CS459/698 Privacy, Cryptography, Network and Data Security

Basics of Cryptography

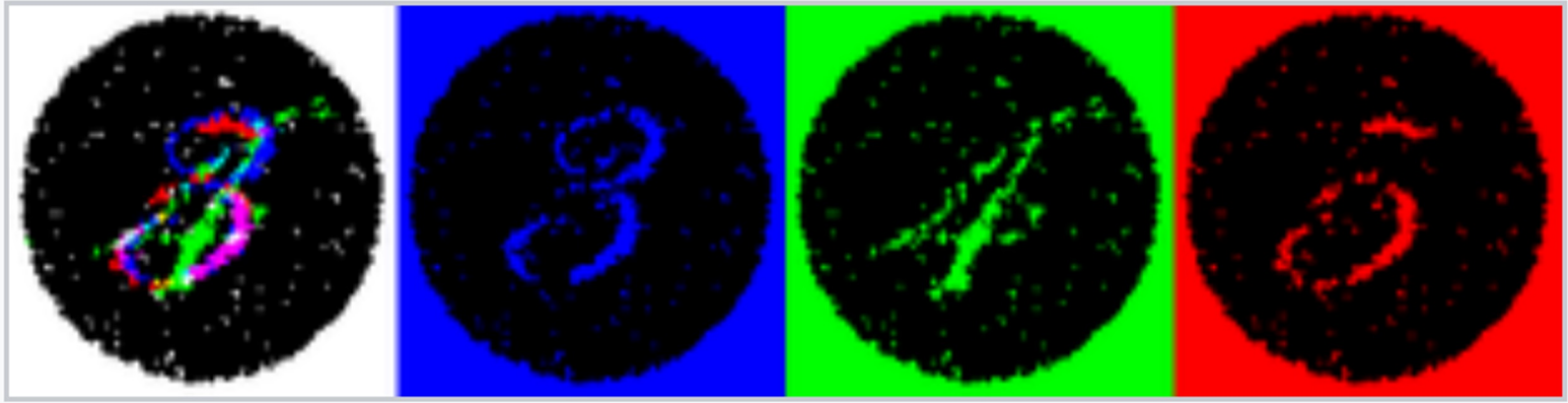
Spring 2025, Monday/Wednesday 2:30pm-3:50pm

Learning Outcomes

- Identify attack techniques and apply them (cryptanalysis)
- Explain building blocks of modern cryptography
- Explain how modern cryptography properties arose

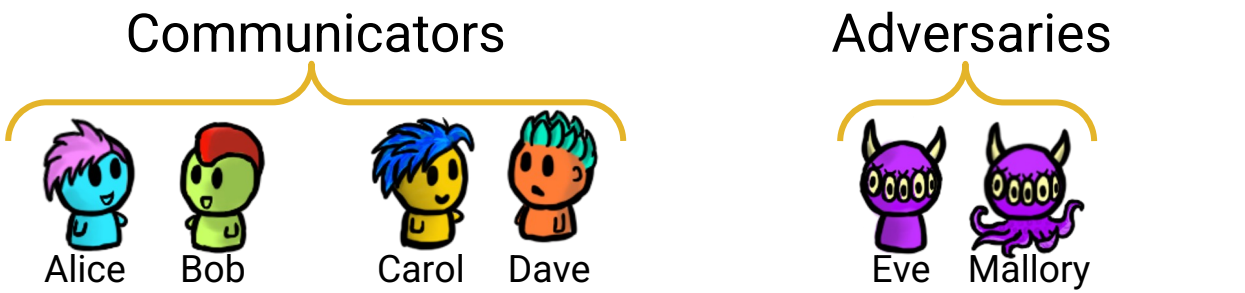
Goal: Basically, know what cryptography tools exist and how to securely use them. Build a foundation of primitives for more complicated “applied cryptography” later.

Steganography - Secretly “hidden” messages



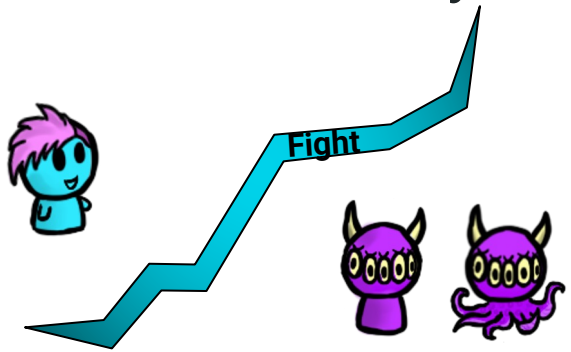
The same image viewed by white, blue, green, and red lights reveals different hidden numbers.

Cryptography - Writing “secret” messages

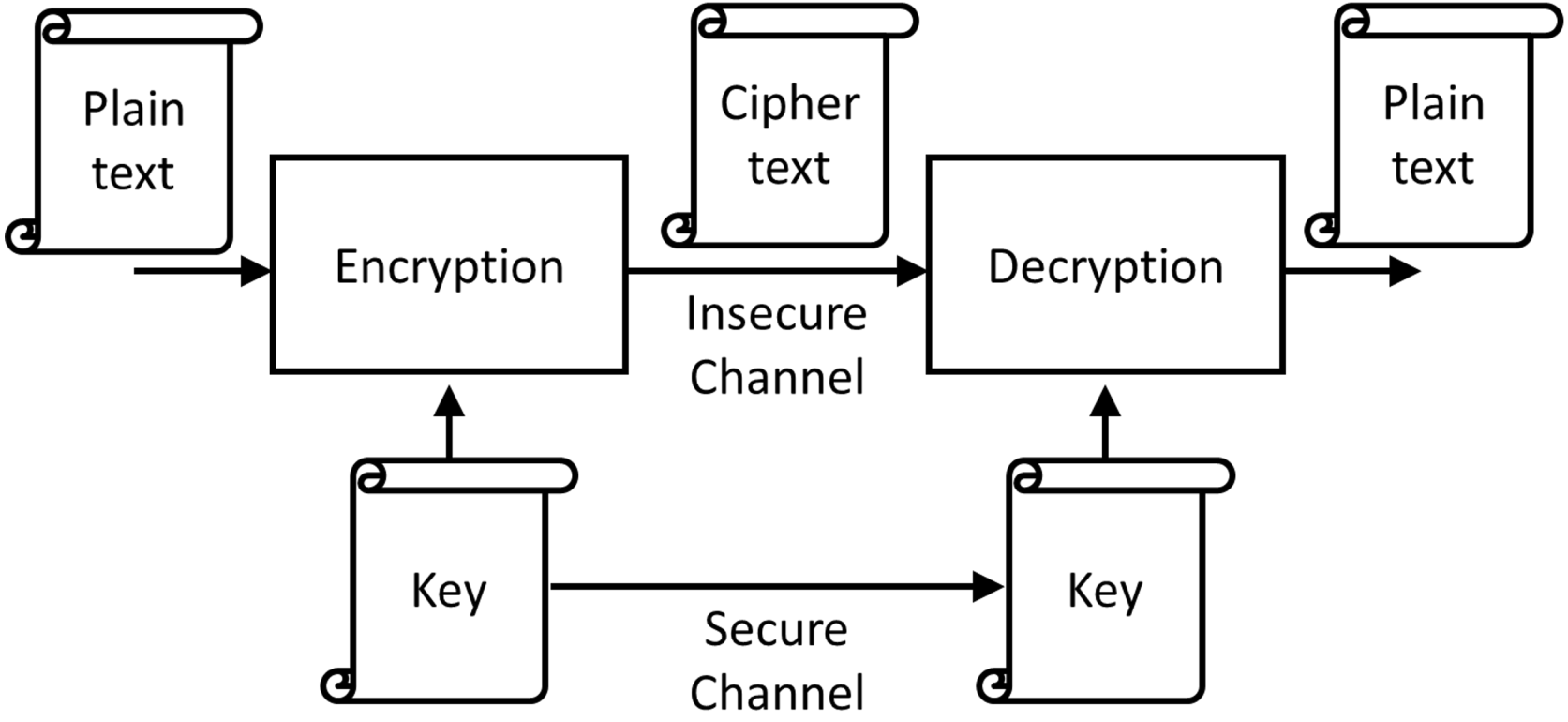


Remember CIA? Different A for Crypto Power

- **C**onfidentiality, prevent Eve **reading** Alice's messages
- **I**ntegrity, prevent Mallory from **changing** Alice's messages
- **A**uthenticity, Prevent Mallory from **impersonating** Alice



Cryptography - Path for Secret Messages



Historical Ciphers: Example One

FUBSWRJUDSKB

CRYPTOGRAPHY

Historical Ciphers: Example One

FUBSWRJUDSKB

CRYPTOGRAPHY

Substitution Cipher (shift 3)
(monoalphabetic)

