3010 Applied Crypto

Dr Tay Kian Boon

NTU, SCSE

2022/2023 Semester 2

Overview

- Intro to AES Block Cipher
- Modes of Encryption
- Intro to Key Management

AES Intro

- The *Advanced Encryption Standard (AES)* is the most widely used block cipher today.
- AES block cipher is also mandatory in several industry standards and is used in many commercial systems, such as
  - Internet security standard IPsec,
  - TLS,
  - the Wi-Fi encryption standard IEEE 802.11i,
  - the secure shell network protocol SSH (Secure Shell), the
  - Internet phone Skype and
  - numerous security products around the world.
- There are hardly any attacks better than brute-force known against AES.

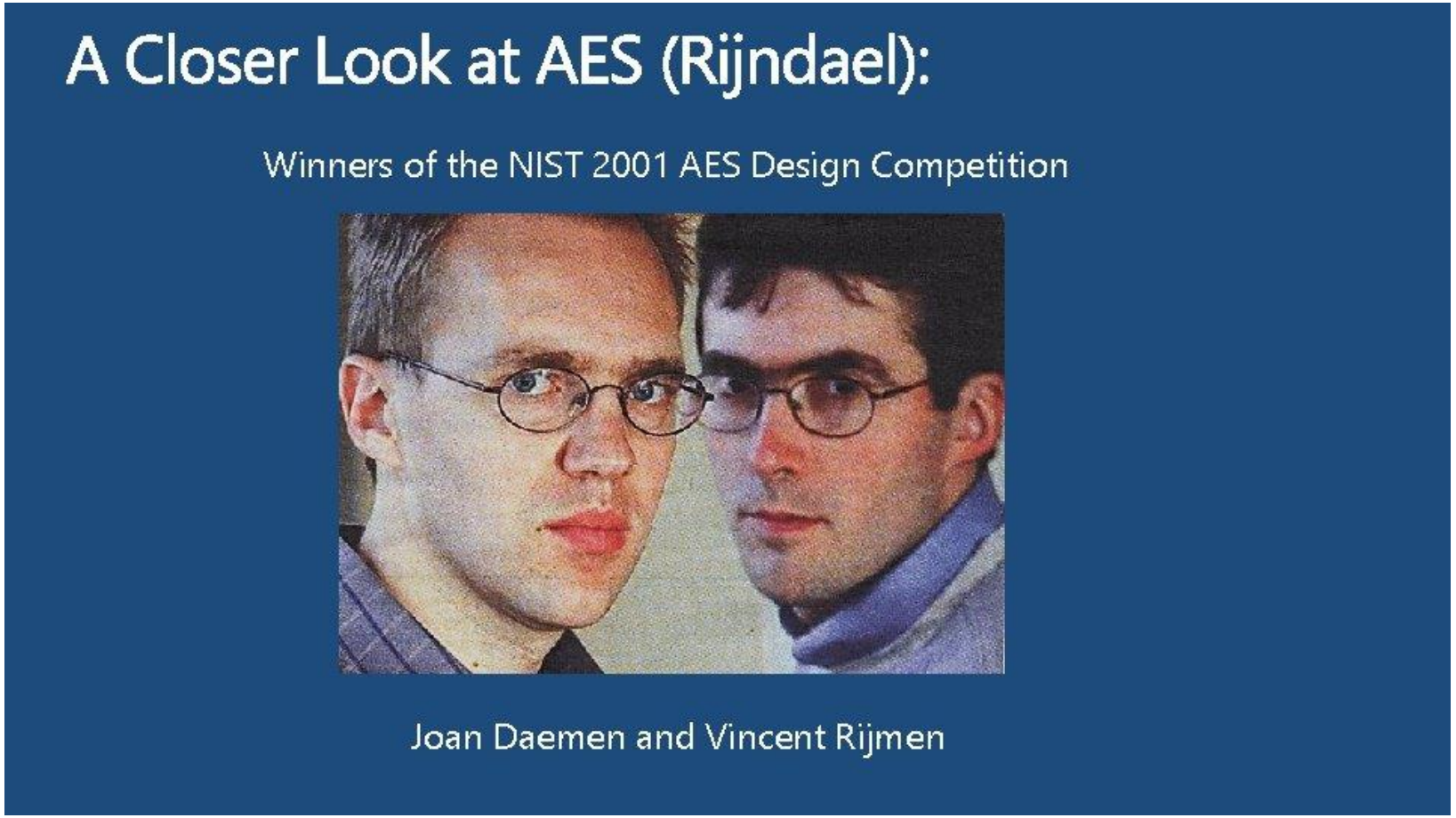
AES Intro: Background

- In 1997 NIST called for proposals for a new *Advanced Encryption Standard (AES)*.
- The selection of the algorithm for AES was an open process administered by NIST.
- In 3 subsequent AES evaluation rounds, NIST & the international scientific community discussed the advantages and disadvantages of the submitted ciphers and narrowed down the number of potential candidates to 5

