CSE467: Computer Security

3. Classical Cryptography

Seongil Wi

Department of Computer Science and Engineering

Notice: Activities

- Period
  - **Activity #1 SaveUNIST**: TBA (after midterm)
  - **Activity #2 HackGPT**: 9/5~12/07 (11:59PM)

Not mandatory, not homework, but participation is highly recommended to upgrade your score!

Activity #2: HackGPT

- Does ChatGPT become a friend or villain?
- **Report NEW <u>security threats</u> that can be caused by ChatGPT!**
  - Read "Large Language Models like ChatGPT say The Darnedest Things", ***CACM'23***
  - DO NOT report known issues (e.g., generating buggy code): a lot of studies already done
  - <u>Describe a concrete and detailed scenario</u>

- Each student can submit **3 scenarios** until **Dec 7**

Image from https://www.the420.in/hackers-abusing-openais-chatgpt-to-build-nefarious-online-tools-israeli-report-claims-amid-microsofts-10-billion-investment-talks/

Activity #2: HackGPT

- Come up with a new malicious use of AI
- Submit your scenario via the email

Image from https://www.the420.in/hackers-abusing-openais-chatgpt-to-build-nefarious-online-tools-israeli-report-claims-amid-microsofts-10-billion-investment-talks/

Submission Format – Submission

- TO: seongil.wi@units.ac.kr
- CC: dy3199@unist.ac.kr
- Title: [HackGPT,ID,Name] Title of the vulnerability
- Content:
  - Input
  - Output
  - Provide a screenshot of the dialog
  - Describe a concrete and detailed scenario (up to 10 sentences)

Image from https://www.the420.in/hackers-abusing-openais-chatgpt-to-build-nefarious-online-tools-israeli-report-claims-amid-microsofts-10-billion-investment-talks/

Activity (2): HackGPT – Evaluation

- Evaluation will be mainly done by TA and professor
- The evaluation criteria are as follows:
  - Severity
  - Creativity
  - Relevance (to this course)

Please refer to the detailed instructions on our course homepage

Image from https://www.the420.in/hackers-abusing-openais-chatgpt-to-build-nefarious-online-tools-israeli-report-claims-amid-microsofts-10-billion-investment-talks/

Recap: CIA Properties

- **C**onfidentiality: information is not made available to unauthorized parties
- **I**ntegrity: information is not modified in an unauthorized manner
- **A**vailability: information is readily available when it is needed

+ Authentication, Non-repudiation

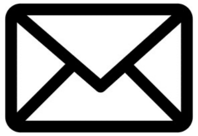
Recap: CIA Properties

- **C**onfidentiality: information is not made available to unauthorized parties
- **I**ntegrity: information is not modified in an unauthorized manner
- **A**vailability: information is readily available when it is needed

+ Authentication, Non-repudiation

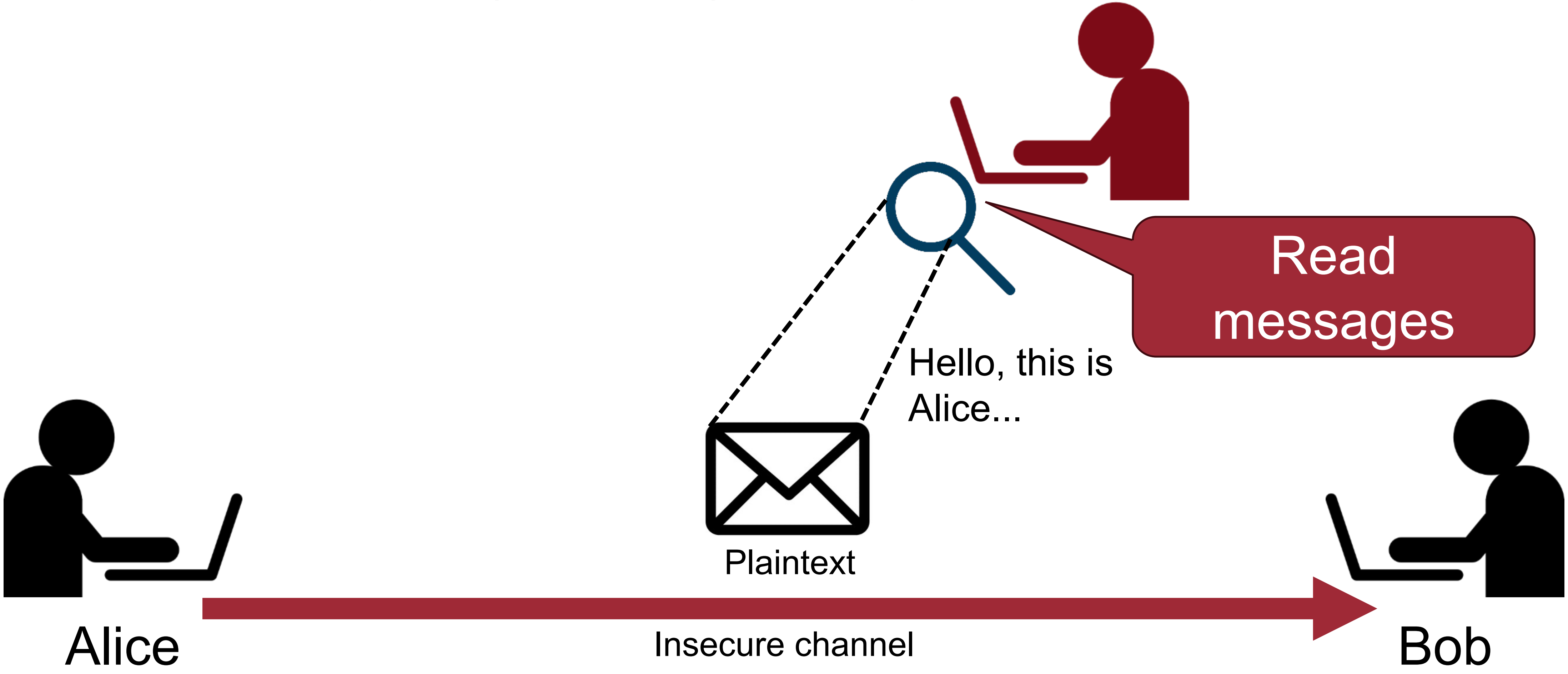
Cryptography!

Basic Terminology

- **Plaintext**: original message 
- **Ciphertext**: coded message 
- **Key**: info used in cipher known only to sender/receiver 
- **Cipher**: algorithm for transforming some texts
  - **Encipher (encrypt):** converting plaintext to ciphertext $\boxed{E}$
  - **Decipher (decrypt):** recovering ciphertext from plaintext $\boxed{D}$