Caesar Cipher

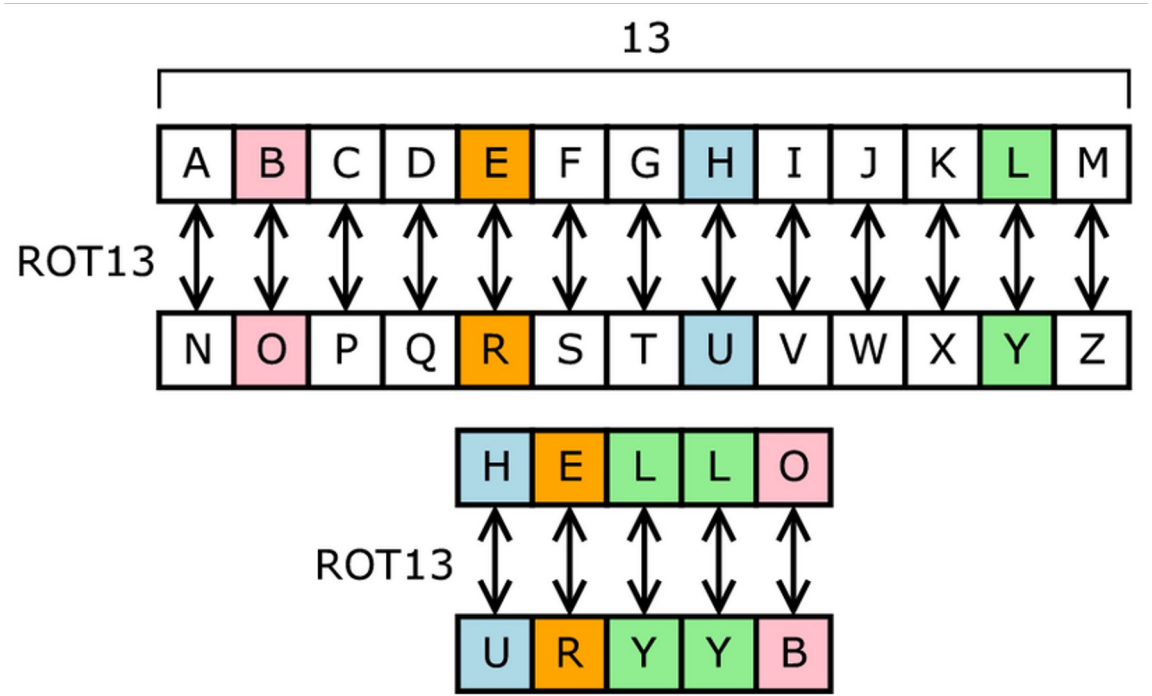
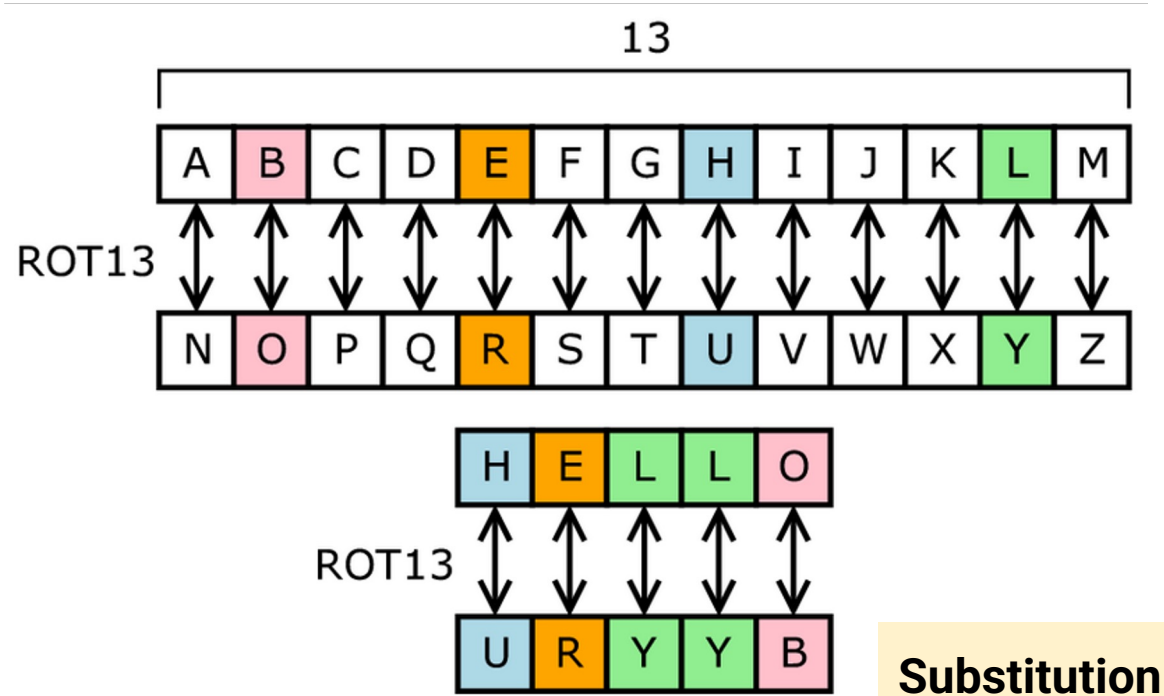


Image source: wikipedia

Caesar Cipher



Substitution Cipher (shift 13)

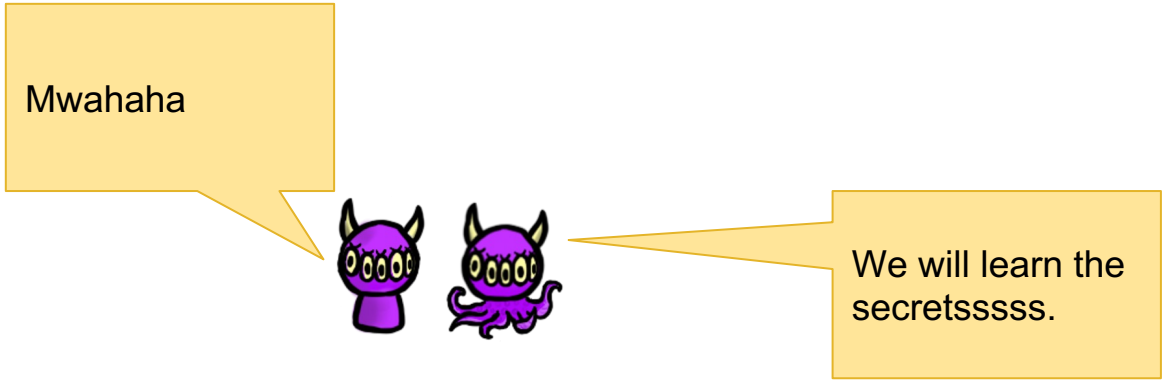
Image source: wikipedia

Shift and Substitution Ciphers

Replace symbols (letters) by others

- Using a rule
  - e.g., y = x + 3 (mod 26), Caesar's cipher Key = 3
- Using a table
  - e.g, Key: table

Cryptanalysis - Analyzing “secret” messages



Historical Ciphers: Example Two

wordplays™|com

Crossword Solver | Scrabble Word Finder | Boggle | Text Twist | Sudoku | Anagram Solver | Word Games

Wordle | Scrabble Help | Words with Friends Cheat | Words in Words | Word Jumbles | Word Search | Scrabble Cheat | Cryptogram

DAILY CRYPTOGRAM

Daily Cryptogram Help

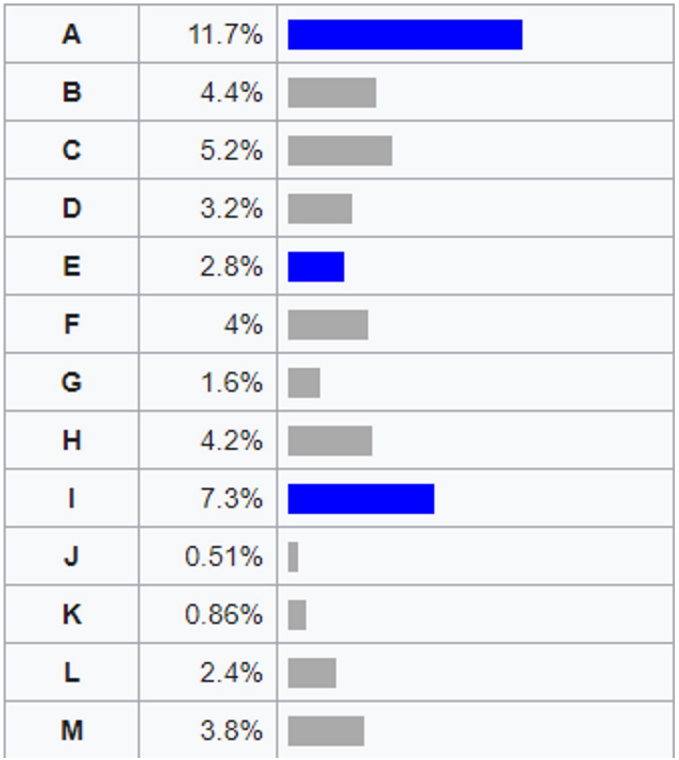
Puzzle #1267 - CATEGORY: DEFINITIONS

Puzzle # Find

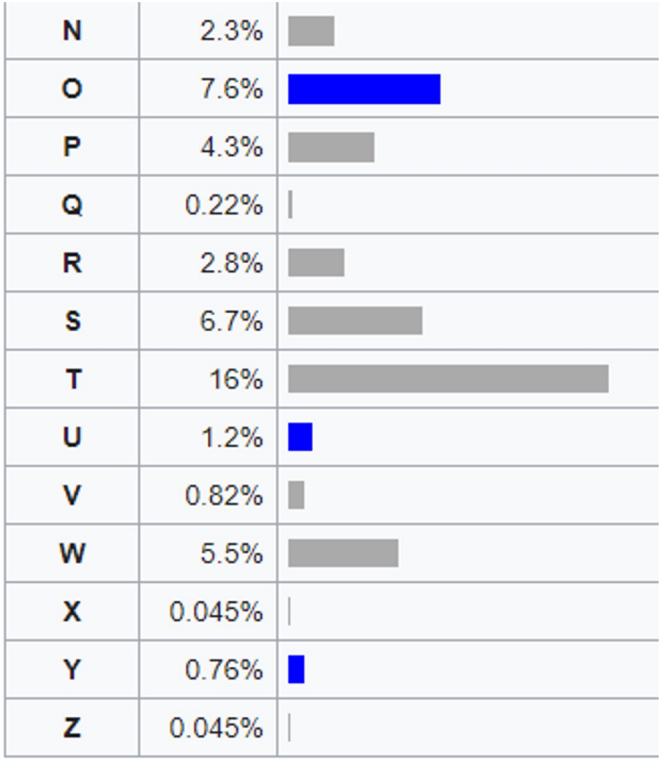
```
      ,   .  :
T V J M G Q P E S M P U ,  G . :  Q F P

P W R E A R M Z Q M G I  C E V R P Y Y  B A E M G I

U F M R F  C P E Y V G G P D  V K K M R P E Y

Y P C Z E Z Q P  Q F P  U F P Z Q  K E V O  Q F P  R F Z K K

- -        .
- - Q F P G  F M E P  Q F P  R F Z K K .
```

