CS 370
Introduction to Security

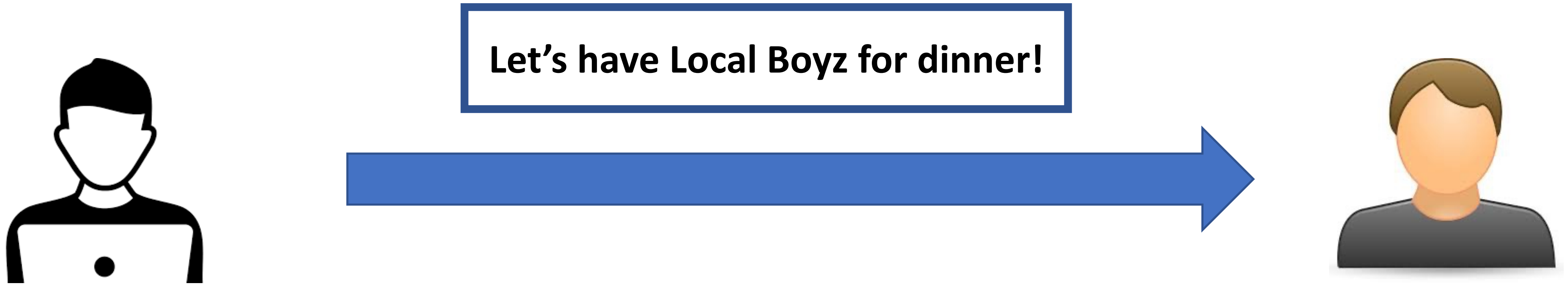
Ancient Cryptography and Cryptography Basics