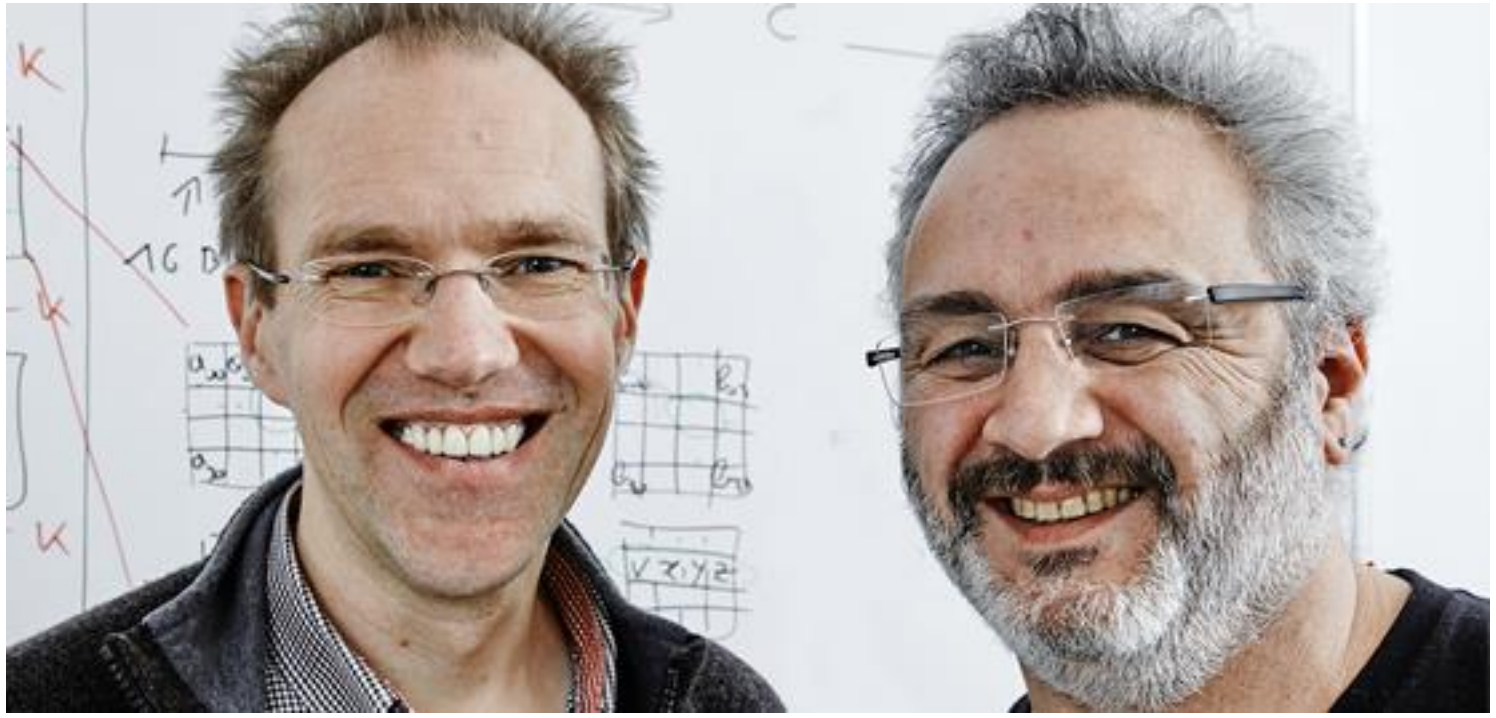
AES Intro: Background

- On August 9, 1999, five finalist algorithms were announced:
  - *Mars* by IBM Corporation
  - *RC6* by RSA Laboratories
  - *Rijndael,* by Joan Daemen and Vincent Rijmen
  - *Serpent*, by Ross Anderson, Eli Biham and Lars Knudsen
  - *Twofish*, by Bruce Schneier, John Kelsey, Doug Whiting, DavidWagner, Chris Hall and Niels Ferguson
- In 2001, NIST declared the block cipher *Rijndael* as the new AES and published it as a final standard (FIPS PUB 197).
- Rijndael was designed by two young Belgian cryptographers.

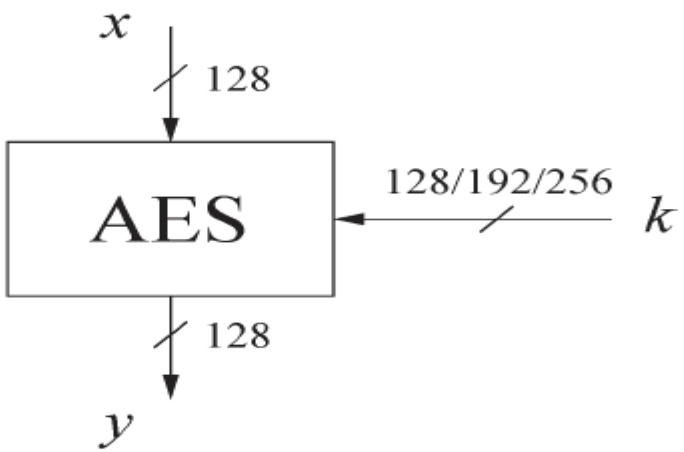
A Closer look at RijnDael



A Closer look at RijnDael (20 years later!)



AES: Overview



The number of rounds depends on the chosen key length:

