Crypto

Bamboofox

Who am I ?

- 毛叡 (Maojui)
- 中央資工 - 4
- Crypto
- DoubleSigma / BFS
- github : maojui

Outline

- 內容
- 背景知識
- 古典密碼學
- 現代密碼學
- Hash
- 各種工具
- CTF 水題

內容

- 一些密碼學的小常識
- 概述 CTF Crypto 常見的題目類型
- CTF Crypto 水題解法
- ~~覺得我太冗我先道歉 QAQ...~~

為什麼從 CTF 講?

因為 CTF 的 Crypto 提供了不錯的情境

- 他減弱了真實情況的安全性 ... 減到讓你可以破解
- 假設存在這些漏洞，你能不能發現、找到，能不能夠破解
- 有些是真實誤用情況：參數亂放、變數打錯 ....
- ~~讓你的論文有應用的時機 ...~~
- ~~讓你有讀論文的理由 ...~~

密碼學在幹嘛？

- 研究如何編制密碼和破譯密碼的技術
- 研究如何隱密地傳遞信息
- 研究如何破解別人的加密方法
- 研究如何玩弄數字讓其他人都無法破解
- ~~研究如何讓同學看不懂你在幹嘛、連隊友都不想理你~~
- 研究一堆跑來跑去的數字有什麼規則可以利用
- 算一堆很難的數學
- ~~CTF 裡沒人愛的那一塊~~

密碼學可以幹嘛

- 可以讓你的聊天內容被竊聽時，也不會馬上外洩
- 可以讓你的產品有點保障
- 可以降低你們公司資料外洩的風險
- ~~可以侵犯他人隱私~~
- ~~可以藏你的惡意程式~~
- ~~可以勒索別人~~

基本觀念

- 真正強的加密演算法，是在提供程式原始碼，及大量明文、密文對仍要花大量時間才能破解。通常是經過了長時間的測試、計算才成形的。（ -> 不要自己設計 .... ）

Kerckhoffs's principle 不是說密碼學演算法都必須公開，而是要確保即使公開也 不會傷害安全性

1. 實務上無法破解(或者要花費很大的成本)
2. 沒有什麼不能夠落入敵手的東西
3. 密鑰不能在記憶、傳遞與改變等方面太過困難
4. 可以數位化
5. 不能太笨重
6. 不能太難懂

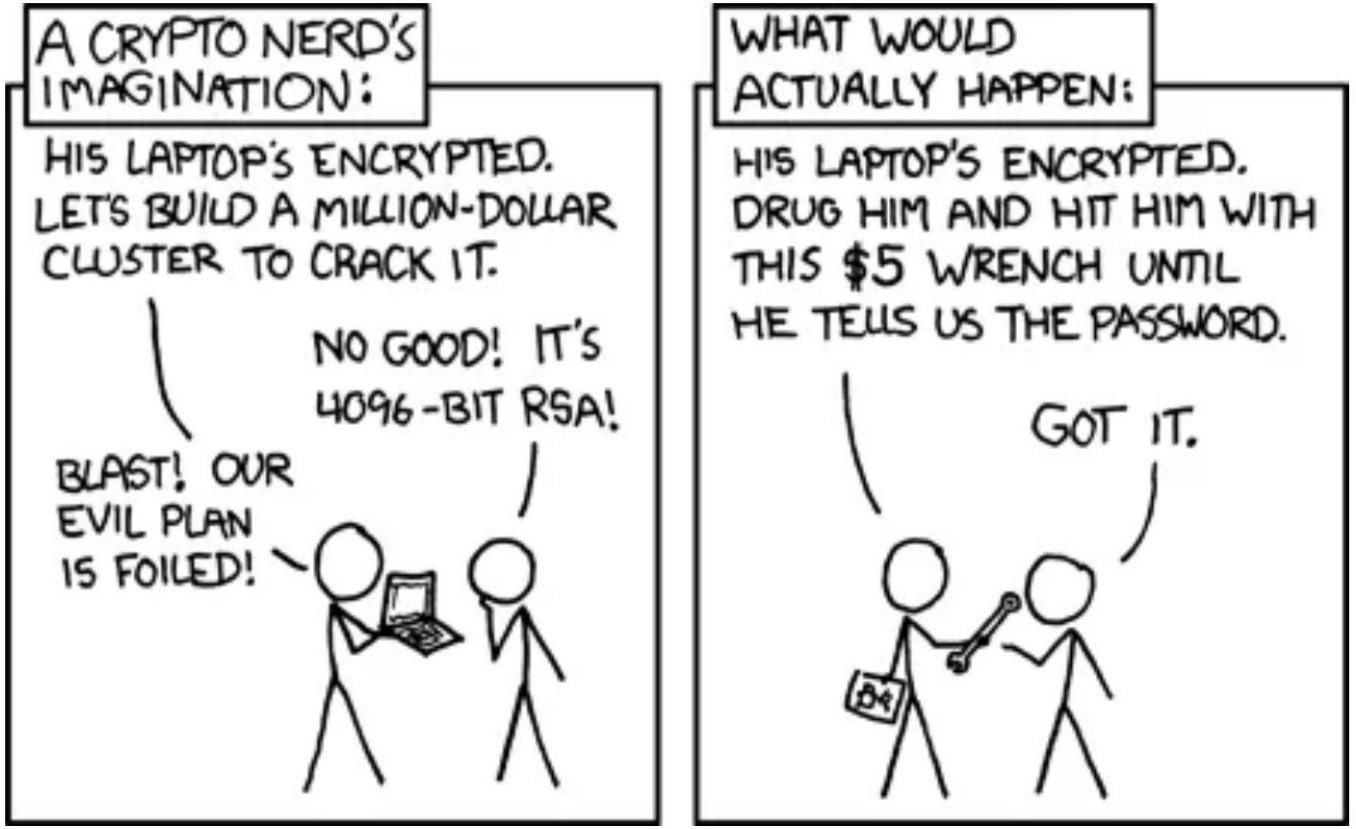
Danger

This is a “Hazardous Materials” module. You should **ONLY** use it if you’re 100% absolutely sure that you know what you’re doing because this module is full of land mines, dragons, and dinosaurs with laser guns.

```
>>> from cryptography.hazmat.backends import default_backend
>>> from cryptography.hazmat.primitives.asymmetric import rsa
>>> private_key = rsa.generate_private_key(
...     public_exponent=65537,
...     key_size=2048,
...     backend=default_backend()
... )
```

基本觀念

密碼不是靠暴力破解，而是 bypass


A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.

密碼學

- 古典密碼學 Classical Cryptography

- 現代密碼學 Modern Cryptography

Classical Cryptography

Classical Cryptography

古典密碼學，因為加密簡單 ~~(破解也簡單)~~ 實用性比較低

他們的存在僅供參考和教學用途

加解密方法和破解方法 大多隨便 google 到處都是

比較常出現在 Misc 或是送分的 Crypto

Classical Cryptography

看到他們的話

1. 根據規則或是提示或是各種資訊找到演算法名字
2. Google ...

~~或直接丟 quipquip ..~~

Classical Cryptography

- Substitution ciphers

  https://maojui.github.io/Crypto/Classical-I/

- Transposition ciphers

  https://maojui.github.io/Crypto/Classical-II/

資料來源：practical cryptography

Substitution ciphers

- 凱薩密碼（ Caesar ）
- ROT13
- 維吉尼爾加密 （ Vigenère cipher ）

凱薩密碼（Caesar）

加密 ： $Ek(m) = (m + k) \bmod 26$

解密 ： $Dk(c) = (c - k) \bmod 26$

凱薩密碼（Caesar）

1. Message = C L A S S I C A L　I S　B O R E D

凱薩密碼（Caesar）

1. Message = C L A S S I C A L　I S　B O R E D

2. key = H (8)

凱薩密碼（Caesar）

1. Message = C L A S S I C A L　I S　B O R E D

2. key = H (8)

3. C(3) + H(8) → K (11)

凱薩密碼（Caesar）

3 12 1 19 19 ……..

C L A S S I C A L I S B O R E D

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

K T I A A Q K I T Q A J W Z M L

11 20 9 1 1 …….

凱薩密碼（Caesar）

破解 … ??

反正只有 26 種 … 暴力解

https://planetcalc.com/1434/

ROT13

Key = 13 的 凱薩

維吉尼爾加密（Vigenère cipher）

	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

維吉尼爾加密（Vigenère cipher）

維吉尼爾密碼 = 一堆凱薩密碼

Message = VigenereCipher

維吉尼爾加密（Vigenère cipher）

維吉尼爾密碼	=	一堆凱薩密碼
Message	=	VigenereCipher
Key	=	code

維吉尼爾加密（Vigenère cipher）

維吉尼爾密碼	=	一堆凱薩密碼
Message	=	VigenereCipher
Key	=	code
→ Extend Key	=	codecodecodeco

維吉尼爾加密（Vigenère cipher）

維吉尼爾密碼	=	一堆凱薩密碼
Message	=	VigenereCipher
Extend Key	=	codecodecodeco（Do caesar）

維吉尼爾加密（Vigenère cipher）

維吉尼爾密碼	=	一堆凱薩密碼
Message	=	VigenereCipher
Extend Key	=	codecodecodeco
Cipher	=	XwjipsuiEwslgf

維吉尼爾加密（Vigenère cipher）

破解 … ??

南京郵電大學 Write up :
https://hackmd.io/KwQwpsDsCcoLQEYDG0AsdUgQEzgI1QAZg4RjDVskA2Q6mIA=

- Index of Coincidence (IC)
- https://en.wikipedia.org/wiki/Index_of_coincidence
- Frequency analysis :
  - calculate the frequency of unigram, bigram, ...
- Natural Language Processing

Transposition ciphers

- Rail Fence Cipher
- Square 類（ 避免 I, J 同時出現 ）

ex:

5x5 = 25 (26-25)

Juice -> Iuice

Rail Fence Cipher (key = 3)

- Example:
- m = WE ARE DISCOVERED. FLEE AT ONCE
- c = WECRL TEERD SOEEF EAOCA IVDEN

```
W . . . E . . . C . . . R . . . L . . . T . . . E
. E . R . D . S . O . E . E . F . E . A . O . C .
. . A . . . I . . . V . . . D . . . E . . . N . .
```

其他古典密碼

ADFGX Cipher	ADFGVX Cipher	Affine Cipher	Autokey Cipher
Beaufort Cipher	Enigma M3 Cipher	Foursquare Cipher	Gronsfeld Cipher
Bifid Cipher	M-209 Cipher	Playfair Cipher	Polybius Square Cipher
Porta Cipher	Railfence Cipher	Substitution Cipher	Rot13 Cipher

破解方法

Online Break !!!

1. 知道大概規則
2. 找到演算法名字
3. Google ...

或直接:

quipquip

XOR

XOR

Flipping bit, Inverting bit ….

Message = ‘a’ , Key = ‘k’ , Cipher = ‘\n’

Message ^ key = Cipher

```
          97   (01100001)   ( ‘a’ )
XOR      107   (01101011)   ( ‘k’ )

--------------------------------------------------------------------------

          10   (00001010)   ( ‘\n’ )
```

A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

XOR

Cipher ^ Key = Message

```
        10     (00001010)  ( '\n' )
XOR    107     (01101011)  ( 'k' )
-------------------------------------------------------------------------
        97     (01100001)  ( 'a' )
```

Cipher ^ Message = Key

```
        97     (01100001)  ( 'a' )
XOR     10     (00001010)  ( '\n' )
-------------------------------------------------------------------------
       107     (01101011)  ( 'k' )
```

=> Message (^ key ^ key) = Message ^ 0 = Message

=> xor eax, eax = 0

=> a ^ b = c, c ^ b = a

XOR

最簡單且省成本的加密

```
def xor(data, key):
    repeat = len(data)//len(key)+1
    return ''.join(chr(ord(a)^ord(b)) for a,b in zip(data,key* repeat ))
```

XOR

```
>> data = 'CaptureTheFlag'
>> key = 'password'

>> encrypted = xor(data,key)
'3\x00\x03\x07\x02\x1d\x170\x18\x045\x1f\x16\x08'

>> decrypted = xor(encrypted,key)
'CaptureTheFlag'
```

XOR

xortool：真D好用的工具，解不開的 xor，懶得解的 Vigenère cipher，直接丟進去即可。

One-Time-Pad @@

One-Time-Pad

用嚴密的演算法得到 truly random data 來加密 (XOR 或者 啥的)

又因為用完即丟, 就算被知道了上一段的 Key

下一段仍然無法預測。

One-Time-Pad

這種加密方法理論上絕對安全，但是實際上會有些問題

1. 安全的共享一個和明文一樣長的 key（幹嘛不直接共享明文）或亂數演算法

2. truly random 不易實現
   a. DVD加密
   b. RC4
   c. 802.11b協定

3. 一次性密碼本成本高，常常因多次使用造成問題 QQ
   a. 蘇聯士兵通訊
   b. Windows NT 的 PPTP 協定

   Many time pad -> Crib Drag

Crib Drag

EX :

m1 = "Hello World" m2 = "the program" key = "supersecret"

You get :

c1: "3b101c091d53320c000910"

c2: "071d154502010a04000419"

Crib Drag

c1 : "3b101c091d53320c000910"

c2 : "071d154502010a04000419"

guess : ‘thethetheth’ ‘hethethethe’ ‘‘ethethethet’ (XOR cipher2)

Crib Drag

cipher-text1: "3b101c091d53320c000910"

cipher-text2: "071d154502010a04000419"

guess	: ‘thethetheth’	‘hethethethe’	‘‘ethethethet’	(XOR cipher2)
out	: ‘sup1jd~lepq’	‘oxa-gubatl|’	‘bi} viopham’	(再 XOR cipher1)

Crib Drag

cipher-text1: "3b101c091d53320c000910"

cipher-text2: "071d154502010a04000419"

guess	: ‘thethetheth’	‘hethethethe’	‘‘ethethethet’	(XOR cipher2)
out	: ‘sup1jd~lepq’	‘oxa-gubatl|’	‘bi} viopham’	(再 XOR cipher1)
get	: ‘Hel8w7L`eya’			

Crib Drag

cipher-text1: "3b101c091d53320c000910"

cipher-text2: "071d154502010a04000419"

guess	: ‘thethetheth’	‘hethethethe’	‘‘ethethethet’	(XOR cipher2)
out	: ‘sup1jd~lepq’	‘oxa-gubatl|’	‘bi} viopham’	(再 XOR cipher1)
get	: ‘Hel8w7L`eya’			

Next …

guess : message1 = ‘HELLO ...’ do again … then again ...