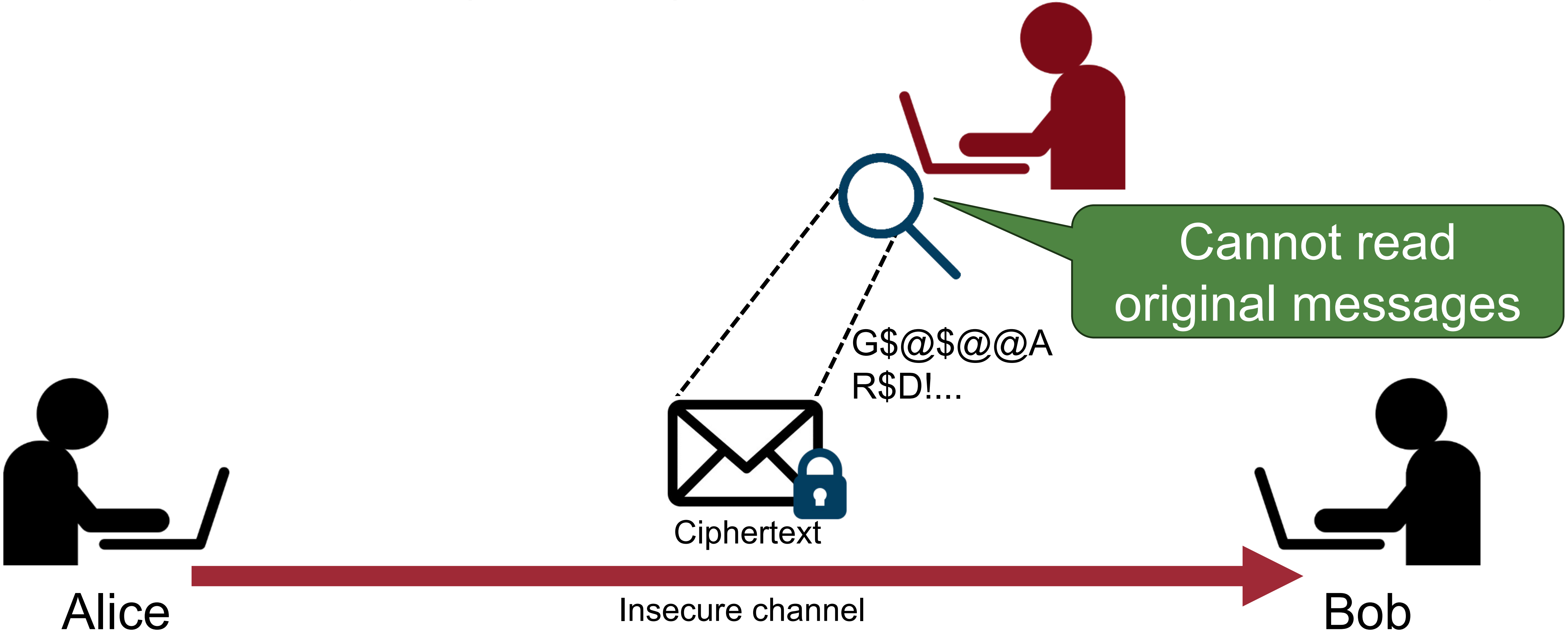
Cryptography

- “Secret writing” in Greek
- Goal: protect your (sensitive) messages/data from eavesdropping



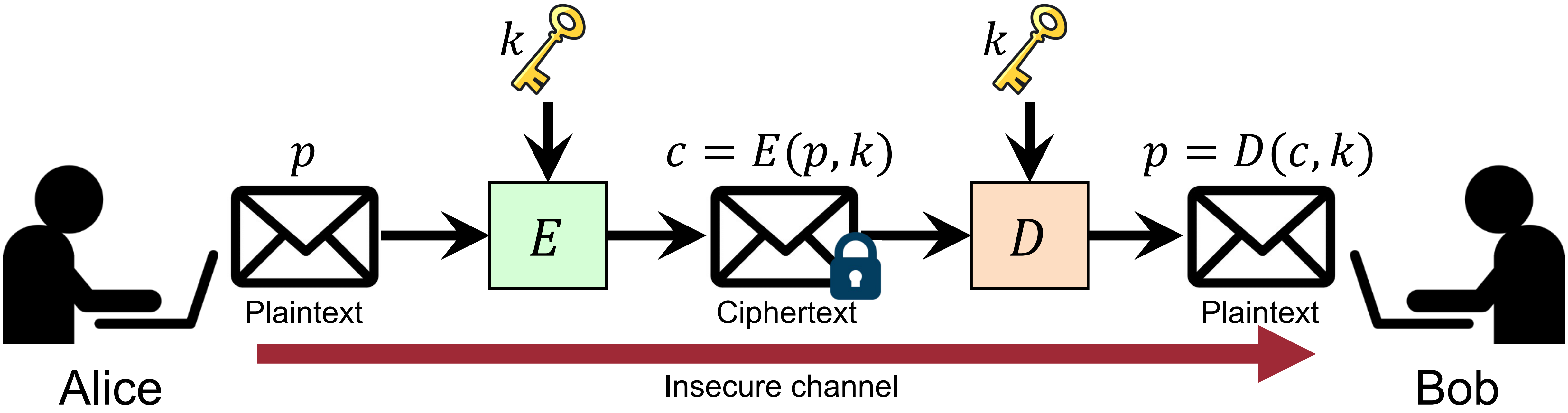
Cryptography

- “Secret writing” in Greek
- Goal: protect your (sensitive) messages/data from eavesdropping

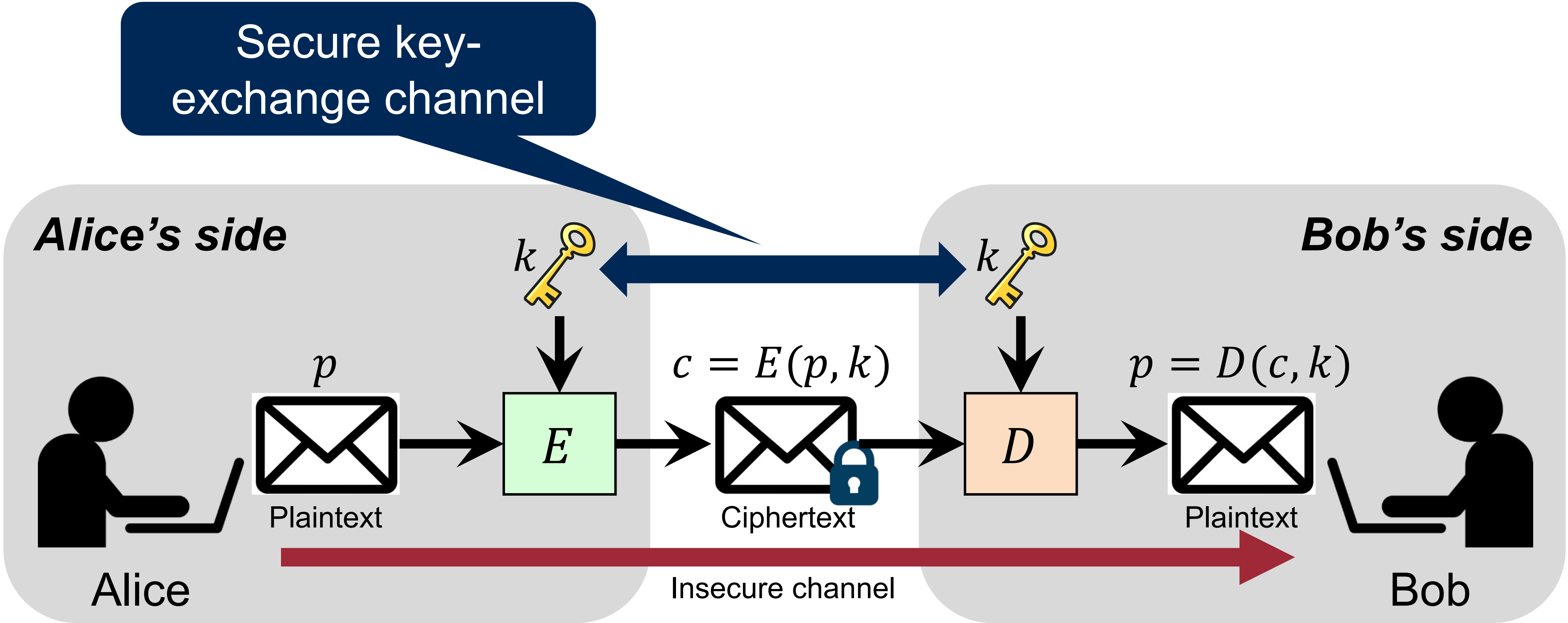


Cryptography

- "Secret writing" in Greek
- Goal: protect your (sensitive) messages/data from eavesdropping
- The most basic building block of computer security
- Two functions: encryption (E) and decryption (D) parameterized by a plaintext (p), ciphertext (c), and cryptographic key (k)



(Symmetric Key) Cryptography



Kerckhoff's Principle

You should always assume that the ***adversary knows the encryption/decryption algorithm***!