Get a Hint | Solve the Puzzle | New Puzzle | Clear

English Frequency

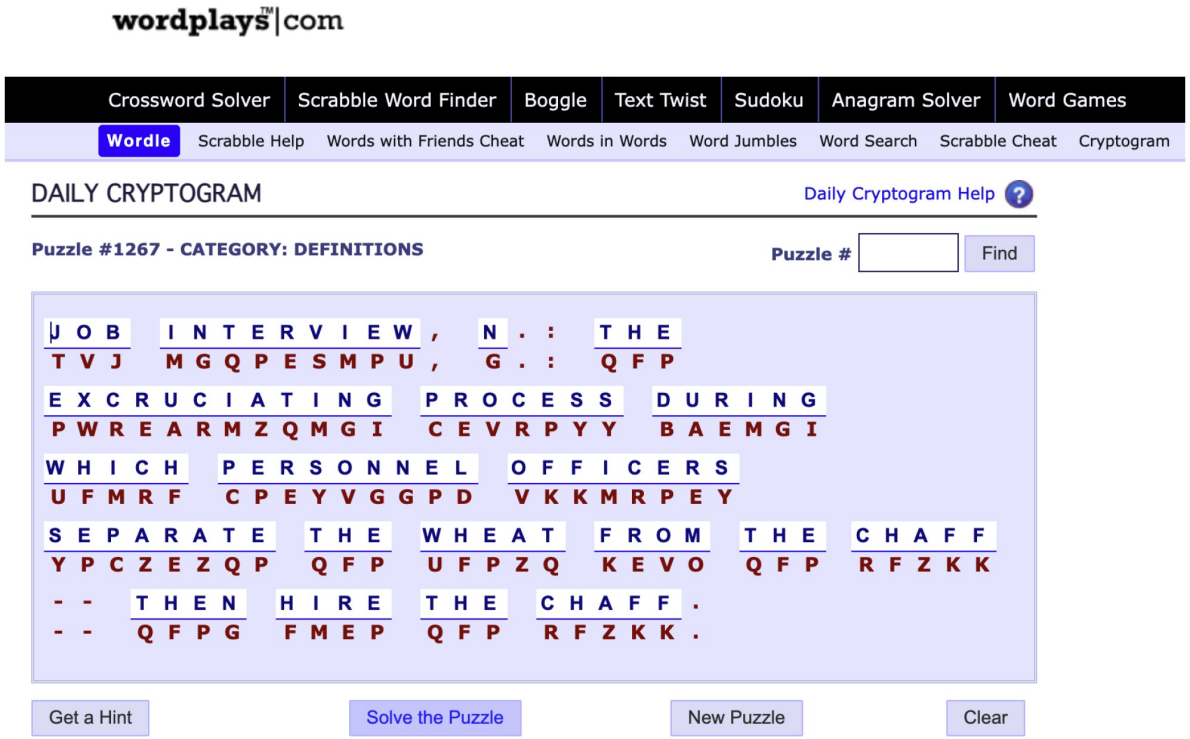


A	11.7%
B	4.4%
C	5.2%
D	3.2%
E	2.8%
F	4%
G	1.6%
H	4.2%
I	7.3%
J	0.51%
K	0.86%
L	2.4%
M	3.8%

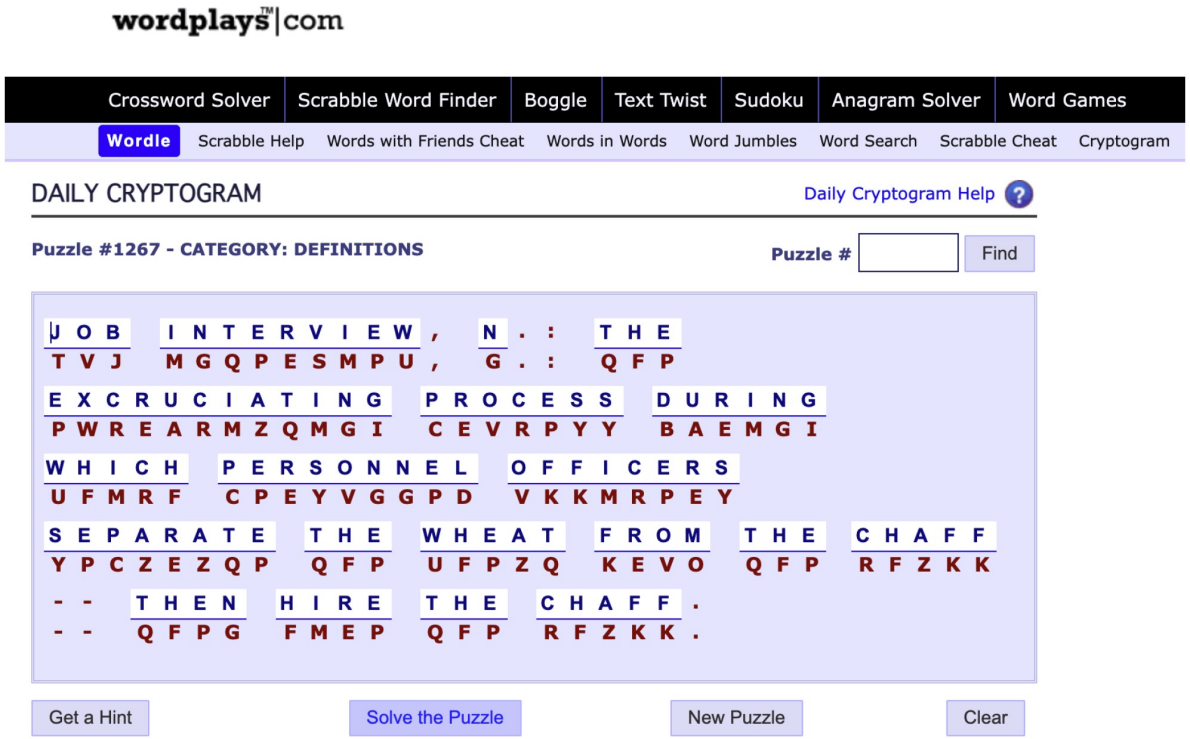


N	2.3%
O	7.6%
P	4.3%
Q	0.22%
R	2.8%
S	6.7%
T	16%
U	1.2%
V	0.82%
W	5.5%
X	0.045%
Y	0.76%
Z	0.045%

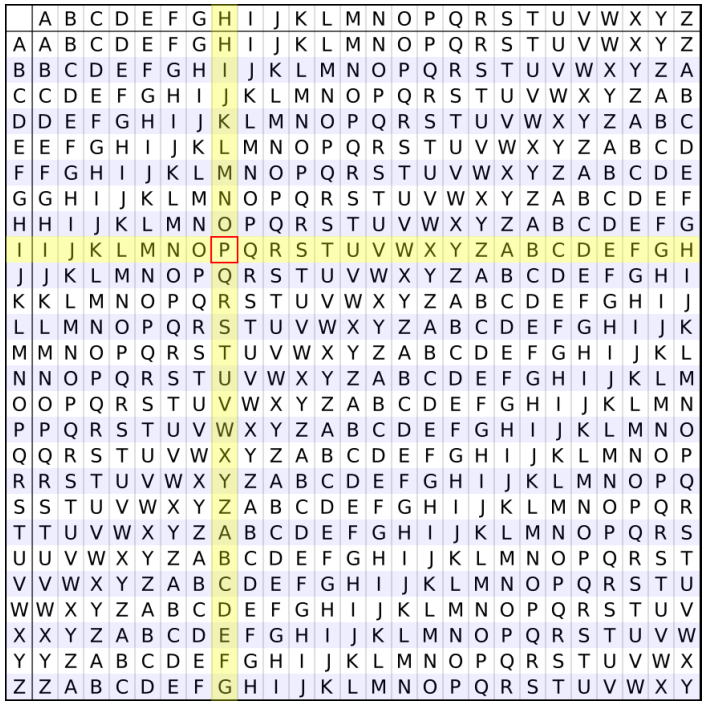
Historical Ciphers: Example Two



Historical Ciphers: Example Two



Historical Ciphers: Example Three – Vigenère



	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
A	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
B	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	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

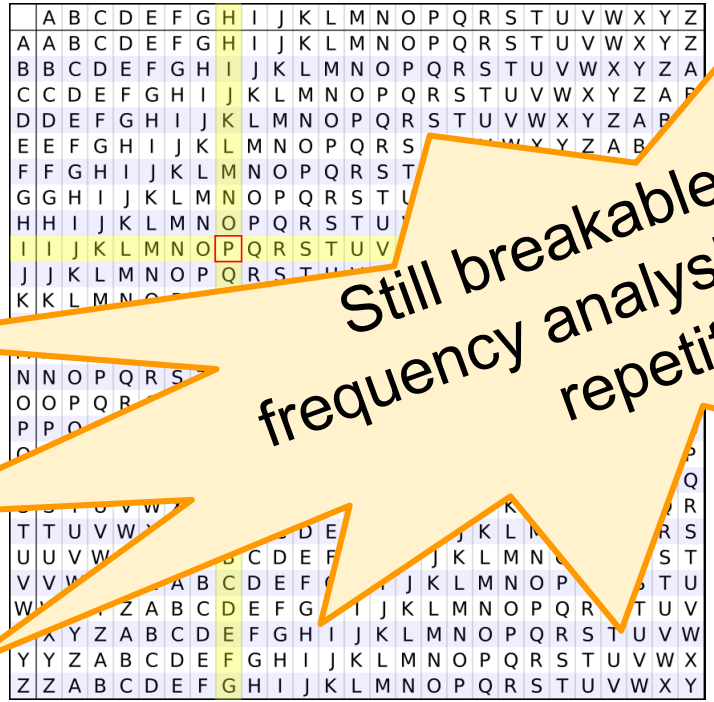
Key: KEYKE

Message: HELLO

Ciphertext: RIJVS

Poly-Alphabetic Substitution Cipher

Historical Ciphers: Example Three – Vigenère



Still breakable through frequency analysis (due to key repetition)

Message: HELLO

Ciphertext: RIJVS

Poly-Alphabetic Substitution Cipher

Kerckhoff Principle

The security of a cryptosystem should solely depend on the secrecy of the key, but never on the secrecy of the algorithms.