Yeongjin Jang

Oregon State University

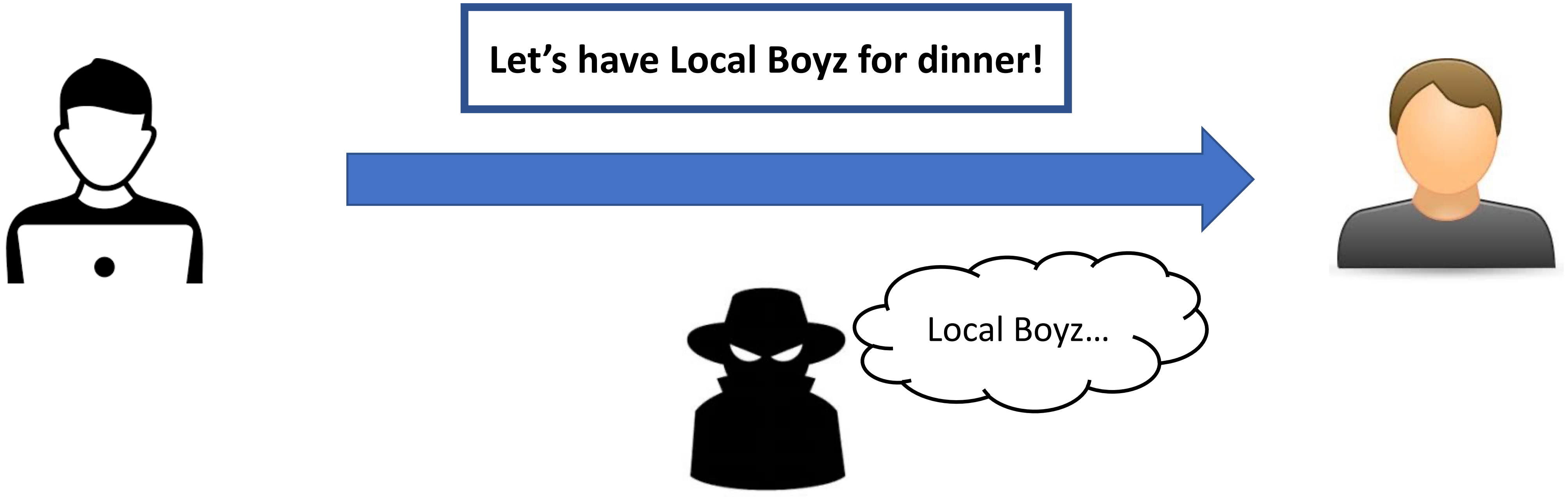
Cryptography

- Suppose you need to communicate with others securely/privately



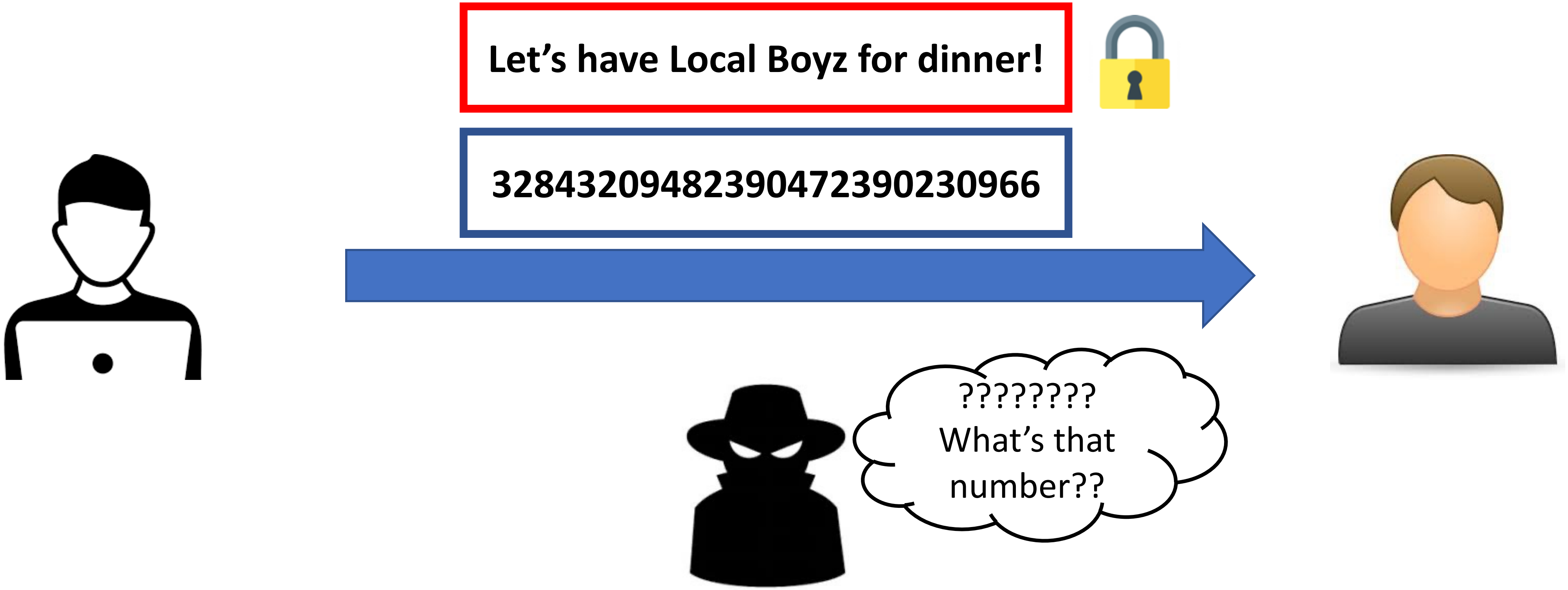
Cryptography

- Others can see the message if you send it over w/o any protection