Modern Cryptography

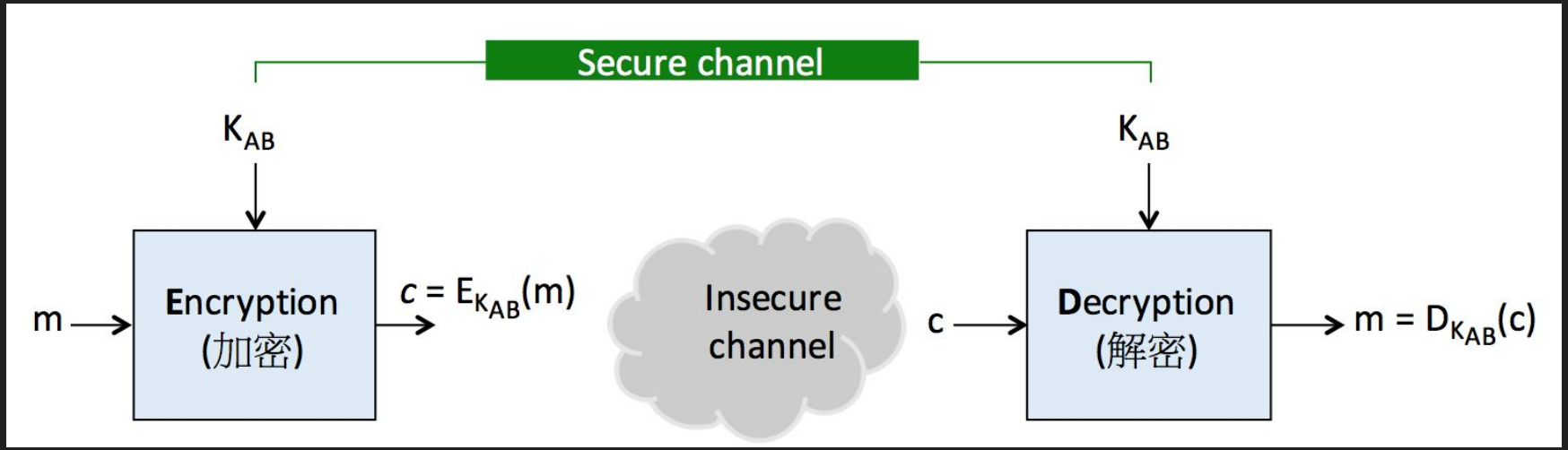
Modern Cryptography

- 對稱式加密 Symmetric
  - 區塊 (Block Ciphers) : AES、DES
  - 串流 (Stream Ciphers) : RC4

- 非對稱式加密 Asymmetric
  - Diffie-Hellman key agreement
  - Public-Private key Encryption: RSA、DSA、ECC
  - 數位簽章 Digital Signature

Symmetric

對稱式加密 Symmetric



對稱式加密 Symmetric

優點:

- 加解密速度較快
- 提供私密性(Confidential) => 由密文推不出明文

缺點:

- 不具有不可否認性 (non-repudiation) => 不能確定加密的人是誰

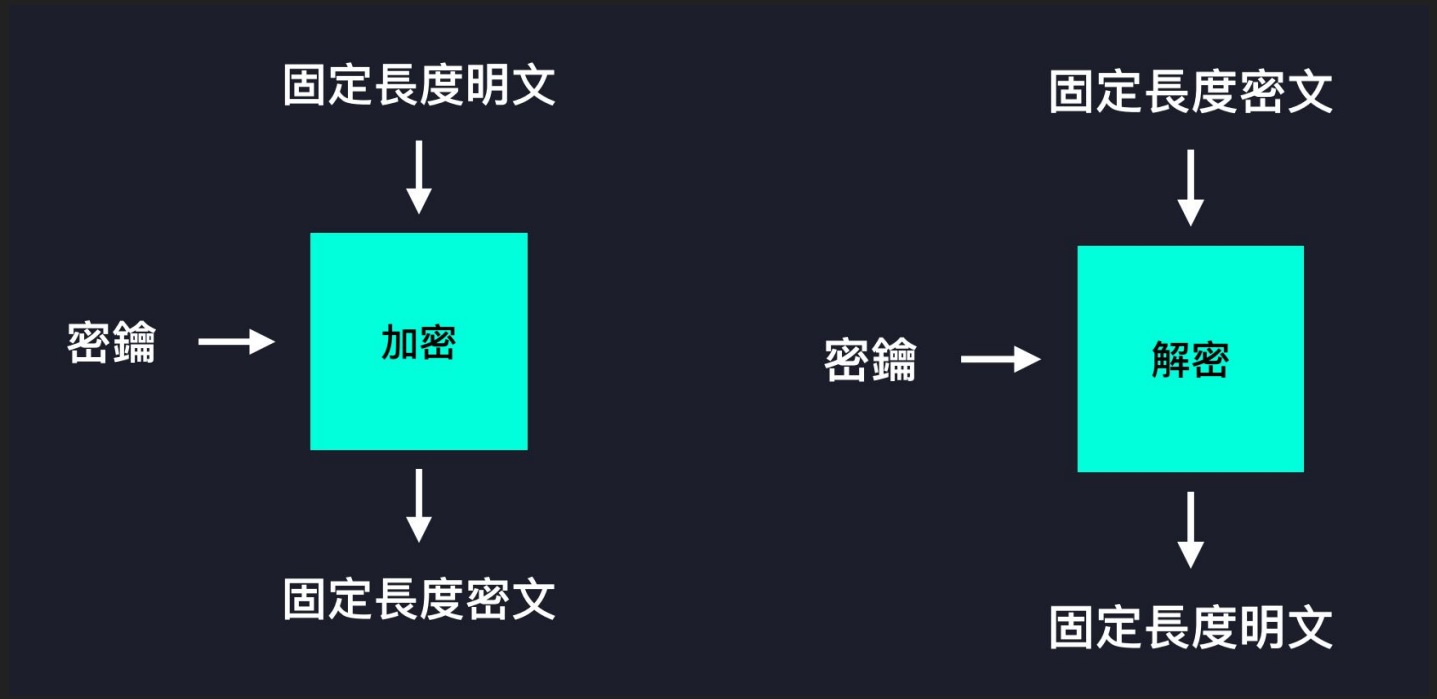
對稱式加密 Symmetric

- Block Cipher
- Stream Cipher

Block Cipher

- Data Encryption Standard (DES)
- Tripple DES (3DES)
- Advenced Encryption Standard (AES)

Block Cipher



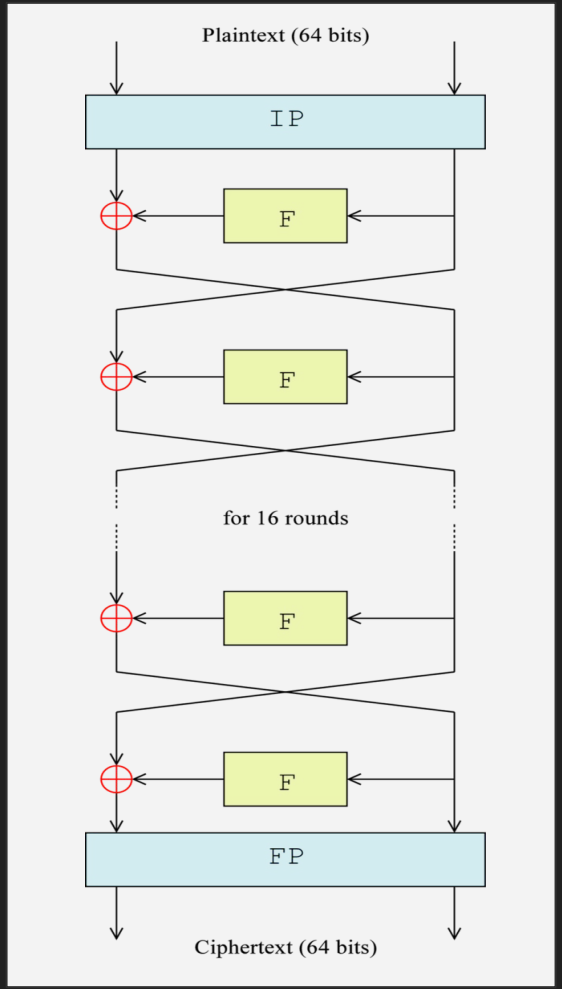
DES (Data Encryption Standard)

Weakness :

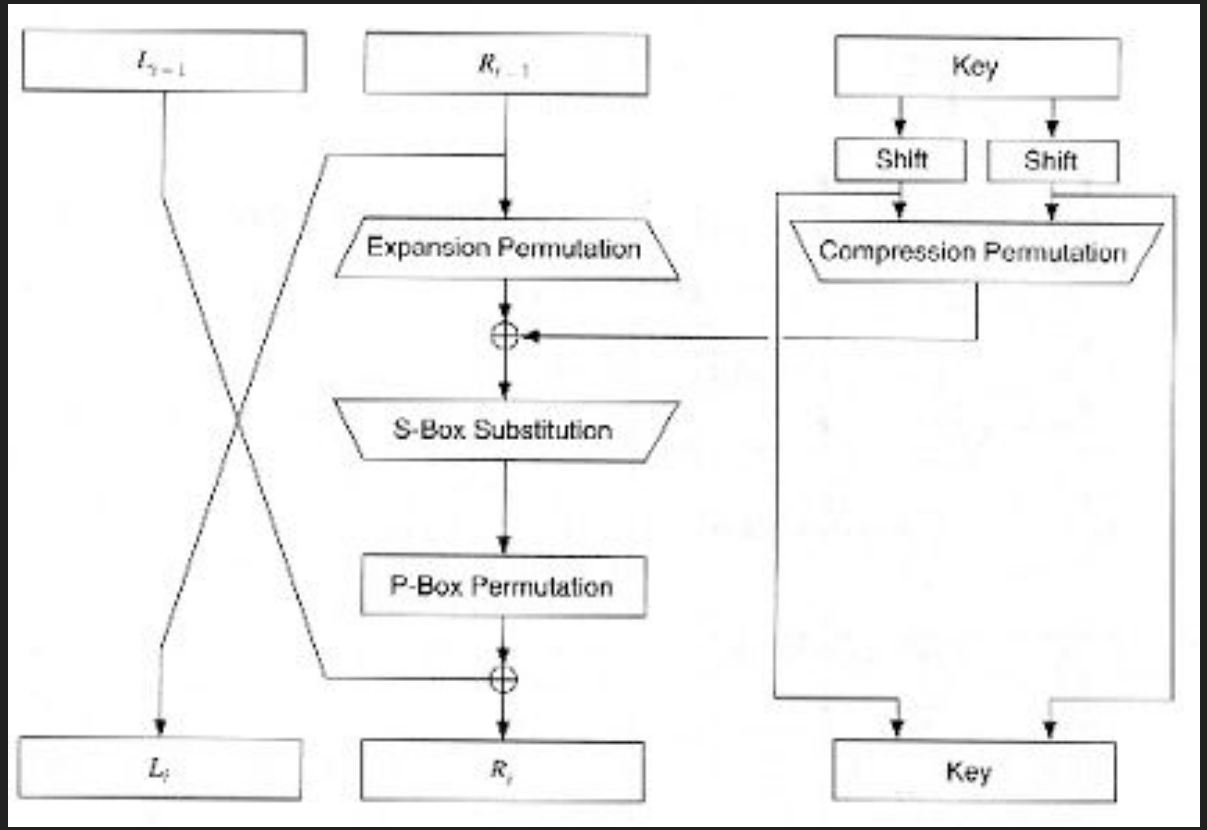
- n=64 (Too small)
- k=56 (Weak keys)

Break

- Brute-force
- linear cryptanalysis (sbox:5th bit) - 2^43 plaintxt
- DES-cracking hardware



DES (Data Encryption Standard)