\- Auguste Kerckhoffs

The resistance of the cipher must be based only on the ***secrecy of the key***

Requirements in Symmetric Encryption

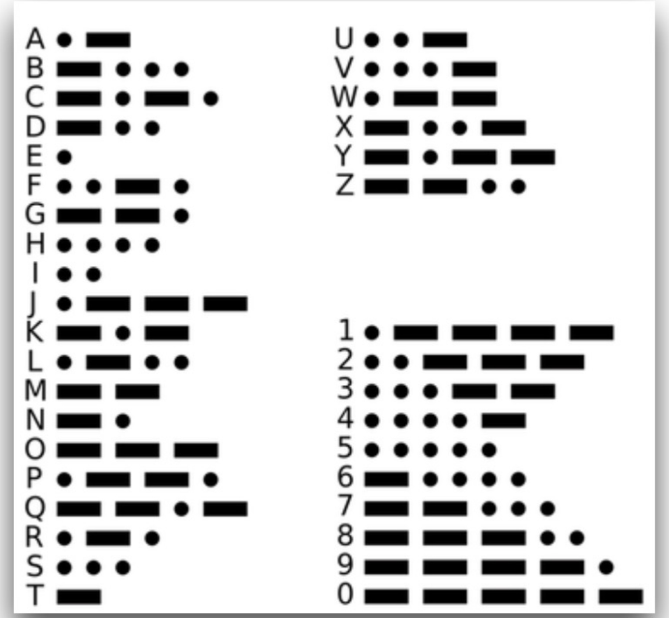
1. A strong encryption algorithm
   - ***Assume encryption algorithm is known***

E D

2. The secrecy of the key
   - A secret key known only to sender / receiver
   - Must be unpredictable

Classical vs. Modern

- Cryptography: “The **art** of writing or solving **codes**”
  (Oxford English Dictionary)
- **Codes**
  - For secret communications: confidentiality
  - *Modern cryptography* includes more: integrity, non-repudiation, secret key exchange, etc.
- **Art**
  - Little theory but ad-hoc designs
  - *Modern cryptography*: science and math



Classical Cryptography

- **CAUTION**: DO NOT use this classical cryptography for any practical uses

- Why do we study classical ones?
  - To highlight the weakness of ad-hoc approaches
  - To demonstrate that simple approaches are unlikely to succeed

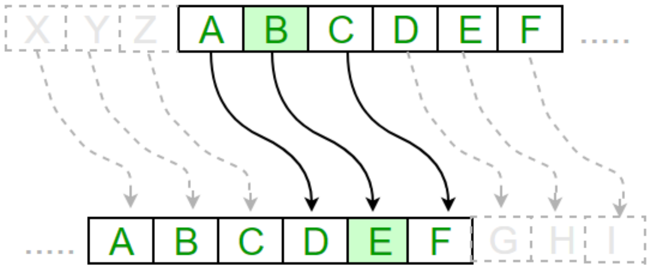
- In this lecture, we will cover
  - Caesar cipher
  - Substitution cipher
  - Vigenere cipher

Classical Cryptography – Caesar Cipher

- Encryption: shift each plaintext character 3 places forward

- Example:
  - Plaintext: `haejunejung`
  - Ciphertext: `kdhmxqhmxqi`

- Q. What is the key?

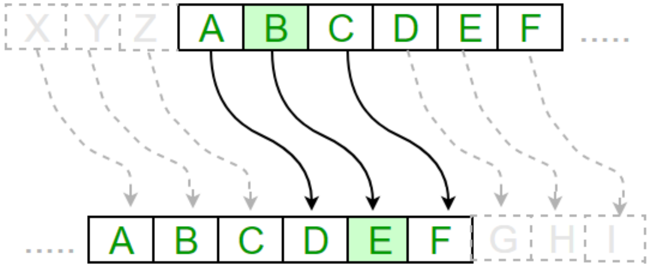


Classical Cryptography – Caesar Cipher

- Encryption: shift each plaintext character k places forward

E $$E(p,k) = (p + k)\ mod\ 26$$

D $$D(c,k) = (c - k)\ mod\ 26$$



- Q. Robust enough?

Problem: Exhaustive Key Search

- Key: a number between 0 and 25
- Given a cipher text: `ovdthufwvzzpislrlfzhylaolyl`
- Can you find the plaintext? How?

Key Value	Possible Plain Text
1	nucsgtevuyyohrkqkeygxkznkxk
2	mtbrfsdutxxngqjpjdxfwjymjwj
3	lsaqerctswwmfpioicwevixlivi
…	…
7	howmanypossiblekeysarethere
…	…

How to make it more robust?

Classical Cryptography – Substitution Cipher

- One-to-one mapping (bijection)
- Example:
  - Plaintext: `seungminlee`
  - Key: Substitution mapping table

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
q	w	e	r	t	y	u	i	o	p	a	s	d	f	g	h	j	k	l	z	x	c	v	b	n	m

  - Ciphertext: `ltxfudofstt`

- Key space?
  - $26! \approx 2^{88}$

- Q. Robust enough?

Classical Cryptography – Vigenere Cipher

- Encryption: poly-alphabetic **shift**

E	$E(p,k) = (p_i + k_i)\ mod\ 26$
D	$D(c,k) = (c_i - k_i)\ mod\ 26$

Invented in 16th century and had been unbreakable for hundreds of years!

- Example
  - Plaintext: `tellhimaboutme`
  - Key (repeated): `cafecafecafeca`
  - Ciphertext: `veqpjiredozxoe`

Problems?

- ***Letters are mapped to different ciphertexts***: smooth out the frequency distribution in ciphertext

Cracking Vigenere Cipher

- When the length t of the key is known:
  - Divide ciphertext into t parts and perform **statistical analysis** for each part

Plaintext:	`tellhimaboutme`
Key (repeated):	`cafecafecafeca`
Ciphertext:	`veqpjiredozxoe`

Each plaintext symbol always maps to the same ciphertext symbol

  - When the length of the key is unknown but the max length T is known:
    - Repeat the above T times
  - What if the length is unknown?