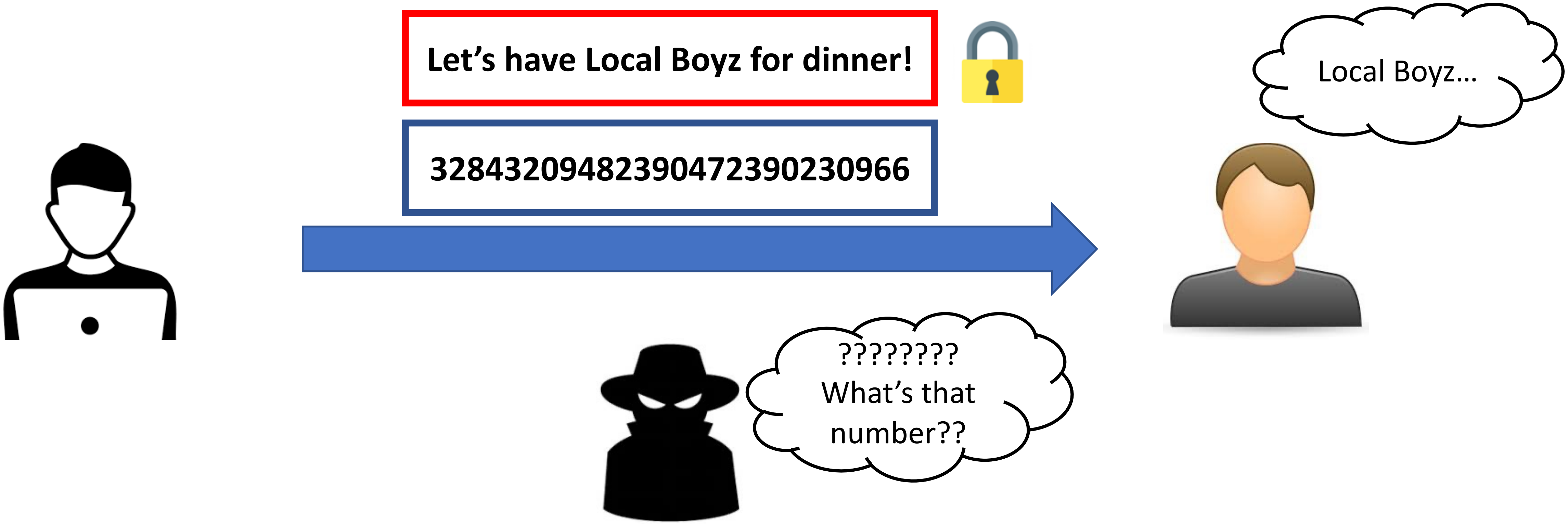
Cryptography

- Cryptography can make our communication secure



Cryptography

- But the other end must know what the plaintext message is



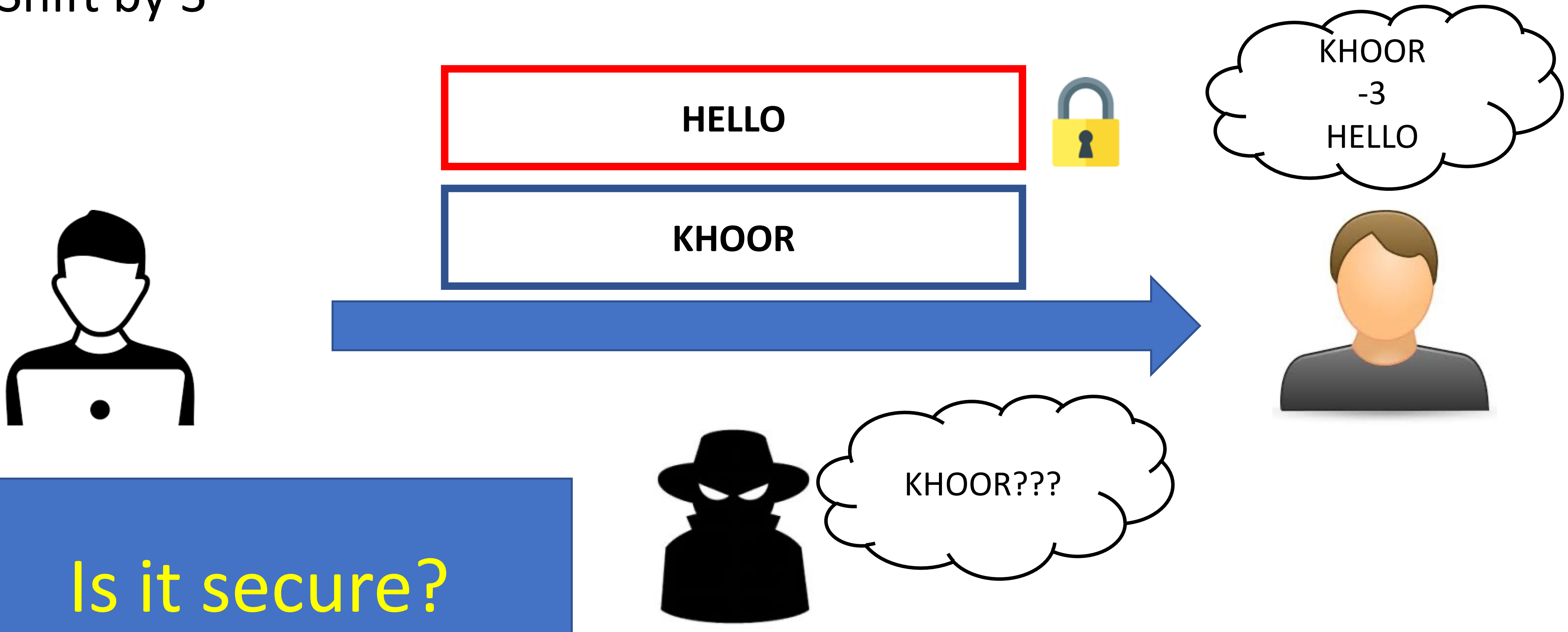
