

Goals of security

CIA: three important aspects

Confidentiality (รักษาความลับ): การจำกัดการเข้าถึงทรัพย์สินแค่บุคคลที่ได้รับอนุญาต

Integrity (ความซื่อสัตย์): การ modified ทำได้เท่าที่ได้รับอนุญาต ภายในกลุ่มที่ได้รับอนุญาต

Availability (เข้าถึง): สามารถเข้าถึง assets ได้ตามต้องการ

Cryptography: ตอบโจทย์ C และ I

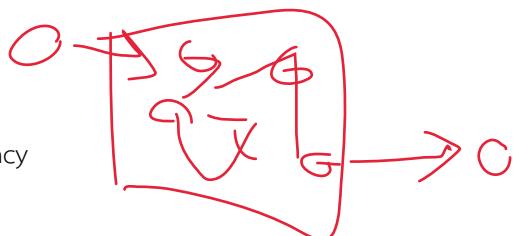
- C: Encryption การเข้ารหัสข้อมูลเป็นการจำกัดการเข้าถึงได้แค่ผู้ที่มีคีย์หรือสิทธิในการถอดรหัส
- I: Cryptographic Hash Function/Message Authentication Code

สิ่งที่ Cryptography provide เพิ่มเติม

Non-repudiation: การห้ามปฏิเสธความรับผิดชอบ มีการใช้ Digital Signature (PKI: public key infrastructure/public key และ private key)

Anonymity

- Anonymous Communication Ex. Tor (PKI)
- Anonymous Digital Cash Ex Crypto Currency



Zero-knowledge proof

- สามารถทำงานได้โดยที่ไม่มีใครรู้ว่าคนที่ทำคือใคร

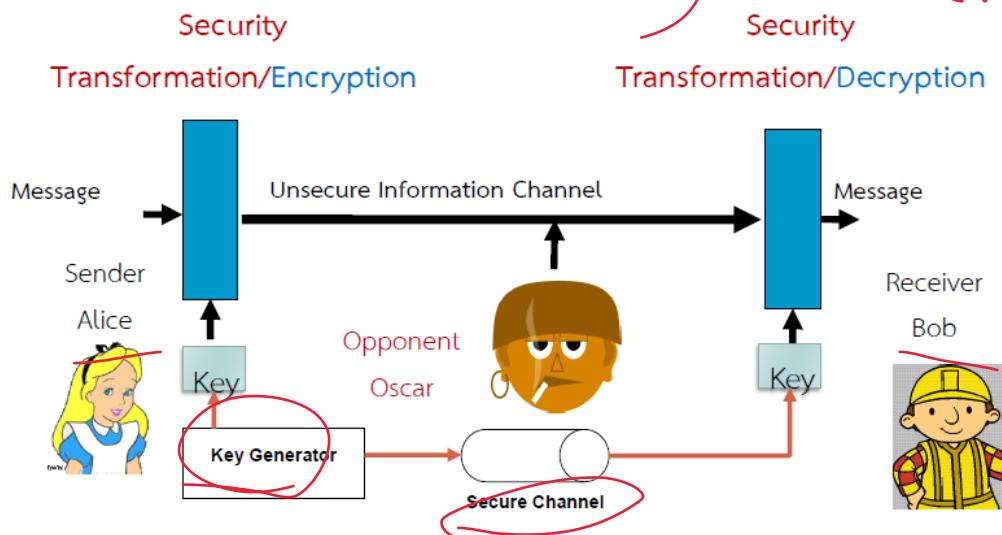
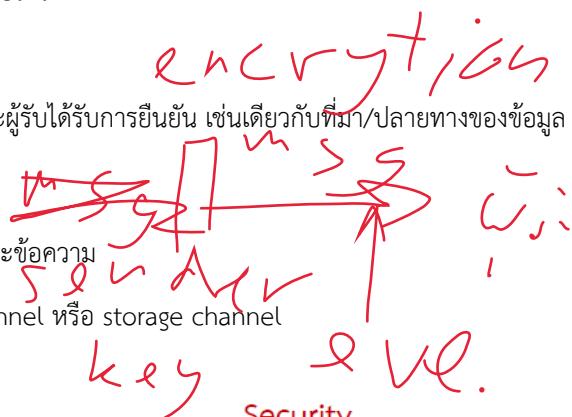
Authentication

- รับรองความถูกต้อง ข้อมูลประจำตัวผู้ส่งและผู้รับได้รับการยืนยัน เช่นเดียวกับที่มา/ปลายทางของข้อมูล

The basic secrecy channels

การสื่อสารประกอบด้วย ผู้ส่ง ผู้รับ ช่องทางสื่อสาร และข้อความ

ช่องทางการสื่อสาร เป็นได้ทั้ง communication channel หรือ storage channel



Encode ≠ Encryption, Decode ≠ Decryption

Encryption และ Decryption ต้องใช้ key

Key dependence

การ transformation แต่ละครั้งจะไม่เหมือนกัน เพราะคีย์ที่ใช้ = คีย์ต้องเป็นความลับ

บางครั้ง Encryption และ Decryption บางครั้งเรียกว่า Enciphering และ Deciphering

Symmetric Key Cryptography

คีย์ที่ใช้ encrypt, decrypt เป็นตัวเดียวกัน มีมานานแล้ว (classical cryptography)

บางที่เรียกว่า Secret key cryptography หรือ Private Key Cryptography

Asymmetric Key Cryptography

คีย์ที่ใช้ encrypt และ decrypt เป็นคนละตัวกัน

อีกชื่อคือ Public Key Cryptography (PKC)

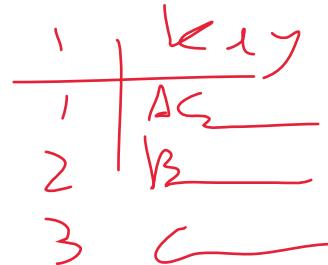
Kirchoff's Principle

ต้องเปิดเผยรายละเอียดของการทำ Encryption และ Decryption ยกเว้น คีย์

Key

Properties of keys

1. จำจ่าย
2. Key space มีขนาดใหญ่ (ป้องกัน exhaustive key search)
3. การลดจำนวนคีย์ใช้ indexed subset เพื่อลดขนาดของคีย์
 - a. Cipher algorithm = substitution, Key = index



Cipher

Monoalphabetic cipher

One-to-One mapping: 1 plaintext alphabet – 1 ciphertext alphabet

Brute force attack: Exhaustive key search เป็นไปได้ ตั้งนั้น ไม่ปลอดภัย

Substitution ciphers

Plaintext 1 ตัว ถูกแทนที่ด้วยตัวอักษรในเซตของ ciphertext ตัวต่อตัว

Additive ciphers

Translation ciphers: การเลื่อนโดยใช้ค่าคงที่ มี Key space = 26

$$Y = X \oplus Z$$

$C = P + Z \text{ mod } 26$ (addition modulo 26)

Multiplicative ciphers

การคูณ plaintext ด้วยค่าคงที่ มี Key space = 12

$$Y = X \otimes Z$$

$$C = P * Z \bmod 26 \text{ (multiplication modulo 26)}$$

ปัญหาคือ ไม่ใช่ทุกค่าที่นำมาใช้ได้ เพราะมี weak key ถึง 14 ตัว {2,4,6,8,10,12,13,14,16,18,20,22,24}

กลยุยเป็นว่าเหลือให้ใช้แค่ 12 ค่า = Exhaustive key search ง่าย

Affine ciphers

ใช้ additive และ multiplicative ร่วมกัน Key space = $12 * 26 = 312$

$$Y = A \otimes X \oplus Z$$

$$X,Y,Z [0-25] \mid A = \text{เลขคี่ } 1-25$$

Key phrase-based ciphers

ใช้คีย์ โดยกำหนดตำแหน่งแรกในตาราง ซึ่งเพิ่มขนาดของคีย์ และทำให้จำคีย์ง่ายขึ้น

- Phase: **bubble bath**
- Starting letter: **e**

a	b	c	d	e	f	g	h	i
W	X	Y	Z	B	U	L	E	A
j	k	l	m	n	o	p	q	r
T	H	C	D	F	G	I	J	K
s	t	u	v	w	x	y	z	
M	N	O	P	Q	R	S	V	

Caesar cipher

การเลื่อนตัวอักษรไปทางขวา 3 ตัวอักษร

Generalized Caesar: การเลื่อนเป็นไปได้ตั้งแต่ 1 – 25 = Shift cipher

Flatten histogram

เมื่อข้อความมีการใช้ตัวอักษรไม่เท่ากัน เช่น มีการใช้สารมากกว่าพยัญชนะ อาจทำให้เห็นแนวโน้มของจำนวนตัวอักษร ดังนั้น ควรซ่อนแนวโน้มดังกล่าวด้วยการทำให้ความถี่ของทุกตัวอักษรมีค่าเท่ากัน มีวิธีทั้งหมด 2 วิธี คือ homophone และ polyalphabetic

a 72 52 98
b 2 5 70

Homophone

- plaintext map เข้ากับชุดของ ciphertext (1 plaintext จะเป็น ciphertext ได้หลายตัว)
- จำนวนตัวอักษรในแต่ละชุดจะแปลงสัดส่วนตามความถี่ของแต่ละภาษา จะแกะยาก
- ปัญหาคือ ถ้าจำนวนข้อความเยอะพอ ยังสามารถวินิจฉัยได้
- อีกปัญหาคือ $\text{len}(C) > \text{len}(P)$

Polyalphabetic

- รวมการแทนที่หลายแบบเข้าด้วยกัน
- Ciphertext หนึ่งตัว ใช้แทน plaintext ได้มากกว่า 1 ตัว (แต่ต้องย้อนกลับได้นะ)

Vigenère cipher

ใช้การแทนที่หลายตัวอักษรรวมกัน (key) : frequency of ciphertext can be flattened by using multiple substitution alphabets

- คีย์ที่มี keyword ที่ใช้

การวินิจฉัย Vigenère cipher

1. หา period (ความยาวของ key phrase)
2. แยก Ciphertext ออกเป็น component ตาม single substitution alphabet
3. แต่ละ component ใช้วิธีการวินิจฉัยเดียวกัน monoalphabetic cipher และใช้ cross component

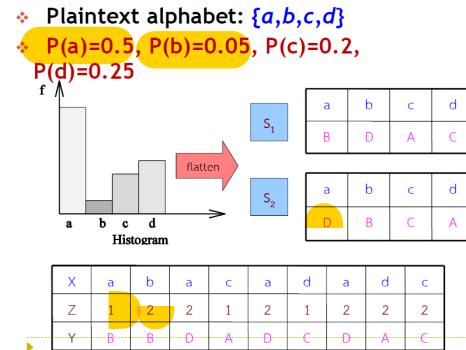
การหา period : The Kasiski method

Two identical plaintexts จะเข้ารหัสได้ Ciphertext ที่เหมือนกัน

- หา ciphertext ที่มี repeated segments
- คำนวณ ระยะห่างระหว่าง repeated segments

The index of coincidence

- การใช้ความยอดของข้อความ ซึ่งเป็นภาษาอังกฤษผ่านสูตร $IC(x) = \frac{\sum_{\lambda=A}^Z f_\lambda(f_\lambda - 1)}{n(n-1)}$
 - f_λ = ความถี่ของแต่ละตัวอักษร
- $IC(x)$ เป็นค่าประมาณของความน่าจะเป็นที่ค่าที่สูงส่องค่าเป็นค่าเดียวกัน
- ใช้ระบุความไม่สมมาตรของ Histogram ได้
- ค่า IC ในตัวข้อความสัมภาษณ์ภาษาอังกฤษคือ 0.036 หรือ 1/26
- ค่า IC ในข้อความภาษาอังกฤษ ความน่าจะเป็นคือ $0.065 \approx \sum_{\lambda=A}^Z p(\lambda)^2$
- สามารถนำมาใช้ในการหาความยาวของคีย์ได้
- Monoalphabetic cipher มีค่า IC plaintext และ ciphertext เท่ากัน
- Ciphertext ของข้อความภาษาอังกฤษที่เข้ารหัสด้วย Monoalphabetic มีค่าเท่ากับ 0.065 (เท่ากับ plaintext)



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

Playfair cipher

เป็นอีกหนึ่งลักษณะของ block cipher โดยจะมีคีย์เป็น matrix 5×5 แต่ละช่องคือตัวอักษร ยกเว้นตัว J

กฎในการเข้ารหัสมี 4 ข้อ

- ถ้าตัวอักษร x1,x2 อยู่ในแถวเดียวกัน ให้ shift right ไป 1 ช่อง ทั้งสองตัว
- ถ้าตัวอักษร x1,x2 อยู่ในคอลัมน์เดียวกัน ให้เลื่อนลงทั้งสองตัวแบบ cyclic
- ถ้าเป็นตัวอักษรเดียวกัน ให้แทนด้วยตัวอักษรไปต่อลงมา
- ถ้า x1,x2 อยู่คุณลักษณะและคอลัมน์ : ถ้า x1 เป็นหลัก ให้เลือกตัวที่อยู่ในแถวเดียวกับ x1 และ ตรงกับคอลัมน์ของ x2 และถ้า x2 เป็นหลัก ให้เลือกตัวอยู่ที่อยู่ในแถวเดียวกับ x2 และ คอลัมน์ตรงกับ x1 (ลักษณะเป็นสามเหลี่ยม เชื่อมกัน)

The diagram shows a 5x5 matrix and a 5x5 grid of ciphertext. The matrix rows are labeled r1 to r5 and columns c1 to c5. The matrix contains the following letters:

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

Arrows indicate shifts and substitutions. The grid below shows the resulting ciphertext:

TH	IS	RI	AT	RI	AL
MP	BH		PN	HO	

Cryptanalysis

Statistical cryptanalysis

- ใช้สถิติวิเคราะห์ความถี่ของตัวอักษร
- ดังนั้นระบบจะสับ ต้องทำให้คุณสมบัติทางสถิติไม่แกะได้ = statistically indistinguishable
- Flatten technique

Note on statistical methods

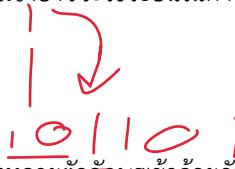
- ใช้ได้เมื่อ ciphertext มีความยาวที่เพียงพอ
- คีย์ที่ยาวจะทำให้ IC และ Kasiski methods ยากขึ้น

Practical Security

Shannon Theorem

- Perfect secrecy: คีย์ยาวเท่า message: ไม่ practical
- Unicity Distance: จำนวนตัวอักษรที่น้อยที่สุดของ ciphertext เพื่อใช้ในการหากคีย์ได้
 - $N_0 = \frac{\log_2 E}{d}$: d = redundancy ของภาษา, E จำนวนคีย์
 - $d = R - r$ bits : R = จำนวนบิตที่น้อยที่สุดสำหรับแต่ละตัวอักษร, r จำนวนบิตเฉลี่ย
 - ภาษาอังกฤษมี R ≈ 4.7, r ≈ 1 – 1.5, d ≈ 3.2 – 3.7 bits
- หมายความว่าใช้ Unicity วัดระดับความปลอดภัย
 - การเพิ่มความปลอดภัย = ต้องเพิ่มขนาดของ key space

- คนโจมตีมี resource ข้อมูล และเวลาที่จำกัด ดังนั้นเราอาจจะใช้วิธีอื่นในการโจมตี เช่น
 - Social attack: เพื่อเอารหัส
 - Plaintext/Ciphertext pair attacks
- Shannon's principle
 - Diffusion: เอ้าแพร่ คือ ใช้การรวมกันของหลายตัวเข้าด้วยกัน
 - Confusion: สร้างความสับสน คือ ทำให้เกิดความซับซ้อนของ P, C และ K
 - แปลว่าทำให้การเข้ารหัสกล้ายเป็น nonlinear **complex function**



The Avalanche effect เดี๋ดดอกไม่สะเทือนทั้งจักรวาล น้ำผึ้งหยดเดียว

- การเปลี่ยนแปลงเล็กน้อยใน P หรือ K ส่งผลให้เกิดการเปลี่ยนแปลงใน Ciphertext
 - เช่น เปลี่ยน P 1 บิต ทำให้ ciphertext เปลี่ยนเกินครึ่งหนึ่ง

Block Cipher

Modern block cipher

สนใจที่ bit

- มีหลักการดังนี้
 - เข้ารหัส n-bit block plaintext
 - ถอดรหัส n-bit block ciphertext
 - ชีบลีย์ที่ใช้เข้ารหัสและถอดรหัสจะเป็น ตัวเดียวกัน และ มีขนาด k บิต (Secret key)
- Decryption algorithm ต้องเป็น **Inverse** ของ encryption algorithm
- ถ้าแบ่ง block แล้วต้องเติมจำนวนบิตให้เต็ม block = padding
- Parameters ที่สำคัญ คือ
 - Block length: ขนาดของ Block
 - Key size: ความยาวของคีย์
- การทำงานเหมือนกับ Substitution cipher หรือ transposition cipher -> ควรจะเป็น Substitution cipher เพื่อไม่ให้ถูก attack ได้ง่าย

Example Scenario

อีฟดักฟัง ได้ Ciphertext ประกอบด้วย bit ตั้งแต่ 1 ถึง 10 bits จาก block cipher ที่ขนาดเท่ากับ 24 bits จะต้องทดลองทั้งหมดกี่ครั้ง

กรณีที่ 1 substitution ไม่รู้ว่า ciphertext อยู่ใน plaintext กี่ตัว

อีฟจะไม่รู้เลยว่า plaintext มีกี่บิต เพราะสิ่งที่ได้รับคือ sub cipher

- เพราะฉะนั้นต้องลองทุก plaintext ที่เป็นไปได้ = 2^{24} เพื่อจะดูว่าบล็อกไหนที่เหมาะสมที่สุด

กรณีที่ 2 transposition เป็นแค่การสลับที่

อีฟรู้ว่าใน plaintext ต้องมีบิตที่ค่าเท่ากันแน่ๆ คือ 10 บิต

- ต้องทำ exhaustive search attack เช่น บล็อกที่มีจำนวนบิตที่มีค่าเป็น 1 อยู่ 10 บิต

คุณสมบัติของ Modern block cipher

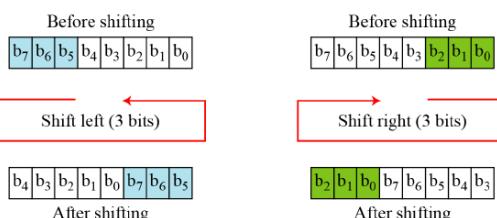
- Key substitution ciphers
 - Transposition units: P-box
 - Substitution units: S-box
 - Other Units เช่น XOR, complement, inverse, Circular Shift, Swap, Split & combine
- ชีวิตรอยalty นำไปสู่ Diffusion และ Confusion

Exclusive or (XOR)

คุณสมบัติ

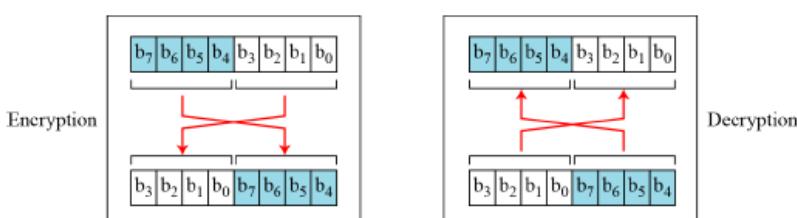
1. สมบัติปิด: ผลลัพธ์ของการ XOR บิต ต้องได้ออกมาเป็นบิต
2. การจัดหมู่
3. การเปลี่ยนกลุ่ม
4. เอกลักษณ์: $x \oplus 0^n = x$
5. Inverse: ตัวเอง $x \oplus x = 0^n$ = Self invertible = involution function

Circular shift

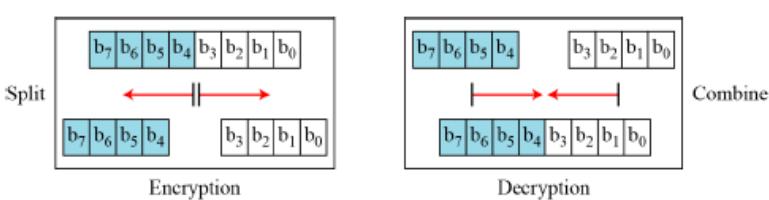


Swap

ให้ลำดับการสลับที่คือ $k = n/2$

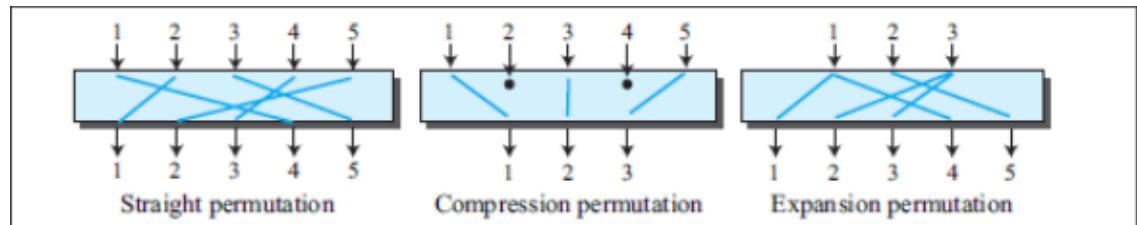


Split & Combine



P-box

- เป็น Keyless transposition
- เป็นแค่การสลับบิตไปมาเฉย ๆ
- มี 3 แบบ



- Straight P-box: เข้าเท่าไหร่ ออกเท่าเดิม
- Compression P-box: ออกมาน้อยกว่าเดิม
- Expansion P-box: ออกมากกว่าเดิม
- มีแค่ Straight ที่เป็นสามารถย้อนกลับได้ (Invertible) แปลว่าใช้ได้แค่ straight P-box สำหรับ encrypt และ decrypt

S-box

- ความสัมพันธ์ระหว่าง input และ output สามารถเขียนเป็นฟังก์ชันได้
- อาจมีหรือไม่มี invertible
 - Invertible เมื่อจำนวนบิตของ input และ output เท่ากัน

Linear S-box

- เป็นฟังก์ชัน ซึ่งสามารถเขียนออกมาในรูปของ matrix ได้

Nonlinear S-box

- ไม่สามารถเขียน input และ output ให้เป็นความสัมพันธ์แบบ linear ได้
- ยากต่อการ attack

ตัวอย่างเพิ่มเติม

- ▶ ตัวอย่างนี้ เป็น S-box แบบ 3×2
 - ▶ โดยมีตัวอย่างดังนี้ ถ้า input เป็น 010 output จะเป็น 01
 - ▶ และถ้า input เป็น 101 output 00.

		Leftmost bit					
		00	01	10	11		
Rightmost bits	0	00	10	01	11		
	1	10	00	11	01		
		Output bits					

Product Cipher

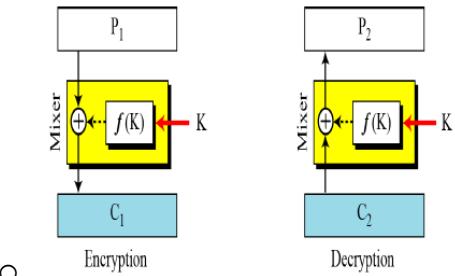
Class of Product ciphers

- Feistel Structure
 - สามารถใช้โครงสร้างเดียวกันในการ encrypt และ decrypt ได้
 - มีสามแบบคือ self-invertible, invertible, non-invertible
- Non-Feistel Structure
 - ไม่สามารถใช้โครงสร้างเดียวกันในการ encrypt และ decrypt
 - ใช้ Invertible component

Evolution of Feistel

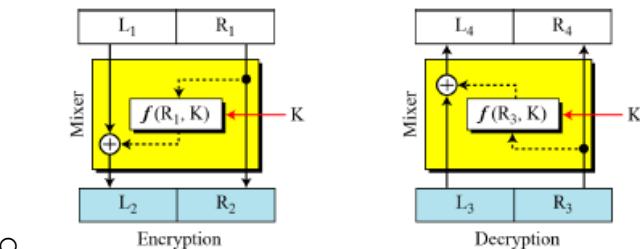
Feistel Encryption

- แรกเริ่มจากการเอา plaintext XOR key : [key มาจากฟังก์ชันของคีย์]



- ง่ายเกินไป ดูเป็น linear

- แบ่ง plaintext เป็นช้ายขวา → ฝั่งขวาเข้าฟังก์ชันของคีย์ → ฝั่งซ้าย XOR กับคีย์ที่ได้, ฝั่งขวาเหมือนเดิม



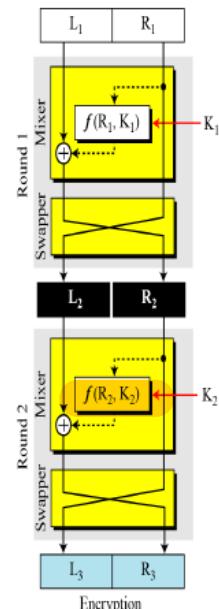
- ปัญหาเกิดขึ้นที่ฝั่งขวา เพราะไม่มีการเปลี่ยนแปลงขั้นเลย

- เพิ่มการ swap หลังจาก XOR กับคีย์แล้ว

- จากเดิมที่จะดึงข้างขวาลงมาเลยเปลี่ยนเป็น
 - สลับช้ายกับขวา ก่อนที่จะเริ่มรอบถัดไป
- เพื่อให้เกิด Diffusion และ Confusion จึงทำหลาย ๆ รอบ
 - เป็น Iterated Cipher

สรุป: Feistel cipher เป็นไซเฟอร์ชนิดหนึ่ง ซึ่งมีการเข้ารหัสดังนี้

- แบ่ง plaintext เป็นสองฝั่ง
- ในแต่ละรอบจะมีการคำนวณดังนี้
 - $L_{i+1} = R_i$
 - $R_{i+1} = L_i \oplus f(R_i, K_i)$
- เมื่อทำการบุกรุบแล้วจะได้ $Ciphertext = (L_{n+1}, R_{n+1})$
- ความปลอดภัยขึ้นอยู่กับฟังก์ชันที่ใช้
- การ Decrypt มีส่วนที่ต้องทำเพิ่มคือ สลับฝั่งของ Ciphertext และใช้คีย์จากตัวสุดท้ายวนไปตัวแรก



Details

- เป็นการรวมกันของ Substitution, permutation และคุณสมบัติอื่น ๆ
- นำเสนอโดย Shannon
 - เพื่อให้ Block cipher มีพื้นที่คุณสมบัติ Diffusion และ Confusion
 - Diffusion: ช่องความสัมพันธ์ของ Ciphertext กับ Plaintext
 - Confusion: ช่องความสัมพันธ์ของ Ciphertext กับ Key
 - ซึ่งเกิดจากการเอา product cipher มาทำงานหลาย ๆ รอบ

การเพิ่ม diffusion, confusion

- การเพิ่ม S-box ทำให้ ciphertext ดูเหมือน random word มา กว่าเดิม เพราะ ความสัมพันธ์ C และ P ดูซ่อนมากขึ้น ทำให้เกิด Diffusion มากขึ้น
- การเพิ่มจำนวนรอบจะช่วยซ่อนความสัมพันธ์ระหว่าง C และ K ทำให้เกิด Confusion มากขึ้น