Key length (bits)	Number of rounds
128	10
192	12
256	14

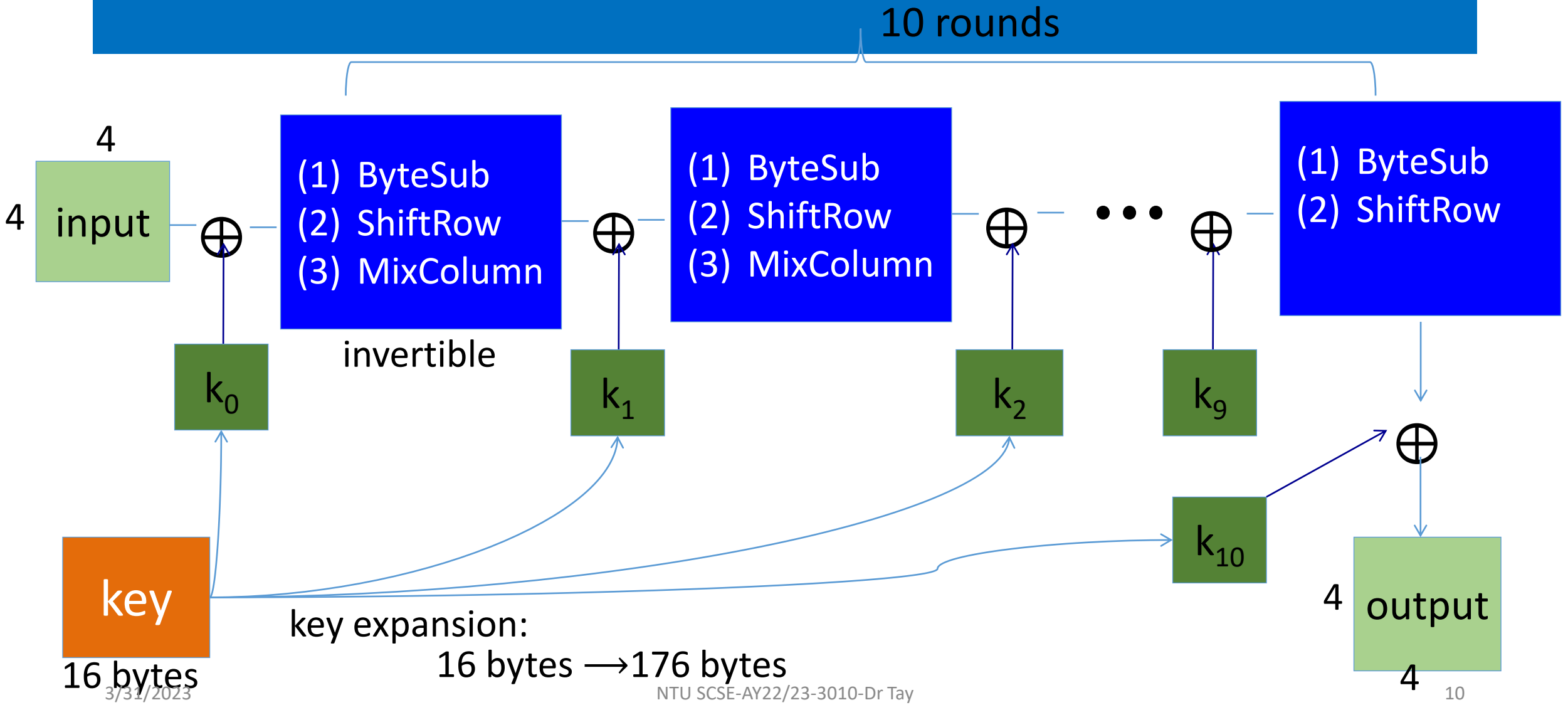
7/28

Chapter 4 of *Understanding Cryptography* by Christof Paar and Jan Pelzl

Fantastic AES animation (under 5 mins only!)

- https://www.youtube.com/watch?v=gP4PqVGudtg

AES-128 schematic



S Box

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
A	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
B	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
C	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
D	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
E	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
F	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

FIGURE 10.4: Rijndael S-box.

and $Y = 0b0101 = 0x5$, or 5. The value in row B and column 5 is 0xd5, which has binary representation 11010101.

Encrypting More Than 1 Block

- So far we only talk about encrypting 1 BLOCK, yes 1 BLOCK!
- Nowadays block size is typically 128-bit.
- Obviously message does not come in such nice block sizes!
- If message is not multiple of 128-bit, we introduce the notion of padding to our last block!

Modes of Encryption

- Now its time to talk about encrypting multiple blocks of fixed size, typically 128-bit long.
- There are several commonly used modes of encryption.
- Will talk about 3 of them
  - ECB (electronic codebook)
  - CBC (cipher block chaining)
  - CTR (counter mode)

• Mode : ECB

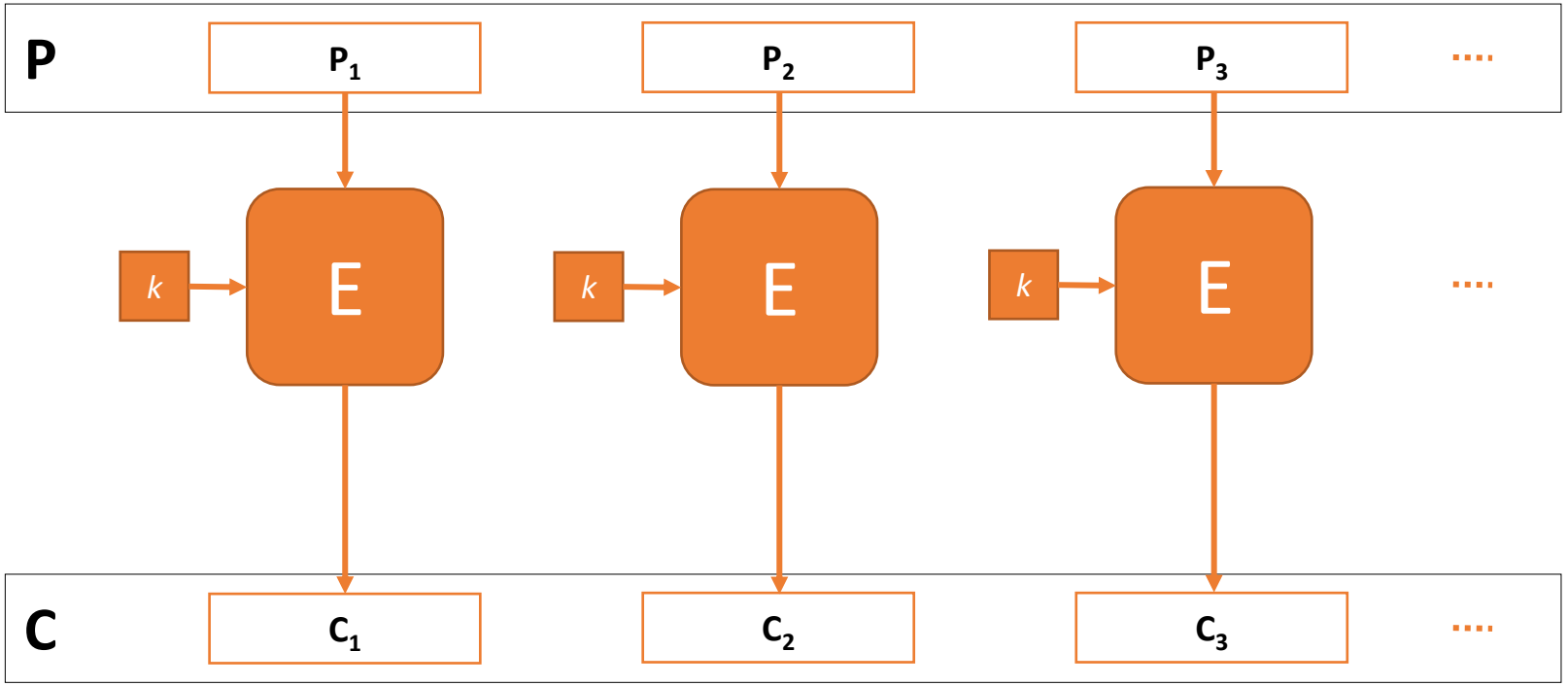
Every block of the plaintext is encrypted independently and identically, using same key *k*.

Encryption

$$C_i = E(k, P_i) = E_k(P_i)$$

• Electronic Code Book : Encryption

- Plaintext **P** considered as a sequence of blocks with size suitable for E_k.



- Parallel Encryption : Yes

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

• Mode : ECB

Every block of the plaintext is encrypted independently and identically, using same key *k*.