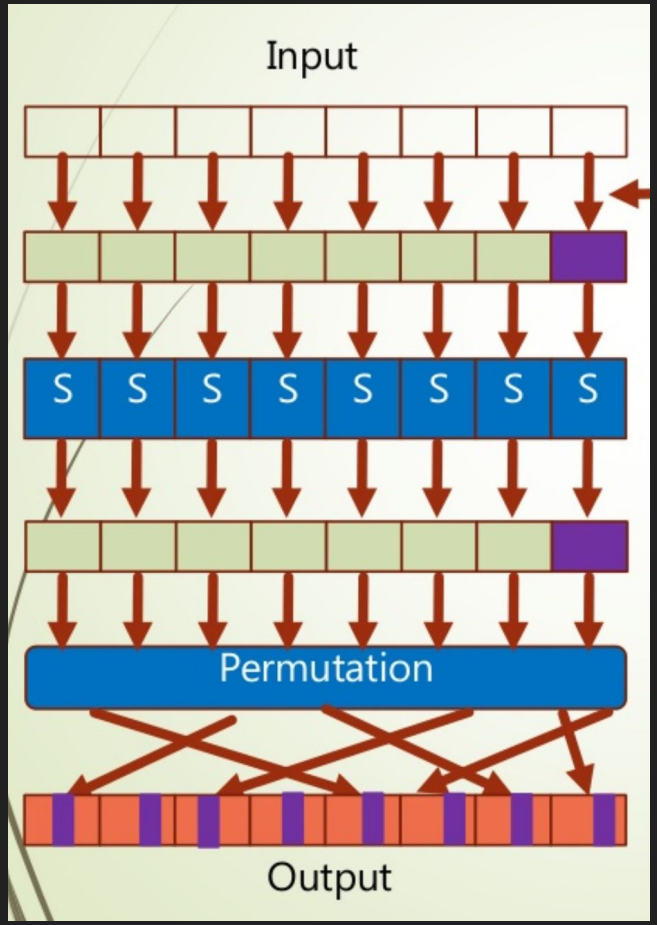
SPN (Substition Permutation Networks)

- Sbox : [14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7]

  output = sbox[input]

- Pbox : [0, 4, 8, 12,
  1, 5, 9, 13,
  2, 6, 10, 14,
  3, 7, 11, 15]

  output = [input[pbox[i]] for i in range(64)]


Input
S S S S S S S S
Permutation
Output

Tripple DES (3DES) → 修正 DES weak keys

- n = 64
- k = 56, 112, 168
- Encryption of P: EK3(DK2(EK1(P)))
- Decryption of C: DK1(EK2(DK3(C)))
- 3 keys, each is 56-bit long
  - Option 1: K1, K2, K3 are independent
  - Option 2: K1, K2 are independent, K3 = K1
  - Option 3: K1 = K2 = K3 => reduced to DES

Linear cryptanalysis

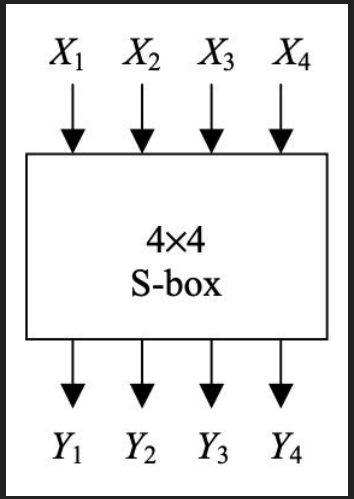
- Known-plaintext 攻擊法
- 利用大量蒐集到的明文/密文對的相關性，對加密法進行攻擊
- 明文/密文對的相關性由線性軌跡 (Linear trails) 所組成，由於線性軌跡的相關係數與Round keys的值有密切關係，透過相關係數的正負號，線性攻擊法就可以找出金鑰值。

優質論文：
https://www.engr.mun.ca/~howard/PAPERS/ldc_tutorial.pdf

Linear cryptanalysis

一個好的 Sbox 從哪裡進來跟從哪裡出去 0,1 的機率應該要一樣為 ½

如果不同, 就存在 bias



bias = 0 (½ - ½)

bias = -⅜
(⅛ - ½)

X_1	X_2	X_3	X_4	Y_1	Y_2	Y_3	Y_4	$X_2 \oplus X_3$	$Y_1 \oplus Y_3 \oplus Y_4$	$X_1 \oplus X_4$	Y_2	$X_3 \oplus X_4$	$Y_1 \oplus Y_4$
0	0	0	0	1	1	1	0	0	0	0	1	0	1
0	0	0	1	0	1	0	0	0	0	1	1	1	0
0	0	1	0	1	1	0	1	1	0	0	1	1	0
0	0	1	1	0	0	0	1	1	1	1	0	0	1
0	1	0	0	0	0	1	0	1	1	0	0	0	0
0	1	0	1	1	1	1	1	1	1	1	1	1	0
0	1	1	0	1	0	1	1	0	1	0	0	1	0
0	1	1	1	1	0	0	0	0	1	1	0	0	1
1	0	0	0	0	0	1	1	0	0	1	0	0	1
1	0	0	1	1	0	1	0	0	0	0	0	1	1
1	0	1	0	0	1	1	0	1	1	1	1	1	0
1	0	1	1	1	1	0	0	1	1	0	1	0	1
1	1	0	0	0	1	0	1	1	1	1	1	0	1
1	1	0	1	1	0	0	1	1	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1	0	1	0
1	1	1	1	0	1	1	1	0	0	0	1	0	1

Input Sum	Output Sum 0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	+8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	−2	−2	0	0	−2	+6	+2	+2	0	0	+2	+2	0	0
2	0	0	−2	−2	0	0	−2	−2	0	0	+2	+2	0	0	−6	+2
3	0	0	0	0	0	0	0	0	+2	−6	−2	−2	+2	+2	−2	−2
4	0	+2	0	−2	−2	−4	−2	0	0	−2	0	+2	+2	−4	+2	0
5	0	−2	−2	0	−2	0	+4	+2	−2	0	−4	+2	0	−2	−2	0
6	0	+2	−2	+4	+2	0	0	+2	0	−2	+2	+4	−2	0	0	−2
7	0	−2	0	+2	+2	−4	+2	0	−2	0	+2	0	+4	+2	0	+2
8	0	0	0	0	0	0	0	0	−2	+2	+2	−2	+2	−2	−2	−6
9	0	0	−2	−2	0	0	−2	−2	−4	0	−2	+2	0	+4	+2	−2
A	0	+4	−2	+2	−4	0	+2	−2	+2	+2	0	0	+2	+2	0	0
B	0	+4	0	−4	+4	0	+4	0	0	0	0	0	0	0	0	0
C	0	−2	+4	−2	−2	0	+2	0	+2	0	+2	+4	0	+2	0	−2
D	0	+2	+2	0	−2	+4	0	+2	−4	−2	+2	0	+2	0	0	+2
E	0	+2	+2	0	−2	−4	0	+2	−2	0	0	−2	−4	+2	−2	0
F	0	−2	−4	−2	−2	0	+2	0	0	−2	+4	−2	−2	0	+2	0

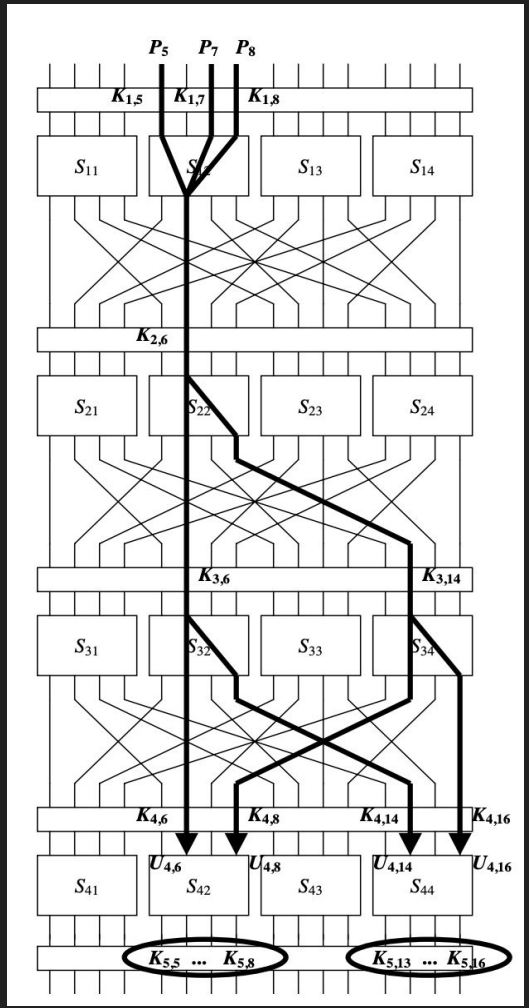
Linear cryptanalysis

一層的 bias 可能很大, 但是也可以在幾 Round 之後減小

右圖的 bias 相當於
½ +2^3(¾ - ½)(¼ − ½)^3 = 15/32 = -1/32

$U_{4,6} \oplus U_{4,8} \oplus U_{4,14} \oplus U_{4,16} \oplus P_5 \oplus P_7 \oplus P_8 \oplus \Sigma_K = 0$

$U_{4,6} \oplus U_{4,8} \oplus U_{4,14} \oplus U_{4,16} \oplus P_5 \oplus P_7 \oplus P_8 = 0$ 的機率為 1/32

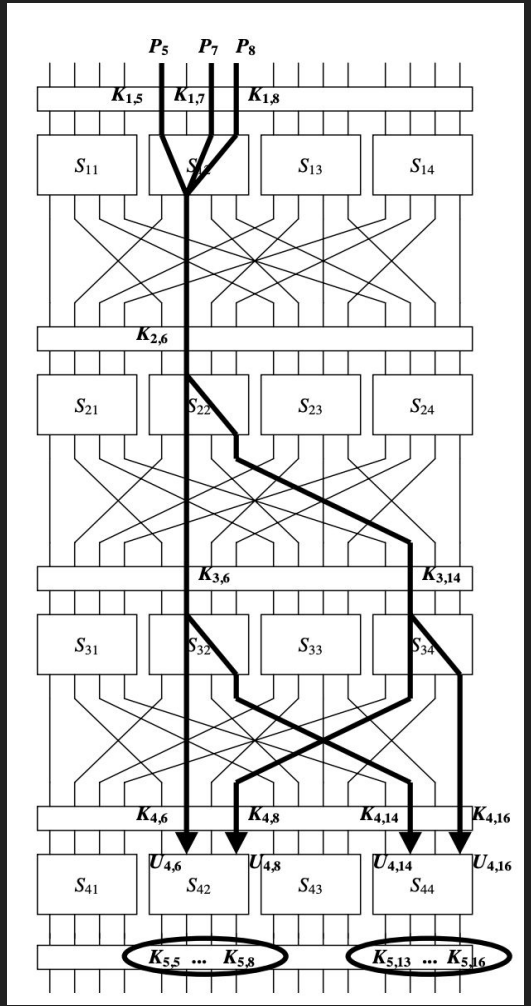


Linear cryptanalysis

亂猜 key 的中間 2bytes

把該 key 加密過的 plaintext 撤過去計算

有誰的 bias 為 1/32，就猜中這倆 bytes 了

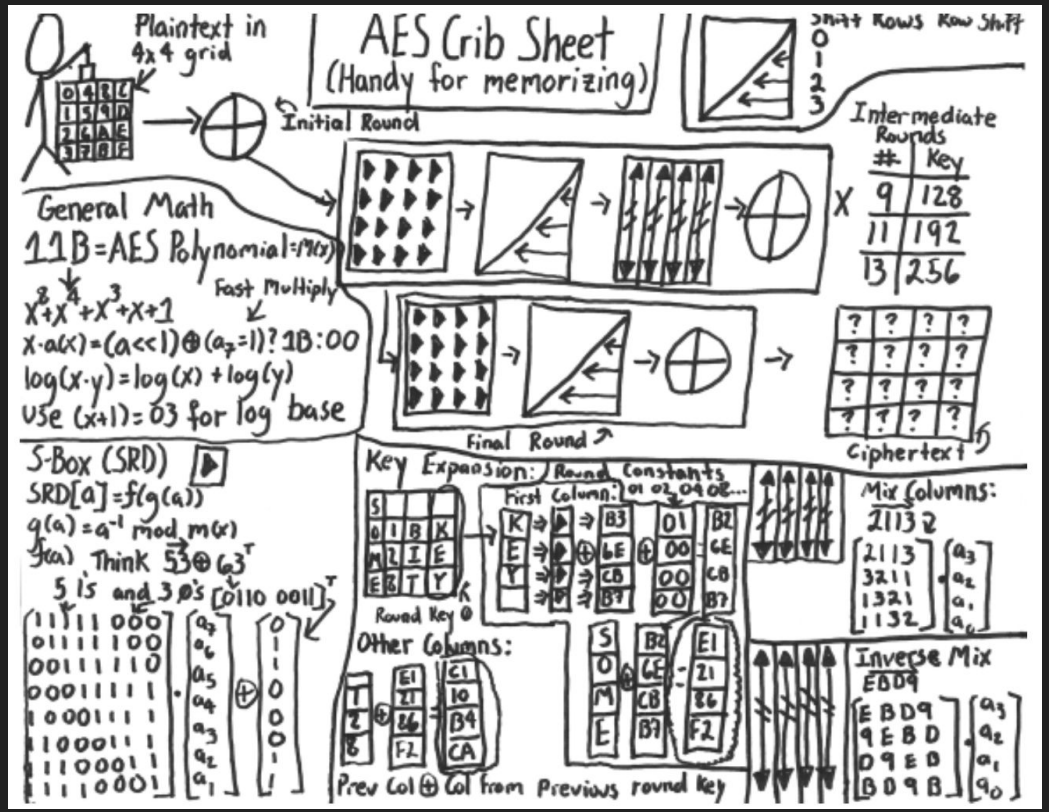


partial subkey $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	| bias |	*partial subkey* $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	| bias |
1 C	0.0031	2 A	0.0044
1 D	0.0078	2 B	0.0186
1 E	0.0071	2 C	0.0094
1 F	0.0170	2 D	0.0053
2 0	0.0025	2 E	0.0062
2 1	0.0220	2 F	0.0133
2 2	0.0211	3 0	0.0027
2 3	0.0064	3 1	0.0050
2 4	**0.0336**	3 2	0.0075
2 5	0.0106	3 3	0.0162
2 6	0.0096	3 4	0.0218
2 7	0.0074	3 5	0.0052
2 8	0.0224	3 6	0.0056
2 9	0.0054	3 7	0.0048