Kasiski's Method

- Goal: extract the length t of the key
- Observations:
  - A repeated substring **may** exist in the ciphertext
  - The distance of the two occurrences **may** be a *multiple of the key length*

- Example:

Plaintext:	`.......THE.....THE.........THE......`
Key (repeated):	`.......ION.....ION.........ION......`
Ciphertext:	`.......BVR.....BVR.........BVR......`
	18 **24**

Friedrich W. Kasiski, ***Die Geheimschriften und die Dechiffrirkunst***, Mittler und Sohn, Berlin, 1863

Properties of Kasiski’s Method

Object	Property
Long ciphertext	
Short plaintext	
Long repeated substrings in a ciphertext	
Short repeated substrings in a ciphertext	

Example


```
LFWKI MJCLP SISWK HJOGL KMVGU RAGKM KMXMA MJCVX
WUYLG GIISW ALXAE YCXMF KMKBQ BDCLA EFLFW KIMJC
GUZUG SKECZ GBWYM OACFV MQKYF WXTWM LAIDO YQBWF
GKSDI ULQGV SYHJA VEFWB LAEFL FWKIM JCFHS NNGGN
WPWDA VMQFA AXWFZ CXBVE LKWML AVGKY EDEMJ XHUXD
AVYXL
```

Example

```
LFWKI MJCLP SISWK HJOGL KMVGU RAGKM KMXMA MJCVX
WUYLG GIISW ALXAE YCXMF KMKBQ BDCLA EFLFW KIMJC
GUZUG SKECZ GBWYM OACFV MQKYF WXTWM LAIDO YQBWF
GKSDI ULQGV SYHJA VEFWB LAEFL FWKIM JCFHS NNGGN
WPWDA VMQFA AXWFZ CXBVE LKWML AVGKY EDEMJ XHUXD
AVYXL
```

- Substring: **LFWKIMJC**
- Position: Idx 0, 72, 144
- Distance: 72

Example

```
LFWKI MJCLP SISWK HJOGL KMVGU RAGKM KMXMA MJCVX
WUYLG GIISW ALXAE YCXMF KMKBQ BDCLA EFLFW KIMJC
GUZUG SKECZ GBWYM OACFV MQKYF WXTWM LAIDO YQBWF
GKSDI ULQGV SYHJA VEFWB LAEFL FWKIM JCFHS NNGGN
WPWDA VMQFA AXWFZ CXBVE LKWML AVGKY EDEMJ XHUXD
AVYXL
```

- Substring: `WMLA`
- Position: Idx 108, 182
- Distance: 74

Example

LFWKI MJCLP SISWK HJOGL KMVGU RAGKM KMXMA MJCVX
WUYLG GIISW ALXAE YCXMF KMKBQ BDCLA EFLFW KIMJC
GUZUG SKECZ GBWYM OACFV MQKYF WXTWM LAIDO YQBWF
GKSDI ULQGV SYHJA VEFWB LAEFL FWKIM JCFHS NNGGN
WPWDA VMQFA AXWFZ CXBVE LKWML AVGKY EDEMJ XHUXD
AVYXL

- Substring: ISW
- Position: Idx 11, 47
- Distance: 36

Analysis

Substring	Distance	Factors (divisors, 약수)
LFWKIMJC	72	2 3 4 6 8 9 12 18 24 36 72
WMLA	74	2 37 74
MJC	66	2 3 6 11 22 33 66
ISW	36	2 3 4 6 9 12 18 36
VMQ	32	2 4 8 16 32
DAV	30	2 3 5 6 10 15

Analysis

	Factors																		
	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
74	O																		
72	O	O	O		O		O	O			O						O		
66	O	O			O					O									
36	O	O	O		O			O			O						O		
32	O		O				O								O				
30	O	O		O	O				O					O					
Total	6	4	3	1	4	0	2	2	1	1	2	0	0	1	1	0	2	0	0

Top candidates

Result

```
LFWKI MJCLP SISWK HJOGL KMVGU RAGKM KMXMA MJCVX
WUYLG GIISW ALXAE YCXMF KMKBQ BDCLA EFLFW KIMJC
GUZUG SKECZ GBWYM OACFV MQKYF WXTWM LAIDO YQBWF
GKSDI ULQGV SYHJA VEFWB LAEFL FWKIM JCFHS NNGGN
WPWDA VMQFA AXWFZ CXBVE LKWML AVGKY EDEMJ XHUXD
AVYXL
```

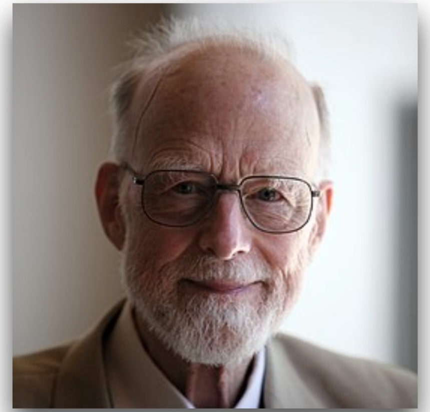
↓ Key: SYSTEM

```
THERE ARETW OWAYS OFCON STRUC TINGA SOFTW AREDE
SIGNO NEWAY ISTOM AKEIT SOSIM PLETH ATTHE REARE
OBVIO USLYN ODEFI CIENC IESAN DTHEO THERW AYIST
OMAKE ITSOC OMPLI CATED THATT HEREA RENOO BVIOU
SDEFI CIENC IESTH EFIRS TMETH ODISF ARMOR EDIFF
ICULT
```

Pop-up Lesson

**“There are two ways of constructing a software design:
One way is to make it so simple that there are obviously
no deficiencies, and the other way is to make it so complicated
that there are no obvious deficiencies.
The first method is far more difficult.”**

- T. Hoare, ACM Turing Award winner (1980)



Principles of Modern Cryptography

- Rigorous approaches to security
- What we need for science?
  - Formal (i.e., rigorous and precise) definitions of security
  - Precise assumptions
  - Proofs of security

Cryptanalysis

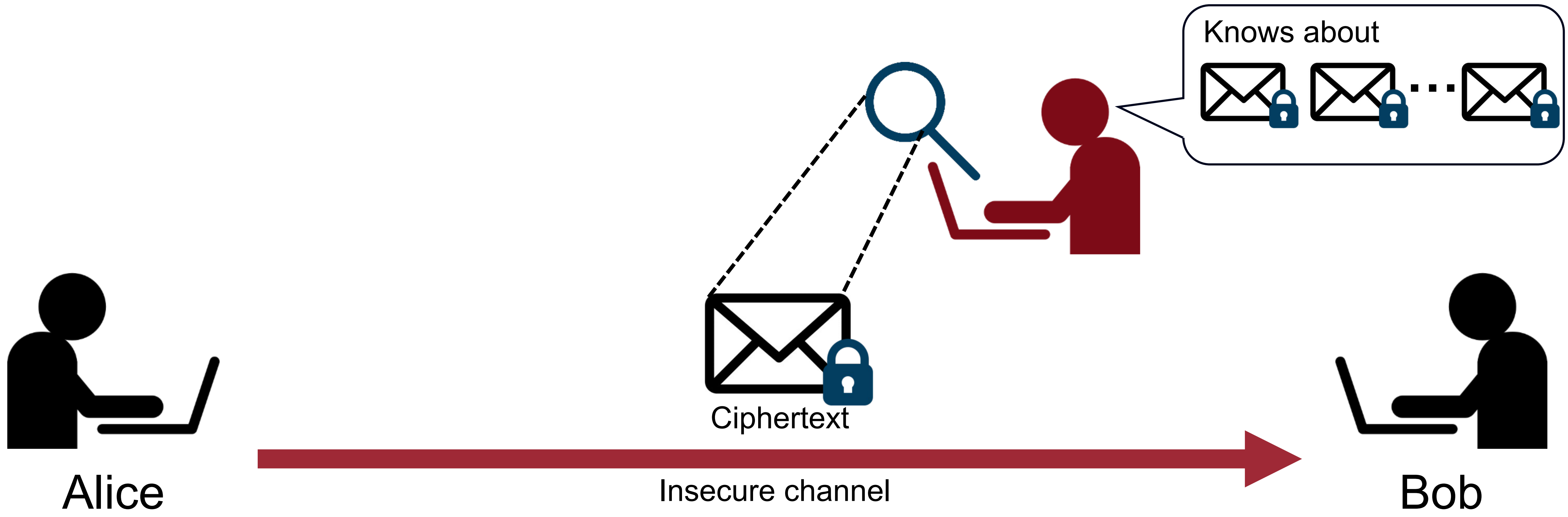
- Study of principles/methods of deciphering ciphertext without using the real key
- Objective: to **recover the key** 🗝 or alternatively to create an algorithm which would allow him to decrypt any ciphertext messages (but without actually knowing the key)
- General approaches
  - Cryptanalytic attack
  - Brute-force attack