Attack on Modern block cipher

สามารถใช้ attack แบบเดียวกับ traditional cipher ได้ ซึ่งในปัจจุบันต้องป้องกันการโจมตีทาง Brute force และ statistical attack ได้ ทำให้เกิดการโจมตีใหม่เอ้าไว้ใช้กับ modern block cipher โดยเฉพาะ

1. Differential Cryptanalysis

a. Chosen plaintext attack

b. ใช้การหาความสัมพันธ์ระหว่าง P และ C ซึ่งจะทำให้สามารถพิสูจน์ได้ว่า

$$C_1 \oplus C_2 = P_1 \oplus P_2$$

$$C_1 = P_1 \oplus K, C_2 = P_2 \oplus K \rightarrow C_1 \oplus C_2 = P_1 \oplus K \oplus P_2 \oplus K$$

X	000	001	010	011	100	101	110	111					
C	11	00	10	10	01	00	11	00					

S-box table

$$C_1 \oplus C_2 = P_1 \oplus P_2$$

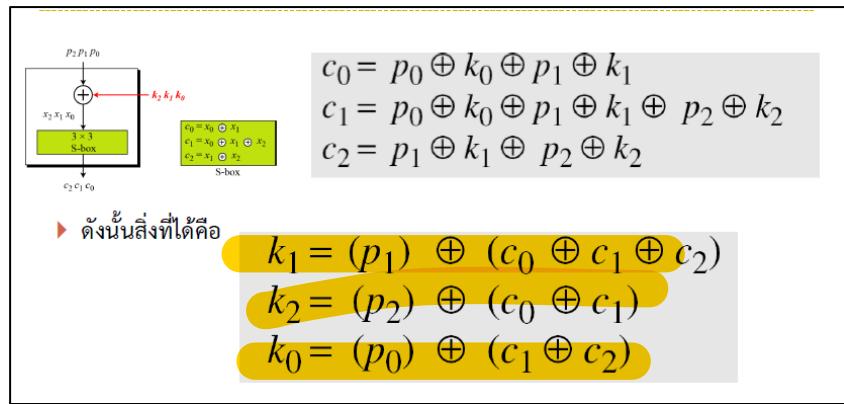
ΔP	ΔC	ΔP	ΔC	ΔP	ΔC	ΔP	ΔC	ΔP	ΔC	ΔP	ΔC	ΔP	ΔC	
P_1 / P_2	000	001	010	011	100	101	110	111						
000	00	001	11	010	01	011	10	100	10	101	11			11
001	001	11	000	00	011	00	010	10	101	01	100	01		00
010	010	01	011	01	000	00	001	00	110	11	111	10		10
011	011	01	010	10	001	00	000	00	111	11	100			10
100	100	10	101	01	110	11	111	11	000	00	001			01
101	101	11	100	00	111	10	110	10	001	01	000			00
110	110	00	111	11	100	01	101	01	010	10	011			11
111	111	11	110	00	101	10	100	10	011	01	010			00

ดังนั้นการเพิ่ม S-box จะทำให้หาความสัมพันธ์ของ C กับ P ยากขึ้น

2. Linear Cryptanalysis

a. Known plaintext attack

Block Cipher Attack	$c_0 = p_0 \oplus k_0 \oplus p_1 \oplus k_1$
Linear cryptanalysis	$c_1 = p_0 \oplus k_0 \oplus p_1 \oplus k_1 \oplus p_2 \oplus k_2$
	$c_2 = p_1 \oplus k_1 \oplus p_2 \oplus k_2$
► 1. หา p_0 จากสมการ 1 $\Rightarrow p_0 = c_0 \oplus k_0 \oplus p_1 \oplus k_1$	
► 2. แทน p_0 ใน 2	
	$c_1 = c_0 \oplus k_0 \oplus p_1 \oplus k_1 \oplus k_0 \oplus p_1 \oplus k_1 \oplus p_2 \oplus k_2$
	$c_1 = c_0 \oplus p_2 \oplus k_2 \Rightarrow k_2 = p_2 \oplus (c_0 \oplus c_1)$
► 3. หา p_1 จากสมการ 3 $\Rightarrow p_1 = c_2 \oplus k_1 \oplus p_2 \oplus k_2$	
► 4. แทน p_1 ใน 2	
	$c_1 = p_0 \oplus k_0 \oplus c_2 \oplus k_1 \oplus p_2 \oplus k_2 \oplus k_1 \oplus p_2 \oplus k_2$
	$c_1 = p_0 \oplus k_0 \oplus c_2 \Rightarrow k_0 = p_0 \oplus (c_1 \oplus c_2)$
► 5. แทนค่า k_0 และ k_1 ใน 2	
	$c_1 = p_0 \oplus p_0 \oplus (c_1 \oplus c_2) \oplus p_1 \oplus k_1 \oplus p_2 \oplus p_2 \oplus (c_0 \oplus c_1)$
	$c_1 = p_1 \oplus (c_0 \oplus c_2) \oplus k_1 \Rightarrow k_1 = p_1 \oplus (c_0 \oplus c_1 \oplus c_2)$



สรุปคือ มีข้อมูล plaintext, ciphertext จากนั้นนำมาหาค่าของคีย์ เป็นการดูคู่ของ Plaintext และ Ciphertext

ซึ่งการโจมตีแบบนี้เป็นสาเหตุให้ครอคแบบโครงสร้าง block cipher ให้มีคุณสมบัติแบบ linear

กรณีที่ไม่สามารถเลือกใช้ Plaintext ได้

- คุณค่าของ P ที่มีอยู่
- คุณค่าของ P นั้นว่าแต่ละตัวคืออะไร และหากความสัมพันธ์ของแต่ละบิตเพื่อเอาไปทำนายว่าควรจะเป็นอะไรต่อไป

DES

เป็น block cipher

input block: 64 bits

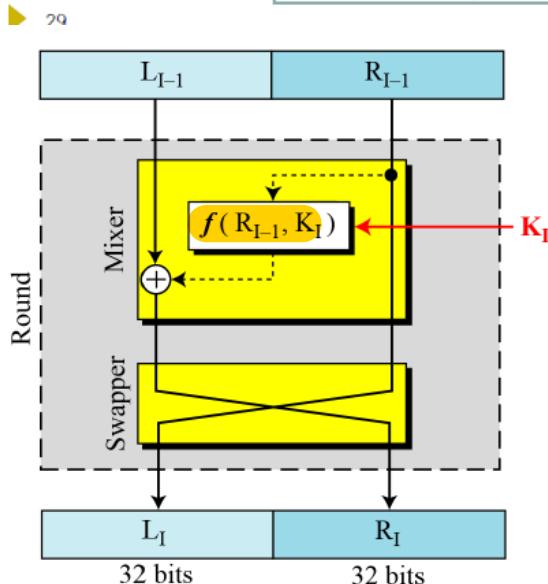
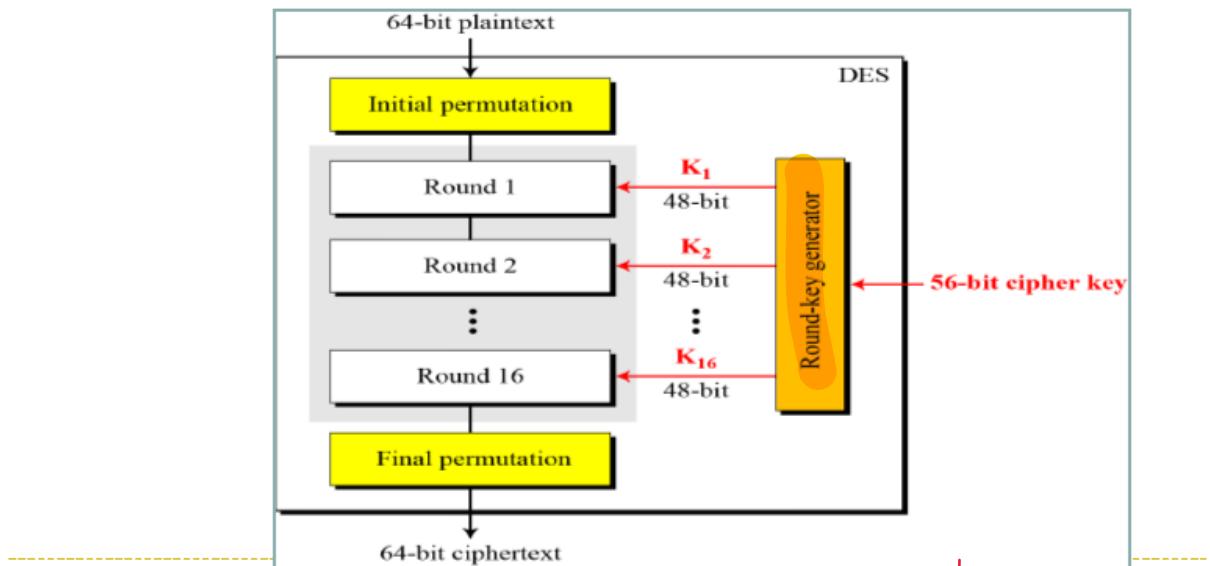
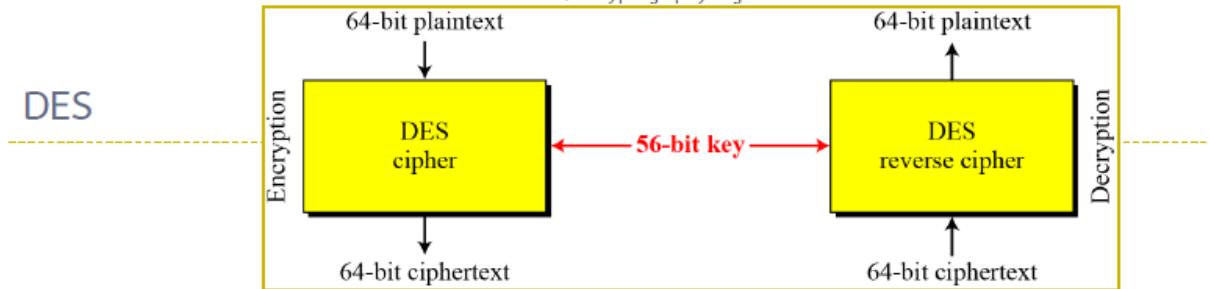
Output block: 64 bits [มีการตัด parity bit ในระหว่างกระบวนการด้วย]

Key 56 bits [แรกเริ่ม 128 บิต แต่มีการไปพัฒนาต่อจนสามารถลดขนาดคีย์ลงมาได้]

Iteration: 16 รอบ

โครงสร้างเป็น Feistel cipher, หัวใจสำคัญคือ f-function เพราะการ permute ไม่ได้มีส่วนให้เกิดความซับซ้อน

การโจมตีด้วย Linear และ Differential มากกว่า brute force (เพราการหา known PT, chosen PT ต้องใช้ถึง 2^{47})



feistel

แต่ละรอบประกอบด้วย 2 Component หลัก

1. Substitution cipher $f_n()$ เป็น S-box ใน
2. Transposition ใช้การสลับที่ ยกเว้นรอบสุดท้าย
ทั้งสองเป็น Involution function : เป็น self-invertible

DES Encryption

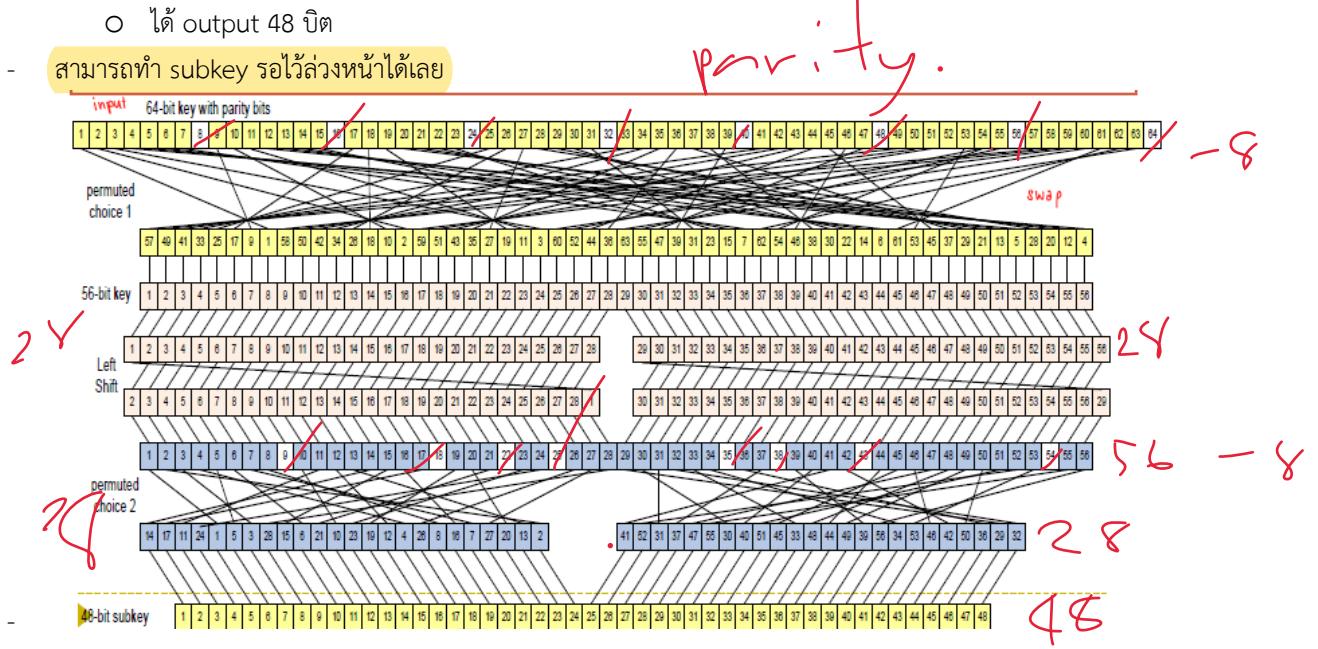
$$DES = (IP)^{-1} F_{16} T F_{15} T \dots F_2 T F_1 (IP)$$