Historical Ciphers: Example Four

LECTURE SECURITY AND CRYPTOGRAPHY I

LENGECDRCUCATRRPUIYHRTPYEYTISAO

Historical Ciphers: Example Four

LECTURES

ECURITYA

NDCRYPTO

GRAPHYI

LENGECDRCUCATRRPUIYHRTPYEYTISAO

Transposition Cipher (Key: 8 letters)

Historical Ciphers: Example Four

LECTURES

ECURITYA

N

GRA

YHRTPYEYTISAO

Shannon's maxim!!!! (design assuming adversaries will learn the algorithm)

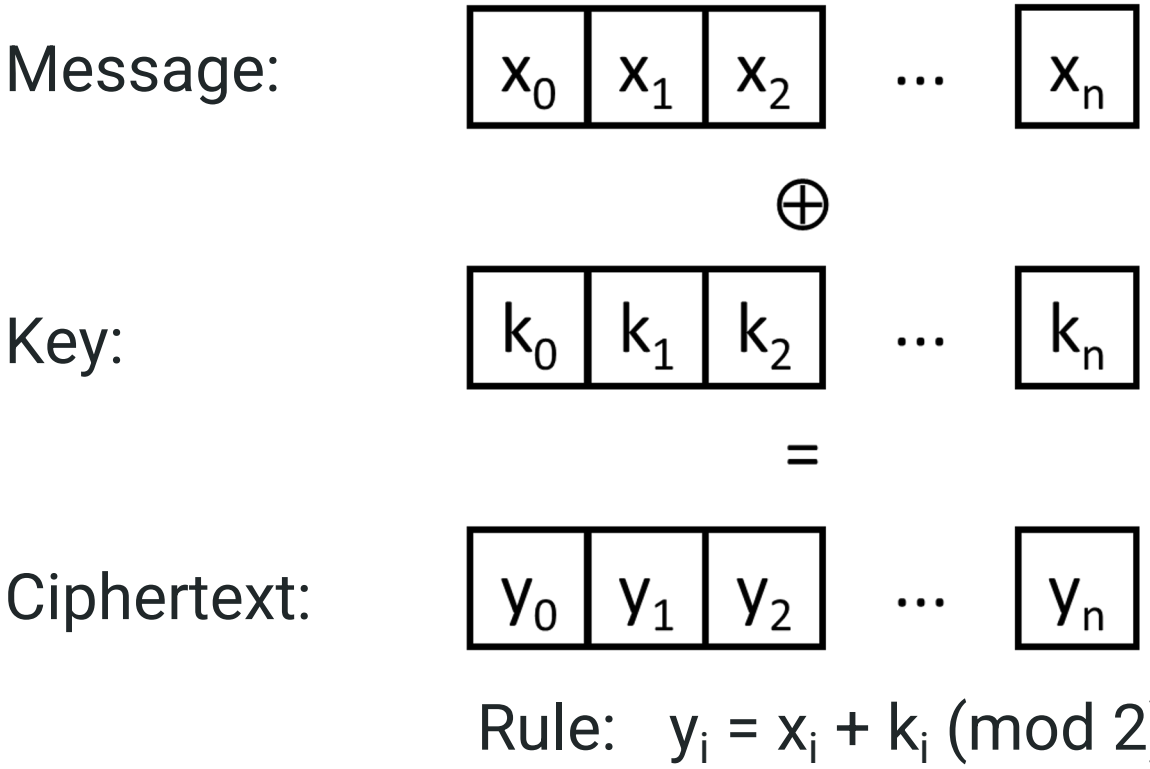
Transposition Cipher (Key: 8 letters)

Shannon's Maxim & Kerkhoff's Principle:

- Security shouldn't rely on the secrecy of the method
- Do use public algorithms with secret "keys"
- The adversaries target is… the key

Idea: Easier to change a "short" key than your whole system. (e.g., Recovery)

Unconditionally Secure: One-Time Pad



Rule: $y_i = x_i + k_i \pmod 2$

Provable Security for One-Time Pad

<Ciphertext is uniformly distributed independent of the plaintext distribution>

$x_i = 0$ with probability p ($x_i = 1$: 1-p),

$k_i = 0$ with probability 0.5 ($k_i = 1$: 0.5),

$y_i = 0$ with probability:

$$p(y_i = 0) = p(x_i = 0)\, p(k_i = 0) + p(x_i = 1)\, p(k_i = 1)$$

$$= 0.5p + 0.5(1-p)$$

$$= 0.5$$

Provable Security for One-Time Pad

Every ciphertext *y* can be decrypted **into every arbitrary plaintext** *x* using the **key** *k*

Consequently the ciphertext cannot contain any information about the plaintext

Encryption is “deniable”

What if it is a Many-Time Pad?

Key: K

$\text{Ciphertext}_1 = \text{message}_1 \oplus K =$ `2c15491000431 30b1000290a1b`

$\text{Ciphertext}_2 = \text{message}_2 \oplus K =$ `3f16421617520 3114c020b1c`

Hmmm… how can I relate these messages together?

What if it is a Many-Time Pad?

Key: K

$\text{Ciphertext}_1 \oplus \text{Ciphertext}_2 =$

$\text{message}_1 \oplus K \oplus \text{message}_2 \oplus K =$

$\text{message}_1 \oplus \text{message}_2 =$ `13030b0617544108014c2b0107`

What if it is a Many-Time Pad?

$message_1 \oplus message_2$ = `13030b061754410801 4c2b0107`

Suppose $message_1$ starts with “Alice” `(414C696365)`

- $message_2$ seems to start with readable text (“Rober”)

Is “Alice” here…?

What if it is a Many-Time Pad?

$message_1 \oplus message_2$ = `13030b061754410801 4c2b0107`

Suppose $message_1$ starts with “Alice” `(416C696365)`

- $message_2$ seems to start with readable text (“Rober”)

Suppose it starts with “Alice and Bob” `(416C69636520616E6420426F62)`

- message2 is fully readable now! (“Robert feline”)

Many-time pad? Messages Lack True Randomness

C_1

C_2

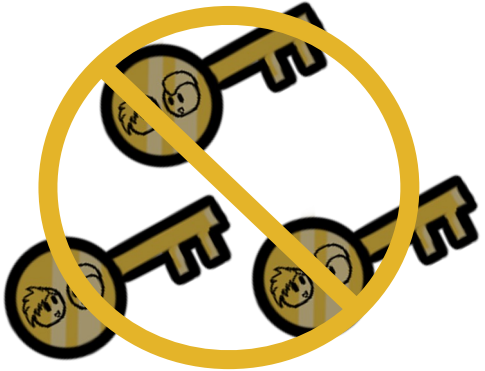
$C_1 \oplus C_2$

M_2

M_1

One-Time Pad - Conditions…

- Key uniformly random
- Only used once
- Key as long as the message



So…Cryptography?

- Simple substitution/transposition is insecure
- One-Time Pad is inefficient
  - Keys as long as messages – think about encrypting GBs of data!

Goal: Securely communicate “a lot” of information on an insecure channel while requiring “limited” communication over a secure channel

Now what?

Substitution is insecure…

Transposition is insecure…

Key reuse using XOR (one-time pad) is insecure…

BUT…

Repeat it often enough and it can be regarded as secure

Now what?

Substitution is insecure…

Transposition is in

Key reuse

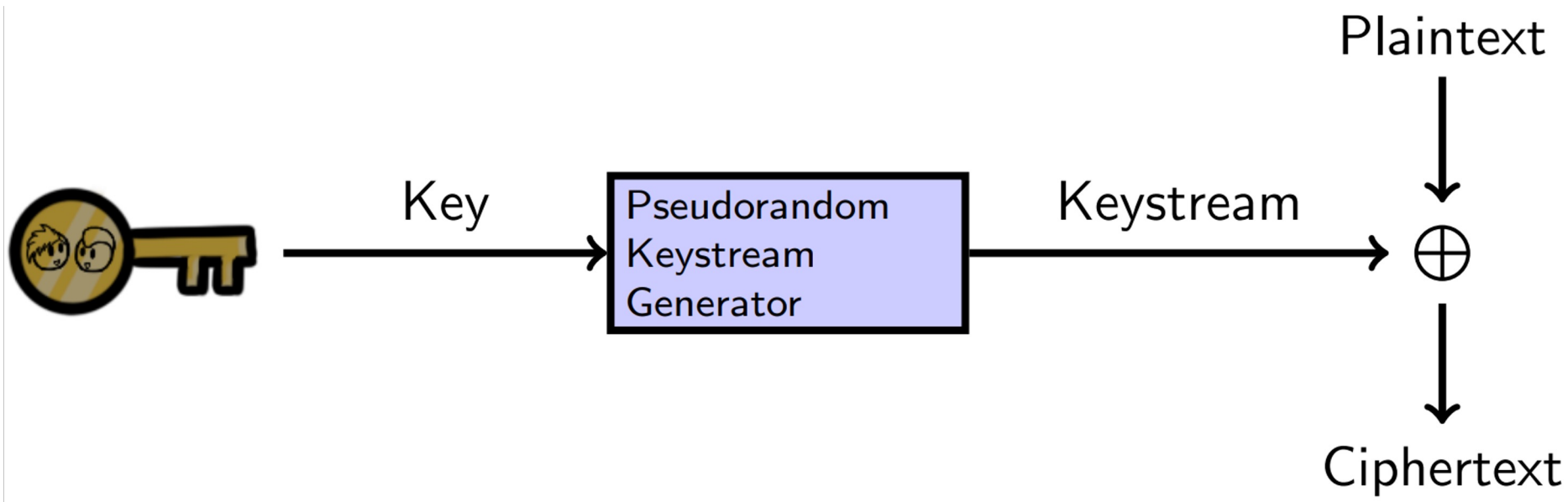
d) is insecure…

Stream Ciphers and Block Ciphers

BUT…

Repeat it often enough and it can be regarded as secure

Stream Cipher?

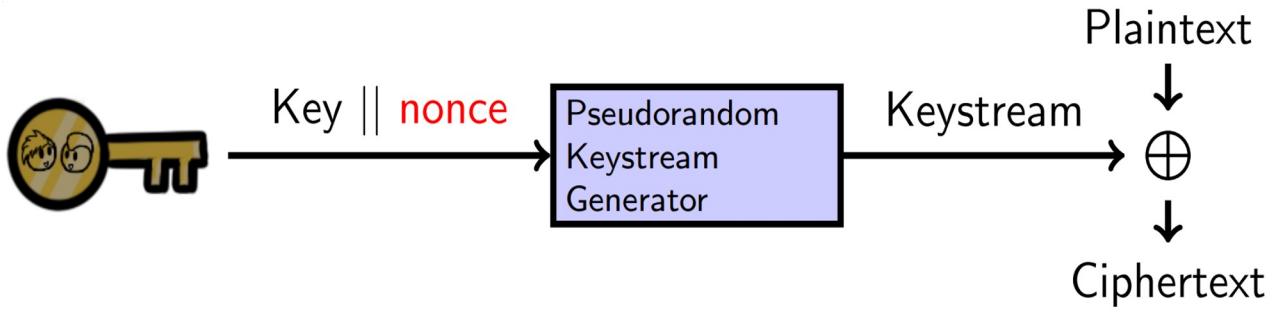


Fun(?) Facts:

- ChaCha increasingly popular (Chrome and Android), and SNOW3G in mobile phone networks.