Cryptanalytic attack - Adversary Assumptions

- What are the adversary capabilities?
- Attacker capabilities (in order of increasing attack power)
  - **Ciphertext-Only Attack**: most basic attack
  - **Known-Plaintext Attack**: attacker obtains certain plaintext/ciphertext pairs
  - **Chosen-Plaintext Attack**: attacker obtains plaintext/ciphertext pairs for plaintext of its choice
  - **Chosen-Ciphertext Attack**: attacker obtains plaintext/ciphertext pairs for ciphertext of its choice

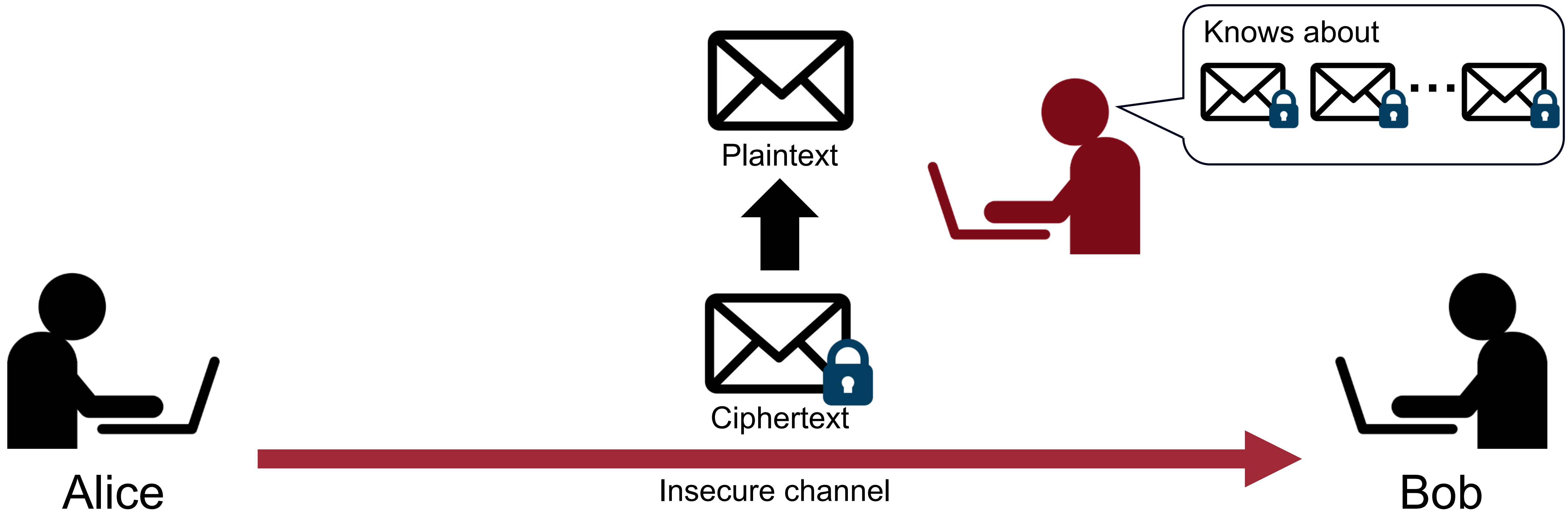
Ciphertext-Only Attack (COA)

- Most basic attack
- The attacker is assumed to have access **only to ciphertexts**



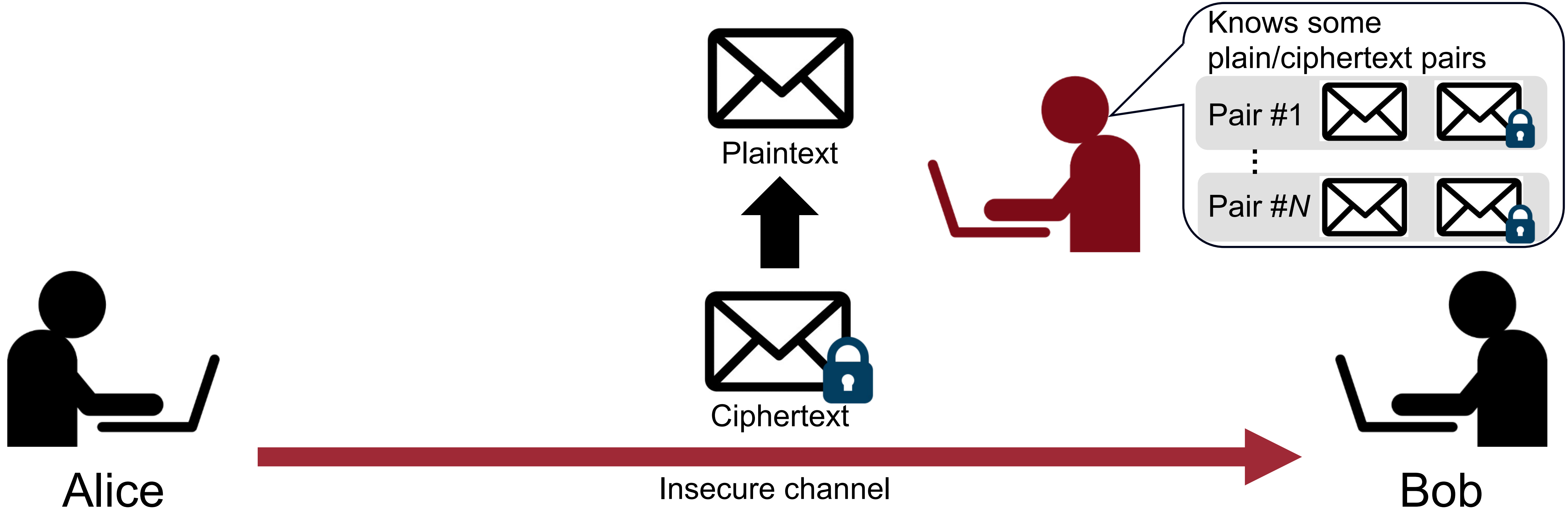
Ciphertext-Only Attack (COA)

- Most basic attack
- The attacker is assumed to have access **only to ciphertexts**
- Can the attacker compute the key from the ciphertext?



Known-Plaintext Attack (KPA)

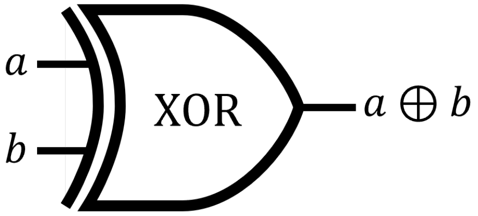
- The attacker is assumed to have access to **multiple plaintexts and their corresponding ciphertexts**
- Can the attacker compute the key from the ciphertext?