DES Decryption

$$DES^{-1} = (IP) F_1 T F_2 T \dots F_{15} T F_{16} (IP)^{-1}$$

Key scheduling

- รับ 64 bits
 - เลือก 56-bits มา permute โดยใช้ Permutated Choice One
 - แบ่ง 2 ช่วง เป็น 28 บิต
- การทำงานแต่ละรอบ
 - ในแต่ละครึ่ง ทำ left-rotate 1 หรือ 2 บิต ขึ้นกับ rotate schedule
 - ในแต่ละครึ่ง เลือก 24 บิต ทำ permute (ใช้ PC2) และรวมเป็น 48 บิต



Initial and Final Permutation

- เป็น Straight P-boxes (เป็น inverse ของกันและกัน)
- บิตตำแหน่งนี่ไปทางซ้าย บิตตำแหน่งนี่คุ้งไปทางขวา

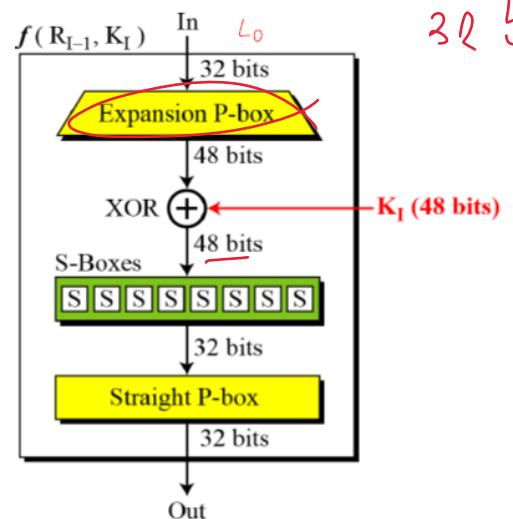
Initial Permutation											Final Permutation										
58	50	42	34	26	18	10	02	40	08	48	16	56	24	64	32						
60	52	44	36	28	20	12	04	39	07	47	15	55	23	63	31						
62	54	46	38	30	22	14	06	38	06	46	14	54	22	62	30						
64	56	48	40	32	24	16	08	37	05	45	13	53	21	61	29						
57	49	41	33	25	17	09	01	36	04	44	12	52	20	60	28						
59	51	43	35	27	19	11	03	35	03	43	11	51	19	59	27						
61	53	45	37	29	21	13	05	34	02	42	10	50	18	58	26						
63	55	47	39	31	23	15	07	33	01	41	09	49	17	57	25						

- จากรูปใน IP ตำแหน่ง (1,1) หมายถึงให้สลับบิตที่ 58 มาที่ตัวแรก บิตที่ 50 สลับมาที่ตัวที่สอง

64 R
32 bit

f-function

- ส่วนสำคัญที่สุดใน DES
- Input 2 ตัว: R, Key
- มี 4 stages
 - Expansion จาก 32 bits เป็น 48 bits
 - Key mixing (XOR with Key)
 - S-box: การันตี Diffusion และ Confusion
 - P-box: เพิ่ม diffusion/avalanche effect



S-box

- มี 8 S-boxes
- ใช้หลักการ 3 ข้อในการออกแบบ
 - การเปลี่ยนหนึ่งบิต output เปลี่ยนอย่างน้อย 2 บิต (Diffusion)

48 / 6 8

- ถ้ามี 1 bit ที่ไม่เปลี่ยน Output ยังคงบาลานซ์ เพราะอีก 5 บิตมีความต่าง (Cryptanalysis ยาก)
- แต่ละแครปจะประกอบด้วย non-linear Boolean function (Confusion)
- Input 6, out 4 = total 32 bits
- ตัดบิตหัวท้ายมาดูแล้วของ S-box ส่วน 4 บิตที่เหลือแทน Column

Table 3.3 Definition of DES S-Boxes

S₁	14 4 13 1 2 15 11 8 3 10 6 12 5 9 0 7 0 15 7 4 14 2 13 1 10 6 12 11 9 5 3 8 4 1 14 8 13 6 2 11 15 12 9 7 3 10 5 0 15 12 8 2 4 9 1 7 5 11 3 14 10 0 6 13
S₂	15 1 8 14 6 11 3 4 9 7 2 13 12 0 5 10 3 13 4 7 11 10 2 8 14 12 0 1 10 6 9 11 5 0 14 8 1 11 3 15 4 2 11 6 7 12 0 5 14 9 13 8 10 1 3 15 7 4 2 11 6 7 12 0 5 14 9
S₃	10 0 9 14 6 3 15 5 1 13 12 7 11 4 2 8 13 7 0 9 3 4 6 10 2 8 5 14 12 11 15 1 1 13 6 4 9 8 15 3 0 11 1 2 12 5 10 14 7 1 10 13 0 6 9 8 7 4 15 14 3 11 5 2 12
S₄	7 13 14 3 0 6 9 10 1 2 8 5 11 12 4 15 13 8 11 5 6 15 0 3 4 7 2 12 1 10 14 9 10 6 9 0 12 11 7 13 15 1 3 14 5 2 8 4 3 15 0 6 10 1 13 8 9 4 5 11 12 7 2 14
S₅	2 12 4 1 7 10 11 6 8 5 3 15 13 0 14 9 14 11 12 4 7 13 1 5 0 15 10 3 9 8 6 4 2 1 11 10 13 7 8 15 9 12 5 6 3 0 14 11 8 12 7 1 14 2 13 6 15 0 9 10 4 5 3
S₆	12 1 10 15 9 2 6 8 0 13 3 4 14 7 5 11 10 15 4 2 7 12 9 5 6 1 13 14 0 11 3 8 9 14 15 5 2 8 12 3 7 0 4 10 1 13 11 6 4 3 2 12 9 5 15 10 11 14 1 7 6 0 8 13
S₇	4 11 2 14 15 0 8 13 3 12 9 7 5 10 6 1 13 0 11 7 4 9 1 10 14 3 5 12 2 15 8 6 1 4 11 13 12 3 7 14 10 15 6 8 0 5 9 2 6 11 13 8 1 4 10 7 9 5 0 15 14 2 3 12
S₈	13 2 8 4 6 15 11 1 10 9 3 14 5 0 12 7 1 15 13 8 10 3 7 12 5 6 11 5 0 14 9 6 7 11 4 1 9 12 14 2 0 6 10 13 15 3 5 8 2 1 14 7 4 10 8 13 15 12 9 0 3 5 6 11

10 15
 NO 10 X
 11 16 16
 10
 0011

- 8+4+2
- ▶ 1. $S_1(011000) \Rightarrow 5 \Rightarrow 0101$
 - ▶ 2. $S_2(001001) \Rightarrow 15 \Rightarrow 1111$
 - ▶ 3. $S_3(010010) \Rightarrow 8+1 \Rightarrow 9 \Rightarrow 10 \Rightarrow 1101$
 - ▶ 4. $S_4(111101) \Rightarrow 8+4+2 \Rightarrow 2 \Rightarrow$
hex
 - ▶ 5. $S(18\ 09\ 12\ 3d\ 11\ 17\ 38\ 39) = 5fd25e03$ (มาได้ใจ ;)

Table 3.3 Definition of DES S-Boxes

S₁	14 4 13 1 2 15 11 8 3 10 6 12 5 9 0 7 0 15 7 4 14 2 13 1 10 6 12 11 9 5 3 8 4 1 14 8 13 6 2 11 15 12 9 7 3 10 5 0 15 12 8 2 4 9 1 7 5 11 3 14 10 0 6 13
S₂	15 1 8 14 6 11 3 4 9 7 2 13 12 0 5 10 3 13 4 7 11 10 2 8 14 12 0 1 10 6 9 11 5 0 14 8 1 11 3 15 4 2 11 6 7 12 0 5 14 9 13 8 10 1 3 15 7 4 2 11 6 7 12 0 5 14 9
S₃	10 0 9 14 6 3 15 5 1 13 12 7 11 4 2 8 13 7 0 9 3 4 6 10 2 8 5 14 12 11 15 1 1 13 6 4 9 8 15 3 0 11 1 2 12 5 10 14 7 1 10 13 0 6 9 8 7 4 15 14 3 11 5 2 12
S₄	7 13 14 3 0 6 9 10 1 2 8 5 11 12 4 15 13 8 11 5 6 15 0 3 4 7 2 12 1 10 14 9 10 6 9 0 12 11 7 13 15 1 3 14 5 2 8 4 3 15 0 6 10 1 13 8 9 4 5 11 12 7 2 14
S₅	2 12 4 1 7 10 11 6 8 5 3 15 13 0 14 9 14 11 12 4 7 13 1 5 0 15 10 3 9 8 6 4 2 1 11 10 13 7 8 15 9 12 5 6 3 0 14 11 8 12 7 1 14 2 13 6 15 0 9 10 4 5 3
S₆	12 1 10 15 9 2 6 8 0 13 3 4 14 7 5 11 10 15 4 2 7 12 9 5 6 1 13 14 0 11 3 8 9 14 15 5 2 8 12 3 7 0 4 10 1 13 11 6 4 3 2 12 9 5 15 10 11 14 1 7 6 0 8 13
S₇	4 11 2 14 15 0 8 13 3 12 9 7 5 10 6 1 13 0 11 7 4 9 1 10 14 3 5 12 2 15 8 6 1 4 11 13 12 3 7 14 10 15 6 8 0 5 9 2 6 11 13 8 1 4 10 7 9 5 0 15 14 2 3 12
S₈	13 2 8 4 6 15 11 1 10 9 3 14 5 0 12 7 1 15 13 8 10 3 7 12 5 6 11 5 0 14 9 6 7 11 4 1 9 12 14 2 0 6 10 13 15 3 5 8 2 1 14 7 4 10 8 13 15 12 9 0 3 5 6 11

18 00011000 $\Rightarrow 12 \Rightarrow 5$
 09 0001001 $\Rightarrow 4 \Rightarrow 15$
 12 $\Rightarrow 00010010 \Rightarrow 9 \Rightarrow 13$
 3d $\Rightarrow 00111101 \Rightarrow 14 \Rightarrow 2$
 11 $\Rightarrow 00010001 \Rightarrow 8 \Rightarrow 5$
 17 $\Rightarrow 00010111 \Rightarrow 5 \Rightarrow 5$
 38 $\Rightarrow 00111000 \Rightarrow 12 \Rightarrow 0$
 39 $\Rightarrow 00111001 \Rightarrow 12 \Rightarrow 3$

DES Decryption

- ทำงานเหมือนตอนเข้ารหัส
- ไม่ต้องย้อนกลับ
- สลับคีย์ เริ่มจากการที่ 16

Avalanche effect in DES

- แข็งแกร่งสุดๆ

- เปลี่ยน 1 bit ที่ input \rightarrow ciphertext เปลี่ยนเฉลี่ย 34 bits
- เปลี่ยน 1 bit ที่ key \rightarrow ciphertext เปลี่ยนเฉลี่ย 35 bits

DES Weak key (ເອກລັກຄ່າ)

- ในจำนวนคีย์ທີ່ໜົດ (2^{56}) มีครึ่ງນึงเป็น Complement (ປົຕຽງຂ້າມ 0,1 ກັນແລະກັນ)
 - ถ้า Ciphertext ของ Plaintext ใช้ Key C complement จะเป็น Ciphertext ของ Plaintext P complement
 - ກລາຍເປັນວ່າໄມ້ຕ້ອງລອງທຸກຄີ່ມ ລອງແຄ່ຄົງນິ້ງ (2^{55}) ກີ່ພອ (Chosen plaintext Assumption)
- ມີ 4 weak keys ເນື້ອນມາ parity bits ອອກຈະມີຄ່າເປັນ 0s, all1s, ທີ່ອ half 0s/ half1s

55

2

Keys before parities drop (64 bits)	Actual key (56 bits)
0101 0101 0101 0101	0000000 0000000
1F1F 1F1F 0E0E 0E0E	0000000 FFFFFFF
E0E0 E0E0 F1F1 F1F1	FFFFFFF 0000000
FEFE FEFE FEFE FEFE	FFFFFFF FFFFFFF

- ກລາຍເປັນວ່າສໍາທຳ encryption 2 ຄັ້ງດ້ວຍ Key ເດືອກກັນ ຈະໄດ້ Original plaintext ທີ່ອແມ້ແຕ່ການທຳ Decryption 2 ຄັ້ງດ້ວຍຄີ່ມເດືອກກັນກີ່ຕາມ

Semi-weak key (Inverse)

- ມີ 12 semi-weak keys
 - เป็น subkey ທີ່ 16 ຮອບມີເພີ່ມ 1 ຄູ່ (2 ຄ່າ)
 - ເປັນຄ່າທີ່ເໝືອນກັນ ແກ່ເຮັງຕ່າງກັນ
 - ຄູ່ຂອງ key ເປັນ inverse ຂອງກັນແລະກັນເນື້ອນມາ parity bit ອອກ