Cryptography in Roman Empire

- CAESAR CIPHER
- Algorithm
    - Shift characters by 3

- ABCDEFGHIJKLMNOPQRSTUVWXYZ
- DEFGHIJKLMNOPQRSTUVWXYZABC
- HELLO
- KHOOR

CAESAR Cipher

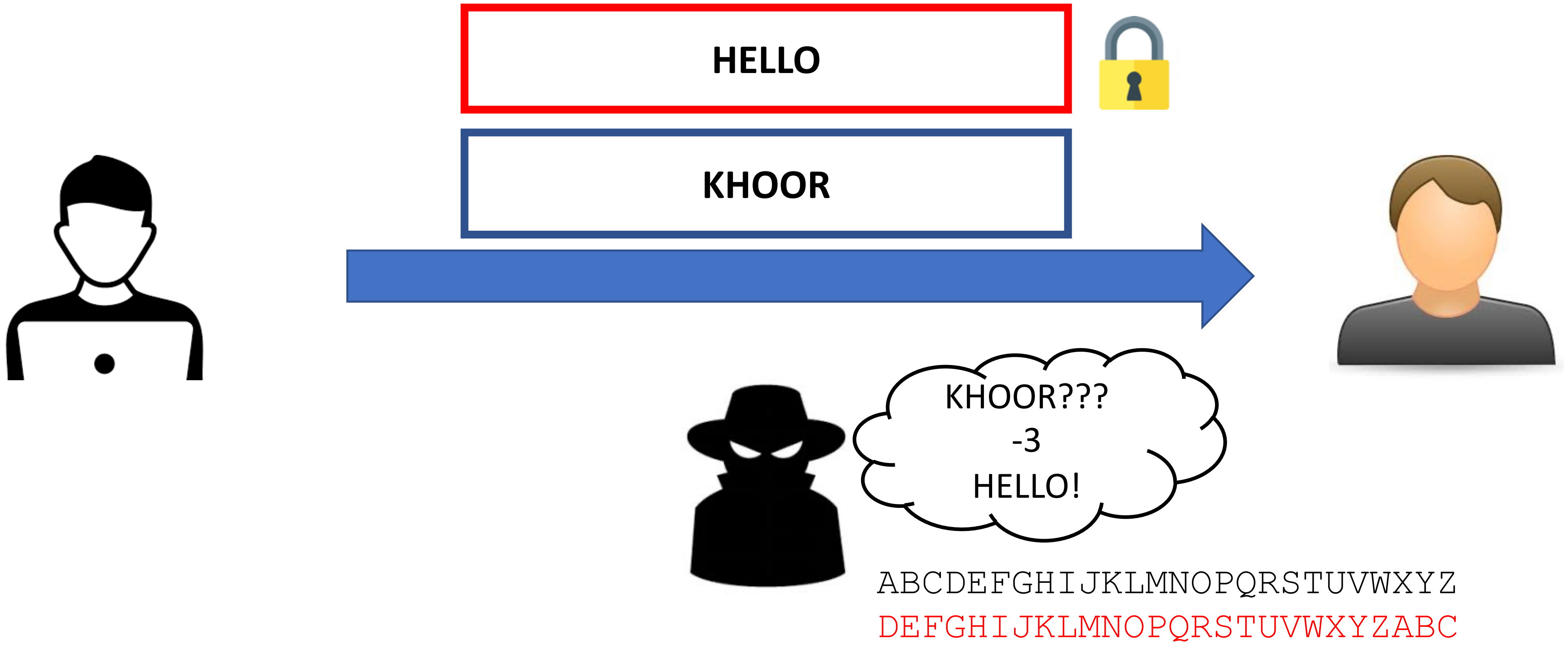
- Shift by 3



Is it secure?