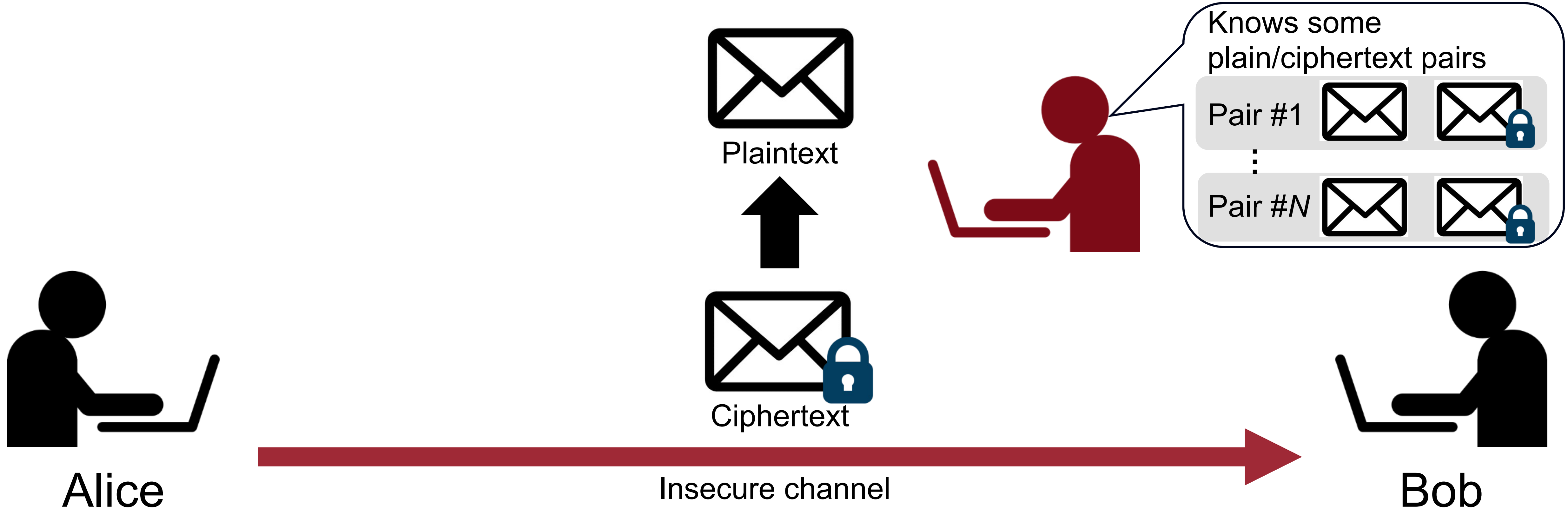
Known-Plaintext Attack (KPA) – Example

$\boxed{E}$

$$E(p,k) = p \oplus k$$

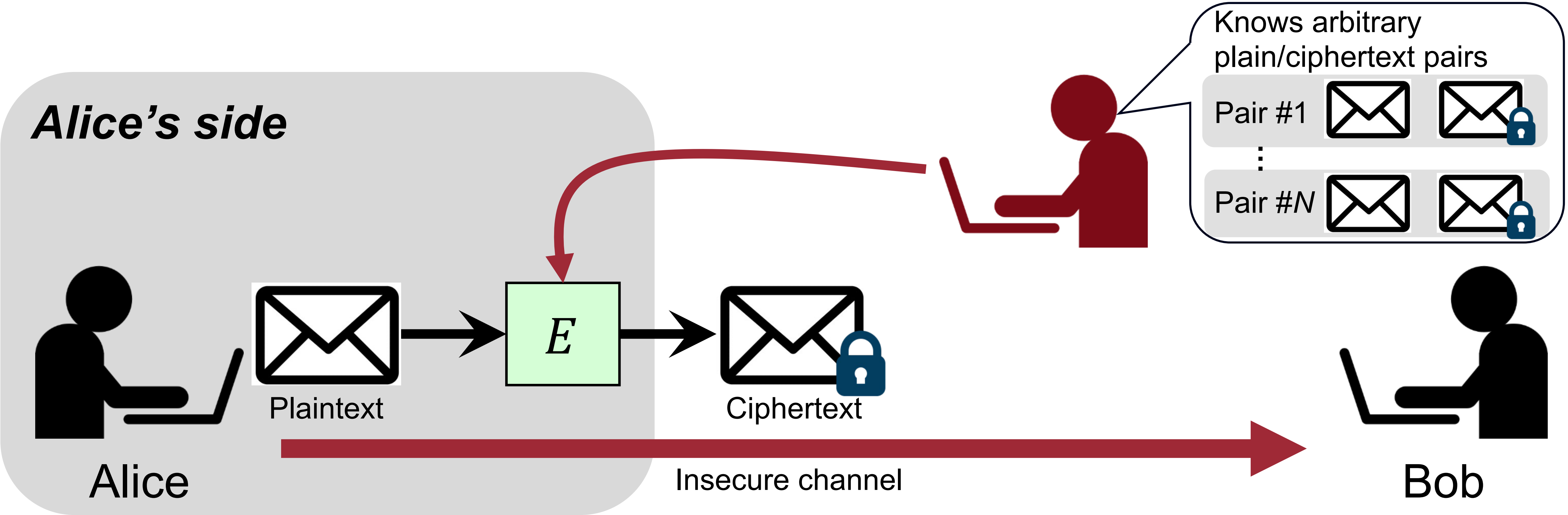


a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0



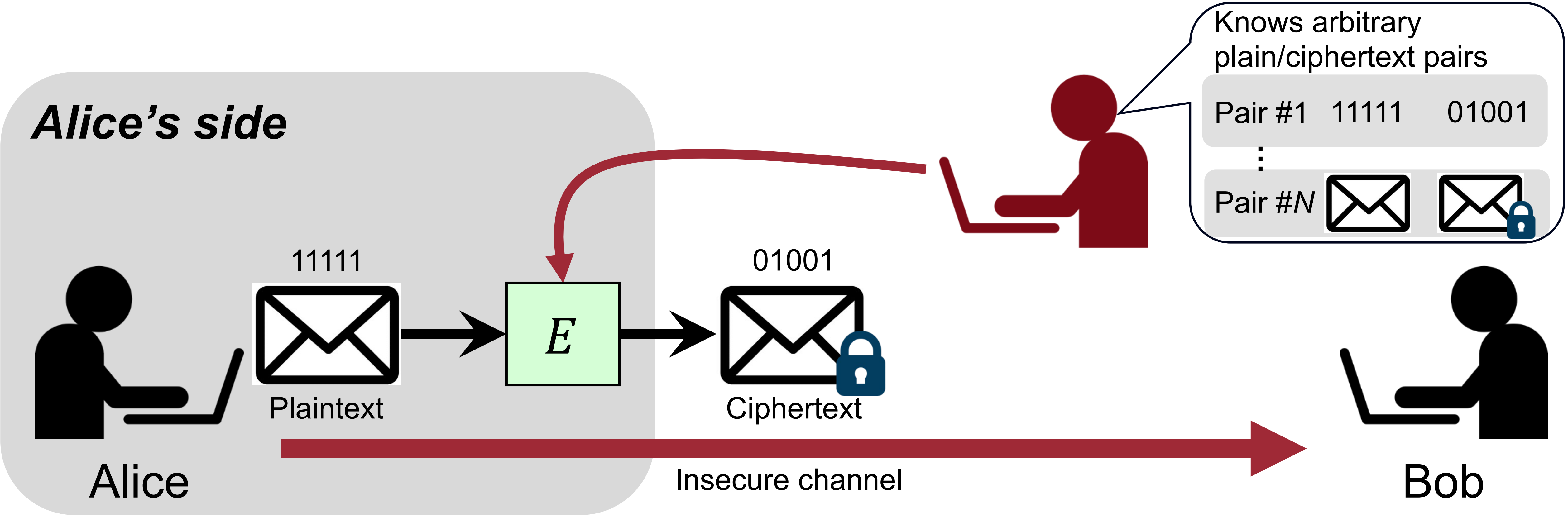
Chosen-Plaintext Attack (CPA)

