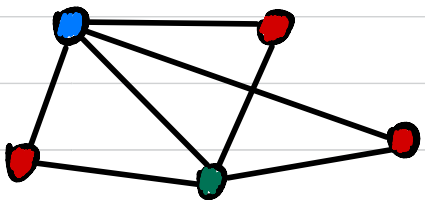
Computational Assumptions

polynomial-time algorithm : $A(x)$

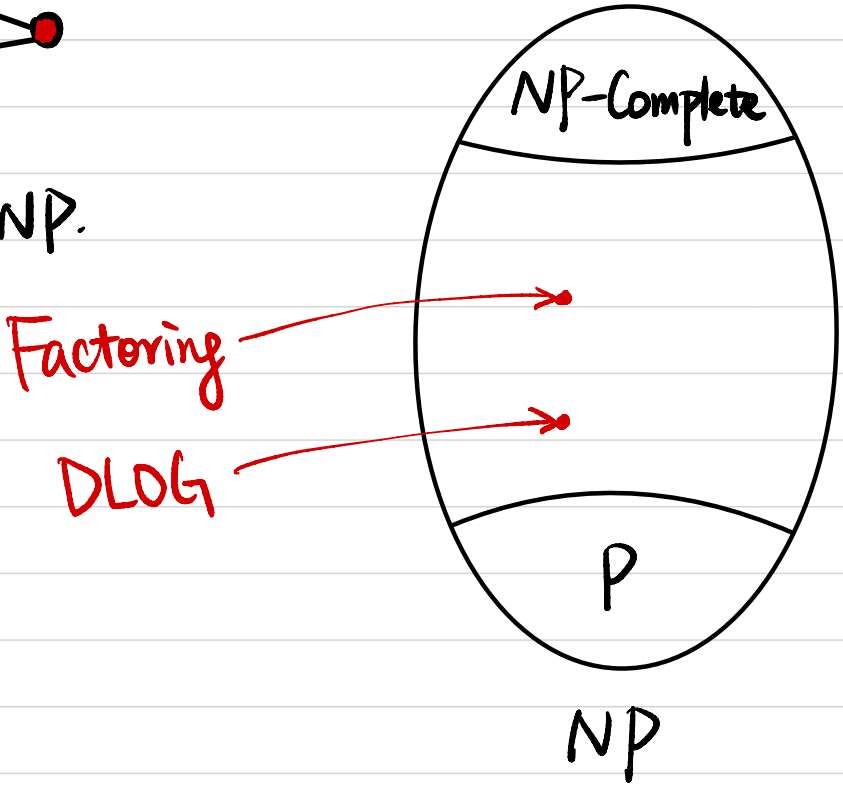
Input x of length n, A's running time $O(n^c)$ for a constant c.

NP Problem : decision problems whose solution can be verified in poly time.