First key in the pair	Second key in the pair
01FE 01FE 01FE 01FE	FE01 FE01 FE01 FE01
1FE0 1FE0 OEOF 0EOF	EO1F EO1F F1OE F1OE
O1EO O1E1 O1F1 O1F1	E001 E001 F1O1 F1O1
1FFE 1FFE OEOF OEOF	FE1F FE1F FEOE FEOE
O11F O11F O1OE O1OE	1F01 1F01 OEOF OEOF
EOF EEOF F1FE F1FE	FEE0 FEE0 FEF1 FEF1

- ເນື້ອນຄູ່ຂອງ semi-weak key ມາ encrypt ຈະໄດ້ plaintext ຕ້າດີມ

DES attack

Differential cryptanalysis

- ໂຄມຕີ່ f-function ກັບ S-box
- ເຮັມຕີ່ f-function ສາມໃຈທີ່ຄູ່ຂອງ input ແລ້ວດູຄວາມຕ່າງກັບ outputs (XOR) $(X \oplus Z) \oplus (X' \oplus Z) = (X \oplus X')$
- ສຽງມີໄຣເປີ່ມຕົວ S-box
- ຄວາມຍາກໃນການໂຈມຕີ່
 - ຈຳນວນໃນການທຳ encryption, ຈຳນວນຂອງຄູ່ Plaintext/Ciphertext ຕ້ອງເພີ່ມພອ

- ถ้ารอบไม่เยื่อง วิธีนี้ดีกว่า แต่ถ้ารอบเยื่องขึ้น Exhaustive key search ดีกว่า
- ต้องเลือก plaintext ถึง 2^{47} ตัว

Linear cryptanalysis

- ใช้ linear approximations $X_{i1} \oplus X_{i2} \dots Y_{j1} \oplus Y_{j2} \dots = Z_{k1} \oplus Z_{k2} \dots$
- ต้องรู้ plaintext 2^{43} ตัว

Other attack

- Timing attack
 - อัลกอริทึมในการเข้ารหัส หรือถอดรหัส ใช้เวลาต่างกันเมื่อ input ต่างกัน

DES upgrade

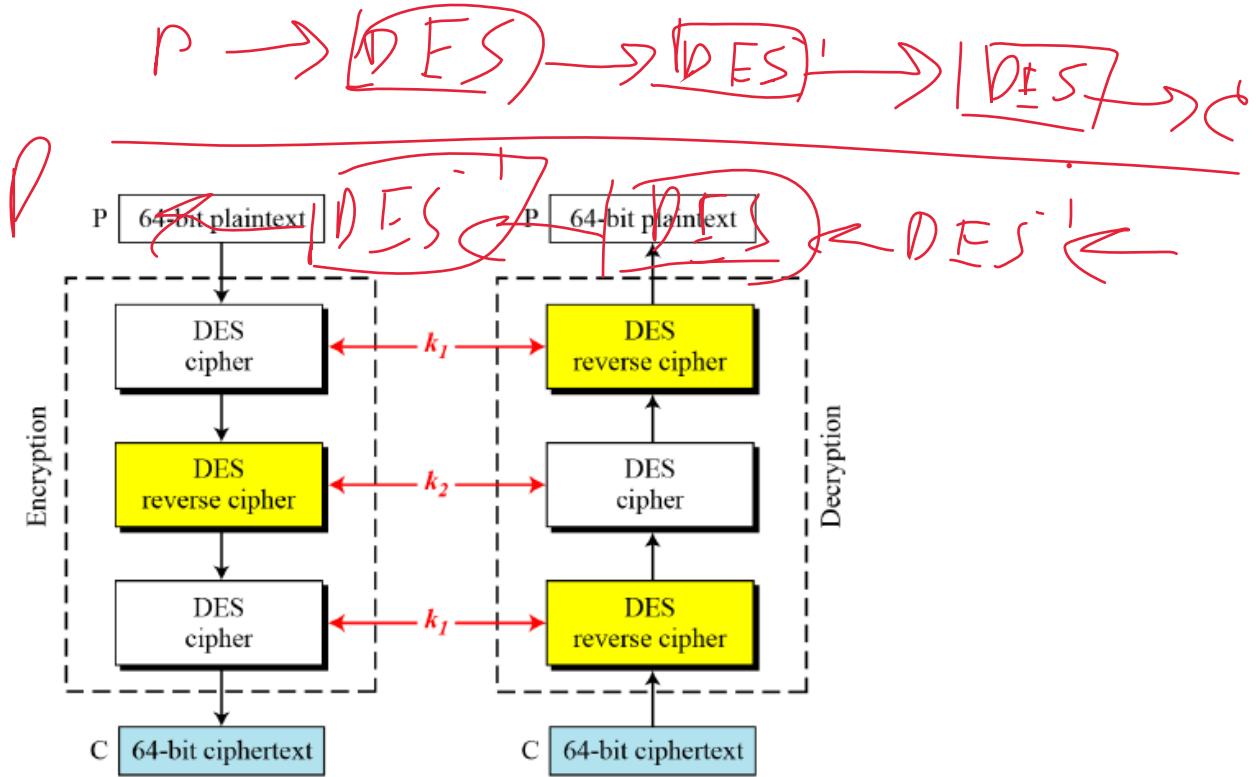
2DES

- เพิ่มขนาดคีย์ โดยการต่อ DES เข้าไปเพิ่ม
 - แต่ ๆ การเพิ่มเป็น 2 ครั้งไม่ได้ทำให้ปลอดภัยขึ้น อีกทั้งขนาดของคีย์ก็เพิ่มมาเป็นแค่ 2^{57} ไม่ใช่ 2^{112}
 - โดน Meet-in-the middle attack: ทำการ Encrypt 1 รอบ, Decrypt 1 รอบ แล้วดูความสัมพันธ์ได้เลย



3DES – EDE2

- Meet-in-the middle ไม่ได้แล้ว
- ทำงานช้ากว่าเดิม 3 เท่าเมื่อใช้งานกับชอร์ฟเวอร์ เพราะแรกเริ่มถูกออกแบบมาโดยเน้นใช้กับฮาร์ดแวร์



AES (Advanced Encryption Standard) / Rijndael

เนื่องจาก 3DES ไม่ practical ในการใช้งานจริง (ช้าและ block size ไม่พอ)

Key: 128, 192, 256 เลือกได้ตามสะดวก
10 12 14

Iterated block ciphers: ทำ Operation เดิมซ้ำๆ

Non-Feistel: แต่ละกระบวนการจะต้องใช้ inverse ยกเว้น AddRoundKey เพราะการ XOR เป็น self-invertible

key scheduling algorithm: สำหรับสร้าง sub-key ของแต่ละรอบ

ไม่มี weak key

Criteria

1. Security: Resistance to cryptanalysis, mathematic basis, Randomness output
2. Algorithm & implementation Characteristic: Computational efficiency (speed), Memory requirement
3. Cost: Flexibility (มีคีย์และไสส์บล็อกให้เลือกได้หลาย), Hardware & software suitability (implement both in hardware & software)

Security Margin: ระยะห่างระหว่างรอบที่ attack ได้กับ จำนวนรอบทั้งหมดของอัลกอริทึม

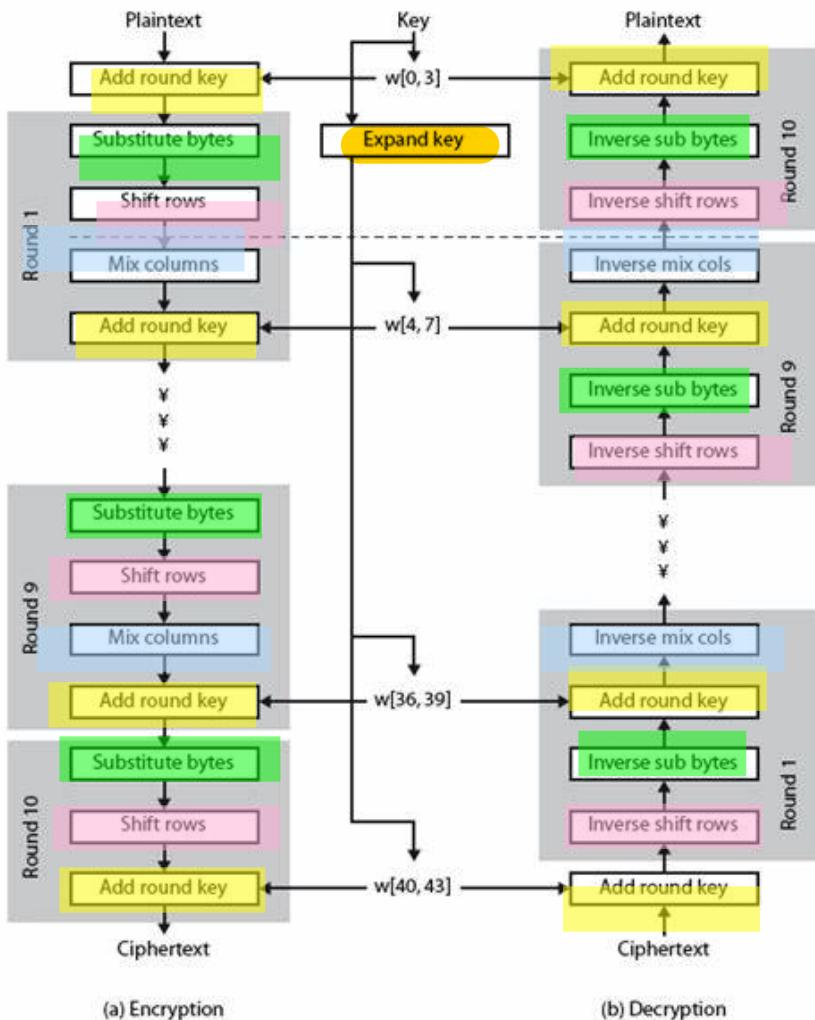
Attack on Rijndael model

- 128-bit keys: 7 ใน 10 รอบ
- 192-bit keys: 8 ใน 12 รอบ
- 256-bit keys: 9 ใน 14 รอบ

Rijndael: Resistance, speed & code compact, design simplicity

Key scheduling: ขนาด block 128 เสมอ, จำนวนรอบขึ้นกับ key size (10:128, 12:192, 14:256)

Diagram



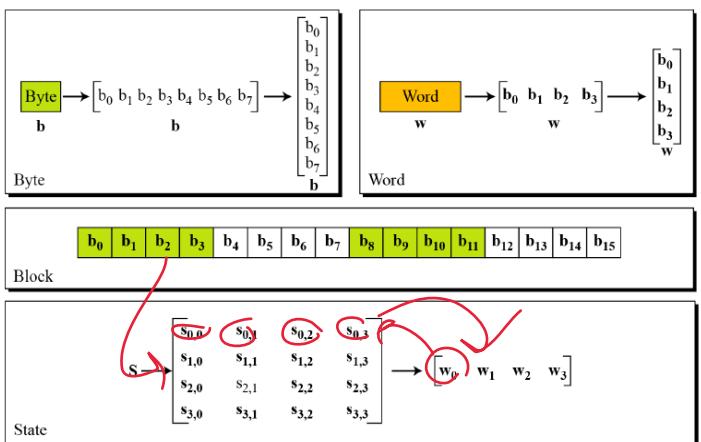
1. Key expansion: สร้าง key ขนาด 128 bits
2. Substitution bytes: ใช้ S box ในการทำ byte-to-byte substitution
3. Shift rows: simple permutation
4. Mix columns: การสับเปลี่ยนโดยใช้ฟังก์ชันทางคณิตศาสตร์ $GF(2^8)$
5. Add round key: XOR block กับ sub key ที่ได้จาก key expansion

Data Unit used in AES

1 byte = 8 bits

1 word = 4 bytes = 32 bits

1 block = 16 bytes = 1 State = 32 bytes = 4 words



The Encryption details

1. แบ่งบล็อก 128 บิต เป็น 32-bit words จำนวน 4 words และ word แบ่งเป็น 4 bytes
2. ได้ matrix 4×4 ซึ่ง 1 ช่อง คือ 1 byte

124

258

3. นำคีย์ 128-bit มาเขียนใน matrix ถ้าคีย์ยาว จำนวน column จะเพิ่มขึ้น เช่น 258 bit keys

4. ลำดับการทำงานของวงจร

a. Add round key ~~612~~

b. R1 – Rn-1

i. Substitute Bytes

ii. Shift Rows

iii. Mix Column

iv. Add round Key

c. Final round

i. Substitute Bytes

ii. Shift Rows

iii. Add round key

5. S-box ของ AES [8 bits (x) to 8 bits (y)]

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x		03	7c	77	7b	f2	6b	6e	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	c5	
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	kd	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	kb	16	

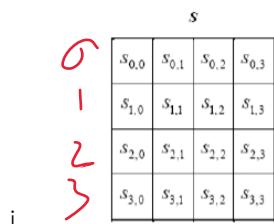
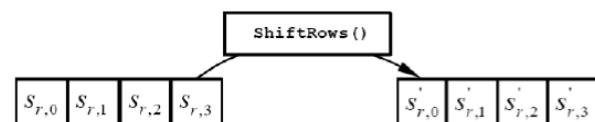
EA	04	65	85
83	45	5D	96
EC	6E	4C	90
5C	33	98	B0
4A	C3	46	E7
F0	2D	AD	C5

a. Figure 7. S-box: substitution values for the byte xy (in hexadecimal format).