Anyone Who Knows the Offset Can Decrypt the Ciphertext

- Shift by 3



CAESAR Cipher is too easy.. Let’s Make it More Complex

- Rotation-based Substitution Cipher
  - Let us choose the offset, not only the 3

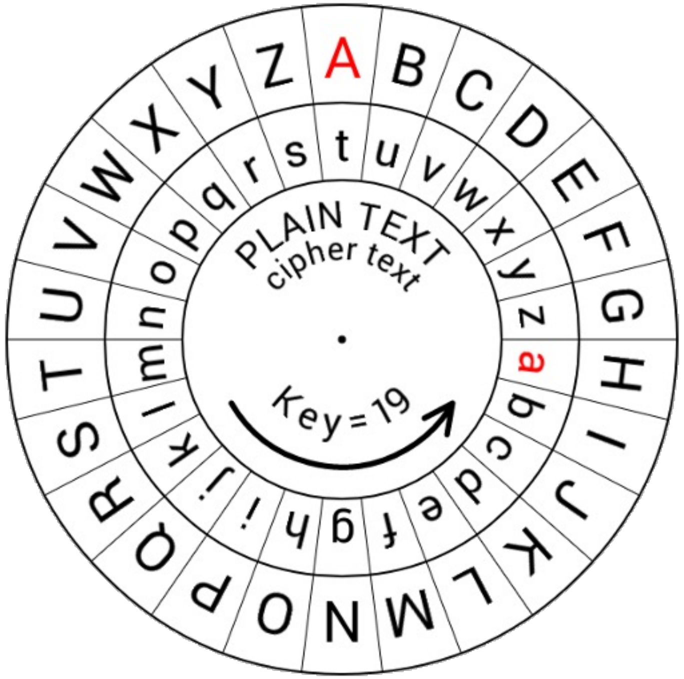


Image from: https://www.amazon.com/Nicolas-Berne-Caesar-Cipher-Wheel/dp/B013KR53SS
https://www.amazon.com/Retroworks-Classic-Caesar-Medallion-Decoder/dp/B004D1L0B0

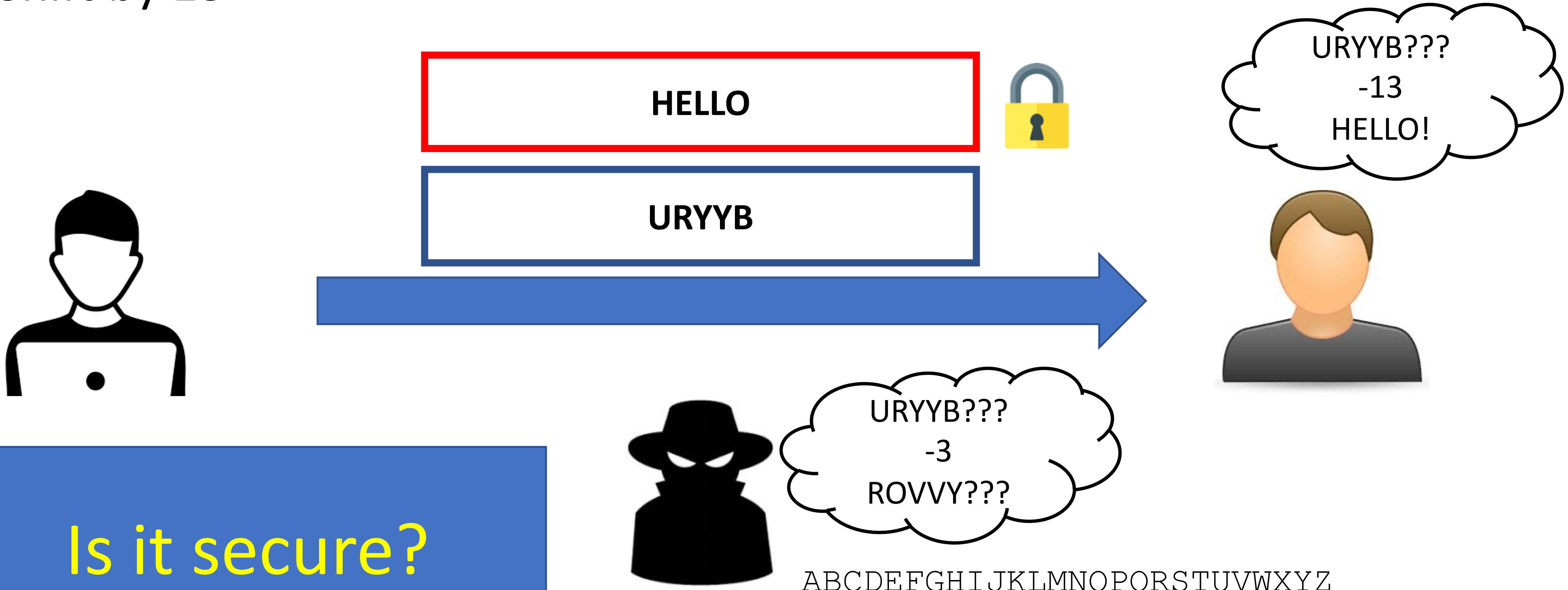
ROT-N Cipher

- We can set whatever N (between 0 and 25) in ROT-N cipher
- If you don't know N, you cannot decrypt it!