Ex: Gragh 3-Coloring



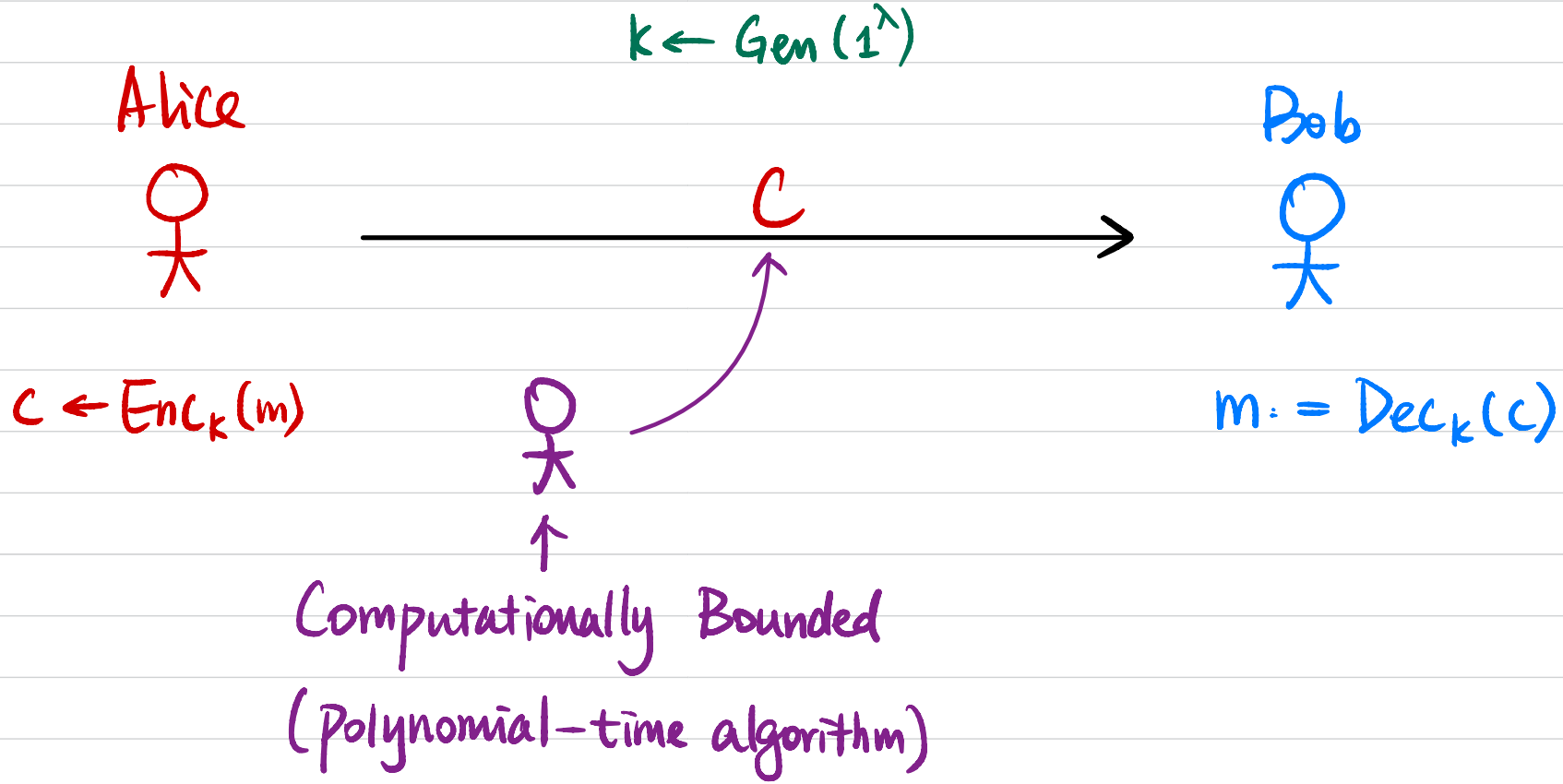
NP-Complete Problems: "hardest" problems in NP.



Is $P = NP$?

Assume NOT !