Advenced Encryption Standard (AES)

- n = 128
- k = 128, 192, 256
- Encrypt & Decrypt :
  - AddRoundKey
  - SubBytes
  - SiftRows
  - MixColumns

Advenced Encryption Standard (AES)

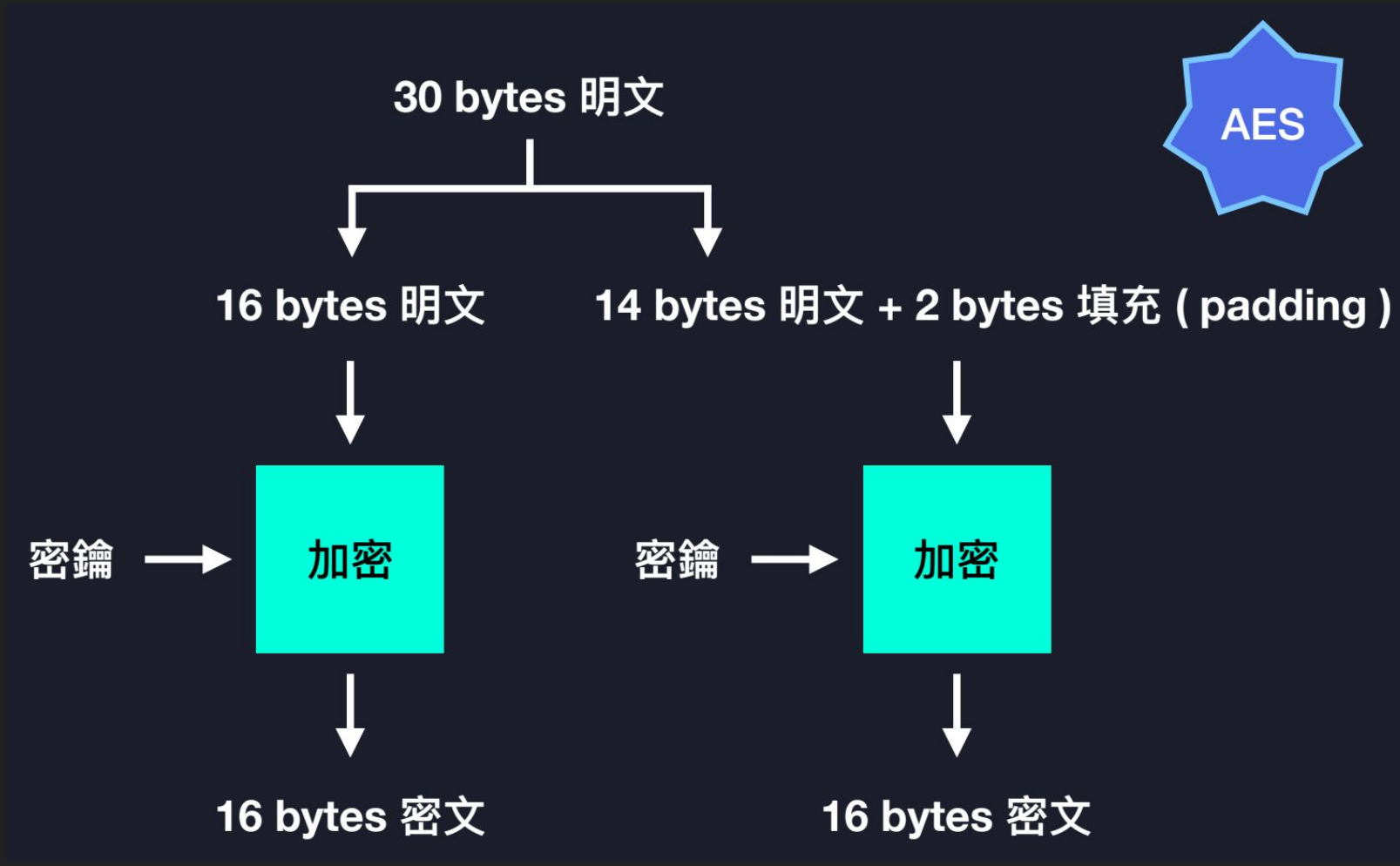


Block Cipher

一次加密一個 Block

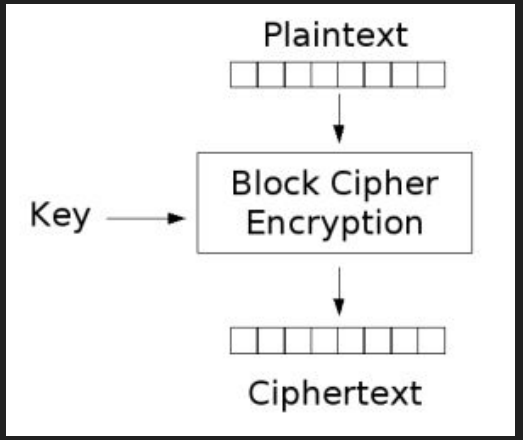
那如果明文很長呢 QAQ ??　→ 切成好幾個 Block

切完有剩呢 OAQ??　→ 加 Padding 補齊


30 bytes 明文
AES
16 bytes 明文
14 bytes 明文 + 2 bytes 填充 (padding)
密鑰
加密
密鑰
加密
16 bytes 密文
16 bytes 密文

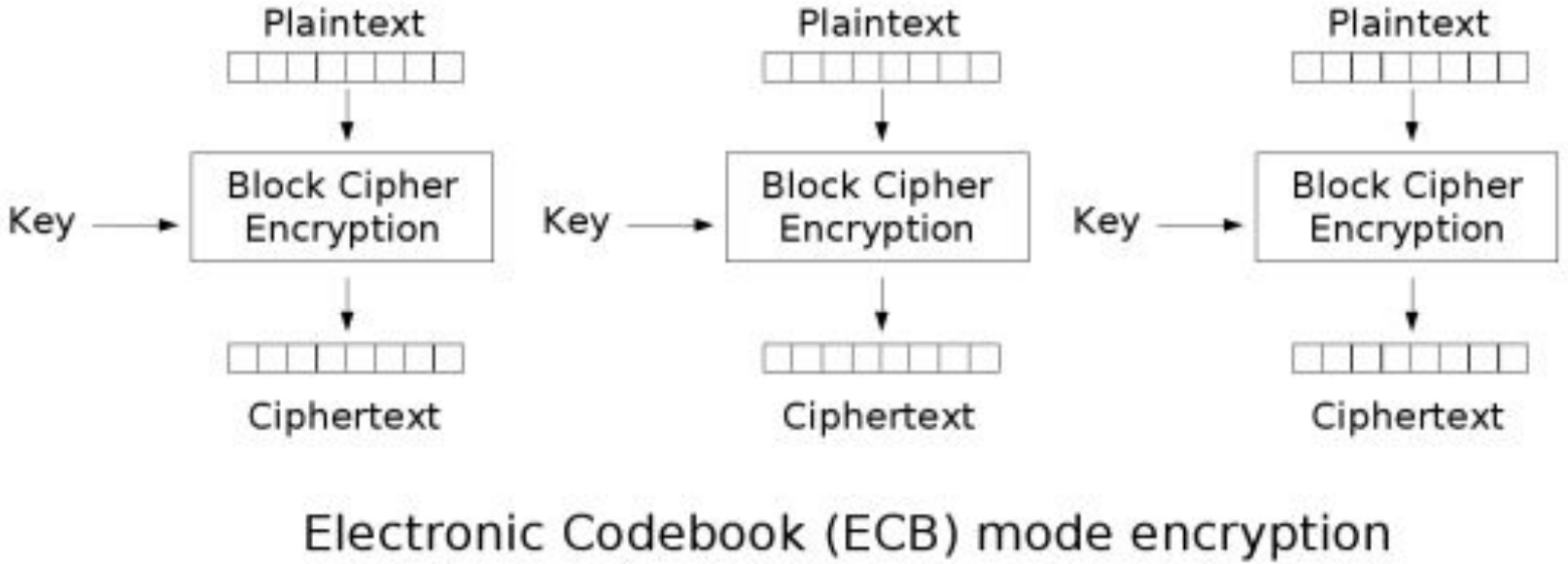
Mode of Operation

- 區塊加密：
  - ECB　(Electronic codebook)
  - CBC　(Cipher-block chaining)
  - CTR　(Counter)
  - PCBC　(Propagating cipher-block chaining)
  - …



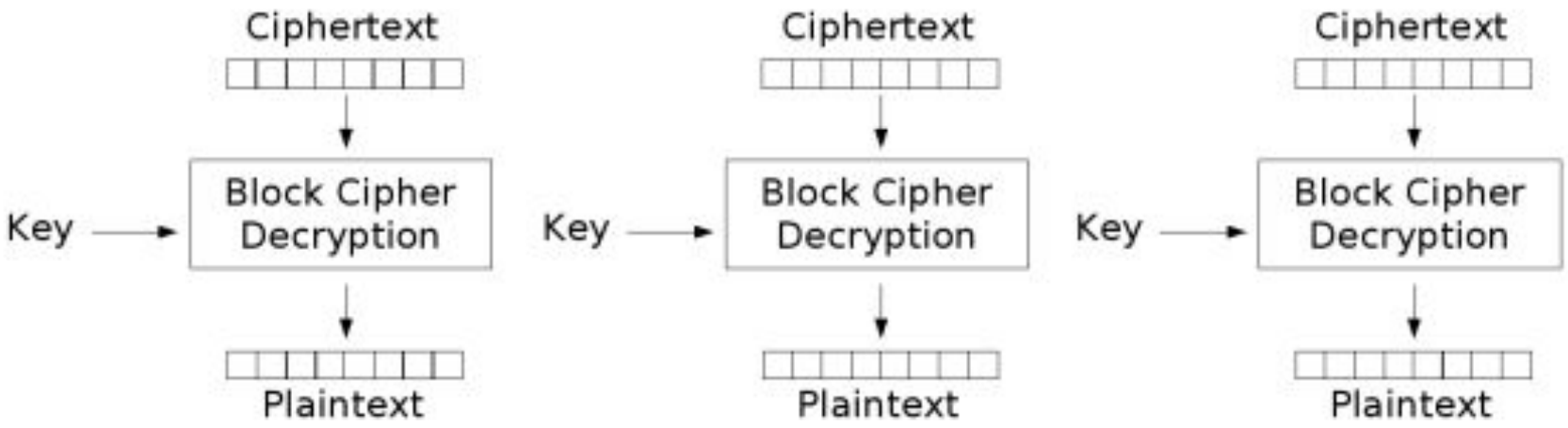
ECB

ECB



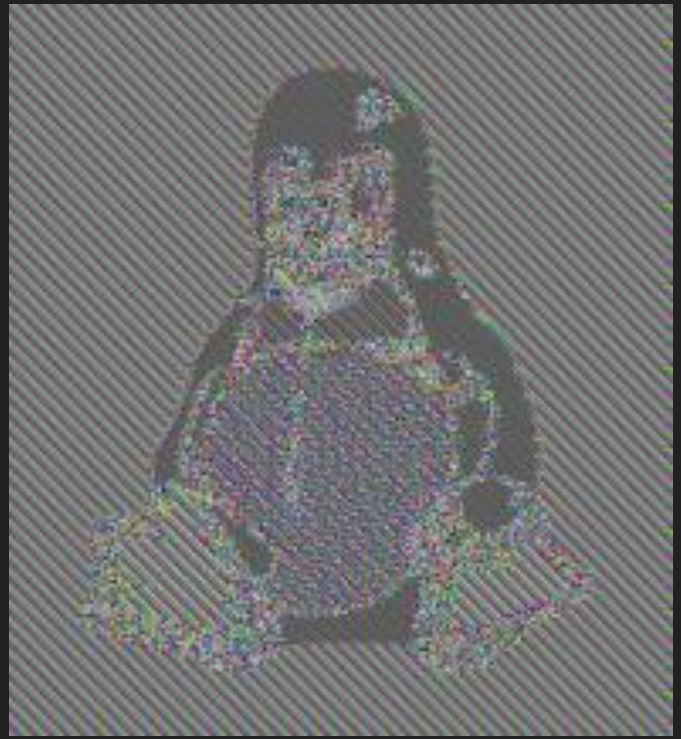
Electronic Codebook (ECB) mode encryption