Encryption

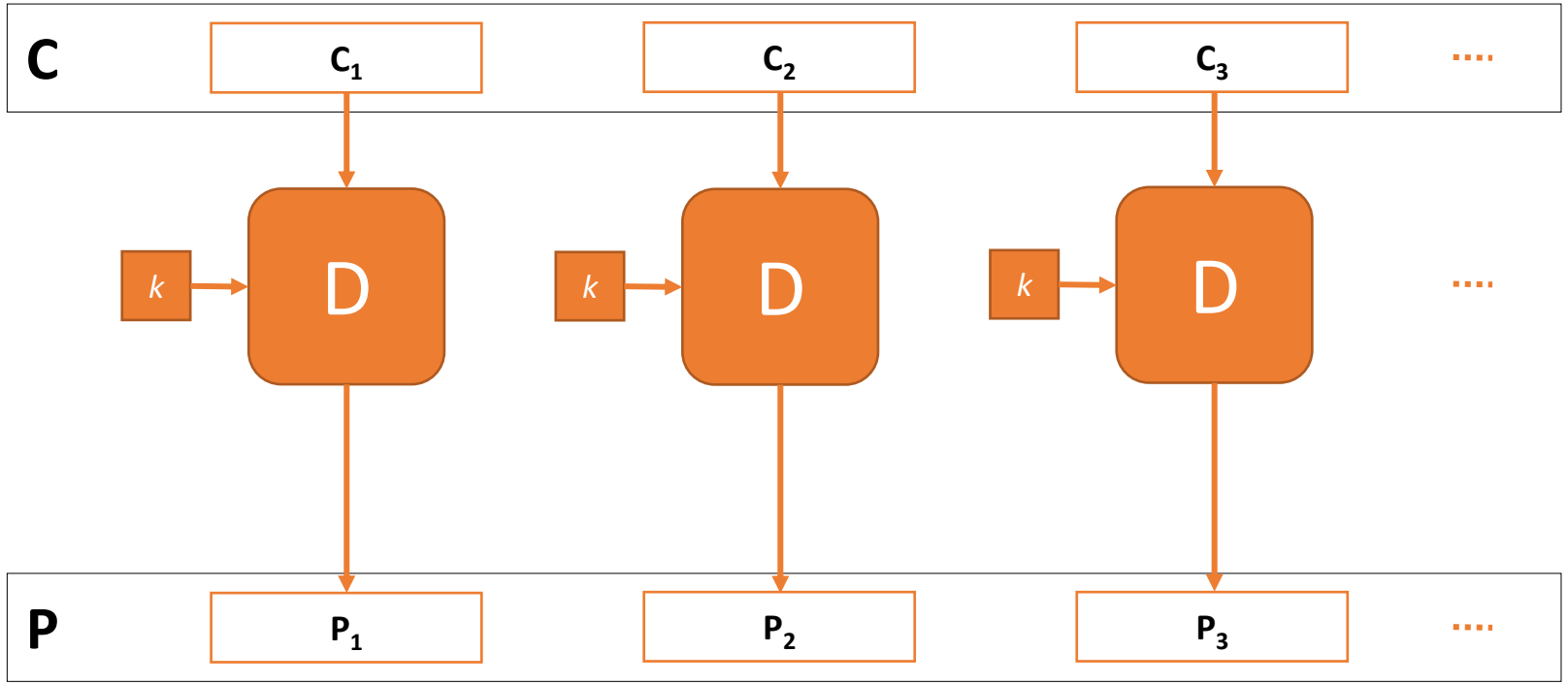
$$C_i = E(k, P_i) = E_k(P_i)$$

Decryption

$$P_i = D(k, C_i) = D_k(C_i)$$

• Electronic Code Book : Decryption

- Ciphertext **C** considered as a sequence of blocks with size suitable for $\mathbf{D}_k$.



- Parallel Encryption : Yes | Parallel Decryption : Yes | Random Read : Yes

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

• Mode : ECB

Every block of the plaintext is encrypted independently and identically, using same key *k*.

Encryption

$$C_i = E(k, P_i) = E_k(P_i)$$

Decryption

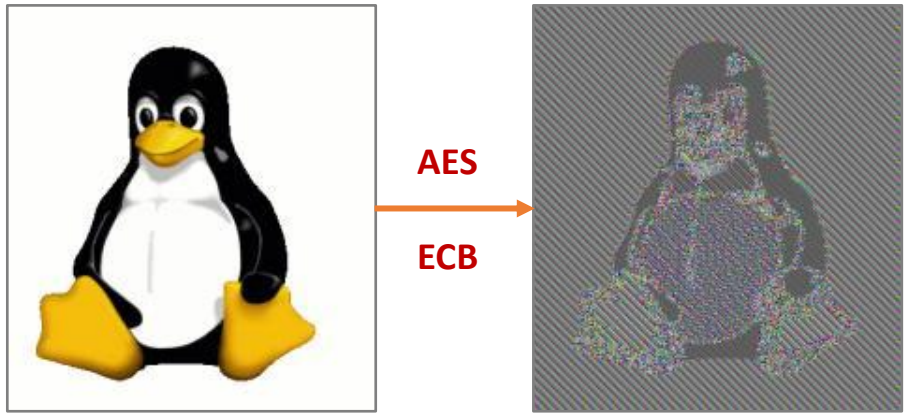
$$P_i = D(k, C_i) = D_k(C_i)$$

• Security and Efficiency Considerations

- **Efficiency**
- Parallel Encryption : Yes | Parallel Decryption : Yes | Random Read : Yes

- **Security**
- E_k is fixed function for fixed *k*, rendering the scheme a simple substitution.

- **Drawbacks of ECB**
  - Fixed "map" for symbols
  - Pattern(s) are preserved
  - Repetition will be visible
  - Frequency will be visible
  - **C** leaks **P**'s "information"



- ***Note : Any deterministic cipher used with a fixed key will behave similarly!***