Stream Ciphers Share Conditions with OTP

- Stream ciphers can be very fast
  - This is useful if you need to send a lot of data securely
- But they can be tricky to use correctly!
  - We saw the issues of re-using a key! (two-time pad)
  - **Solution:** concatenate key with nonce (which does not need to be a secret)



Fun(?) Facts:

- WEP, PPTP are great examples of how not to use stream ciphers

Bit by bit…. but do you have to?

- Weakness of streams…one bit at a time?
  - What happens in a stream cipher if you change just one bit of the plaintext?

Bit by bit…. but do you have to?

- Weakness of streams…one bit at a time?
  - What happens in a stream cipher if you change just one bit of the plaintext?

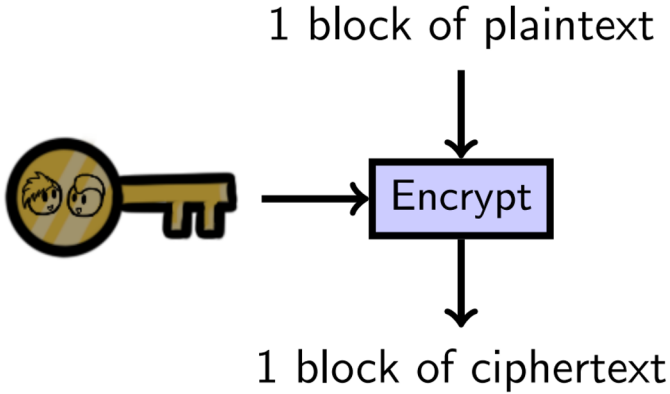
A: You only change a bit in the ciphertext

Bit by bit…. but do you have to?

- Weakness of streams…one bit at a time?
  - What happens in a stream cipher if you change just <u>one bit</u> of the plaintext?

A: You only change a bit in the ciphertext

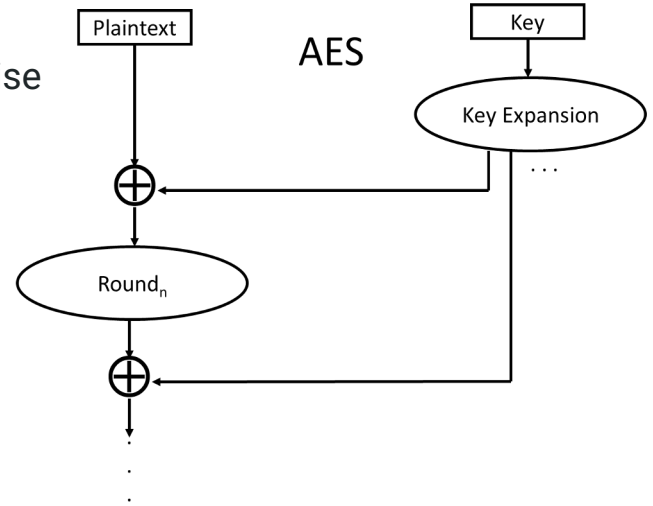
Q: Can we do better?



Block ciphers!!!

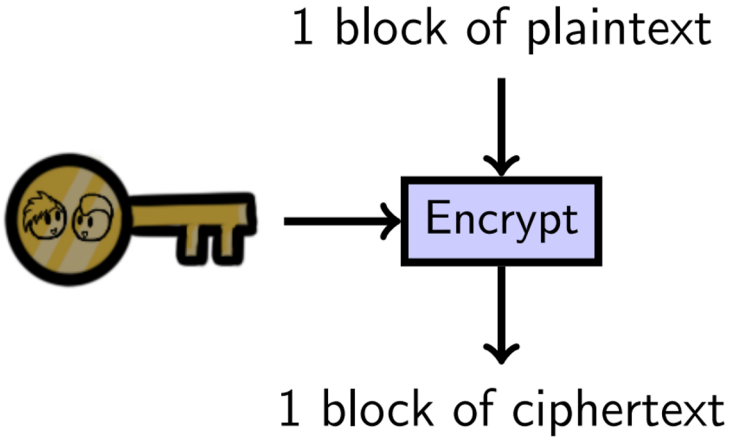
Block Ciphers

- Welcome, block ciphers
  - Block ciphers operate on the message one block at a time
  - Blocks are usually 64 or 128 bits long
- **AES,** the current standard
  - You better have a very…very good reason to choose otherwise

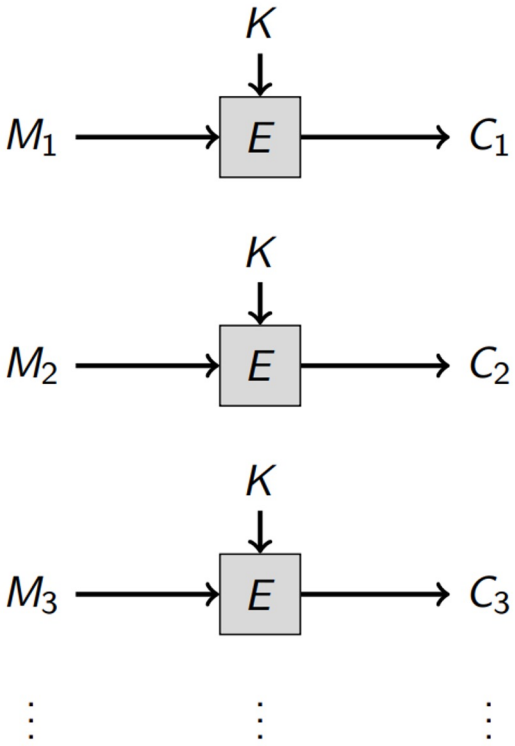


Two Catches with Block Ciphers

- Message is **shorter** than one block?
  - Requires padding
- Message is **longer** than a block?
  - Requires modes of operation

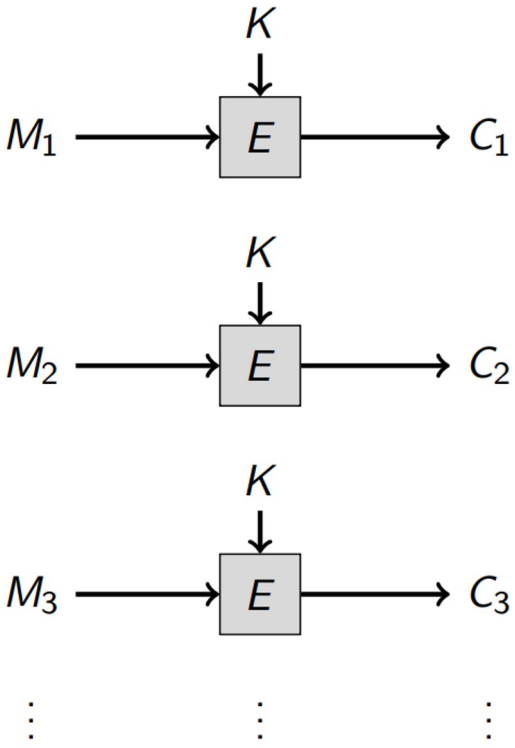


Block Ciphers and Modes of Operation: ECB Mode



- ECB: Electronic Code Book
- Encrypts each successive block separately

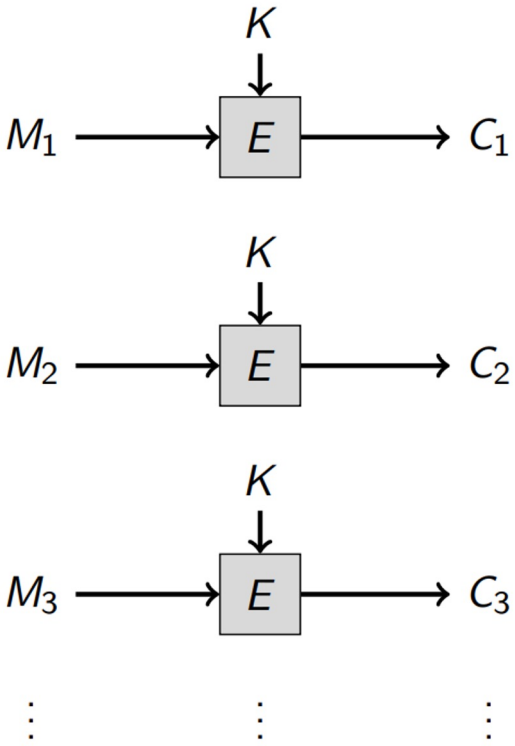
Block Ciphers and Modes of Operation: ECB Mode



- ECB: Electronic Code Book
- Encrypts each successive block separately

Q: What happens if the plaintext M has some blocks that are identical, $M_i = M_j$?

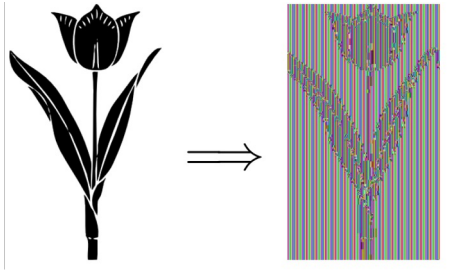
Block Ciphers and Modes of Operation: ECB Mode



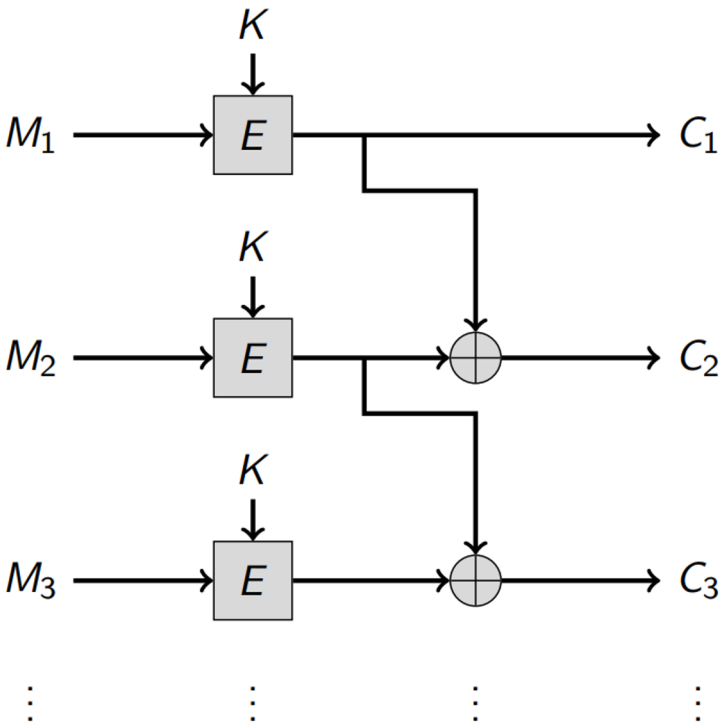
- ECB: Electronic Code Book
- Encrypts each successive block separately

Q: What happens if the plaintext M has some blocks that are identical, $M_i = M_j$?