6. Shift rows transformation: คิดที่ shift

a. แค่เรกไม่ shift

b. สามแถวหลังทำ left-shift ตามลำดับ 1 2 และ 3 ไป 1 ตามลำดับ



shift
↓
shift row

7. Mix Column: คิดที่ columm

a. Linear mixing ของ word $(a_0, a_1, a_2, a_3) \rightarrow (b_0, b_1, b_2, b_3)$ ผ่าน operation $GF(2^8)$

b. $f(a_0, a_1, a_2, a_3) = (a_1 \oplus a_2 \oplus a_3) \oplus g((a_0 \oplus a_1) \ll 1)$

i. $(a_0 \oplus a_1) \ll 1$ จะได้ 9-bit เพื่อนำไป & กับ 100_x

c. $g(x) = (x \& 100_x) ? (x \oplus 1B_x) : x$

i. ถ้า $x \& 100_x$ และตัวหน้าสุด = 1 ให้ $x \oplus 1B_x$ XOR กับ $1B_x$

mix col
↓
key

$$f(a_1, a_2, a_3, a_0) = (a_2 \oplus a_3 \oplus a_0) \oplus GF(a_0 \oplus a_1)$$

ii. ตัดตัวหน้าสุดทิ้ง เพื่อให้เหลือแค่ 8 บิต

8. Add Round key (Key mixing): XOR กับ sub key ของรอบนั้น

- ▶ $f(a,b,c,d) = b \text{ xor } c \text{ xor } d \text{ xor } g((a \text{ xor } b) \ll 1)$,
- ▶ where $g(x) = x \text{ xor } 1b_x$ if $x \& 100_x$
else $g(x) = x$
- ▶ $f(87,6E,46,A6)$
 - ▶ $6E \text{ xor } 46 \text{ xor } A6 \text{ xor } (g(87 \text{ xor } 6E) \ll 1)$
 - ▶ $g(87 \text{ xor } 6E \ll 1) \quad 11101001 \rightarrow \text{เต็มบิต} = 011101001 \rightarrow \text{shift} \rightarrow 111010010$
 - ▶ $[10000111 \text{ xor } 01101110] \ll 1$
 - ▶ $111010010 \Rightarrow 111010010 \& 000100000000$
 - ▶ ดังนั้น เลือก $x \text{ xor } 1b_x$ (ตัด 1 ทิ้ง เพราะจะเลือกใช้แค่ 8 bits)
 - ▶ $111010010 \Rightarrow 11010010$
 - ▶ $11010010 \text{ xor } 00011011 \Rightarrow 11001001$
 - ▶ $01101110 \text{ xor } 01000110 \text{ xor } 10100110 \text{ xor } 11001001 \Rightarrow 01000111 \Rightarrow 47$

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

The Decryption details

- การถอดรหัสใช้ Inverse ของ transformation ทั้ง 4 operation และ sub-keys ใน reverse order
 - Inverse ของ S-box
 - Inverse ของ Mix column
 - Inverse ของ Shift Rows

1. Inverse Shift Rows

- a. แก้ 2,3,4 shift right ไป 1,2,3 ครั้งตามลำดับ

S

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

S'

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,3}$	$S_{1,0}$	$S_{1,1}$	$S_{1,2}$
$S_{2,2}$	$S_{2,3}$	$S_{2,0}$	$S_{2,1}$
$S_{3,1}$	$S_{3,2}$	$S_{3,3}$	$S_{3,0}$

b.

2. Inverse Sub bytes

- a. ใช้ inverse S-box ได้จาก Inverse transformation

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

b. ทำ multiplicative inverse ใน $GF(2^8)$

3. Inverse Mix Column

a. เป็น linear operation คล้ายกับ Mixcolumn แต่เปลี่ยนชื่อเป็น d()

$$s'(x) = a^{-1}(x) \otimes s(x) :$$

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \leq c < Nb.$$

$$d_1(a_0, a_1, a_2, a_3) = a_0 \oplus a_1 \oplus a_2 \oplus a_3$$

$$d_2(a_0, a_1, a_2, a_3) = g(d_1(a_0, a_1, a_2, a_3) \ll 1) \oplus (a_0 \oplus a_2)$$

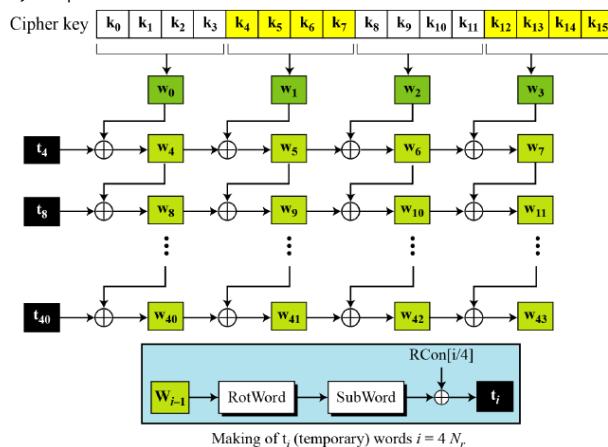
$$d_3(a_0, a_1, a_2, a_3) = g(d_2(a_0, a_1, a_2, a_3) \ll 1) \oplus (a_0 \oplus a_1)$$

$$b. \quad d_4(a_0, a_1, a_2, a_3) = g(d_3(a_0, a_1, a_2, a_3) \ll 1) \oplus (a_1 \oplus a_2 \oplus a_3)$$

Key scheduling

- คีย์แต่ละรอบ คิดจาก Cipher key ในอัลกอริทึม scheduling มีสองขั้นตอน

- Key expansion



- Round Key Selection

- Input N_k -word key = $(32N_k\text{-bits})$
- Key Generation มีหลักการดังนี้
 - จำนวนบิตทั้งหมด = ขนาดของบล็อก * (จำนวนรอบ + 1)

- เท่าบล็อก 128 บิต (4 word) และทำ 10 รอบ คีย์ที่ใช้มีจำนวน $128 \times 11 = 1408$ บิต
- Cipher key จะถูกขยายเป็น Expanded Key
- Round key ดึงมาจาก Expanded key
 - คีย์แรก คือ 4 words แรก $[w_0, w_1, w_2, w_3]$
 - คีย์รอบแรก คือ 4 words ถัดไป $[w_4, w_5, w_6, w_7]$
 - คีย์ของรอบสุดท้าย คือ ห้าย คือ 4 words สุดท้าย $[w_{40}, w_{41}, w_{42}, w_{43}]$

Attack on AES

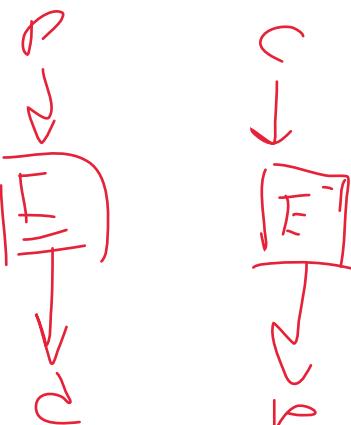
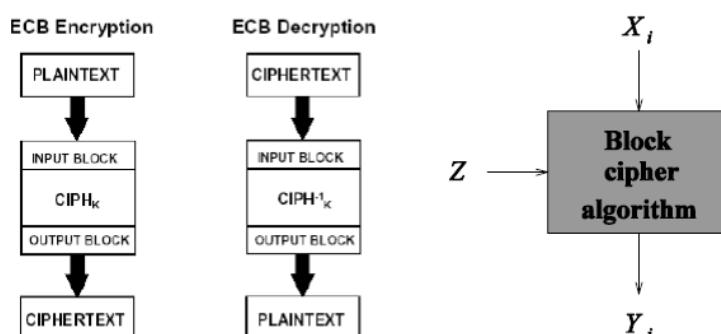
- AES ป้องกันการโจมตีทั้ง differential attack และ linear attack
- Square attack: เป็นการทดลองโจมตีที่เร็วกว่า brute force ซึ่งก็ยังซ้ำมากอยู่ดีสำหรับการจะมาใช้จริง

ECB mode

Electronic codebook: ใหม่ดที่ง่ายที่สุด

C.A : 08

~~P | 00|01|10|1
E | 10|00|11|0~~



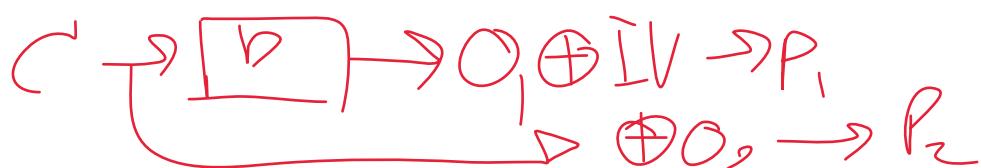
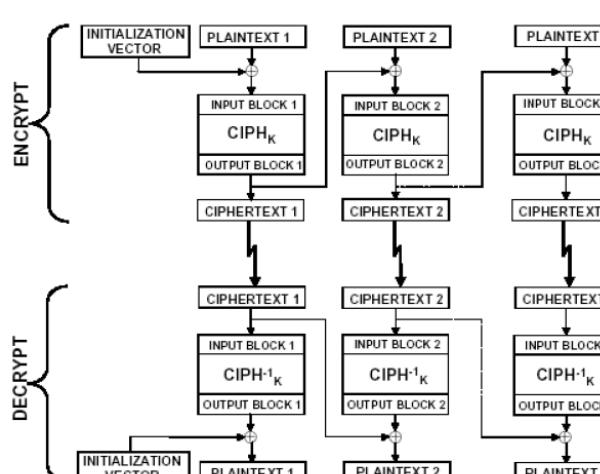
Problem: เทืนโครงสร้างอยู่บ้าง

ใช้คีย์เดียวกันทุกบล็อก, ไม่ปลอดภัยสำหรับข้อความยาวๆ

Plaintext เดียวกันได้ ciphertext เหมือนกัน

CBC mode

Cipher Block Chaining:



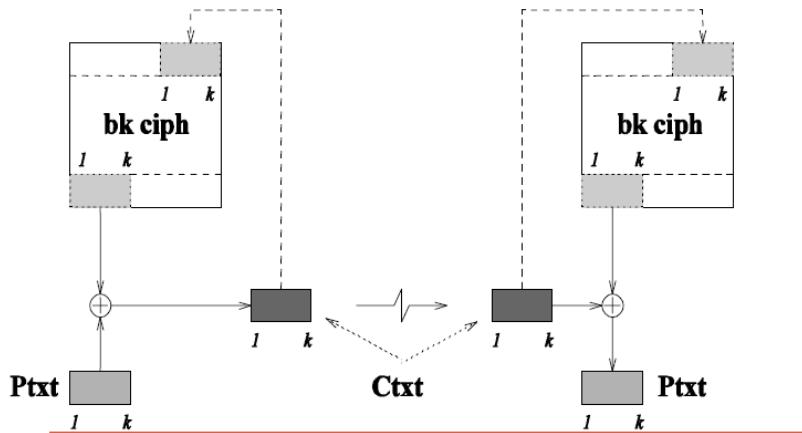
แก้ปัญหา plaintext เดิมได้ ciphertext เดิม มี Integrity

ข้อ ทำแบบ parallel ไม่ได้

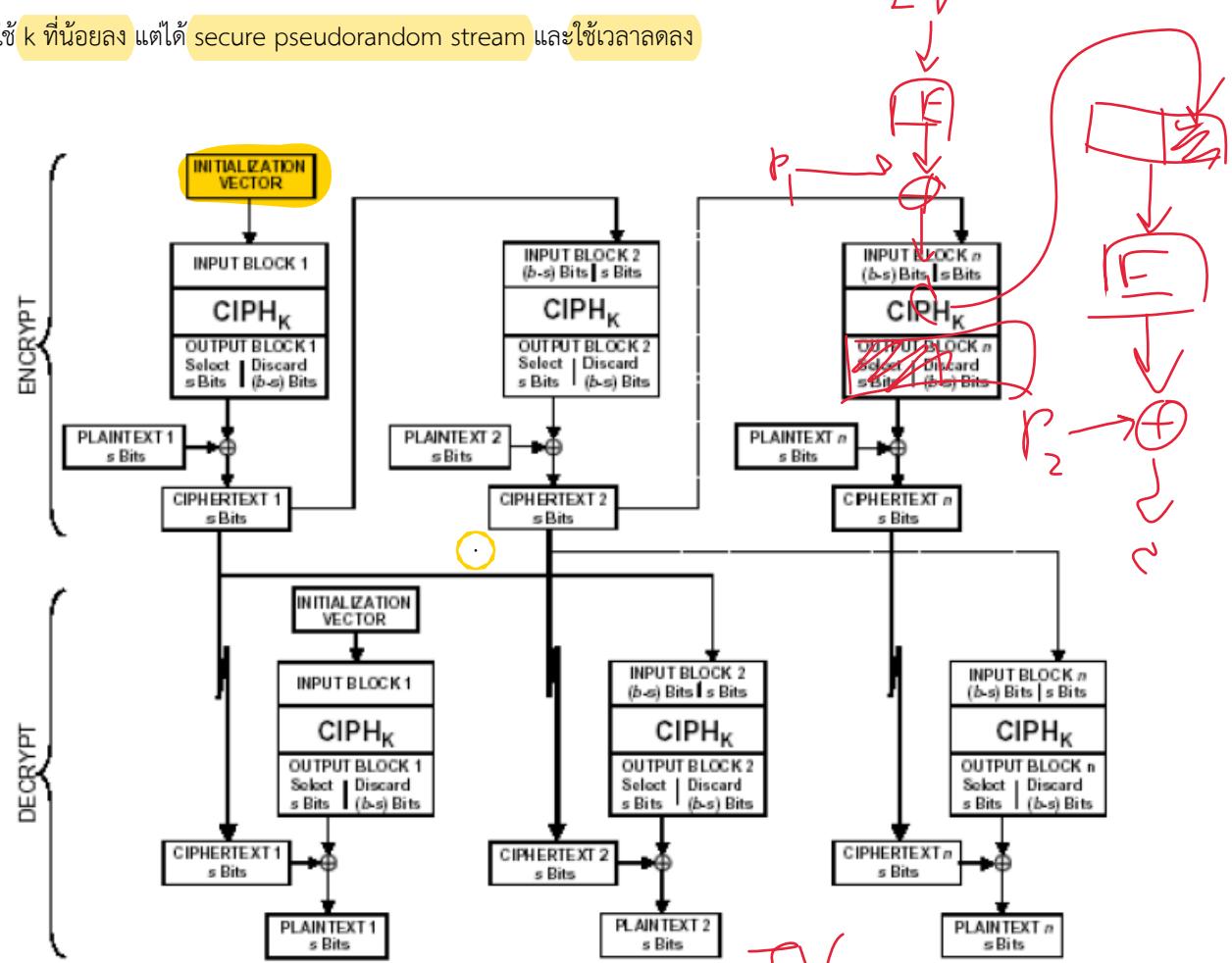
CFB mode

IV

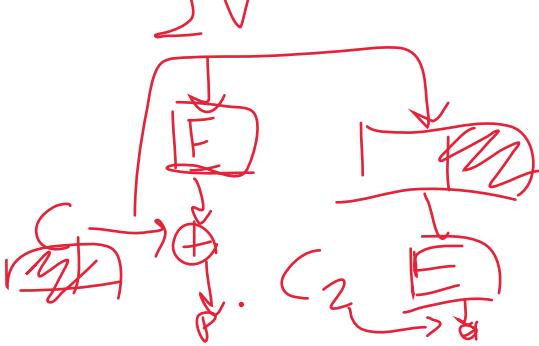
Cipher Feedback: pseudorandom generator, left shift by k-bits, with k-bit feedback