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

• Mode : CBC

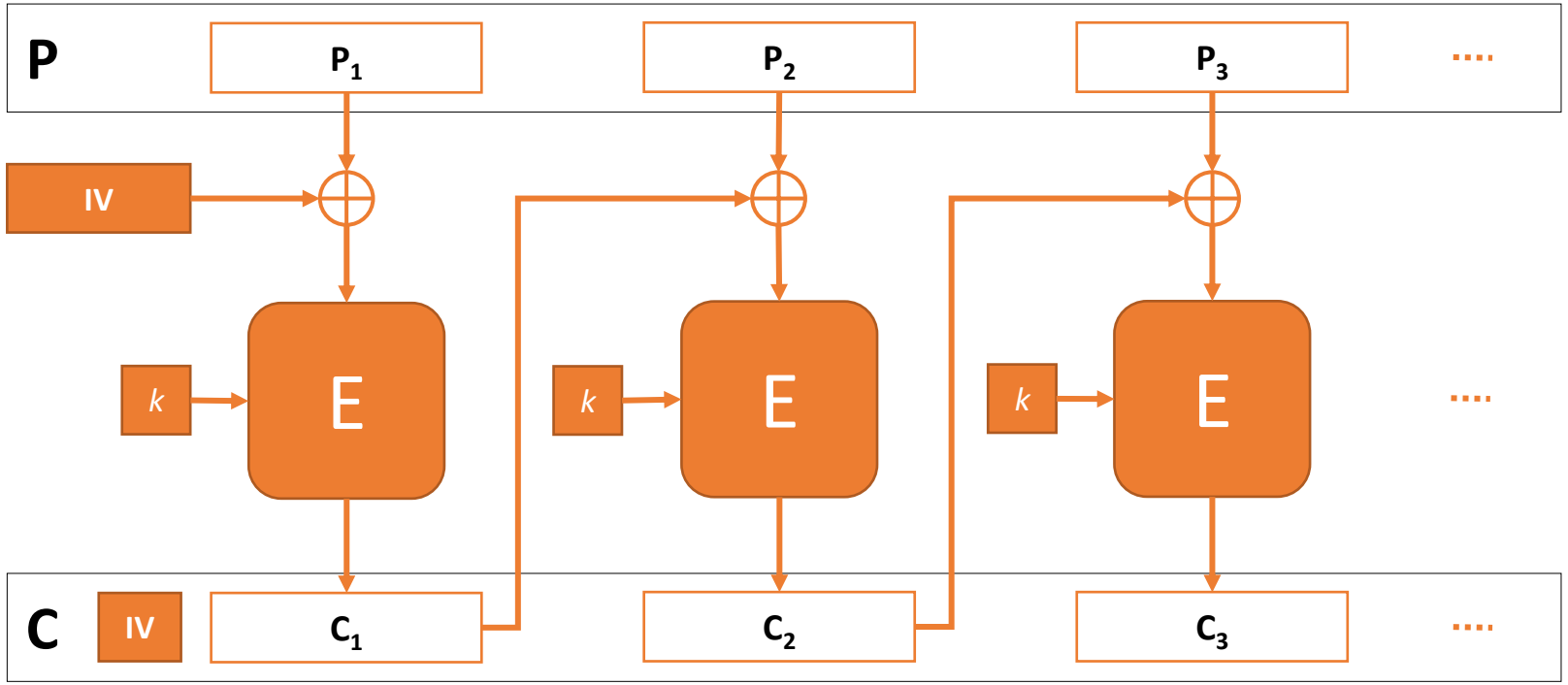
Each block of plaintext is XOR-ed with previous block of ciphertext before it is encrypted using E_k.

Encryption

$C_0 = IV \quad C_i = E_k(P_i \text{ XOR } C_{i-1})$

• Cipher Block Chaining : Encryption

- Plaintext **P** considered as a sequence of blocks with size suitable for $\mathbf{E}_k$.



- Parallel Encryption : No

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

• Mode : CBC

Each block of plaintext is XOR-ed with previous block of ciphertext before it is encrypted using E_k.

Encryption

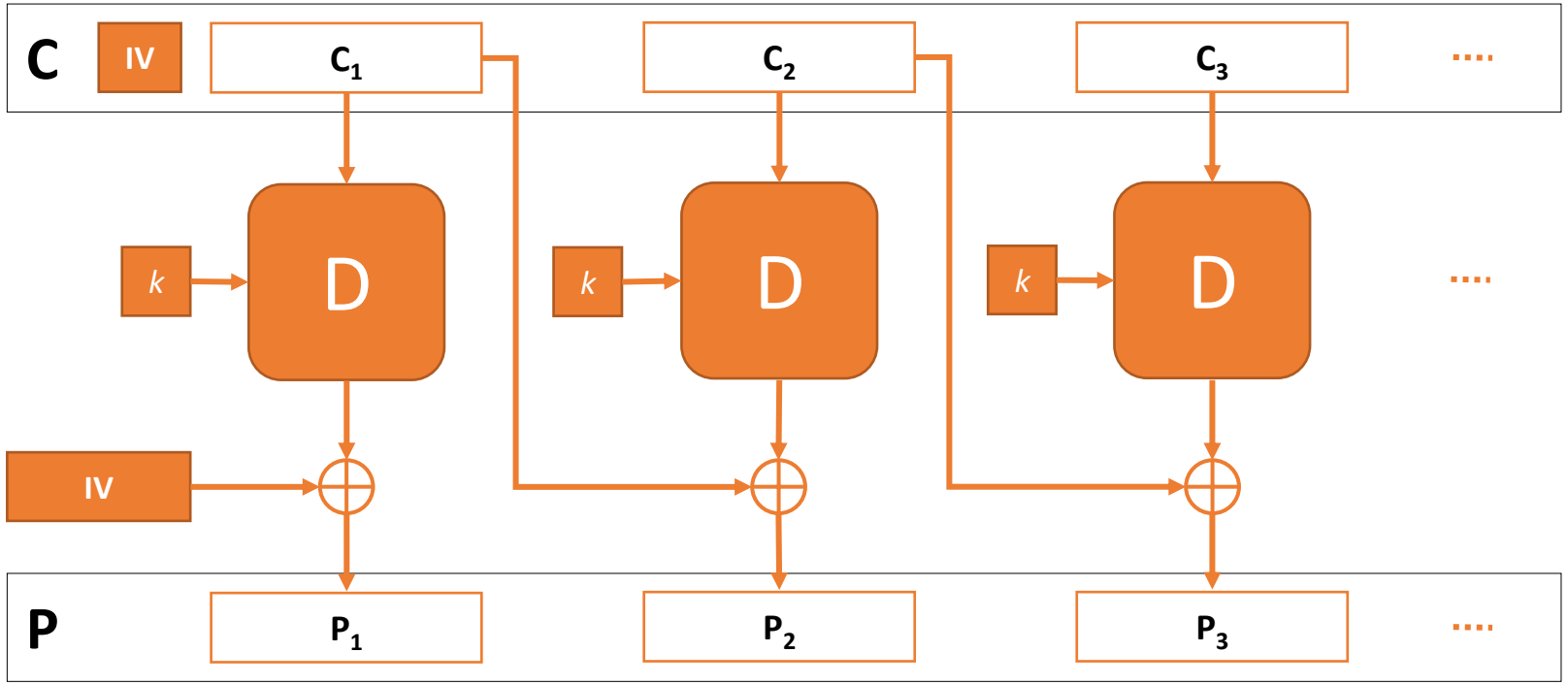
$C_0 = IV \quad C_i = E_k(P_i \text{ XOR } C_{i-1})$

Decryption

$C_0 = IV \quad P_i = D_k(C_i) \text{ XOR } C_{i-1}$

• Cipher Block Chaining : Decryption

- Ciphertext **C** considered as a sequence of blocks with size suitable for $\mathbf{D}_k$.