A: $C_i = E_K(M_i)$, $C_j = E_K(M_j) \Rightarrow C_i = C_j$



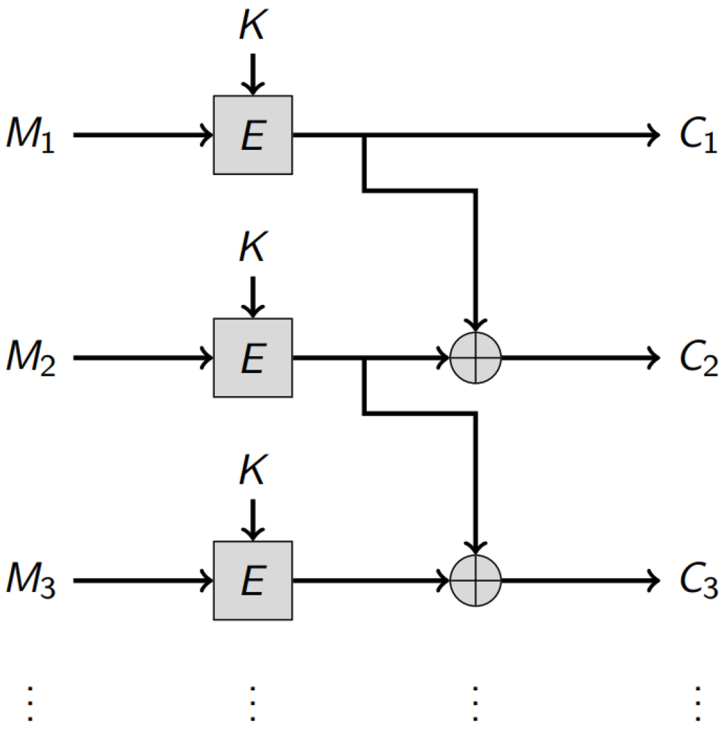
Attempt 1: Fixing ECB_1



- Provide “feedback” among different blocks, to avoid repeating patterns…

Q: Fix repeating patterns? Are there other issues?

Attempt 1: Fixing ECB_1

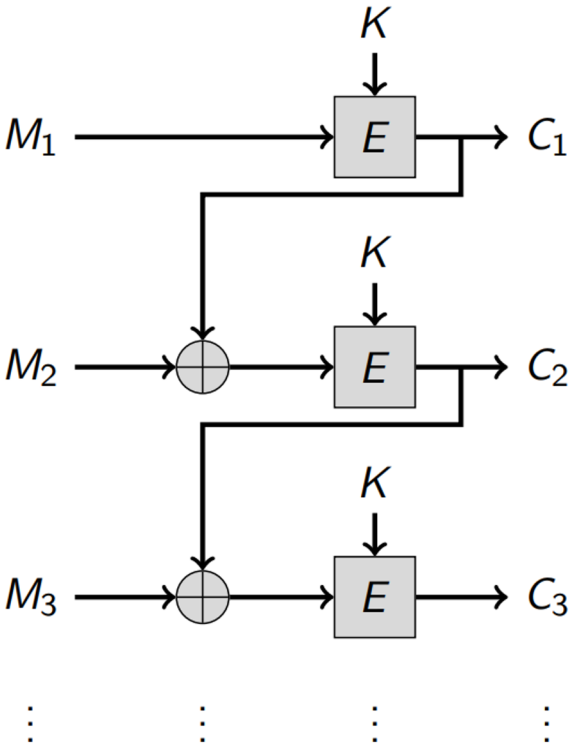


- Provide “feedback” among different blocks, to avoid repeating patterns…

Q: Fix repeating patterns? Are there other issues?

A: Yes. But We can un-do the XOR if we get all the ciphertexts. This basically does not improve compared to ECB.

Attempt 2: ECB_2!!!

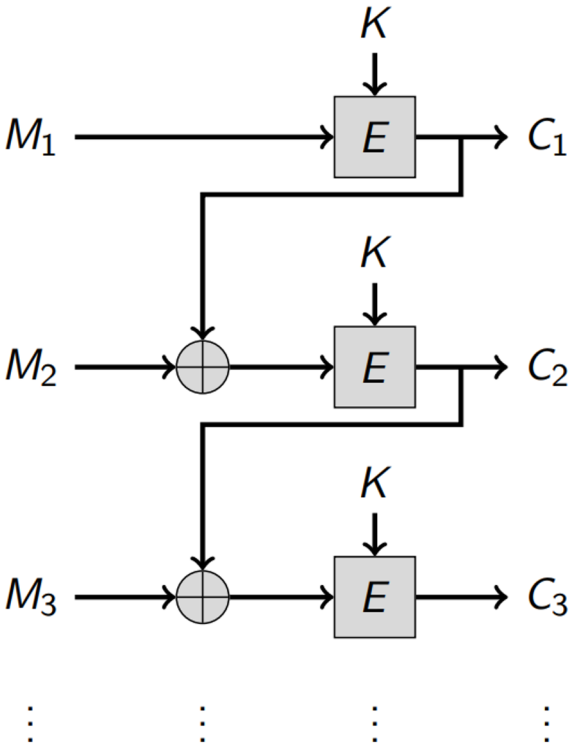


Q: Spot the difference?

Q: Is it fixed this time?

Q: Does this avoid repeating patterns among blocks?

Attempt 2: ECB_2!!!



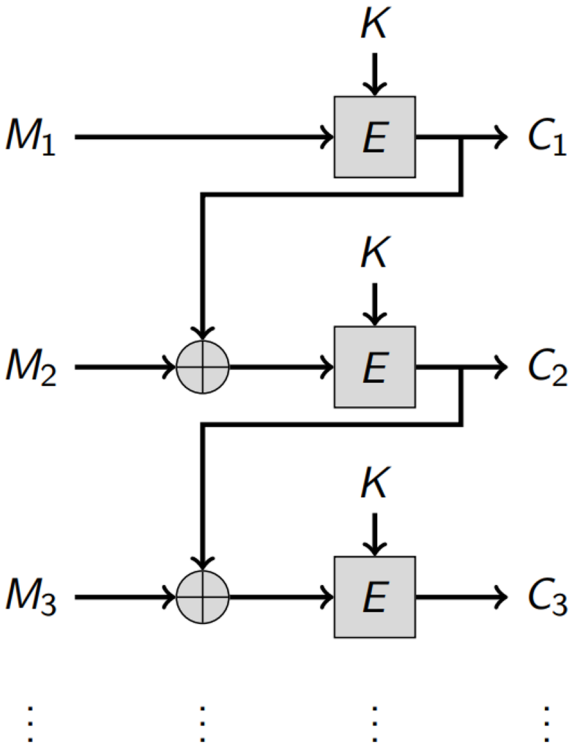
Q: Spot the difference?

Q: Is it fixed this time?

Q: Does this avoid repeating patterns among blocks?

Q: What would happen if we encrypt message M (i.e., M1|M2|M3) twice with the same key?

Attempt 2: ECB_2!!!



Q: Spot the difference?

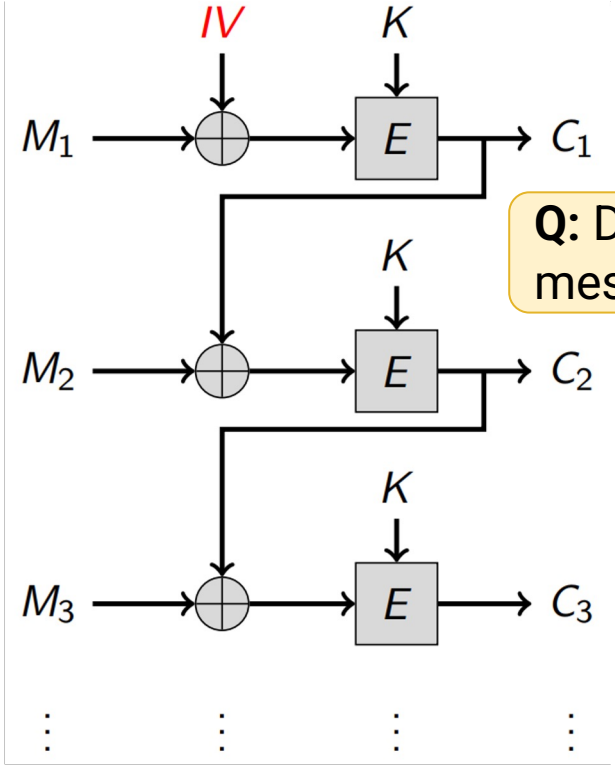
Q: Is it fixed this time?

Q: Does this avoid repeating patterns among blocks?

Q: What would happen if we encrypt message M (i.e., M1|M2|M3) twice with the same key?

A: for M = N,
$C = E_K(M), Y = E_K(N) \Rightarrow C = Y$

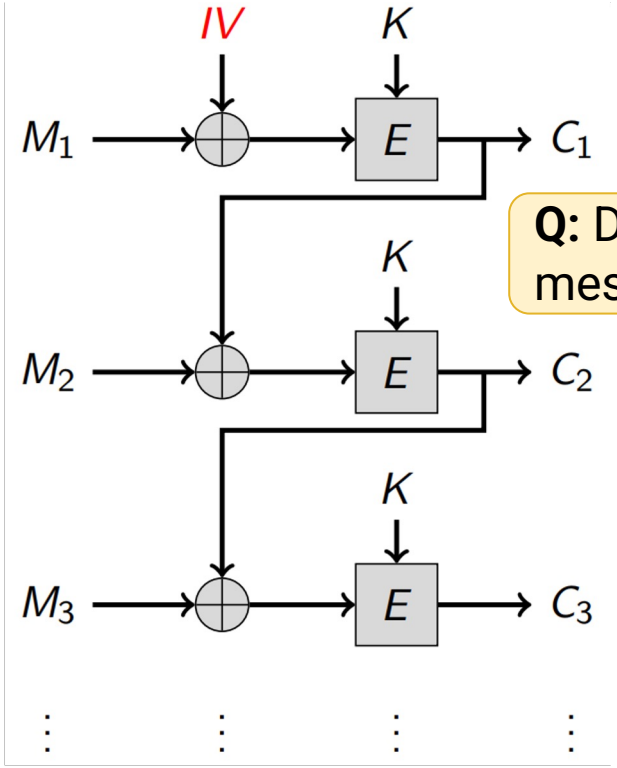
New Plan: Cipher Block Chaining (CBC) Mode



Q: Does this solve the issue of encrypting equal blocks?