Anyone Who Knows the Offset Can Decrypt the Ciphertext

ABCDEFGHIJKLMNOPQRSTUVWXYZ
NOPQRSTUVWXYZABCDEFGHIJKLM

- Shift by 13



ABCDEFGHIJKLMNOPQRSTUVWXYZ
DEFGHIJKLMNOPQRSTUVWXYZABC

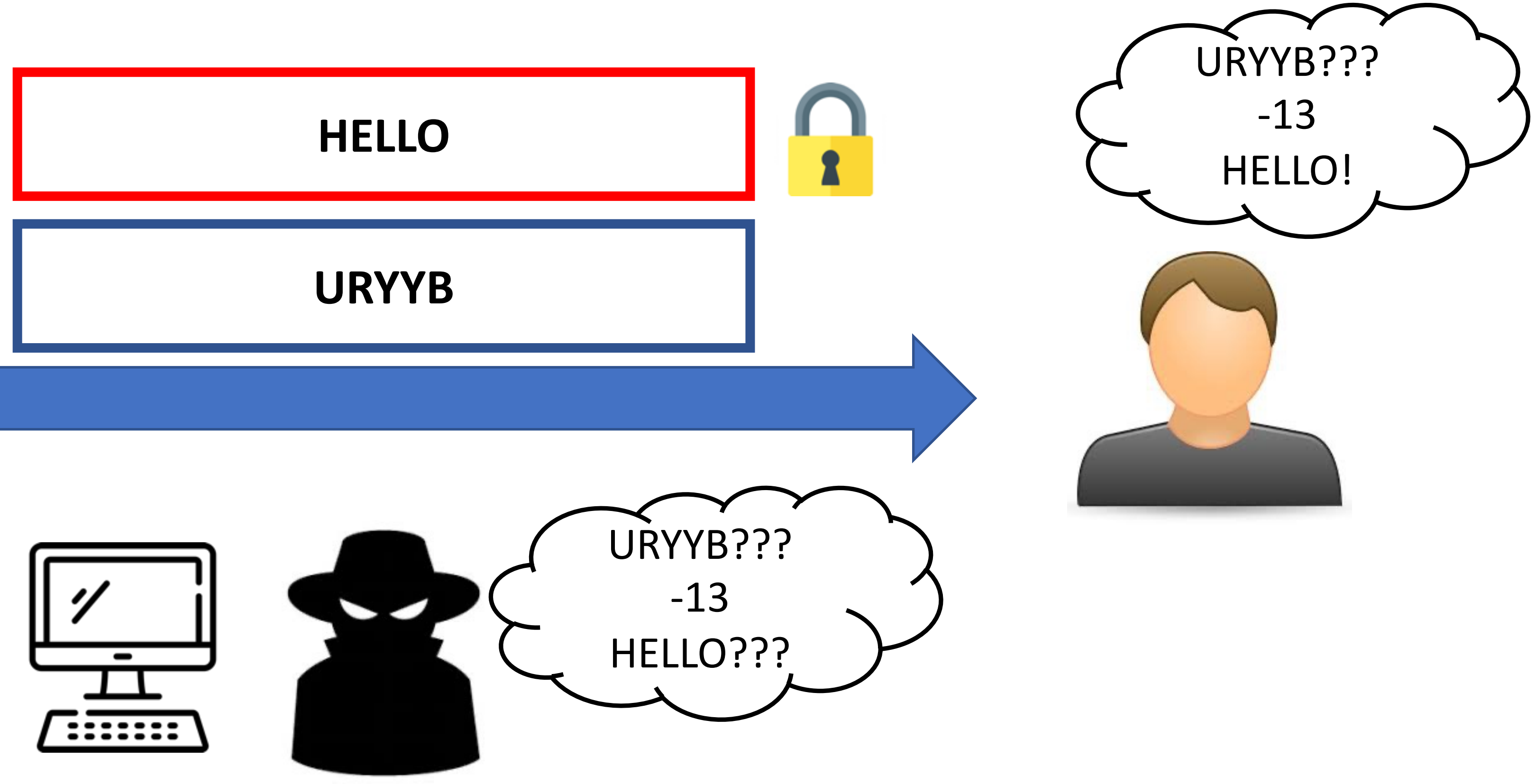
Is it secure?