- Parallel Encryption : No | Parallel Decryption : Yes | Random Read : Yes

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

• Mode : CBC

Each block of plaintext is XOR-ed with previous block of ciphertext before it is encrypted using E_k.

Encryption

$C_0 = IV \quad C_i = E_k(P_i \text{ XOR } C_{i-1})$

Decryption

$C_0 = IV \quad P_i = D_k(C_i) \text{ XOR } C_{i-1}$

• Security and Efficiency Considerations

- **Efficiency**
- Parallel Encryption : No | Parallel Decryption : Yes | Random Read : Yes

- **Security**
- E_k is a fixed function for fixed *k*, but the input changes for each block.

  - Even if the key *k* remains fixed, only one *IV* can encrypt many blocks.
  - If the key *k* remains fixed, one may just change *IV* for a new plaintext.
  - The pair (*k, IV*) must not repeat for the lifetime of the mechanism.

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

• Mode Variations

Each block of plaintext is XOR-ed with previous block of ciphertext before it is encrypted using E_k.

Encryption

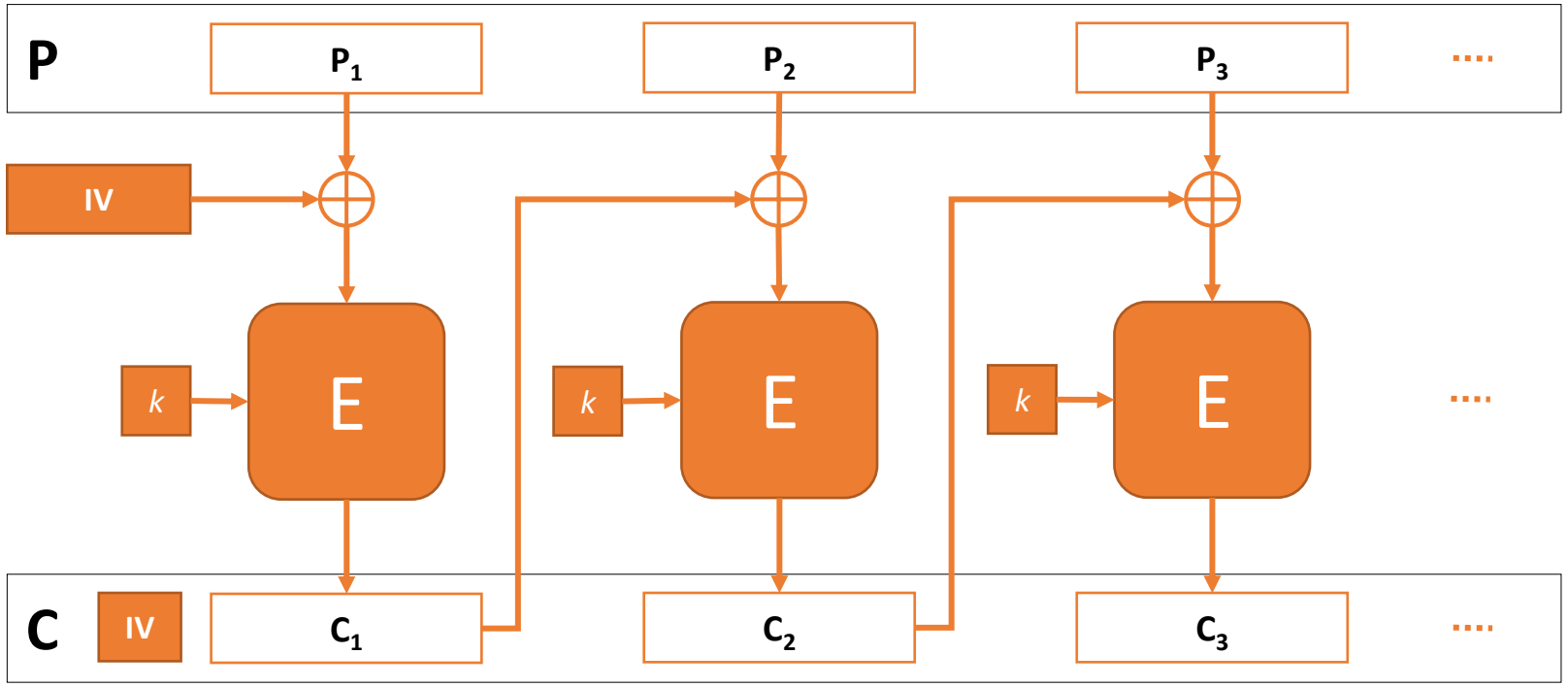
$C_0 = IV \quad C_i = E_k(P_i \text{ XOR } C_{i-1})$

Decryption

$C_0 = IV \quad P_i = D_k(C_i) \text{ XOR } C_{i-1}$

• CBC Mode : Options for IV

- Note : You must release IV as a part of the Ciphertext, if it is not known.
- **Random IV** : Choose IV randomly for each plaintext if the key *k* is fixed.



Counter Mode

- Lastly we talk about the Counter Mode
- It uses a block cipher as a stream cipher
- The key stream is computed in a blockwise fashion.
- The input to the block cipher is a counter which assumes a different value every time the block cipher computes a new key stream block.
- We have to be careful how to initialize the input to the block cipher. We must prevent using the same input value twice. Otherwise, if an attacker knows one of the two plaintexts that were encrypted with the same input, he can compute the key stream block and thus immediately decrypt the other ciphertext.