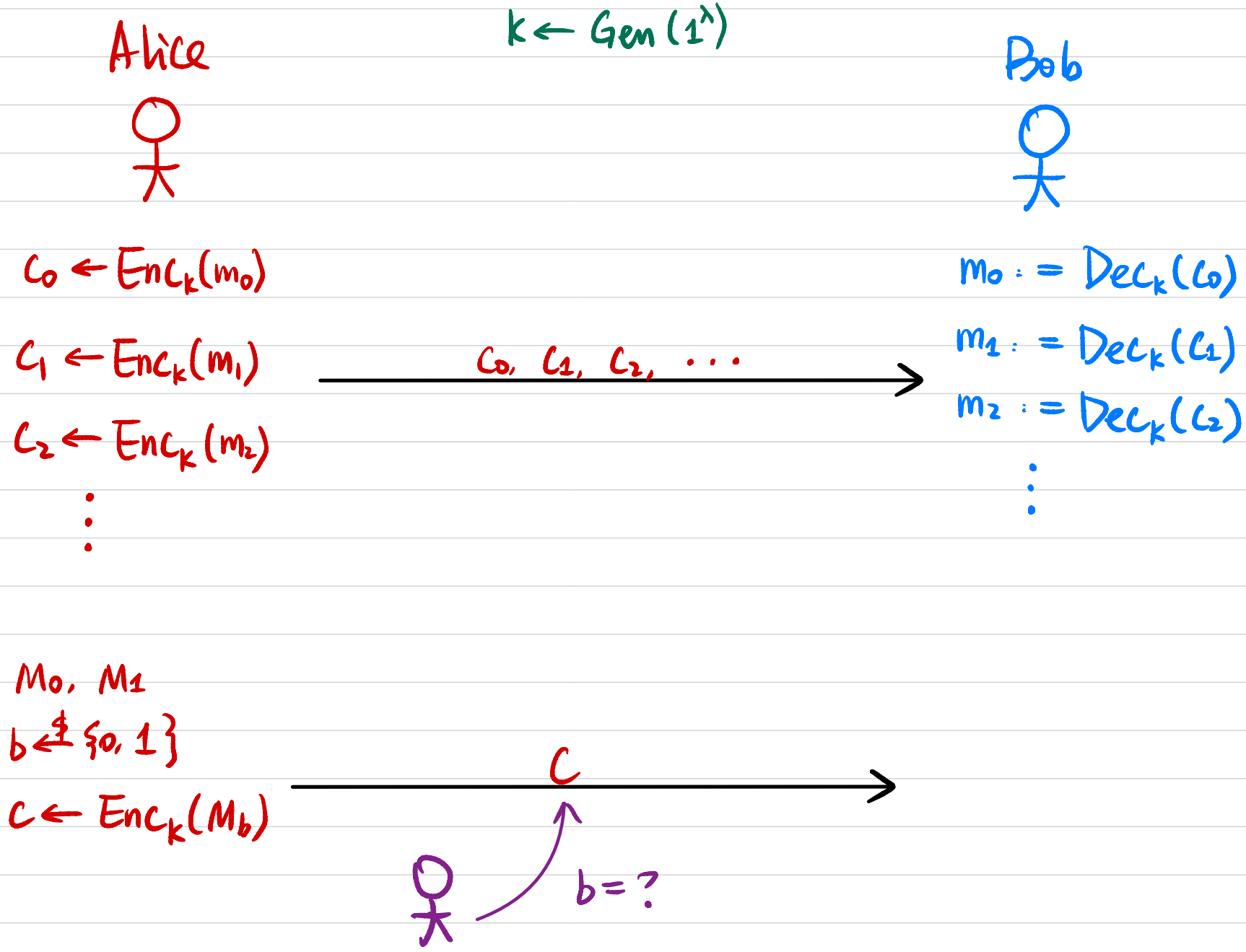
Computational Security



$\forall$ probabilistic poly-time (PPT) $\mathcal{A}$. $Enc_k(m_0) \overset{c}{\approx} Enc_k(m_1)$

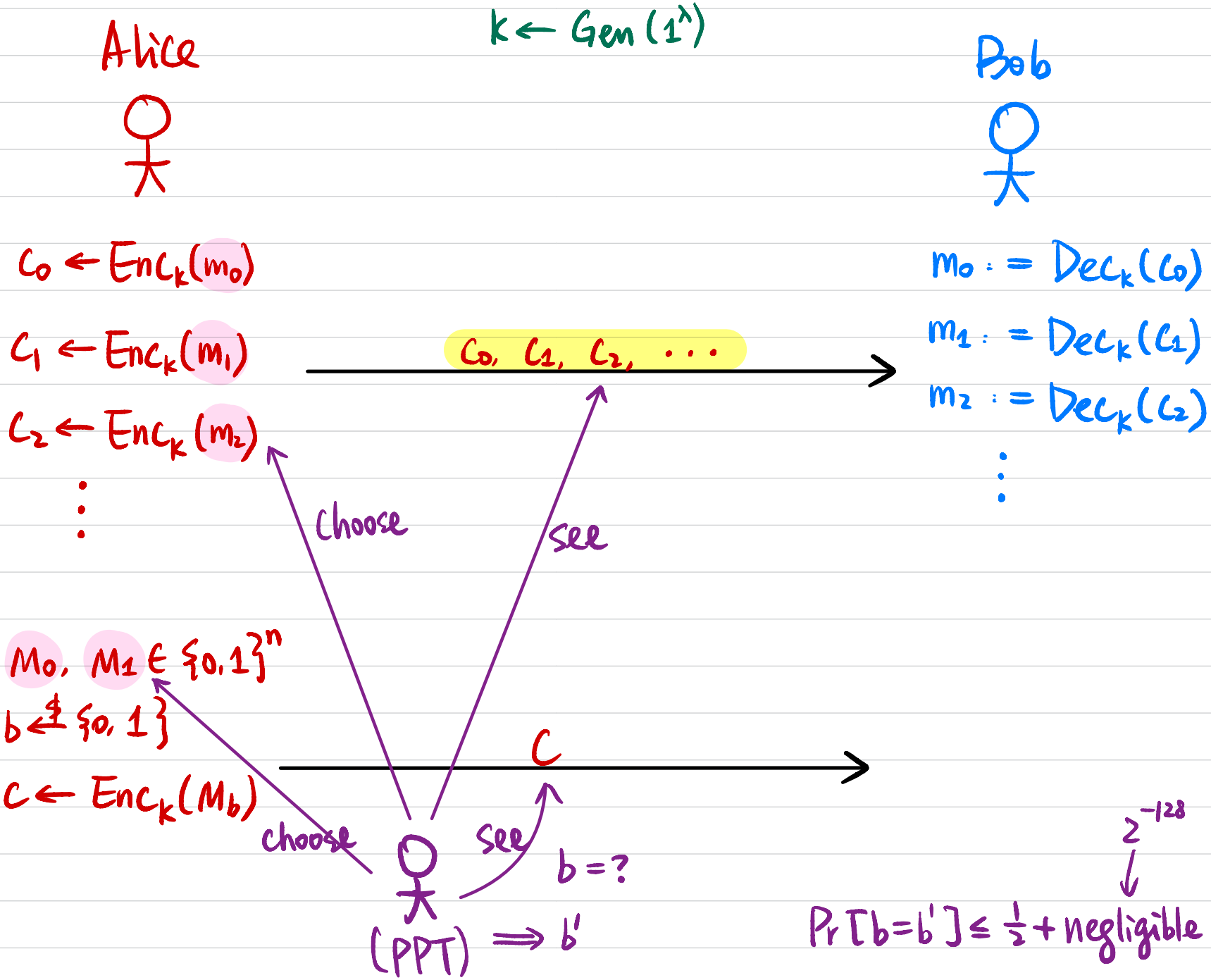
"Computationally indistinguishable"