Anyone Who Knows the Offset Can Decrypt the Ciphertext

ABCDEFGHIJKLMNOPQRSTUVWXYZ
NOPQRSTUVWXYZABCDEFGHIJKLM

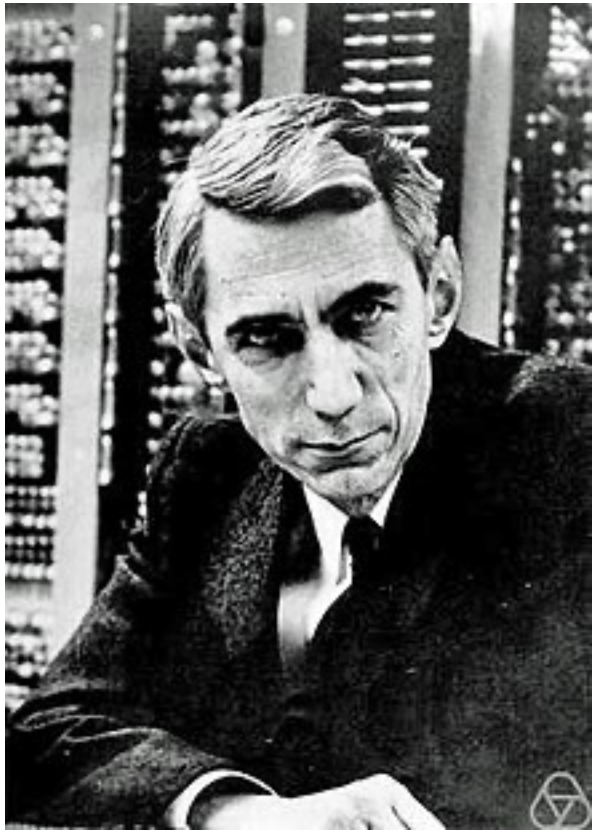
- Shift by 13

```
URYYB
00: URYYB
01: TQXXA
02: SPWWZ
03: ROVVY
04: QNUUX
05: PMTTW
06: OLSSV
07: NKRRU
08: MJQQT
09: LIPPS
10: KHOOR
11: JGNNQ
12: IFMMP
13: HELLO
14: GDKKN
15: FCJJM
16: EBIIL
17: DAHHK
18: CZGGJ
19: BYFFI
20: AXEEH
21: ZWDDG
22: YVCCF
23: XUBBE
24: WTAAD
25: VSZZC
```



What is a Secure Cryptography?

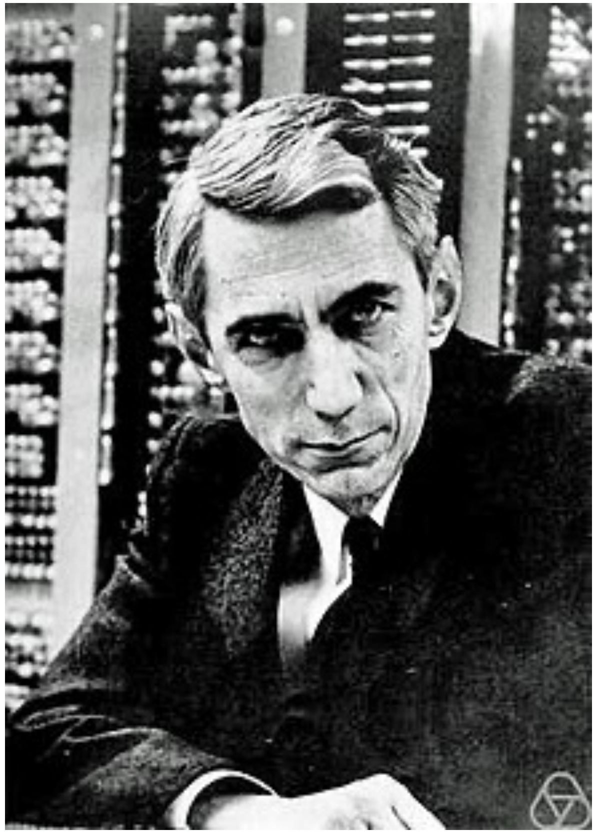
- Shannon's Intuition
  - If attackers cannot distinguish a message M from
  - A random number R
  - Then it is perfectly secure



Claude Shannon (1916 ~ 2001)
A Father of Information Theory and Modern Cryptography

What is a Secure Cryptography?