ECB



Electronic Codebook (ECB) mode decryption

ECB



ECB

- Encryption Oracle + Cut-and-paste

ECB

Encryption Oracle Attack :

- E : for ECB mode， E_k(P || A || S)
- 你不知道 key, Padding, secret 但是可以控制 A
- Oracle : return EK(P || A || S)

ECB

Example :

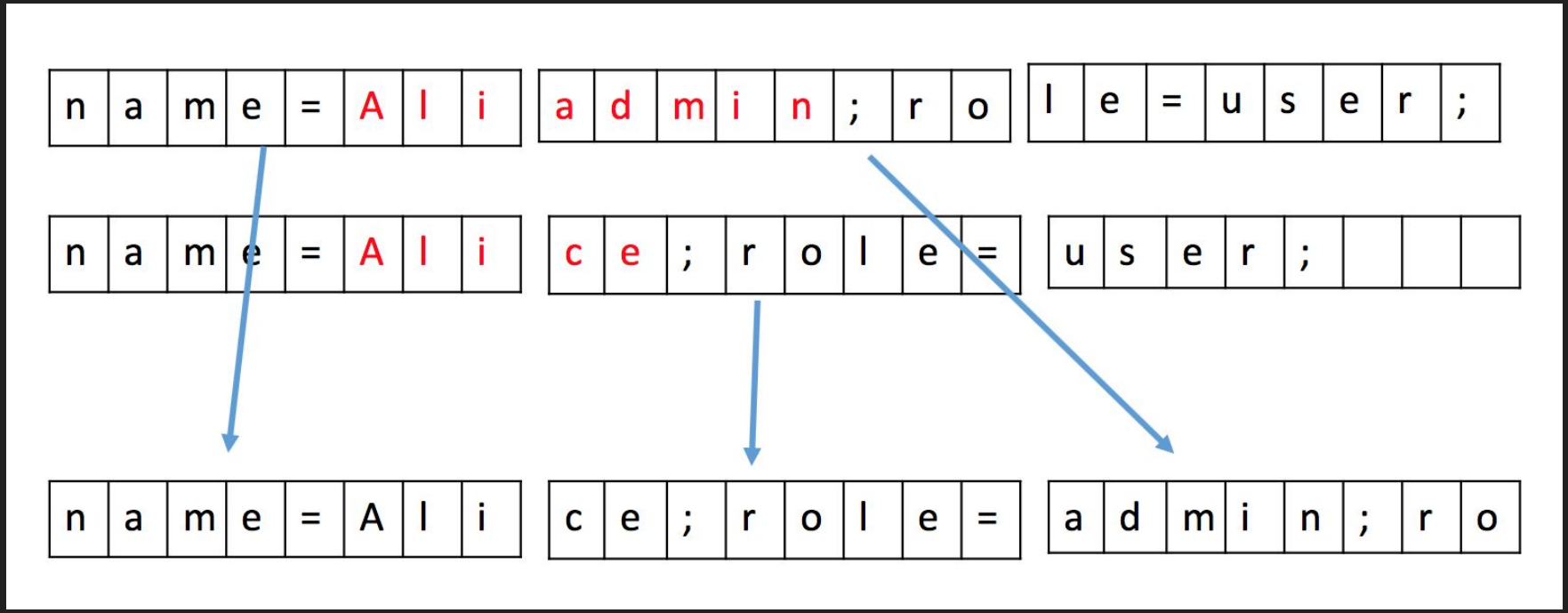
- A cookie is EK(name = Alice || role = user), username is alice

  Ek(P | name = Alice || role = user | S)

- user-controllable but not the role

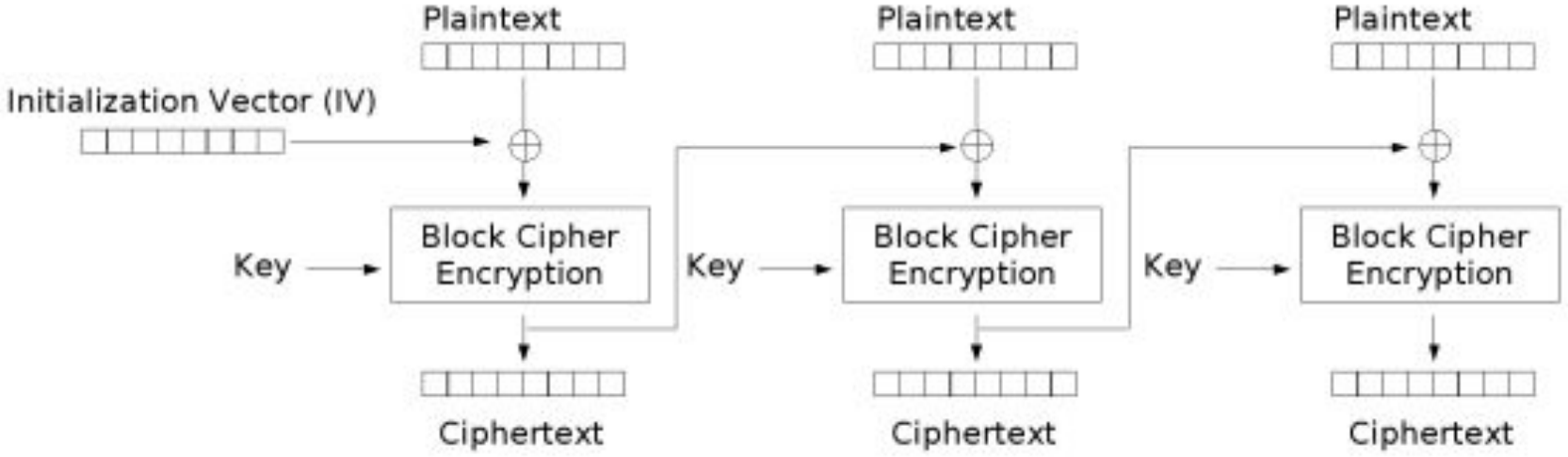
- You want to obtain EK(name = Alice || role = admin)

ECB


name=Ali
admin;ro
le=user;
name=Ali
ce;role=
user;
name=Ali
ce;role=
admin;ro

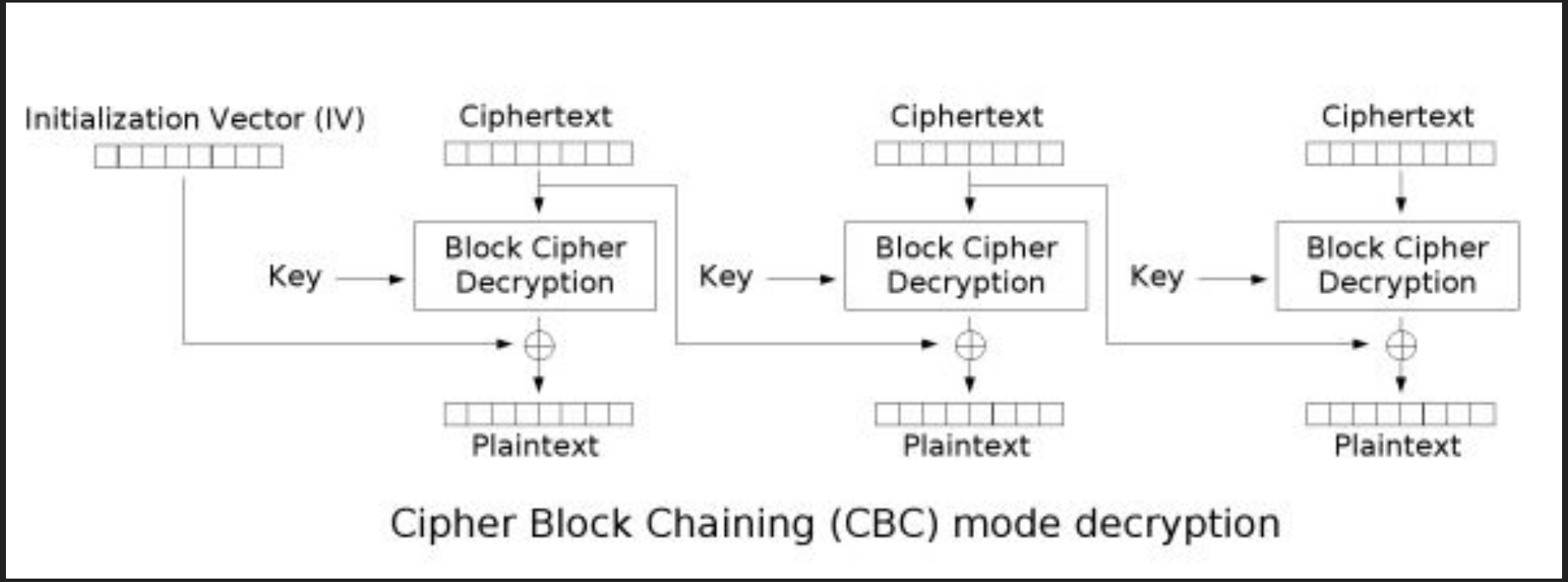
CBC

CBC

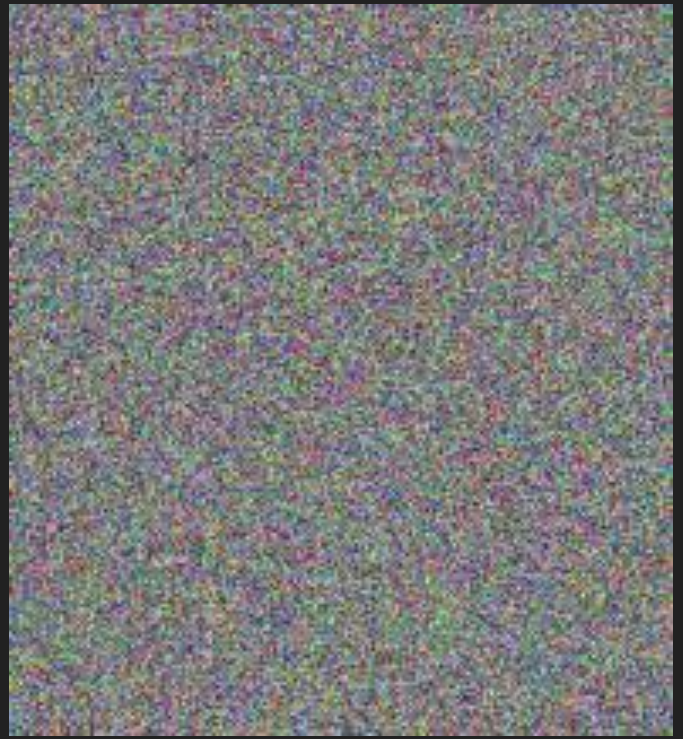


Cipher Block Chaining (CBC) mode encryption

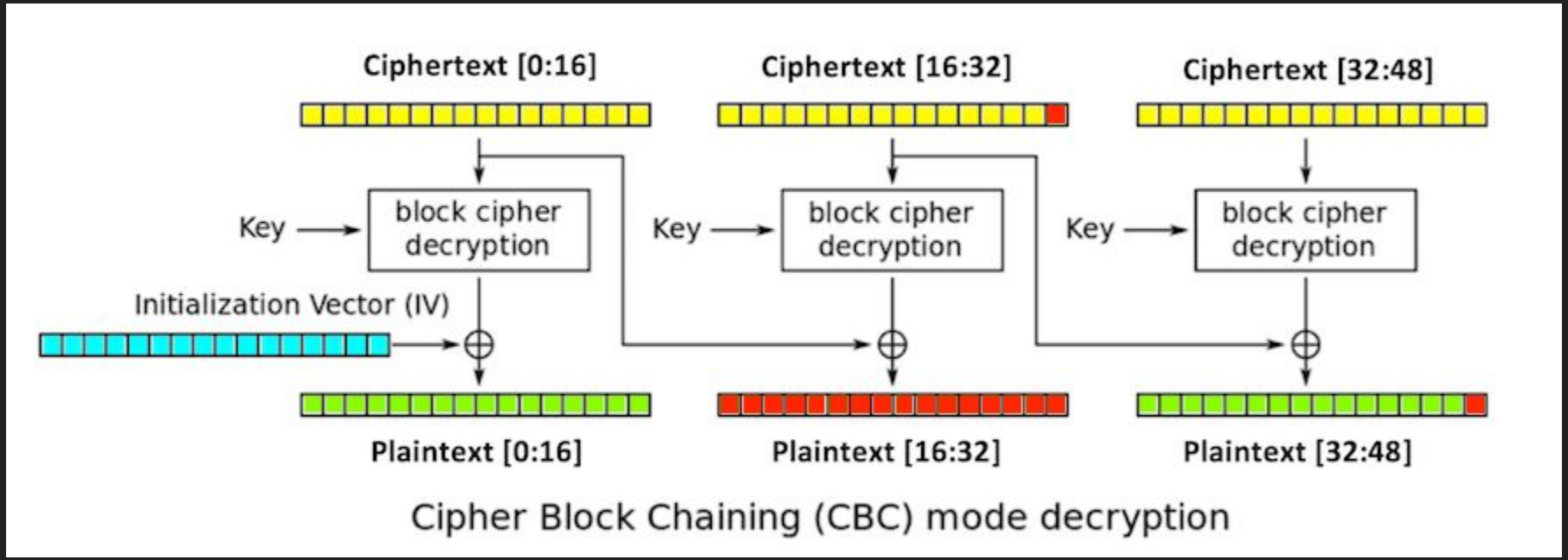
CBC


Initialization Vector (IV)
Ciphertext
Ciphertext
Ciphertext
Key
Block Cipher Decryption
Key
Block Cipher Decryption
Key
Block Cipher Decryption
Plaintext
Plaintext
Plaintext
Cipher Block Chaining (CBC) mode decryption