Q: Does this solve the issue of encrypting equal messages/plaintexts?

New Plan: Cipher Block Chaining (CBC) Mode

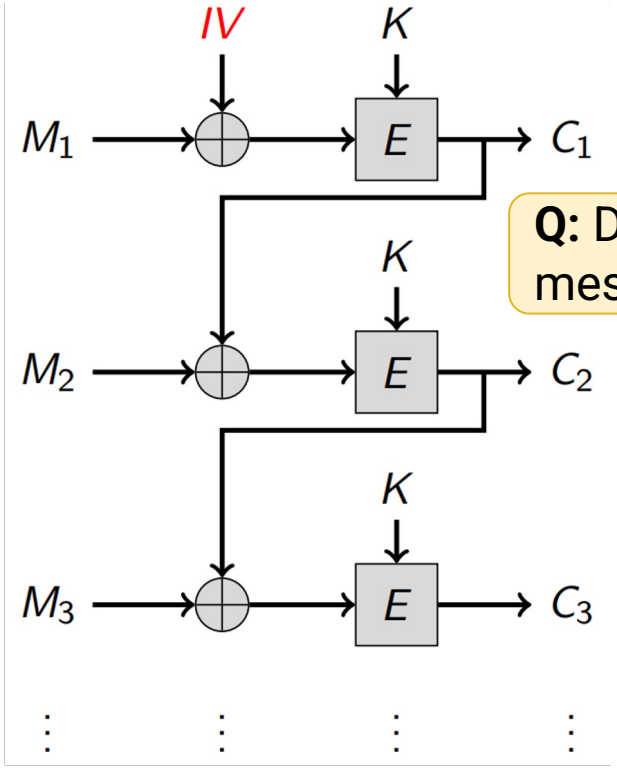


Q: Does this solve the issue of encrypting equal blocks?

Q: Does this solve the issue of encrypting equal messages/plaintexts?

A: Yes!!!

New Plan: Cipher Block Chaining (CBC) Mode



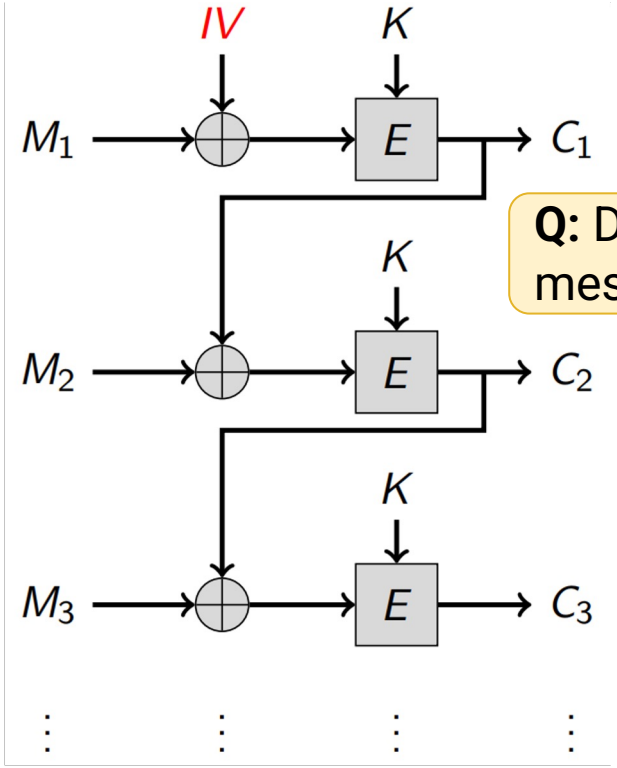
Q: Does this solve the issue of encrypting equal blocks?

Q: Does this solve the issue of encrypting equal messages/plaintexts?

A: Yes!!!

Q: Can we share IV in the clear?

New Plan: Cipher Block Chaining (CBC) Mode



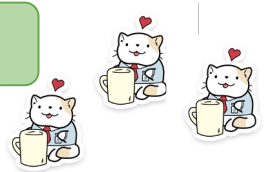
Q: Does this solve the issue of encrypting equal blocks?

Q: Does this solve the issue of encrypting equal messages/plaintexts?

A: Yes!!!

Q: Can we share IV in the clear?

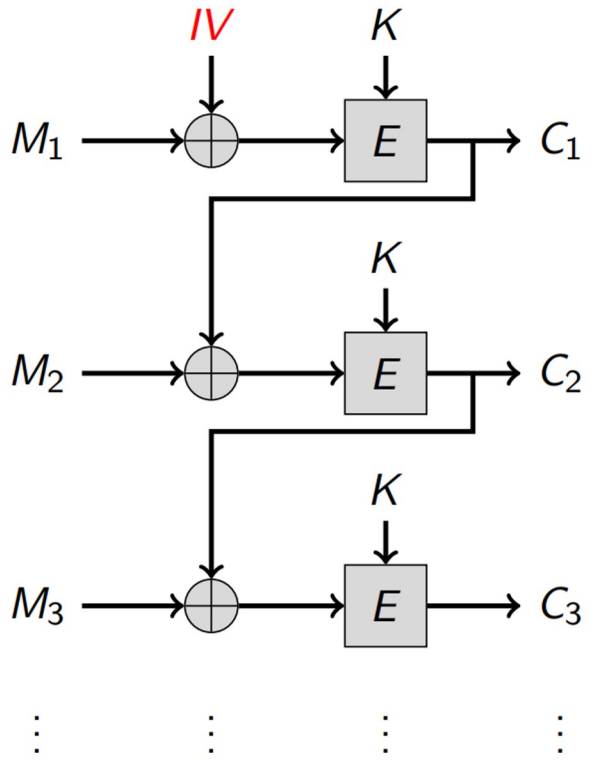
A: Yes!!!



IV, an initialization vector, nonce, salt. (1 for each message)

Recall CBC Mode for Block Ciphers:

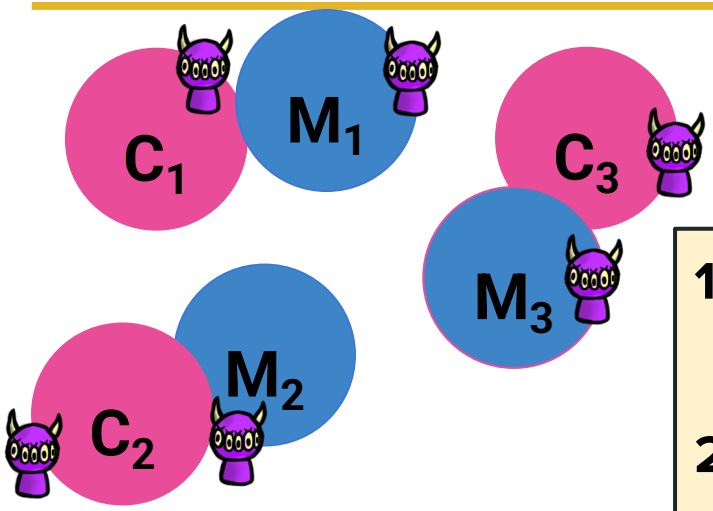
1. Generate a secret key *K*
2. Encrypt M using *K* and a generated *IV*
3. Decrypt *C* using *K* and the *IV* to get M



Security Goal: indistinguishability under adaptive chosen ciphertext attack (IND-CCA2)

Cipher Security, IND-CCA2

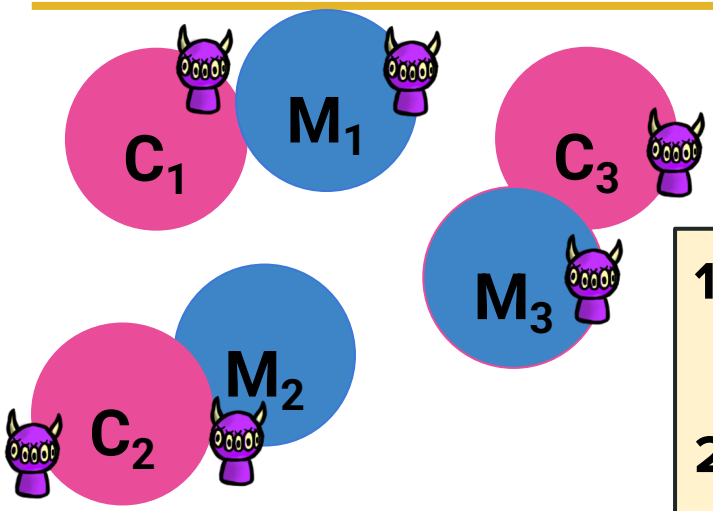
Indistinguishability under Adaptive Chosen Ciphertext Attack



1. Assume an oracle that can decrypt ciphertexts fed by the adversary
2. The adversary asks the oracle to decrypt multiple chosen ciphertexts
3. Finally, the adversary attempts to decrypt a new ciphertext by itself

Cipher Security, IND-CCA2

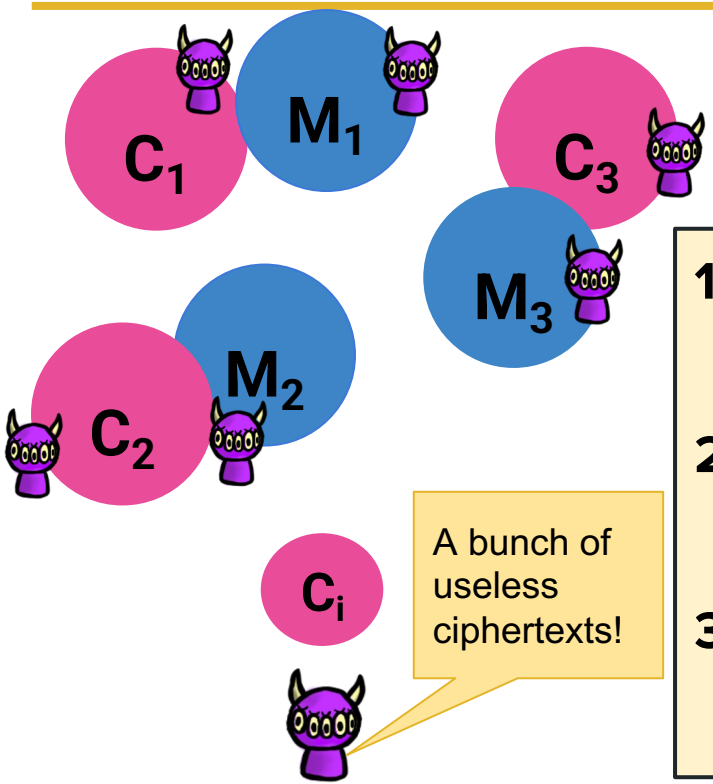
Indistinguishability under Adaptive Chosen Ciphertext Attack