- The attacker is able to define his own plaintext, feed it into the encryption algorithm, and analyze the resulting ciphertext



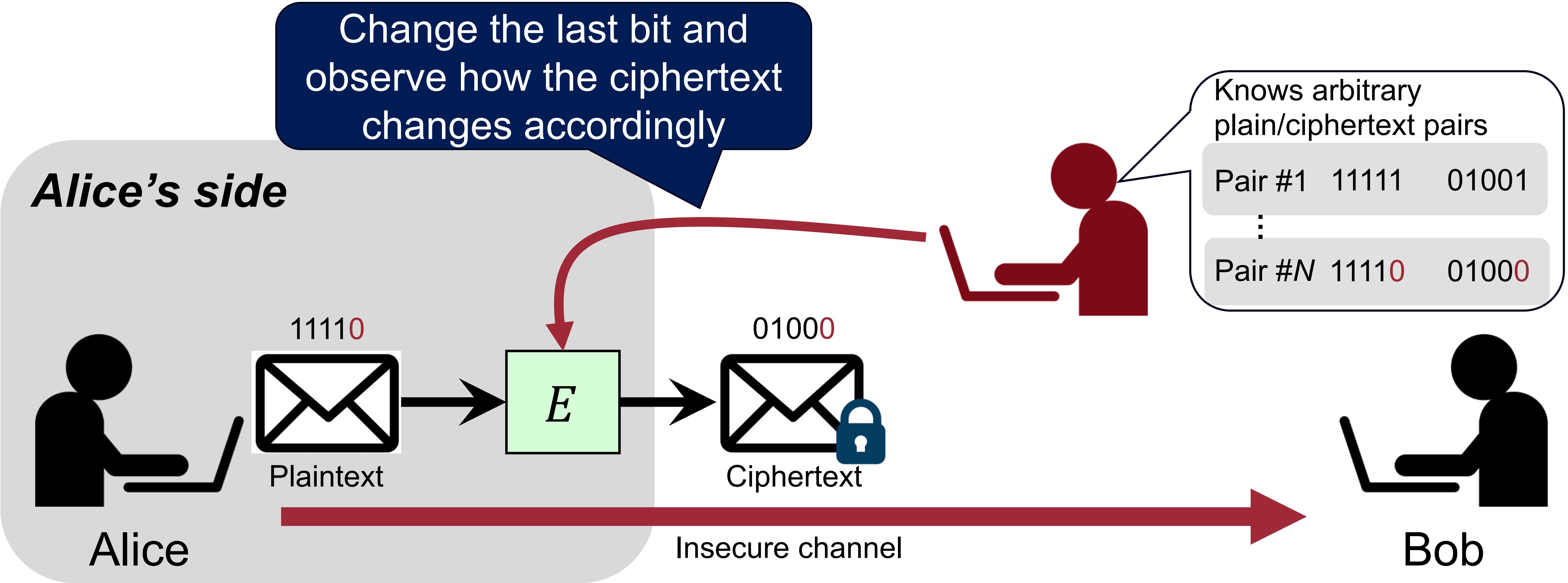
Chosen-Plaintext Attack (CPA) – Example

- The attacker is able to define his own plaintext, feed it into the encryption algorithm, and analyze the resulting ciphertext



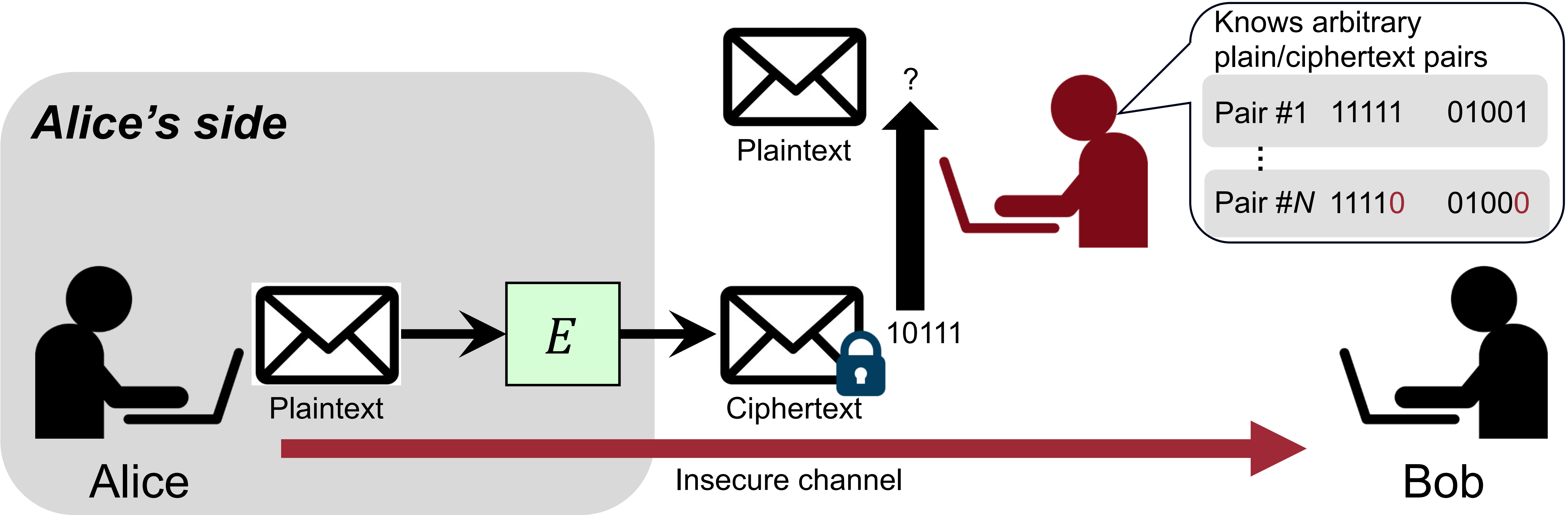
Chosen-Plaintext Attack (CPA) – Example

- The attacker is able to define his own plaintext, feed it into the encryption algorithm, and analyze the resulting ciphertext



Chosen-Plaintext Attack (CPA) – Example

- The attacker is able to define his own plaintext, feed it into the encryption algorithm, and analyze the resulting ciphertext
- Can the attacker compute the key from the ciphertext?