- More formally
  - A message M has a distribution D
  - D is known to adversary (English, etc..)
  - Adversary observes Ciphertext C which is Enc(M)
  - Knowledge of adversary before observing C
    - Distribution of D
  - Knowledge of adversary after observing C
    - Distribution of D | C
- Shannon Secrecy
  - Distribution of D == Distribution of D|C
  - Then the scheme is perfectly secure
- This intuitively means
  - Observing many Cs does not give any information to adversary



Claude Shannon (1916 ~ 2001)
A Father of Information Theory
and Modern Cryptography

Definition of Perfect Secrecy

- For every pair of m1, m2 in M and for all c,
- Pr [k <- KG : Enc (m1, k) = c] = Pr[k <- KG: Enc(m2, k) = c]

- Implications
  - Probability that the encryption of m1 with k resulting in c
  - is the same as
  - Probability that the encryption of m2 with k resulting in c

  - Adversary cannot distinguish which message the c corresponds to

XOR Cipher

- A simple XOR Cipher is with perfect secrecy

- Scheme
  - For a message M with length L
  - Get a random key K with length L
  - Compute ciphertext C
    - $C = M \oplus K$

XOR Cipher

- A simple XOR Cipher is with perfect secrecy

- Scheme
  - $C = M \oplus K$

- Example

Message	H (0x48)	E (0x45)	L (0x4c)	L (0x4c)	O (0x4f)
Key	A (0x41)	B (0x42)	C (0x43)	D (0x44)	E (0x45)
Ciphertext	0x9	0x7	0xf	0x8	0xa

XOR Cipher

- Decryption

Message	H (0x48)	E (0x45)	L (0x4c)	L (0x4c)	O (0x4f)
Key	A (0x41)	B (0x42)	C (0x43)	D (0x44)	E (0x45)
Ciphertext	0x9	0x7	0xf	0x8	0xa
Decrypt	H	E	L	L	O

```
>>> chr(0x41^0x9)
'H'
>>> chr(0x42^0x7)
'E'
>>> chr(0x43^0xf)
'L'
>>> chr(0x44^0x8)
'L'
>>> chr(0x45^0xa)
'O'
```

Example in Bits

For example, the string "Wiki" (`01010111 01101001 01101011 01101001` in 8-bit ASCII) can be encrypted with the repeating key `11110011` as follows:

```
  01010111 01101001 01101011 01101001
⊕ 11110011 11110011 11110011 11110011
  -----------------------------------
= 10100100 10011010 10011000 10011010
```

And conversely, for decryption:

```
  10100100 10011010 10011000 10011010
⊕ 11110011 11110011 11110011 11110011
  -----------------------------------
= 01010111 01101001 01101011 01101001
```

Image from: https://en.wikipedia.org/wiki/XOR_cipher

How is It Perfectly Secure?

- Key must be selected randomly
    - The distribution of K is random

- Ciphertext distribution is independent to the message distribution
    - $C = M \oplus K$
    - No matter how you choose M, if you choose K randomly, then it's good

CAVEAT

- Re-using the key make the scheme weak

- Suppose the attacker knows
  - HELLO -> 0x9, 0x7, 0xf, 0x8, 0xa

- They can calculate the key by

```
>>> chr(ord('H')^0x9)
'A'
>>> chr(ord('E')^0x7)
'B'
>>> chr(ord('L')^0xf)
'C'
>>> chr(ord('L')^0x8)
'D'
>>> chr(ord('O')^0xa)
'E'
```

Generic Version of XOR Cipher

- One-time Pad
  - https://en.wikipedia.org/wiki/One-time_pad

The resulting ciphertext will be impossible to decrypt or break if the following four conditions are met:[1][2]

1. The key must be at least as long as the plaintext.
2. The key must be random (uniformly distributed in the set of all possible keys and independent of the plaintext), entirely sampled from a non-algorithmic, chaotic source such as a hardware random number generator. It is not sufficient for OTP keys to pass statistical randomness tests as such tests cannot measure entropy, and the number of bits of entropy must be at least equal to the number of bits in the plaintext. For example, using cryptographic hashes or mathematical functions (such as logarithm or square root) to generate keys from fewer bits of entropy would break the uniform distribution requirement, and therefore would not provide perfect secrecy.
3. The key must never be reused in whole or in part.
4. The key must be kept completely secret by the communicating parties.

Tutorials for the Challenges!