1. **Assume an oracle that can decrypt ciphertexts fed by the adversary**
2. **The adversary asks the oracle to decrypt multiple chosen ciphertexts**
3. **Finally, the adversary attempts to decrypt a new ciphertext by itself**

Cipher Security, IND-CCA2

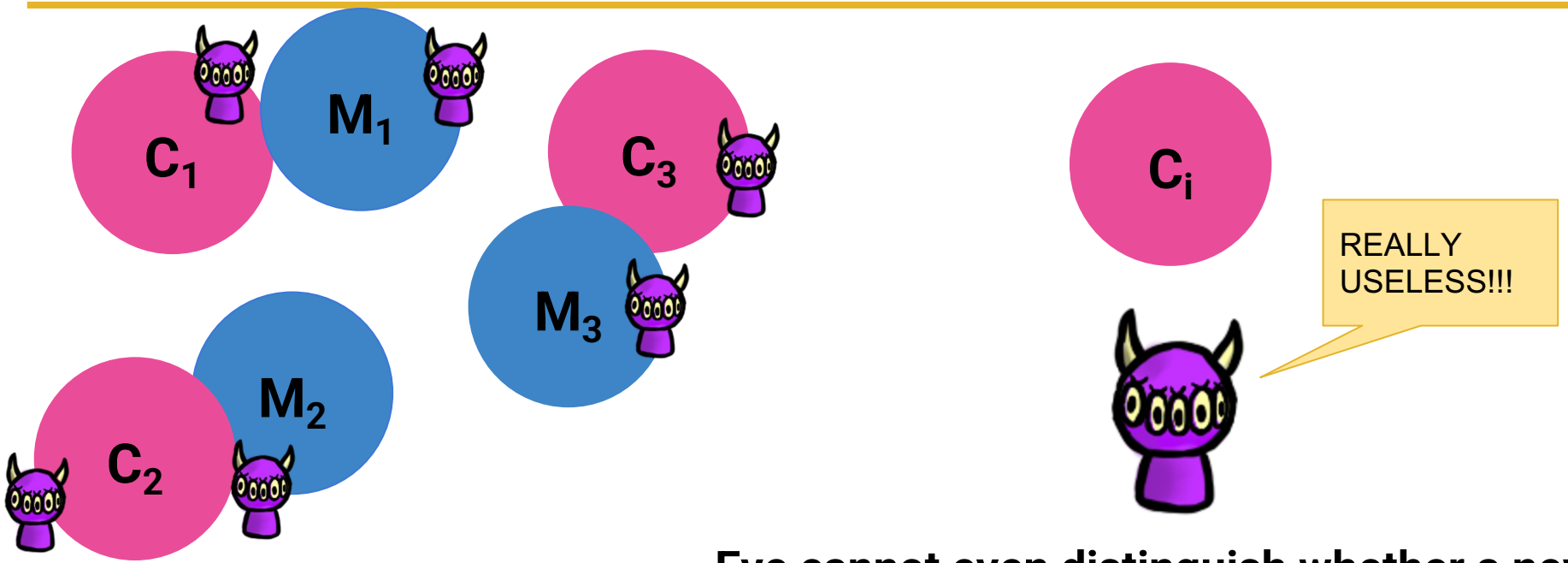
Indistinguishability under Adaptive Chosen Ciphertext Attack



1. **Assume an oracle that can decrypt ciphertexts fed by the adversary**
2. **The adversary asks the oracle to decrypt multiple chosen ciphertexts**
3. **Finally, the adversary attempts to decrypt a new ciphertext by itself**

Cipher Security, IND-CCA2

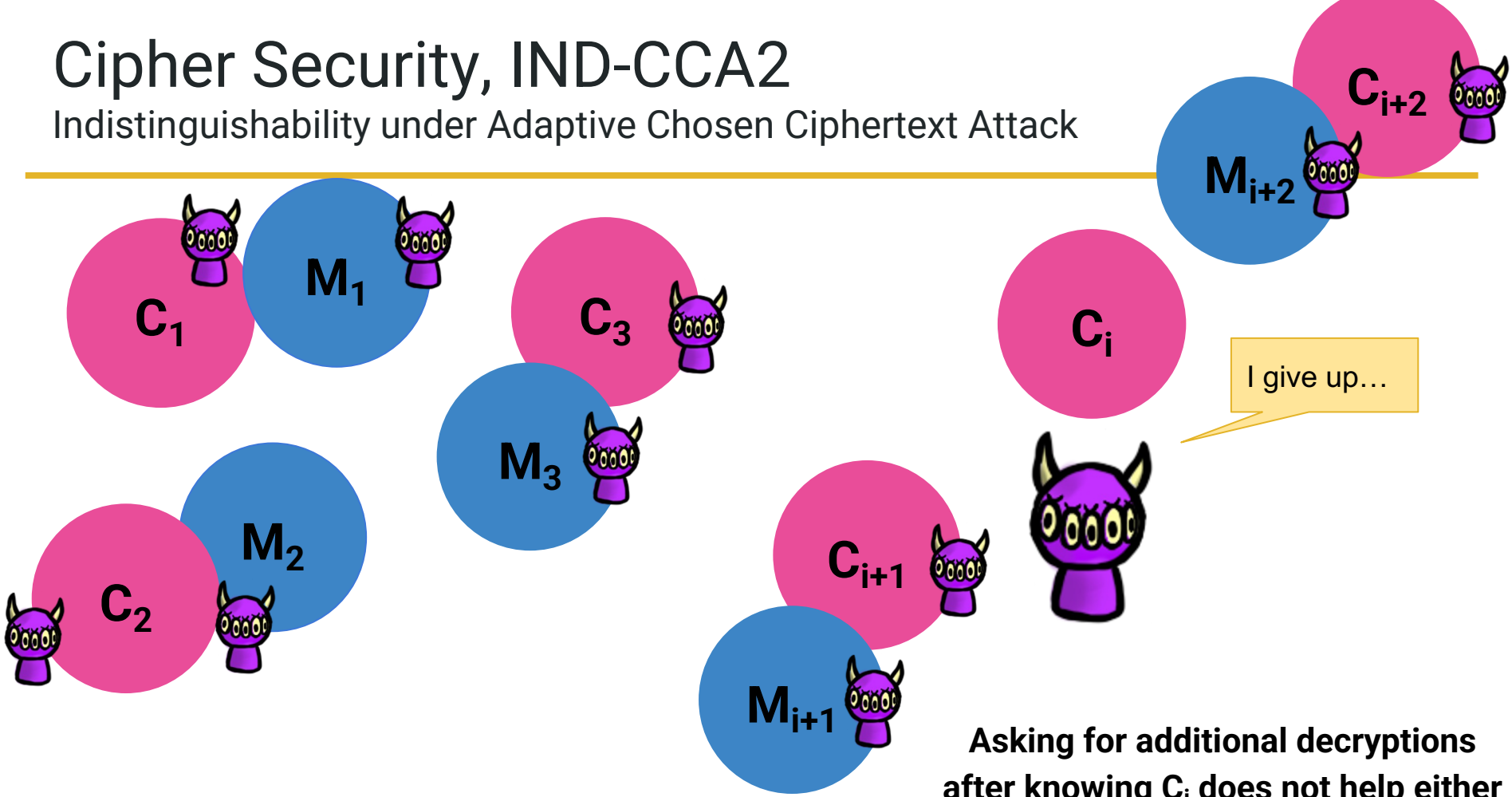
Indistinguishability under Adaptive Chosen Ciphertext Attack



Eve cannot even distinguish whether a new C_i is generated from M_1, M_2, or M_3

Cipher Security, IND-CCA2

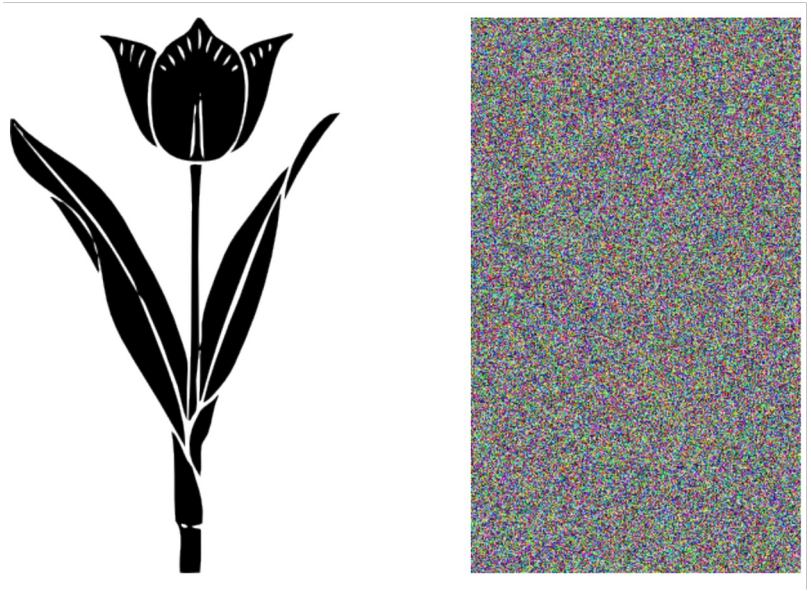
Indistinguishability under Adaptive Chosen Ciphertext Attack



Asking for additional decryptions after knowing C_i does not help either

Plenty of Modes of Operation

- e.g., Cipher Block Chaining **(CBC)**, Counter **(CTR)**, and Galois Counter **(GCM)** modes
- Patterns in the plaintext are no longer exposed because these modes involve some kind of “feedback” among blocks.
- But you need an ***IV***



So…now what?

- Alice and Bob still need to share the secret key… But how?
  - Meet in person; diplomatic courier…
- In general this is very hard

Or, we invent new technology!!

Spoiler Alert: Already been invented…

Stay tuned!