Chosen-Plaintext Attack (CPA) – Example

- ✓ The bit we vary is consistently negated
- ✓ As one bit varies, the remaining ones are left unchanged

	Plaintext	Ciphertext
Try #1	11111	01001
Try #2	11110	01000
Try #3	11101	01011
Try #4	11011	01101
Try #5	10111	00001
Try #6	01111	11001

Chosen-Plaintext Attack (CPA) – Example

- ✓ The bit we vary is consistently negated
- ✓ As one bit varies, the remaining ones are left unchanged

	Plaintext	Ciphertext
Try #1	11111	01001
Try #2	11110	01000
Try #3	11101	01011
Try #4	11011	01101
Try #5	10111	00001
Try #6	01111	11001
		10111

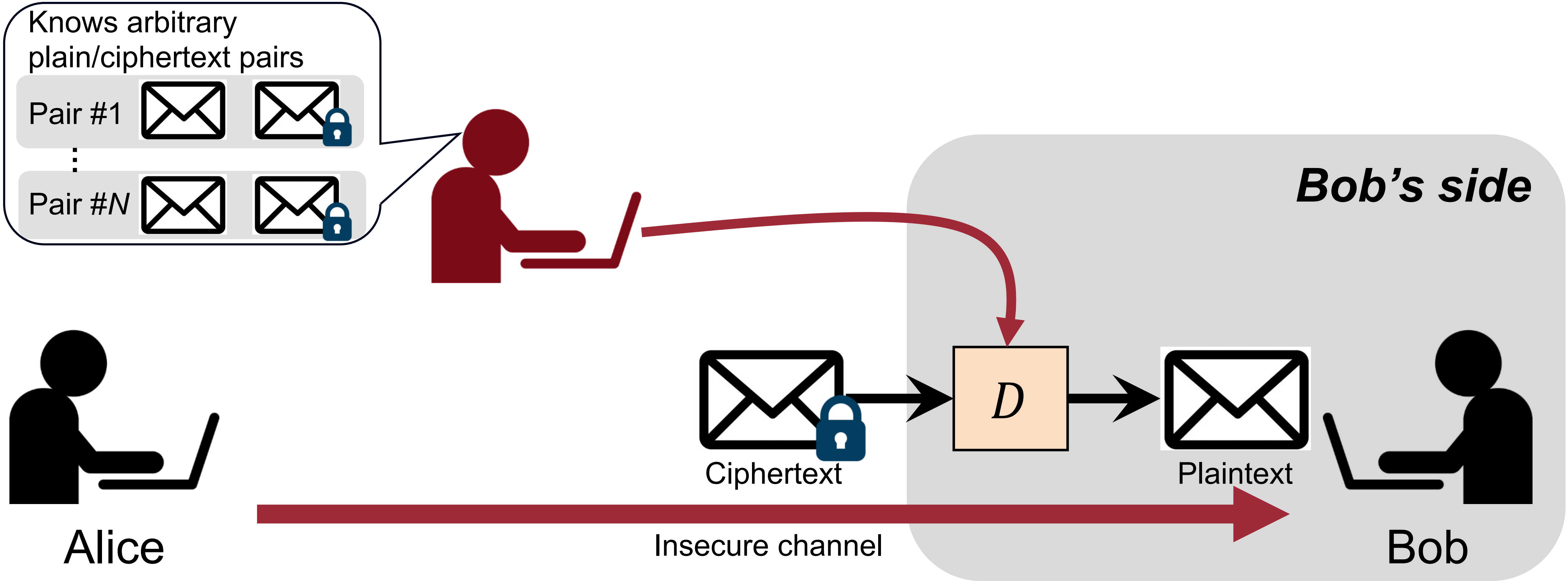
Chosen-Plaintext Attack (CPA) – Example

- ✓ The bit we vary is consistently negated
- ✓ As one bit varies, the remaining ones are left unchanged

	Plaintext	Ciphertext
Try #1	11111	01001
Try #2	11110	01000
Try #3	11101	01011
Try #4	11011	01101
Try #5	10111	00001
Try #6	01111	11001
Final	00001 ←	10111

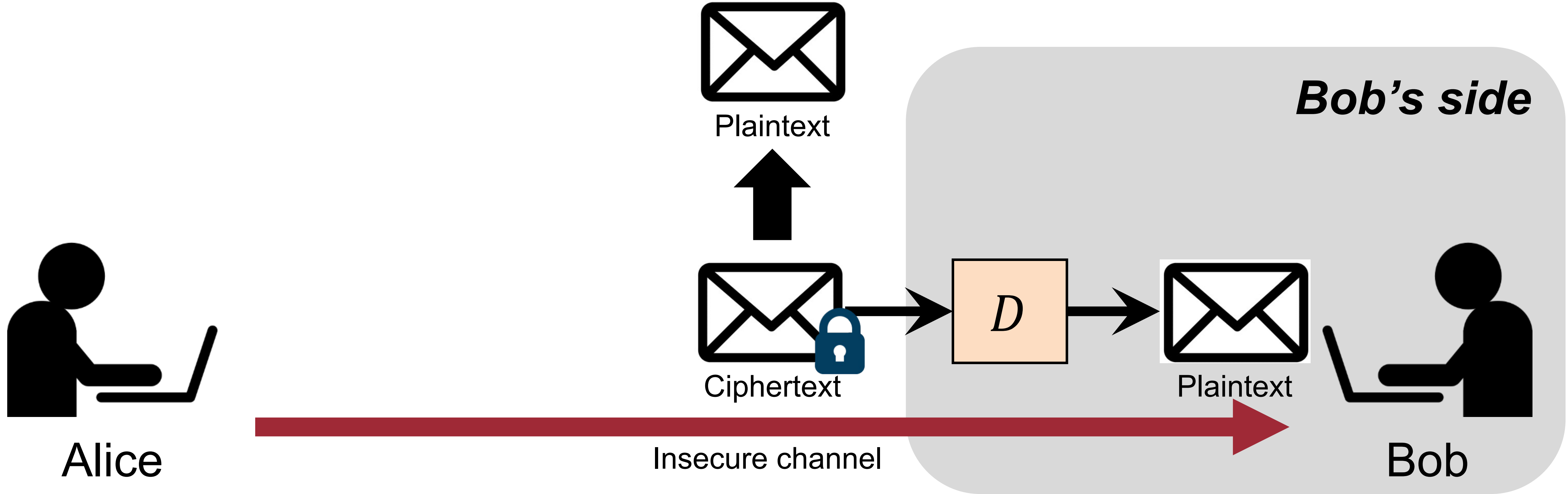
Chosen-Ciphertext Attack (CCA)

- The attacker is assumed to have access to the plaintexts for all ciphertexts other than the target



Chosen-Ciphertext Attack (CCA)

- The attacker is assumed to have access to the plaintexts for all ciphertexts other than the target
- Can the attacker compute the key from the ciphertext?



Brute Force Search

- Always possible to simply try every key!
- Therefore, the key should be secure against **exhaustive key search**!

Key size (Bits)	# of alternative keys	Time required at 1 decryption/μs	Time required at 10^6 decryption/μs
32	$2^{32} = 4.3\times10^{9}$	2^{31}μs = 5.8 minutes	2.15 milliseconds
56	$2^{56} = 7.2\times10^{16}$	2^{55}μs = 1,142 years	10.01 hours
128	$2^{128} = 3.4\times10^{38}$	2^{127}μs = 5.4 × 10^{24} years	5.4 × 10^{18} years
168	$2^{168} = 3.7\times10^{50}$	2^{167}μs = 5.9 × 10^{36} years	5.9 × 10^{30} years

Summary

- Classical cryptography: ad-hoc design & informal proof
  - Caesar's cipher, Substitution cipher, Vigenere cipher

- Modern cryptography: rigorous design & formal proof
  - Security guarantee
  - Threat model:
    - Ciphertext-Only Attack
    - Known Plaintext Attack
    - Chose-Plaintext Attack
    - Chose-Ciphertext Attack

    \+ Brute Force Search

Question?