CBC



Bit flipping attacks



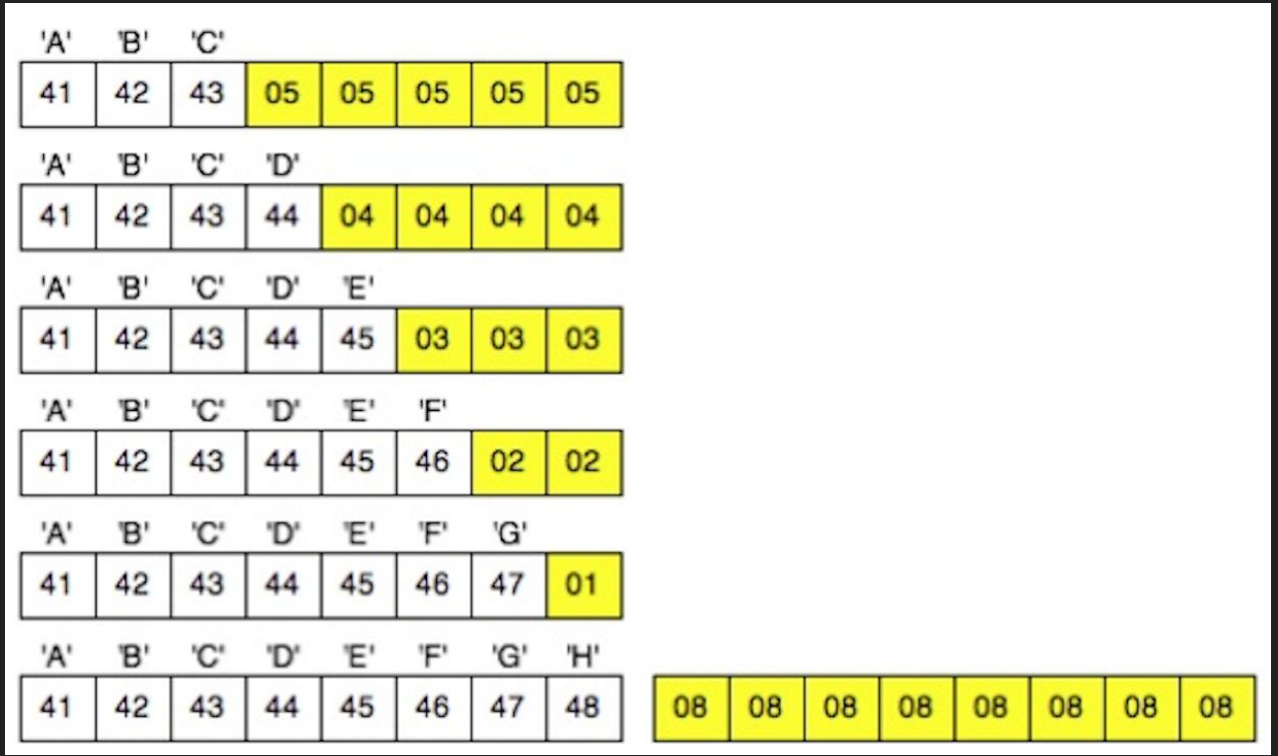
CBC

Padding Oracle Attack => 利用 PKCS #7 Padding 挖出 明文

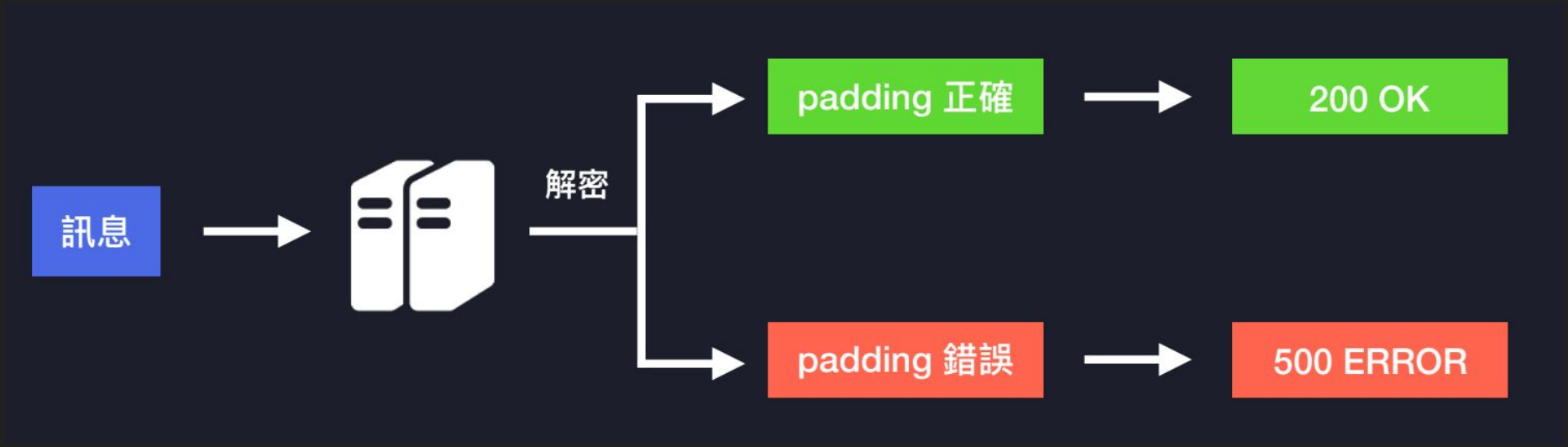
加解密 圖

我的小筆記

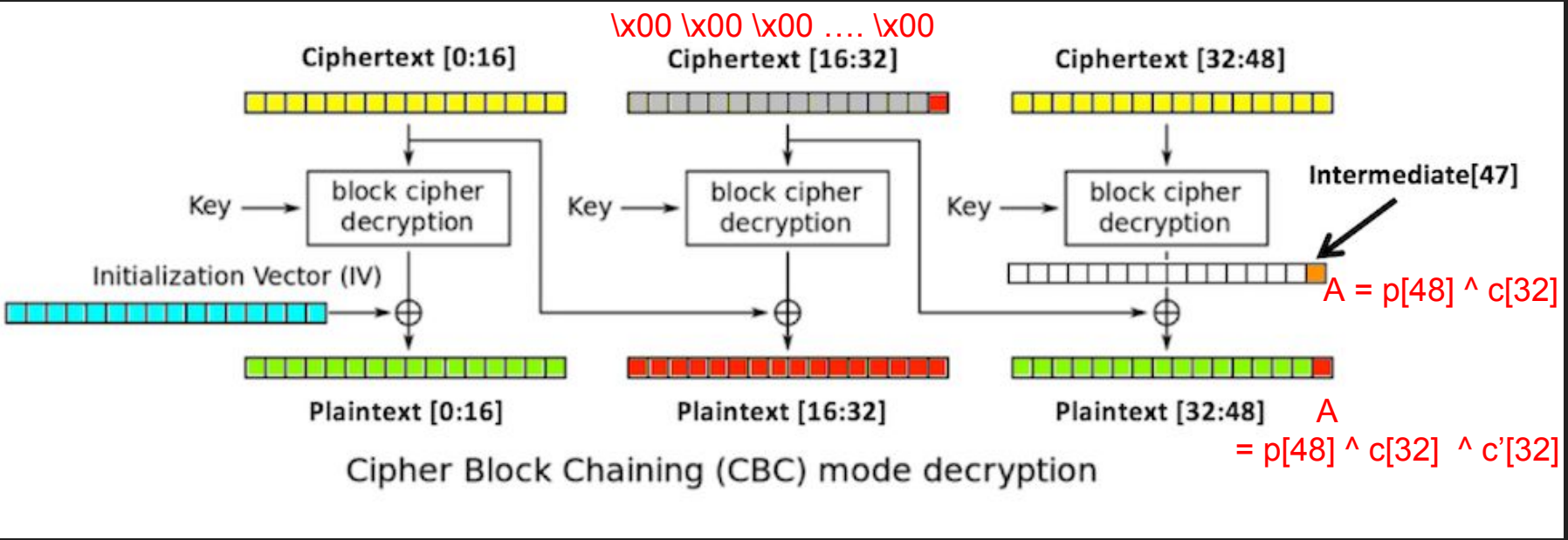
PKCS #7 Padding



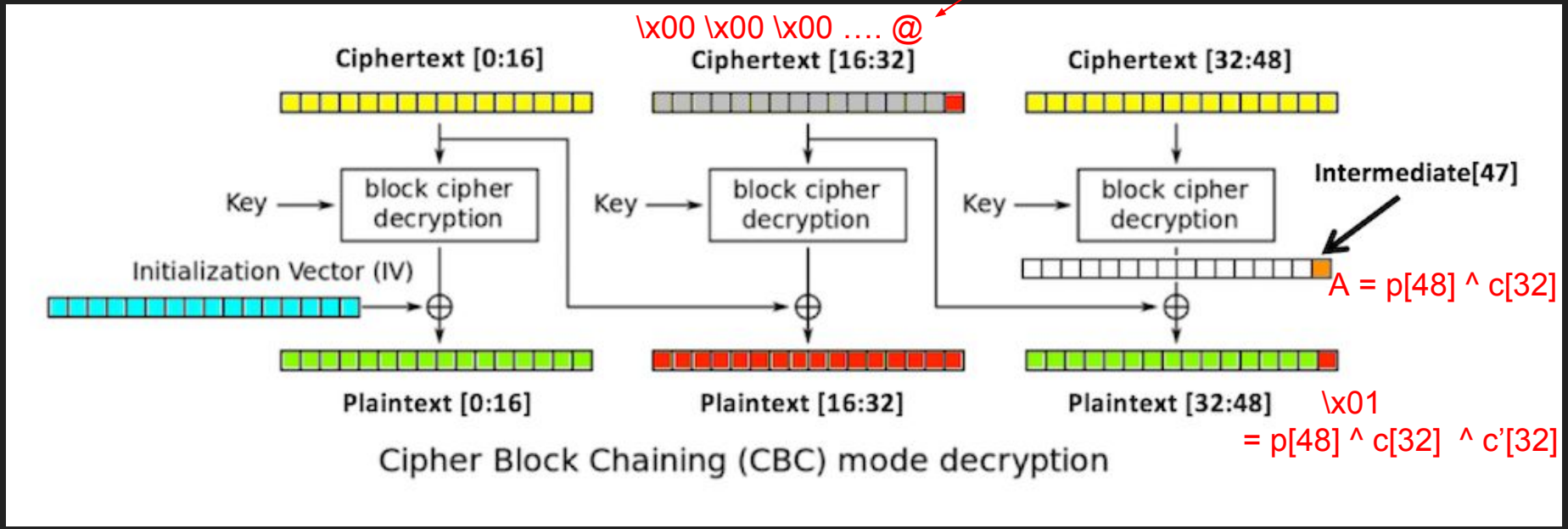
Padding Oracle



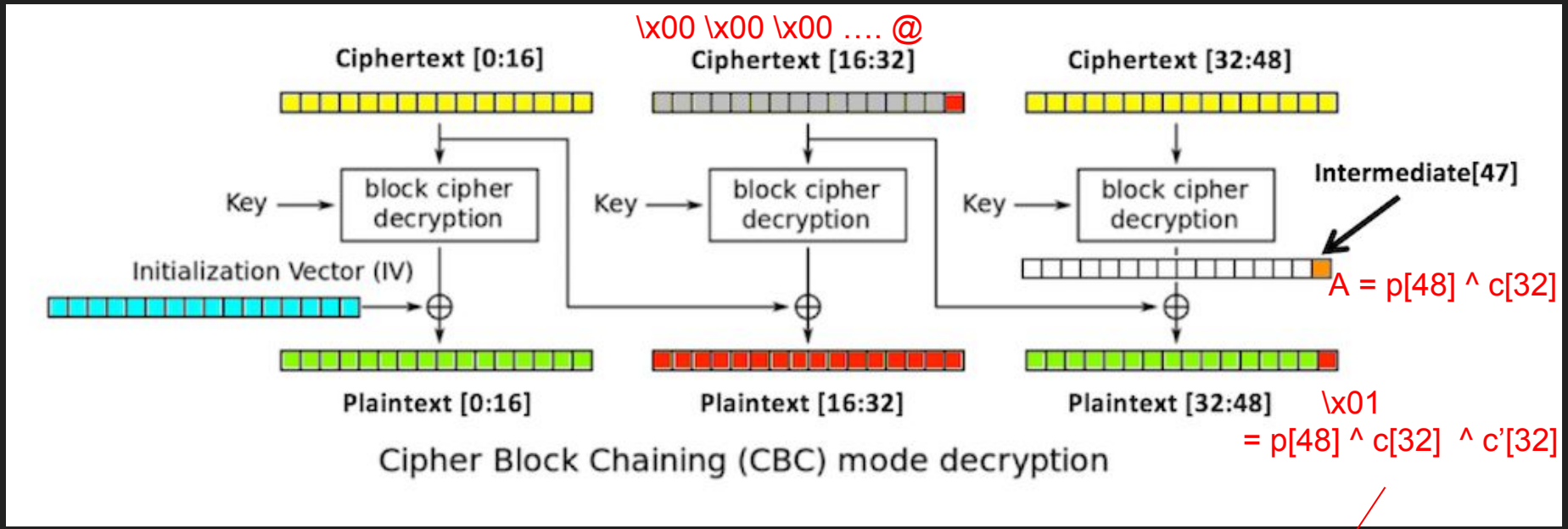
Padding Oracle



Padding Oracle



Padding Oracle



下一格

找到 p[48]
c'[32] = p[48] ^ c[32] ^ \x02

\x01 = p[48] ^ c[32] ^ c'[32]
p[48] = \x01 ^ c[32] ^ c'[32]