Computational Security



Computational Security

Chosen-Plaintext Attack (CPA) Security



Security Parameter

$k \leftarrow \text{Gen}(1^\lambda)$ $\quad \underbrace{11\cdots1}_{\lambda}$

λ: security parameter

① adversary runs in time $\text{poly}(\lambda)$

② distinguishing advantage $\text{negligible}(\lambda)$ $\leftarrow 2^{-\lambda}$

$\uparrow$

$\text{negligible}(\lambda) \ll \frac{1}{\lambda^c}$ $\forall$ constant c

Set parameters in practice:

Computational security parameter $\lambda = 128$

Best algorithm to break the scheme (e.g. find secret key) takes time $\sim 2^\lambda$.

How long does 2^{128} CPU cycles take? $\sim 10^{22}$ years

Apple M4 chip: ~ 4.5 GHz

$(4.5 \times 10^9$ CPU cycles/s$)$

Age of universe: $\sim 10^{10}$ years

Construction for SKE

From pseudorandom function / permutation (PRF/PRP)

Practical construction for PRF/PRP: block cipher

Standardized implementation: AES

Computational Assumption: "The construction is secure" (heuristics)

Best attack is brute-force search (classical / quantum)

Constructions for PKE

RSA Encryption: Factoring / RSA Assumption

El Gamal Encryption: Discrete Logarithm / Diffie-Hellman Assumption

Lattice-Based Encryption Schemes (Post-Quantum Security)

Thm (Informal): It's impossible to construct PKE from SKE in a black-box way.

Basic Number Theory

- $a \mid b$: a divides b ($b = a \cdot c$)
- Primes: an integer $p > 1$ that only has 2 divisors: 1 & p.
- Modular Arithmetic:

  $a \bmod N$: remainder of a when divided by N

  $a \cdot b \bmod N = (a \bmod N) \cdot (b \bmod N) \bmod N$.

  $a \equiv b \pmod N$: a and b are congruent modulo N

How to compute $a^b \bmod N$ for a, b, N of n bits? Time complexity? O(n)

Ex: $5^{10} \bmod 7$ $\qquad$ $219426^{736459} \bmod 392643$

$5 \bmod 7 = 5$

$5^2 \bmod 7 = 4$

$5^4 \bmod 7 = (5^2 \bmod 7) \cdot (5^2 \bmod 7) \bmod 7 = 2$

$5^8 \bmod 7 = (5^4 \bmod 7) \cdot (5^4 \bmod 7) \bmod 7 = 4$

$5^8 \cdot 5^2 \bmod 7 = 2$

Basic Number Theory

- gcd(a,b): greastest common divisor

How to compute gcd(a,b)? Time complexity?

a, b both of n bits

Euclidean Algorithm

gcd(17, 12) = 1
17 mod 12 = 5
12 mod 5 = 2
5 mod 2 = 1
2 mod 1 = 0

gcd(18, 12) = 6
18 mod 12 = 6
12 mod 6 = 0

- gcd(a, N) = 1 : a & N are coprime

⇒ ∃b s.t. $a \cdot b \equiv 1 \pmod N$: a is invertible modulo N,
b is its inverse, denoted as a^{-1}.

How to compute b?