• Mode : CTR

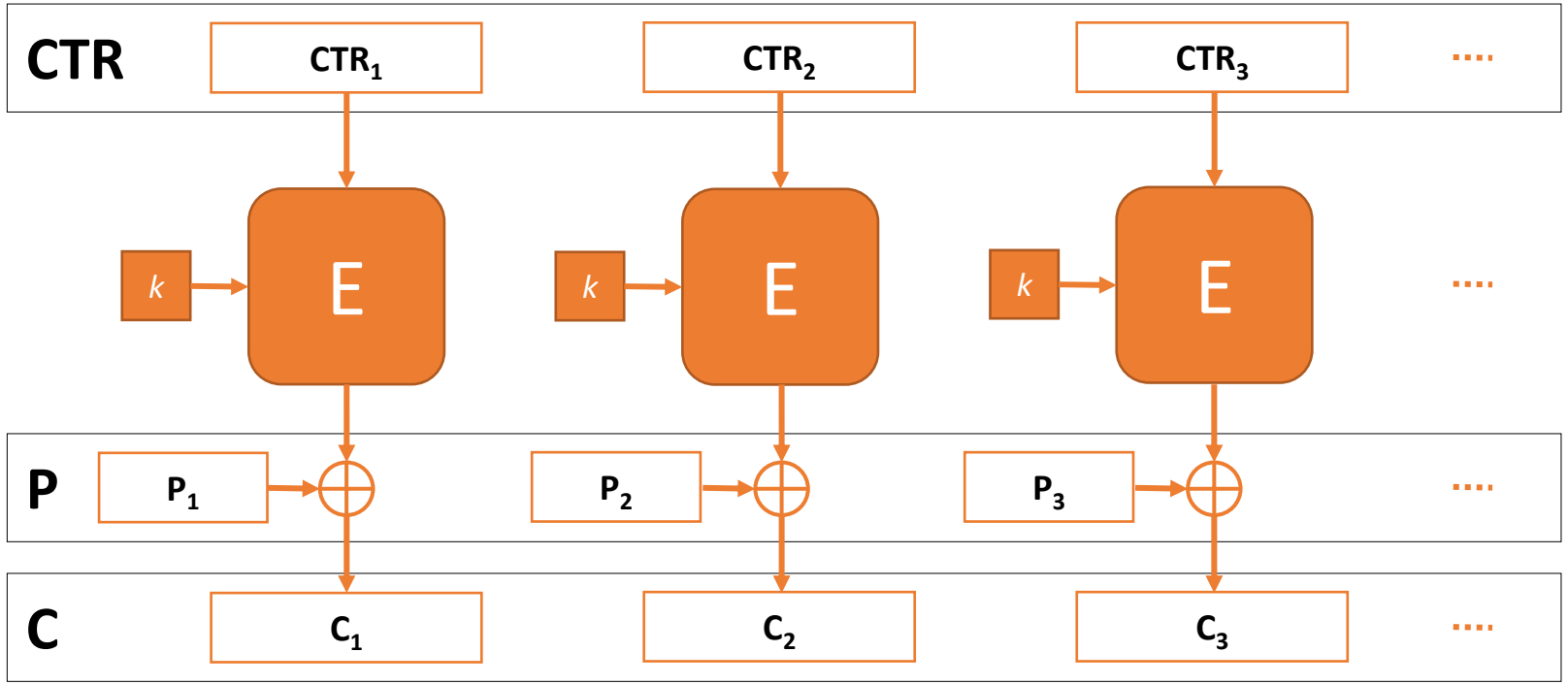
Plaintext is encrypted by XOR-ing with a stream generated using **E**, key *k* and a (nonce) counter CTR.

Encryption

$$C_i = P_i \text{ XOR } E_k(CTR_i)$$

• Counter Mode : Encryption

- Plaintext **P** considered as a sequence with $\mathbf{E}_k$ generating encryption pad.



- Parallel Encryption : Yes

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

• Mode : CTR

Plaintext is encrypted by XOR-ing with a stream generated using **E**, key *k* and a (nonce) counter CTR.

Encryption

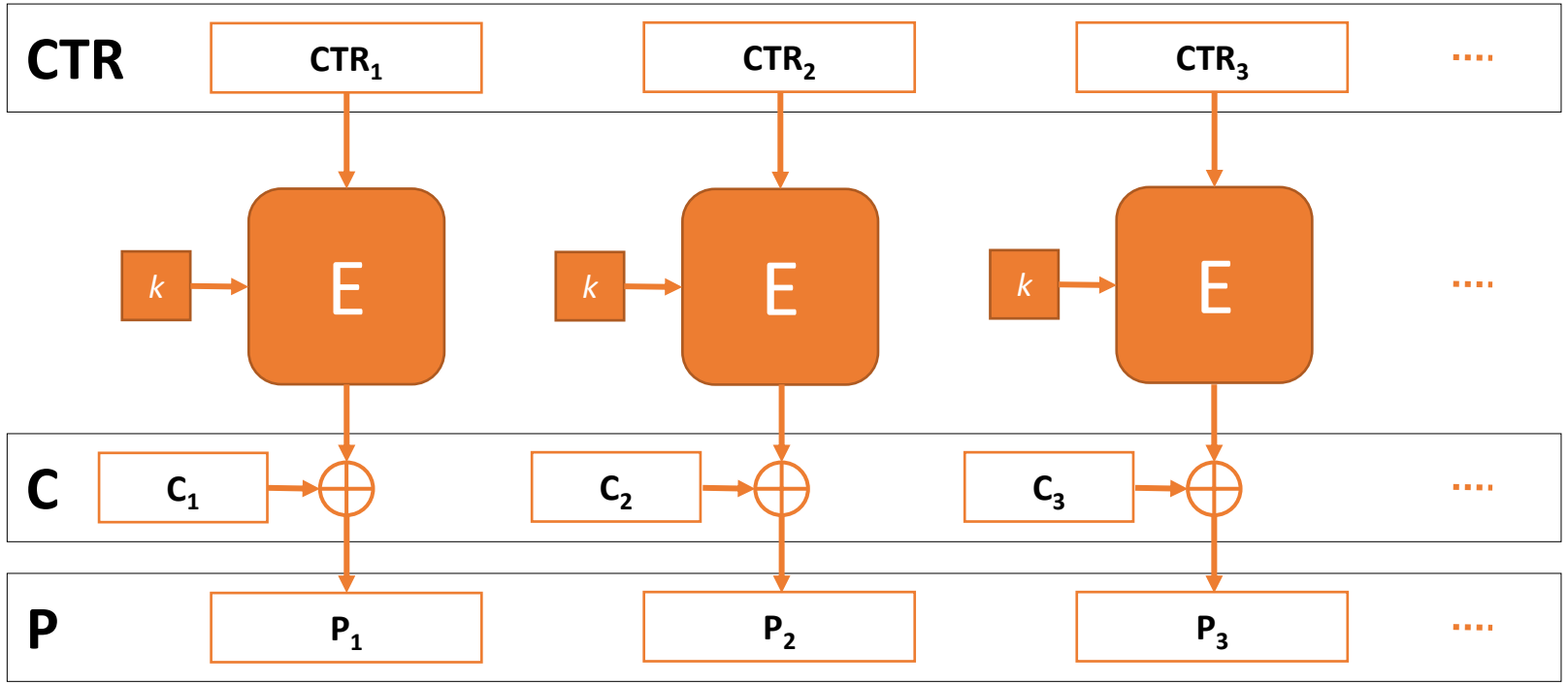
$$C_i = P_i \text{ XOR } E_k(CTR_i)$$

Decryption

$$P_i = C_i \text{ XOR } E_k(CTR_i)$$

• Counter Mode : Decryption

- Ciphertext **C** considered as a sequence with $\mathbf{E}_k$ generating decryption pad.