Padding Oracle

三層迴圈 :

猜 0 - 255 直到猜出一個 byte

一次解出一個 byte 直到解完一個 block

一次解出一個 block 直到解完所有 blocks

- 第一個區塊 :

我們需要前一個區塊來解目前的區塊

所以我們需要知道原始 IV 和能夠操控 IV 才能解出第一個區塊

Padding Oracle

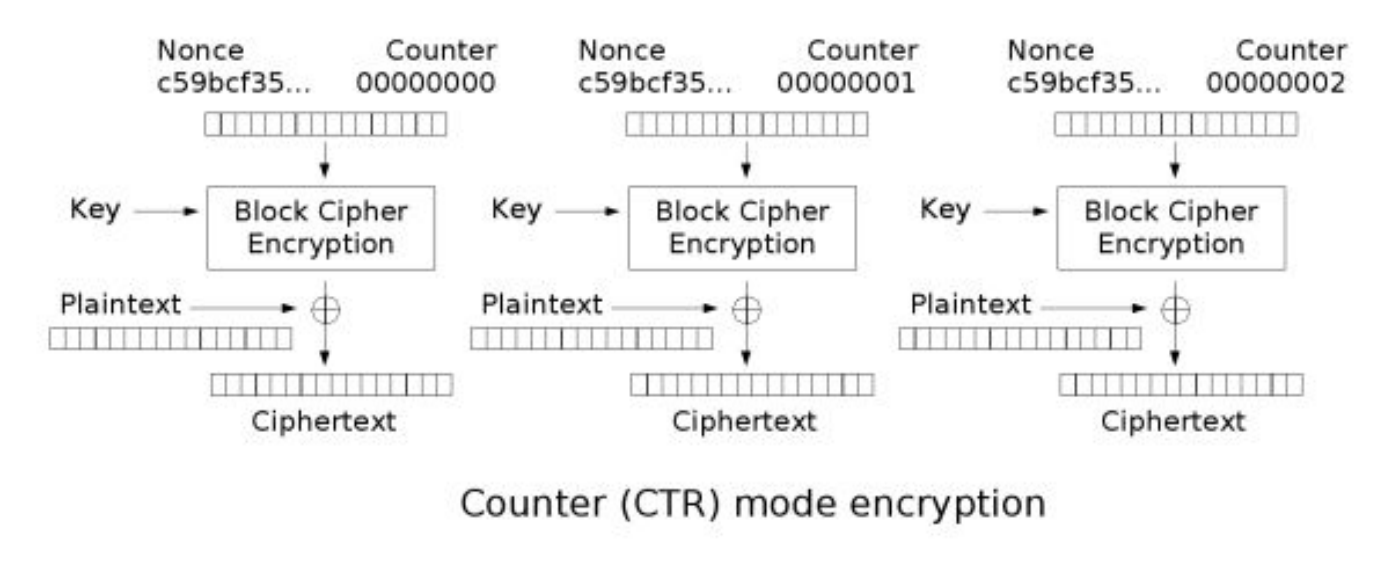
嘗試次數 :

- 解出一個 byte 最多需要 256 次嘗試
- 解出一個 block (16 bytes) 最多需要 4096 次嘗試
- 看他有幾個 Block … 一個一個解回去

CTR

CTR

- counter 從 0 開始遞加
- nonce 為隨機生成 (IV)



對稱式加密 Symmetric

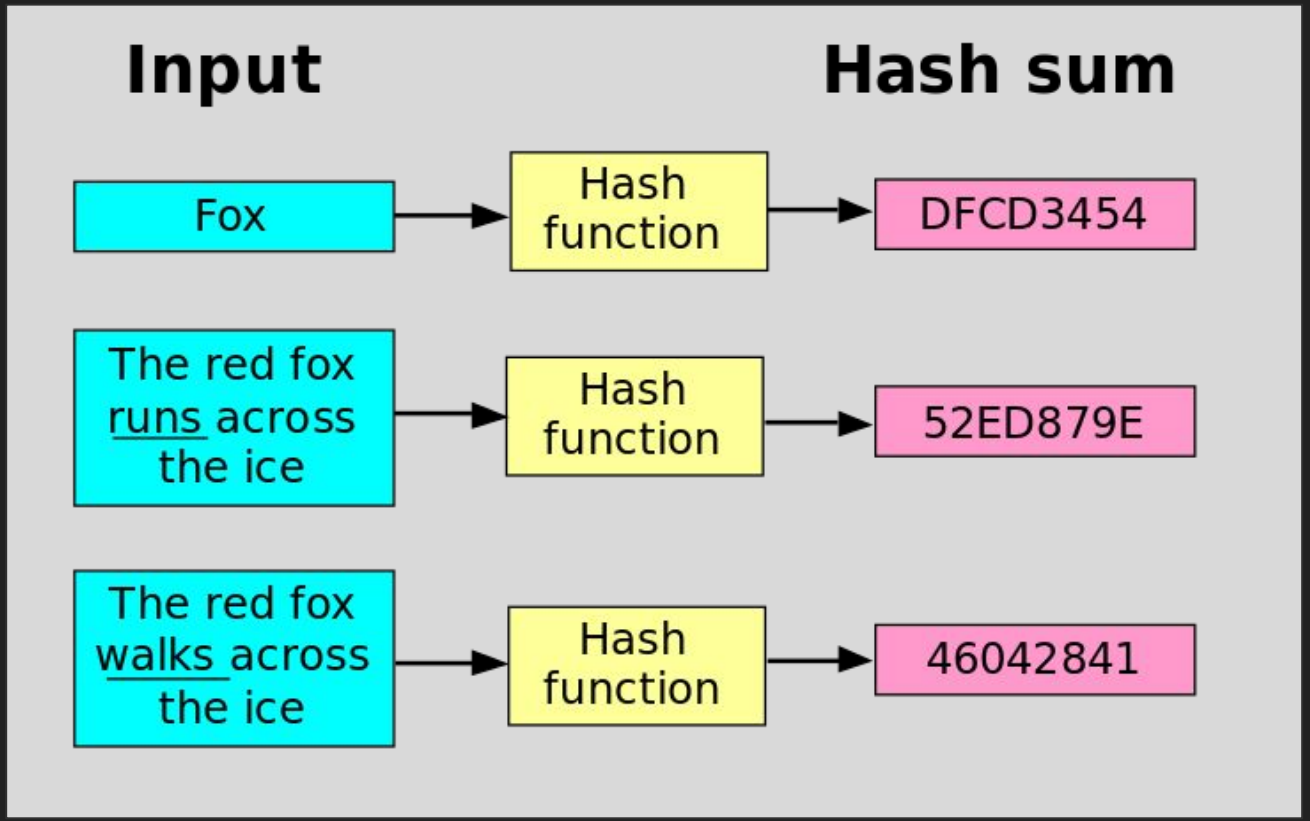
- Stream Cipher : (ex : RC4)

  A (synchronous) stream cipher is an algorithm which maps some fixed-length key to an arbitrary-length key-stream

  => 很像 One-Time-Pad …

Hash 夤乱入

Hash Function



Hash Function

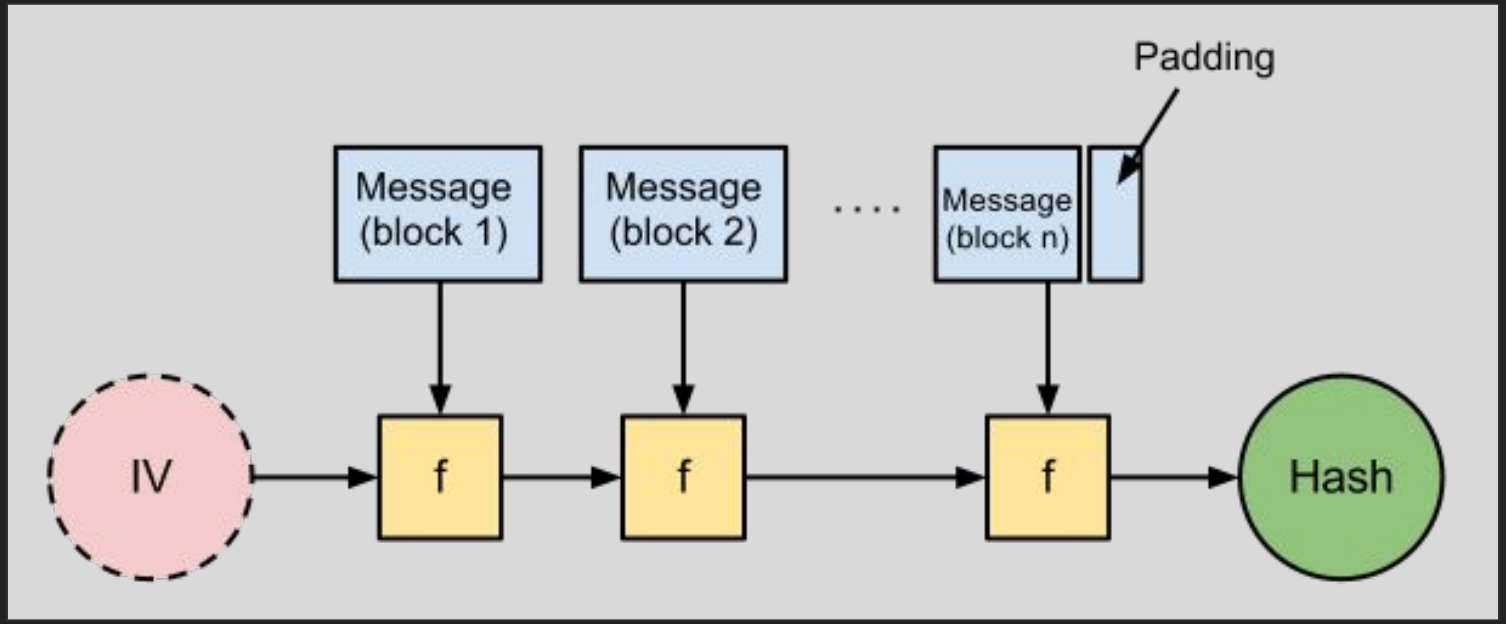
Hash function 就像 Block cipher 一段一段 hash 起來

他們各自也有自己的 IV (起始向量) 叫做 magic number

ex: md5 0x67452301, 0xefcdab89, 0x98badcfe, 0x10325476

Hash Function

剩下不夠的 + Padding 補齊，最後補上該 message 的長度



Hash Function

Hash(message)

=> Hash(message + padding + length)

=> Hash(message + (\x80 + \x00 … \x00) + \x02\x28)

Hash Function

Hash(m1) → digest (d1)

把 d1 當 magic number 再繼續 Hash(m2) 相當於

Hash(m1 + padding + length + m2)