Extended Euclidean Alg.

a N
gcd(17, 12) = 1
17 mod 12 = 5
12 mod 5 = 2
5 mod 2 = 1
2 mod 1 = 0

5 = 17 − 12 × 1
2 = 12 − 5 × 2
1 = 5 − 2 × 2

gcd(a, N) = 1
⇓
$1 = a \cdot x + N \cdot y$
⇓ mod N
$1 \equiv a \cdot x$

Basic Number Theory

$\mathbb{Z}_N^* := \{a \mid a \in [1, N-1],\ \gcd(a, N) = 1\}$

Euler's phi (totient) function $\phi(N) := |\mathbb{Z}_N^*|$

Ex: N is prime. $\phi(N) = N-1$.

$N = p \cdot q$ (p, q are primes) $\phi(N) = (p-1) \cdot (q-1)$.

Euler's Theorem $\forall a, N$ where $\gcd(a, N) = 1$, $a^{\phi(N)} \equiv 1 \pmod N$.

Corollary If $d \equiv e^{-1} \bmod \phi(N)$, then $(a^d)^e \equiv a \bmod N$.

$\Downarrow$

$d \cdot e \equiv 1 \bmod \phi(N)$

$\Downarrow$

$d \cdot e = \phi(N) \cdot c + 1$

$\Downarrow$

$a^{de} \equiv a^{\phi(N) \cdot c + 1} \bmod N$

$\equiv 1^c \cdot a \bmod N$

$\equiv a \bmod N$

RSA Assumption

- **Factoring Assumption:**

  How? ↙

  Generate two n-bit primes p, q $(p \neq q)$

  Compute $N = p \cdot q$

  Given N, it's computationally hard to find p & q (classically).

- **RSA Assumption:**

  Generate two n-bit primes p, q $(p \neq q)$

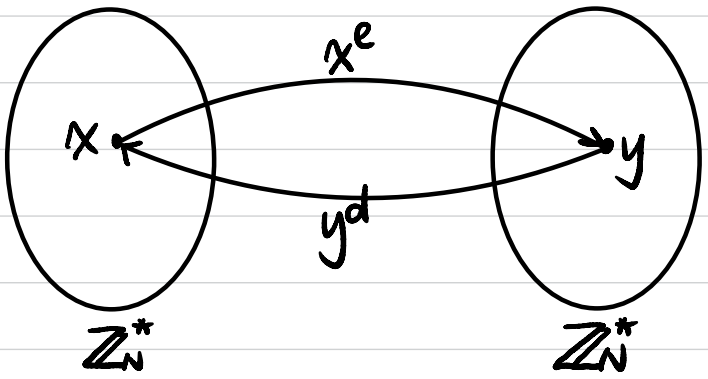
  Compute $N = p \cdot q$, $\phi(N) = (p-1)(q-1)$

  Choose e s.t. $\gcd(e, \phi(N)) = 1$

  Compute $d = e^{-1} \bmod \phi(N)$

  Given N & a random $y \xleftarrow{\$} \mathbb{Z}_N^*$, it's computationally hard to find x s.t.

  $$x^e \equiv y \pmod{N}$$



"Plain" RSA Encryption

$\lambda = 128$

$n = O(\lambda)$ $\quad n = 1024$, key length 2048

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

  Generate two n-bit p, q $(p \neq q)$

  Compute $N = p \cdot q$, $\phi(N) = (p-1)(q-1)$

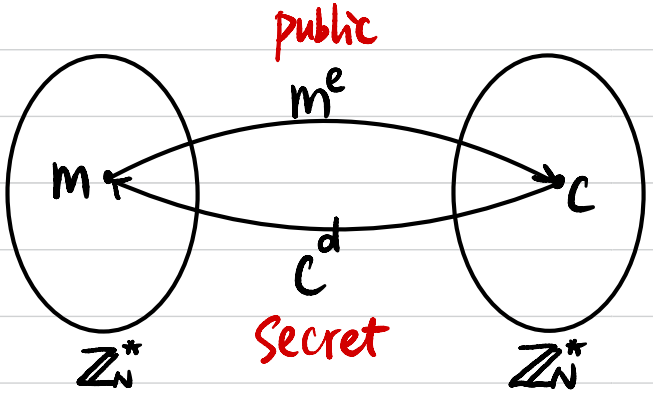
  Choose e s.t. $\gcd(e, \phi(N)) = 1$

  Compute $d = e^{-1} \bmod \phi(N)$

  $pk = (N, e)$ $\quad sk = d$

- $\text{Enc}_{pk}(m)$: $c = m^e \bmod N$

- $\text{Enc}_{sk}(c)$: $m = c^d \bmod N$



Any security issue?