ใช้ k ที่น้อยลง แต่ได้ secure pseudorandom stream และใช้เวลาลดลง

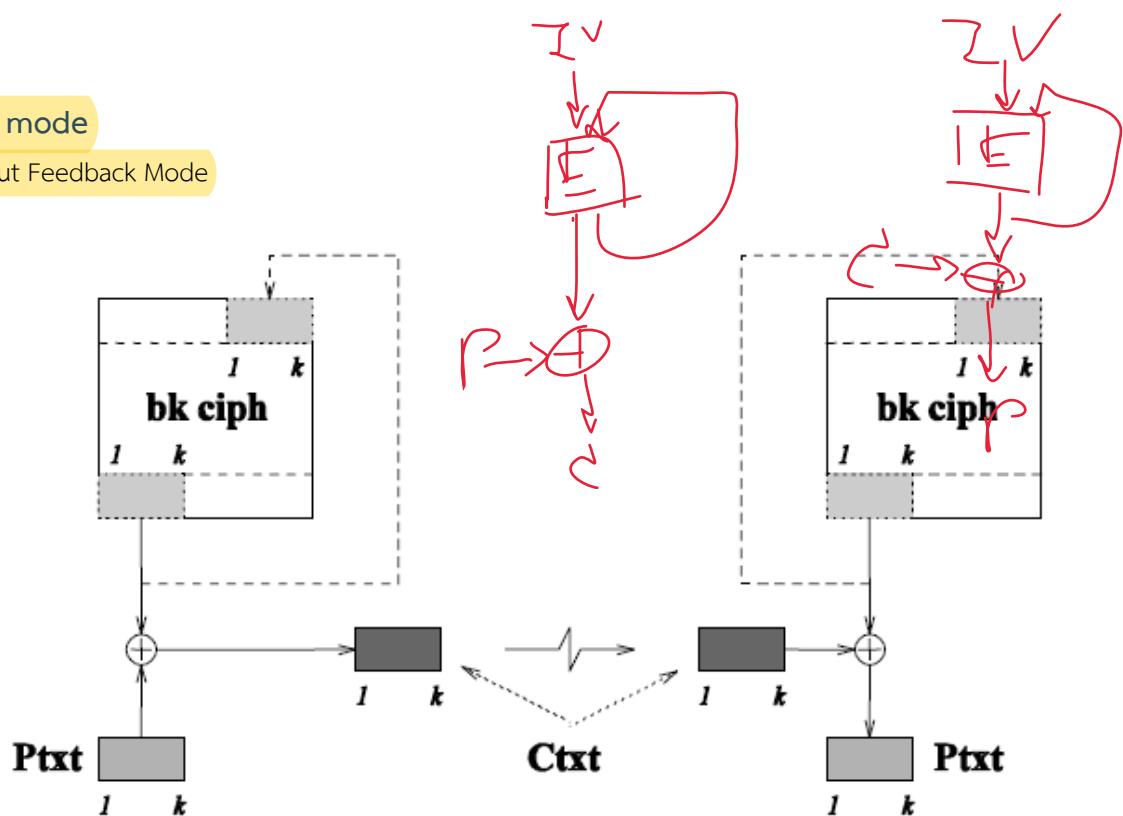


Block cipher algorithm \rightarrow pseudorandom generator



OFB mode

Output Feedback Mode



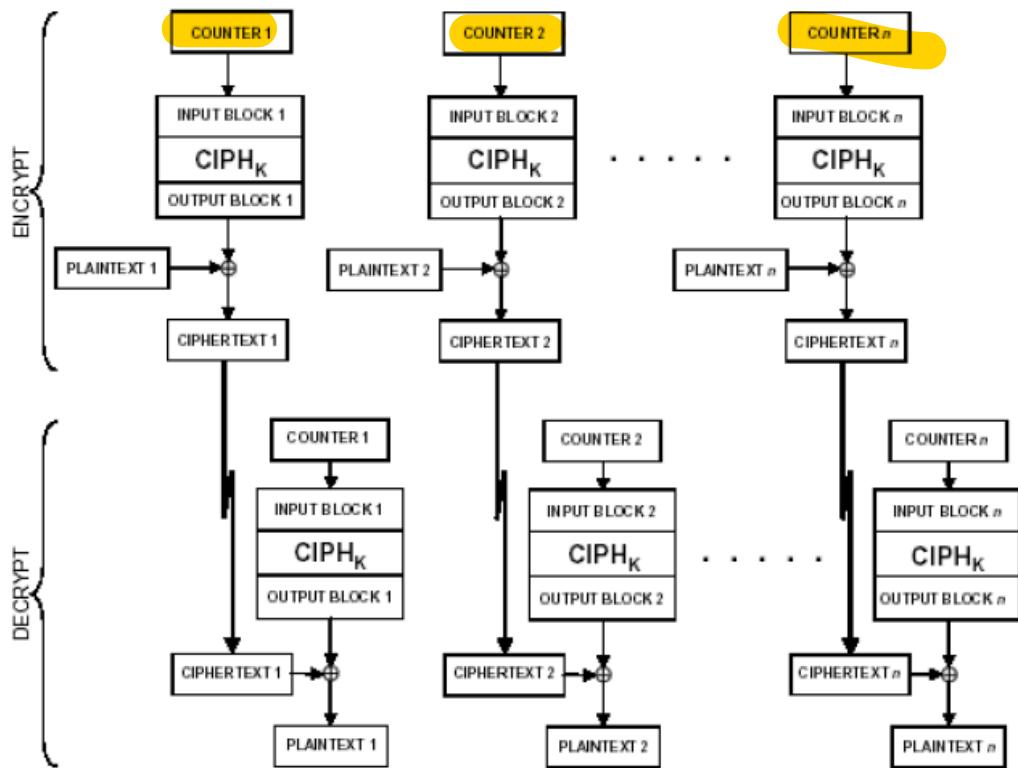
CTR mode

เร็วขึ้น แต่ไม่ยืนยันว่ามี integrity

- มี input block เพิ่มขึ้นมา เรียกว่า counter
- Ciphertext = plaintext XOR output block
- Counter: block: sequence ต่างกันหมดทุกอัน
 - Counter จะถูก initialized ไว้ก่อน แล้วไปเพิ่มทีละหนึ่ง
- Decryption: ใช้ลำดับเดิมของ counter

0000

1111

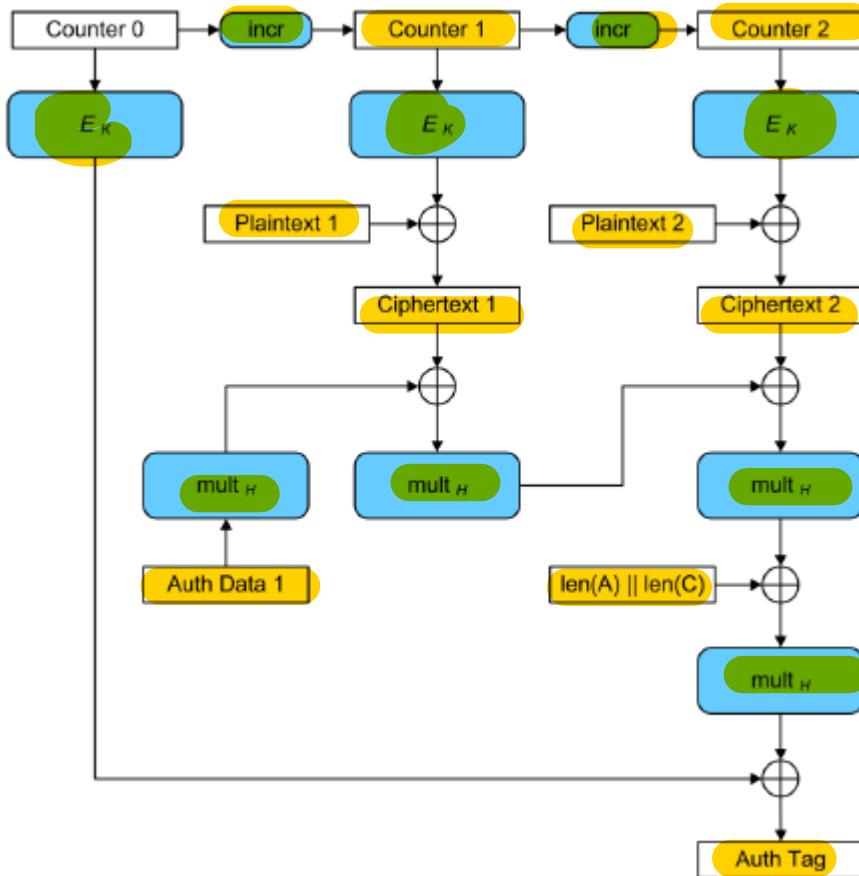


- สามารถทำแบบ parallel ได้
- สามารถทำ output block รอไว้ก่อนได้เลย

GCM mode

- เป็น Authenticated Encryption algorithm
- ใช้ CTR รวมกับ Galois mode of authentication

ได้ทั้ง confidentiality และ Integrity



การเลือกใช้ mode ของ Block cipher

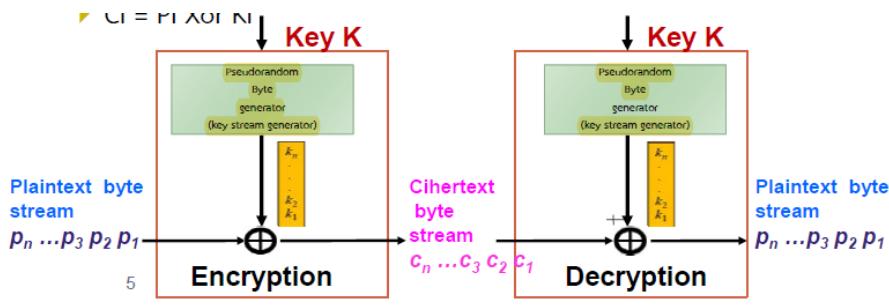
- ECB: Secure transmission of single value เช่น ส่งคีย์ในการ encryption
- CBC, CTR: กรณี block-oriented transmission
- OFB, CFB: Stream-oriented transmission

Padding

- กรณีการแบ่งบล็อกของ plaintext แล้วมีขนาดของ block size เล็กกว่า cipher block size จำเป็นจะต้องมีการเติมบล็อกสุดท้ายให้เท่ากัน = เติมให้เต็ม
 - ตัวอย่างเช่น
 - เติม 100...000 ในบล็อกสุดท้ายให้เติมขนาด
 - ปกติใช้คณิตศาสตร์ในการเติม เช่น การสุ่มเติมข้อมูลในบล็อกสุดท้าย
- rightmost.