- Parallel Encryption : Yes | Parallel Decryption : Yes | Random Read : Yes

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

• Mode : CTR

Plaintext is encrypted by XOR-ing with a stream generated using **E**, key *k* and a (nonce) counter CTR.

Encryption

$$C_i = P_i \text{ XOR } E_k(CTR_i)$$

Decryption

$$P_i = C_i \text{ XOR } E_k(CTR_i)$$

• Security and Efficiency Considerations

- **Efficiency**
- Parallel Encryption : Yes | Parallel Decryption : Yes | Random Read : Yes

- **Security**
- E_k is fixed function for fixed *k*, but the input is different for each counter.

  - If the key *k* remains fixed, the counter must change for every block.
  - The counter must not repeat for any block encrypted with same key.
  - The pair (k, CTR_i) must not repeat for the lifetime of the mechanism.

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

• Mode Variations

Plaintext is encrypted by XOR-ing with a stream generated using **E**, key *k* and a (nonce) counter CTR.

Encryption

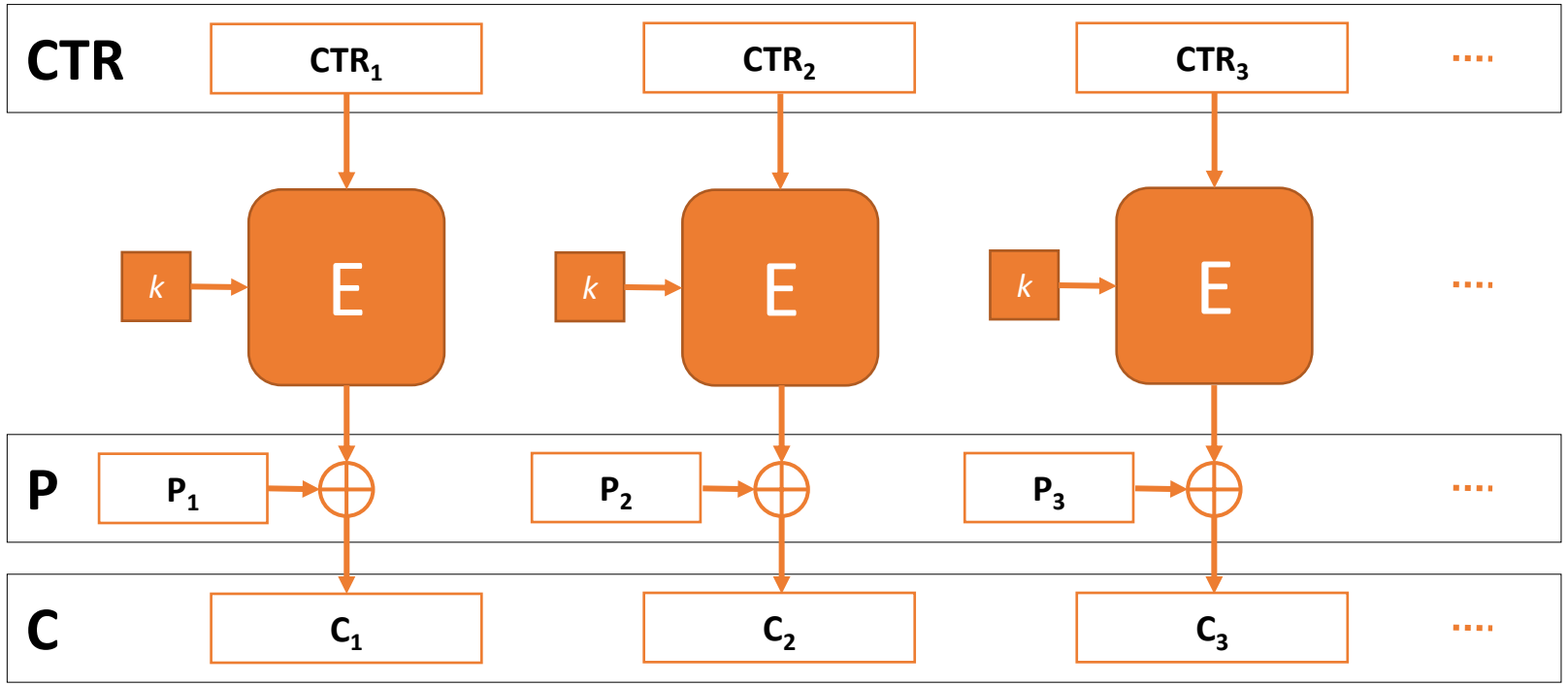
$$C_i = P_i \text{ XOR } E_k(CTR_i)$$

Decryption

$$P_i = C_i \text{ XOR } E_k(CTR_i)$$

• Counter Mode : Options for CTR

- **Deterministic Counter** : $CTR_1 = 0$, $CTR_2 = 1$, $CTR_3 = 2$, …
- **Random Counter (IV)** : $CTR_1 = IV$, $CTR_2 = IV + 1$, $CTR_3 = IV + 2$, …
- **Random Counter (Nonce)** : Choose IV = [*x*-bit Nonce || *y*-bit Counter]
- **Galois Counter (GCM)** : 96-bit Nonce fixed, 32-bit Counter increments



KEY MANAGEMENT

Impt Questions on KEYS

- We have not talked abt how to
- Generate keys
- Store keys
- Life cycle of keys
- How many people to be in charge of different keys
- Destroy keys etc etc

Impt Questions on KEYS

- Sometimes Eve doesn't have to break the algorithms.
- She doesn't have to rely on subtle flaws in the protocols.
- She can use their keys to read all of Alice's and Bob's message traffic without lifting a cryptanalytic finger.
- In the real world, key management is the hardest part of cryptography.
- Designing secure cryptographic algorithms and protocols isn't easy, but you can rely on a large body of academic research.

Impt Questions on KEYS

- Keeping the keys secret is much harder.
- Cryptanalysts often attack both symmetric and public-key cryptosystems through their key management.
- Why should Eve bother going through all the trouble of trying to break the cryptographic algorithm if she can recover the key because of sloppy key storage procedures?
- Why should she spend $10 million building a cryptanalysis machine if she can spend $1000 bribing a clerk?

Impt Questions on KEYS

- It's a whole lot easier to find flaws in people than it is to find them in cryptosystems.
- Alice and Bob must protect their key to the same degree as all the data it encrypts.
- If a key isn't changed regularly, this can be an enormous amount of data.
- Unfortunately, many commercial products simply proclaim "We use AES" and forget about everything else.
- Wont say more, but just want to highlight these key issues.