Stream Cipher

- เป็นการ encrypt ตัวอักษรทั้งหมดภายในรอบเดียว
 - เช่น Vigenère cipher



Two-time pad (insecure)

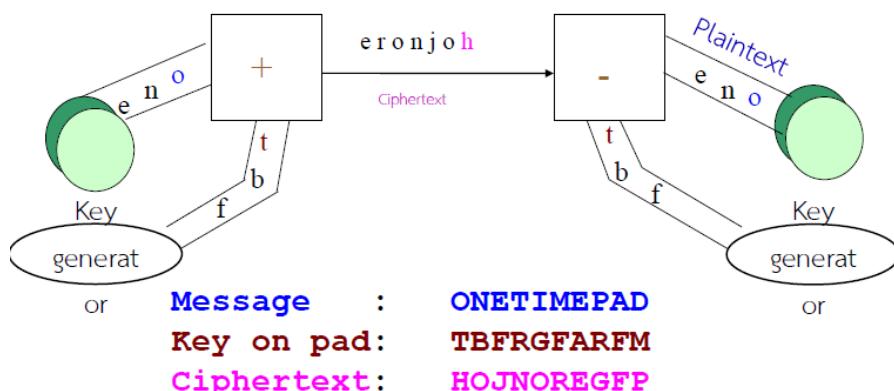
$$C_1 \leftarrow m_1 \oplus PRG(k)$$

$$C_2 \leftarrow m_2 \oplus PRG(k)$$

$$C_1 \oplus C_2 \leftarrow m_1 \oplus m_2$$

เพราะเมื่อคีย์มาเจอกันจะเหลือแค่ข้อความที่ส่ง ด้วยความที่

One-time pad



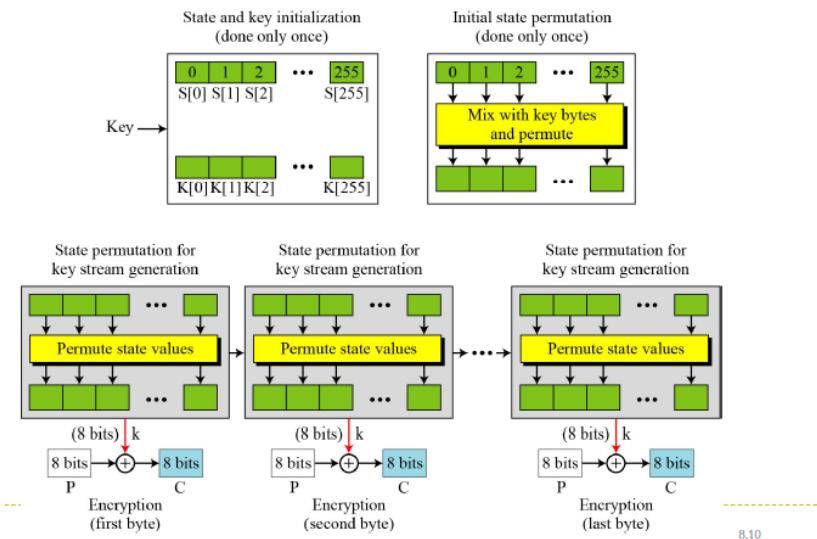
$$\begin{aligned} O + T \mod 26 &= H \\ N + B \mod 26 &= O \\ E + F \mod 26 &= J \dots \end{aligned}$$

- คีย์มีความยาวเท่า plaintext
- มีการสุ่มคีย์ด้วย seed ที่เหมือนกันระหว่างผู้ส่งและผู้รับ
- ไม่สามารถ crack ได้ = **Perfect secrecy** แต่ว่า **ไม่ Practical**
 - Perfect secrecy โดยติด้วย ciphertext only attack
 - ใช้อัลกอริธึม ใช้ตัวอักษร郁郁ฯ ยาวๆ ตามสูตร Unicity distance $N_0 = \frac{\log_2 E}{d}$
 - Practical/Computational Security: ป้องกันภัยในระยะเวลาที่กำหนดได้
 - พยายามใช้คีย์ให้ใหญ่มากพอดี
 - ถ้าจะแกะคีย์ได้ก็ต้องใช้อัลกอริธึมที่ใหญ่กว่า exponential time

RC4

- ประสิทธิภาพเกิดจาก Pseudo-random number generator (PRG)
- Encryption, Decryption ด้วยกระบวนการเดียวกัน
- Key มีได้มากถึง 256 bytes
- ทำงานกับ Byte
- ใช้งานบน WEP protocol, SSL/TLS
 - Wired Equivalent Privacy = Wireless
 - Secure Sockets Layer/Transport Layer Security (ใช้กับ data บน network ยุคก่อน)

Figure 8.10 The idea of RC4 stream cipher



RC4: Encryption/Decryption

- ▶ Byte by byte processing.

$i = 0$

$j = 0$

loop until all message encrypted

Get the next input byte

$i = (i + 1) \bmod 256$

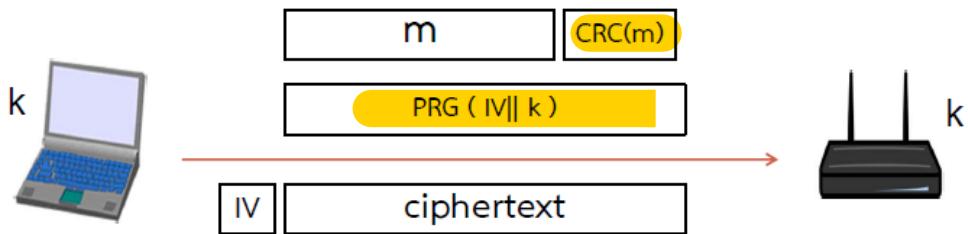
$j = (j + S[i]) \bmod 256$

$\text{swap}(S[i], S[j])$

$k = S[(S[i] + S[j]) \bmod 256]$

output: XOR k with input byte

- การใช้งานจริงของ RC4



- ถ้าให้ IV มีขนาด 24 bits
 - จะเกิดการใช้ IV ซ้ำหลังจากใช้ไปรอบบิตแล้ว ทำให้เกิด PRG เดิม ทำให้ถูกโจมตีได้
- การโจมตี RC4
 - Two-time pad: ใช้ IV ครบบิต
 - ต้องตัด output 1024 bytes แรกทิ้ง เพราะไม่ใช่ byte สุ่ม

eSTREAM

- มี 2 คลาส สำหรับใช้บน Software และ hardware

Profile 1 (SW) (128 bit keys)	Profile 2 (HW) (80 bit keys)
HC-128	Grain v1
Rabbit	Mickey v2
Salsa 20/12	Trivium
SOSEMANUK	

Source: <http://www.ecrypt.eu.org/stream/>

- ปัจจุบันมีตัวที่ถูกพัฒนาขึ้นมาจากการ Salsa20 ชื่อ ChaCha



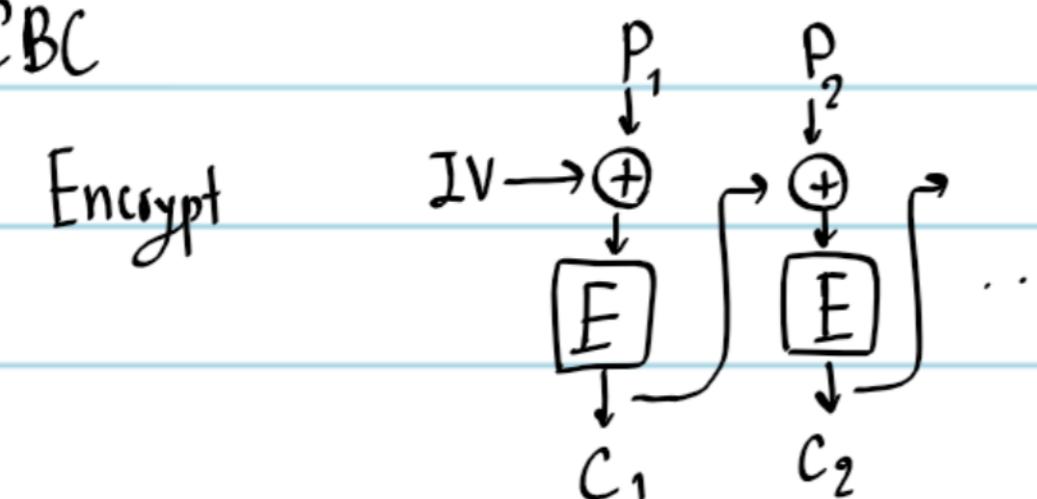
- first Secure

Tümü AZ, IV = 111 Encrypt input 000 001 010 011 100 101 110 111

ascii output 111 110 011 100 001 000 101 010

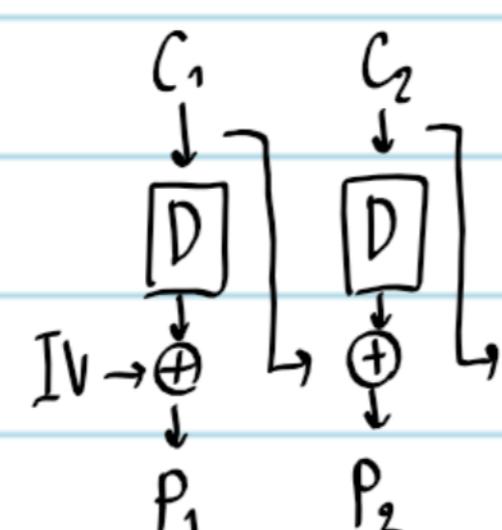
$$AZ = 65_{\text{decimal}} = 01000001_2 \quad 90 = 01011010_2$$

CBC



P ₁	010	000	010	101	101	000	<i>padding</i>
IV	111	000	111	000	000	000	
⊕	101	000	101	101	101	000	
[E]	000	111	000	000	000	111	
C	000	111	000	000	000	111	
Output	<u>00011100</u>	<u>00000001</u>	<u>11000000</u>				

Decrypt

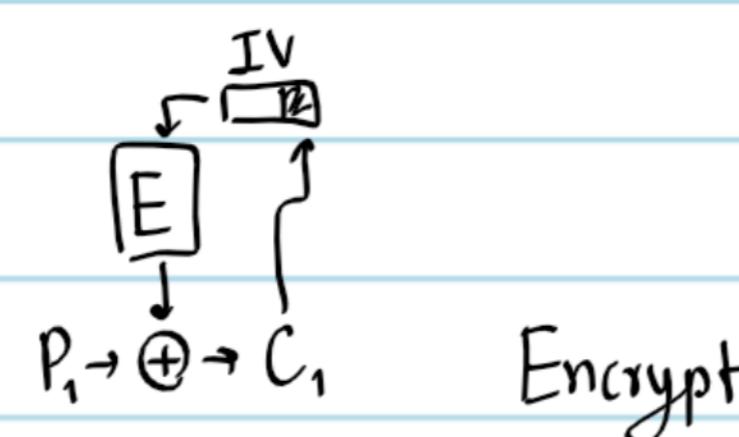


C	000	111	000	000	000	111	000	000
D	101	000	101	101	101	000	101	101
⊕	111	000	111	000	000	000	111	000
IV	111	000	111	000	000	000	111	000

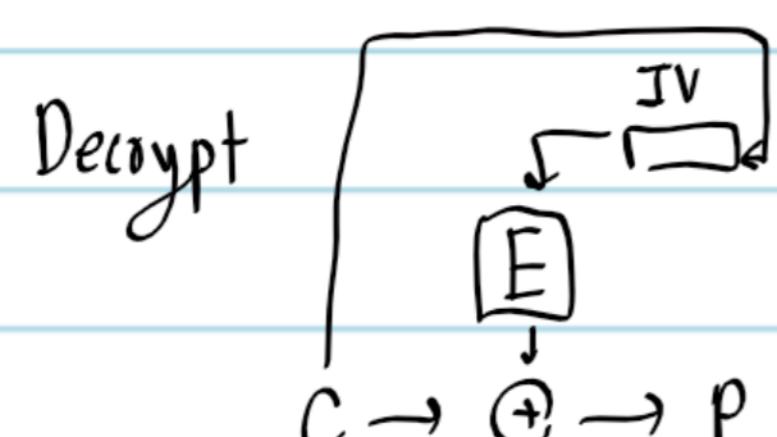
P	010	000	010	101	101	000	010	101
---	-----	-----	-----	-----	-----	-----	-----	-----

Output 0100 0001 0101 1010 00

A Z O

CFB - 2 bit 0100000101011010₂

IV	111	<u>100</u>	000	011	111	100	001	101
E	010	001	111	100	010	001	110	000
P ⊕	01	00	01	01	01	10	10	

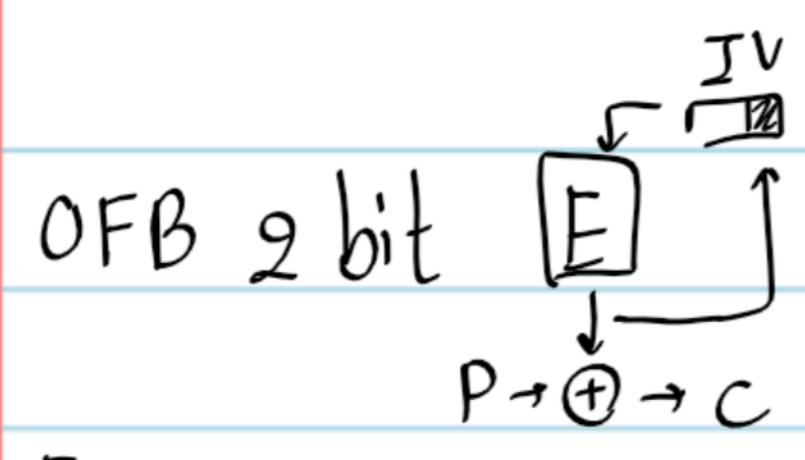
output 0000111100010110

IV	111	<u>100</u>	000	011	111	100	001	101
E	010	001	111	100	010	001	110	000
P ⊕	01	00	01	01	01	10	10	

P 01 00 00 01 01 01 10 10

0100000101011010₂

$Z \rightarrow 0101 \underline{1010} \underline{1}$

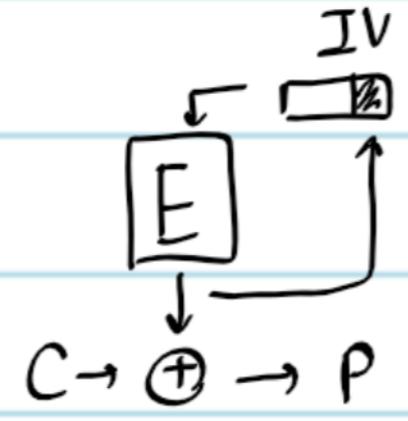


Encrypt

IV	111	→ 101	→ 100	000	011
E	010	000	001	111	100
O	⊕	00	00	- 11	10
P	01	01	10	10	10
C	00	01	10	01	00

output 0001 1010 00

Decrypt



IV	111	→ 101	→ 100	000	011
E	010	000	001	111	100
O	⊕	00	00	11	10
C	00	01	10	01	00
P	01	01	10	10	10

0101 1010 1