Hash Function

Length Extend Attack (LEA) (for : MD5, SHA-1, and SHA-2)

tool : hashpumpy

Hash Function

Sha-1 Google 已經破解了，隨便 google 都有

https://github.com/maojui/cryptools/blob/master/cryptools/hash.py

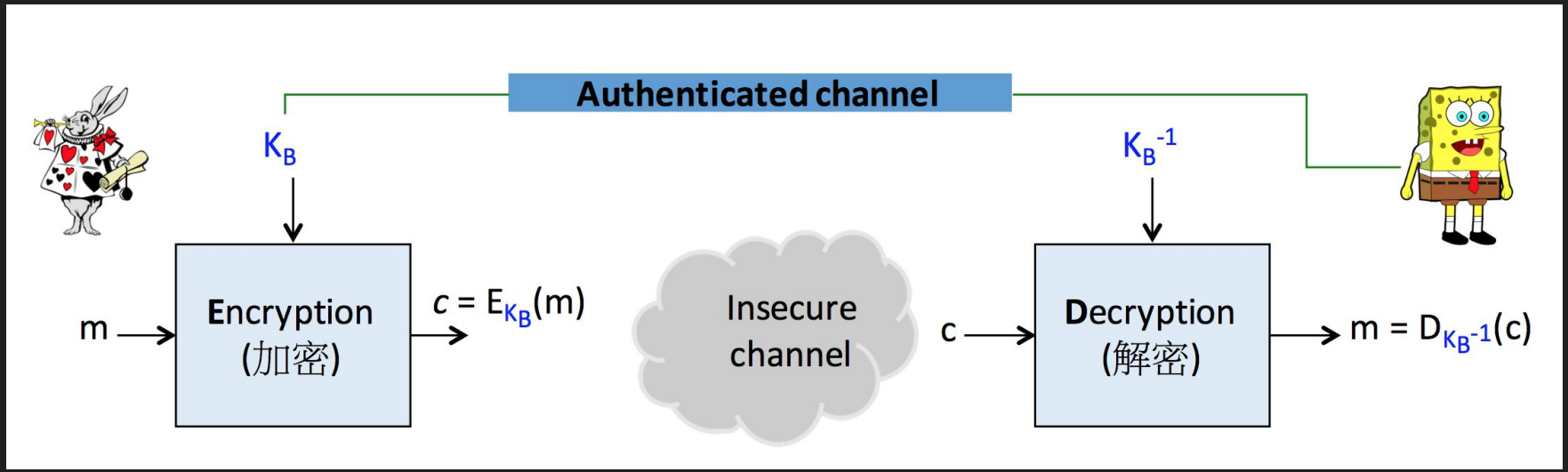
偶爾會出現要 sha1-collision 的東西，可以用上面的 code 生

Asymmetric

非對稱式加密 Asymmetric

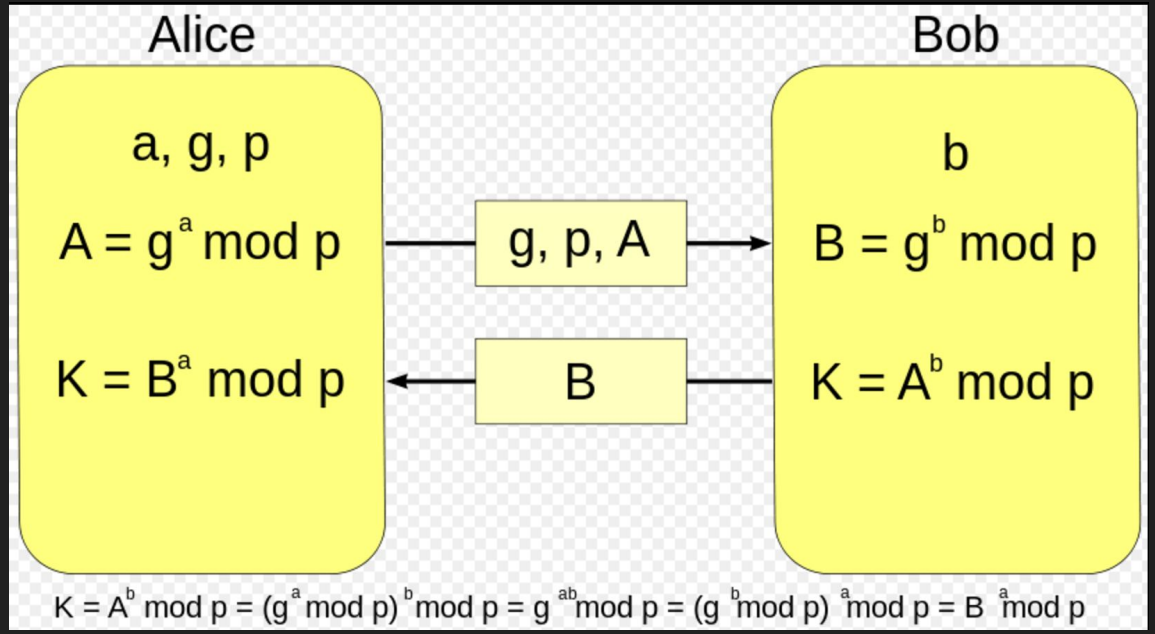
1. Diffie-Hellman key exchange (D-H)
2. Public-Private key Encryption: RSA、DSA、ECC
3. 數位簽章 Digital Signature

非對稱式加密 Asymmetric



Diffie-Hellman Key Exchange

一種交換密鑰的方法：



Diffie-Hellman Key Exchange

最簡單，最早提出的這個協定使用一個質數*p*的整數模n乘法群以及其原根*g*。下面展示這個演算法，綠色表示非秘密資訊, **紅色粗體**表示秘密資訊：

愛麗絲				鮑伯		
秘密	非秘密	計算		計算	非秘密	秘密
	p, g				p, g	
a						b
		$g^a \bmod p$	→		…	
	…		←	$g^b \bmod p$		
	$(g^b \bmod p)^a \bmod p$		=		$(g^a \bmod p)^b \bmod p$	

1. 愛麗絲與鮑伯協定使用 p=23以及base g=5.
2. 愛麗絲選擇一個秘密整數**a=6**, 計算A = $g^a \bmod p$並行送給鮑伯。
   - A = $5^6 \bmod 23 = 8$.
3. 鮑伯選擇一個秘密整數**b=15**, 計算B = $g^b \bmod p$並行送給愛麗絲。
   - B = $5^{15} \bmod 23 = 19$.
4. 愛麗絲計算**s** = $B^a \bmod p$
   - $19^6 \bmod 23 = \mathbf{2}$.
5. 鮑伯計算**s** = $A^b \bmod p$
   - $8^{15} \bmod 23 = \mathbf{2}$.

Key = (g^a mod p)^b mod p = (g^b mod p)^a mod p = g^ab (mod p)

Diffie-Hellman Key Exchange

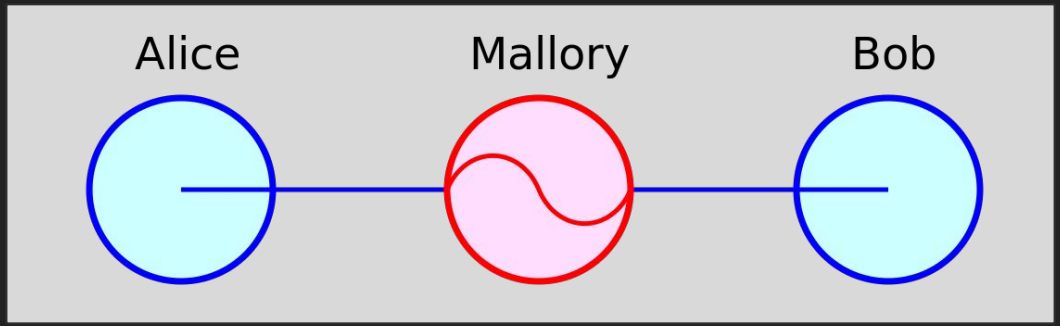
中間就算被監聽：對方得到

1. A = g^a mod p
2. B = g^b mod p
3. g
4. p

他還是算不出 Key : g^ab mod p

Man In the Middle (MITM) Attack

可是如果他從一開始就偽裝成你交換的對象 …



Public-Private key Encryption

想要有自己的 Key，又不想因為通訊方外洩造成問題 ….

Public Key 加密過的，只有擁有 Private Key 的人可以解開 (訊息一定傳給你)

Private Key 加密的東西，只有 Public Key 可以解 (訊息一定是你傳的)

RSA

RSA

安全在哪？

安全在 N 雖然是合數，但是兩個質數的因數分解超爆幹難。

懂？

RSA

143 ???

11

x

13

RSA

1170641952412170398116946265057141558063481451550883039403123662923123909687906067930709171046588009959324156890476924399115987295002514512363284816380910931716305629337272132394183012044490174099598958998619361237859128776321152368680682400628993229488902869915929771211832452759162619159165065590896006336429407912156339430725772722328799417647528719004078962609671506974158553546017125450440580097181090355574200228461658491598791887490113266745870667017599133806709959025364960543973145607357341171413726935927742274278393455749736696875747685465785473573661801920021305943466907987813525752912764583229198691108 7 = ?????

Few months after ….

954743383122182109981129184902800414535070725794414584115518873474871238570904895412887570699907979739183742978630763508896862733967700680758046910927850575369393685099575922619148509575866097579439313437983366735924891601050728831198401309562403718775566301641666244801893834127825572674381709323078499806131226132564107395268868527539619191499425854537558807345623690061500650566319551988022788494588051040797696327016863514305311556471075442985629111024971314985814162100965835181718613769104530486967377098003874351604915301248349705385502663609461030322772686933279156437066093699663251587510736261674588881808991

RSA

60184674974596457388468934439303153863105758281119186954191555882407617701197168377596294604210433709917216605003491845751550128271213913158214624480255154670030717734451574595231695934693298966846749512180060307658765270308040304979612168993991205620483672058782959006971989580331942290999451563888609561424074059477555963304933240270330274895817903146112115707357375950511698780169991970512927077532494026513542002542040715678355421919523725066872464136212521203684143100937780211726591874023899581215752593444234622986813399702513757464299227954319098372101095871794798579802052432871093490275342311340492781483864025211563890757868907787845759775419308581286066377698959676226531183831623894141990568934057410993864338850477377396369718972844676434524746002352327433261507326257089751052328688315875928612930234590055825104500555967726629138848486672021648450257631453937825959072164834999055836914539356804411674227504463371210979863582104027640305992891287895236562961351133732358449421695875668781546659407402141170937349850220186019115388913271632434849457877406142694268344648300354122919712208023449219914022590238603979338960607659631840920767390168792142920468504079098011471868223425800378313755677719055499154919850884 7

????? Few years after ….

24868527299409573118412055445854802684980398583271411750996183428073272716055101197810887561599854614584078660659576569512070046584394167232238039767734849045345505721478985002648019087626287067890772935698784905446410158456683022652846014580200764140362330075570390194836996130991712682111266983312740065958842967676518568470639229946516648262711099089221711320664499165880268724819876938541311778941498687944196281776743156979438647726657002485863683365126630557963248334036748366679424282702858840603887056931610631536455557543791916911142441893585366659569090914285566944840330358742845142040589111873515992365211

x

24201141567408118229470005659804041719037780307396011831645310392970614048856106670844792186753027299139186993695958066969706793991661239660078740788921441594228207791250197481597532194248243597428124895028832942313238305480214477311193486488900924329116881756043237566912114754173294728571914795726121447971394855509325002312151319510311901834278526216083320055645990701023519411945141476991683825234562976011209911293453217364887195291316508544133945091924202765800013718226993683007014880935463797024403622662824582607243551782670844280680811490430292299235468766035258517261777688658894610305771271862432176551677

RSA

要怎麼做？

1. 首先 找出兩個超大質數 p,q (2048 bit)
2. N = p*q
3. e = 任意質數
4. 找到 phi = (p-1) * (q-1) 必須跟e 互質
5. d = invmod(e,phi)　　ed mod phi = 1
6. 把 p,q 丟掉

public key : (N,e)
private key : (N,d)

mod (%)

mod (%)

模 …. 啪 … 取餘數 … ?

7 ÷ 2 = 3 … 1 : 7 % 2 = 1 寫成： 7 ≡ 1 (mod 2)

93 ÷ 11 = 8 … 5 : 93 % 11 = 5 寫成： 93 ≡ 5 (mod 11)

關於 mod 的特性太多了，有興趣請 Google 數論

他好像就把前面的數字套入一個循環中，永遠不會超出來

mod (%)

費馬小定理:（當p為質數）

$$a^{p-1} \equiv 1 \pmod{p}$$

ex : mod 7

```
def fm(p) :
    for _ in range(1,p) :
        print( pow( _,p-1,p) )
```

mod (%)

歐拉 (Euler) 定理：

$$a^{\varphi(n)} \equiv 1 \pmod{n}$$

歐拉函數：

n 為質數：Φ(n) = n-1　即為 費馬小定理

ex： n = 7, Φ(n) = 6

n 為合數 (n = p^a * q^b * r^c * ...)：Φ(n) = [(p-1) * p^(a-1)] * [(q-1) * q^(b-1)] * ….

ex： n = 1125 (3*3*5*5*5), Φ(n) = (2*3) * (4* 5*5) = 600

n = p * q　　phi = (p-1)*(q-1)

mod (%)

歐拉定理 :

$$a^{\varphi(n)} \equiv 1 \pmod{n}$$

重要特性:

ex : n = 7, Φ(n) = 6, a<n

a^19 (mod 7) ≡ a^(19 mod Φ(7)) (mod 7)

a^(19 mod 6) (mod 7) ≡ a^1 (mod 7)

: pow(5,19,7) >> 5

mod (%)

歐拉定理：

$$a^{\varphi(n)} \equiv 1 \pmod{n}$$

可以幹嘛? 反函數!!

如果可以找到一個 d

使得 ed mod Φ(n) = 1

那麼 當我們遇到 a^e, 將他乘上 d 次方, 變成:a^ed (mod n) 相當於 a^1 (mod n)

就可以 “ 把 a^e 還原成 a “

c^d = (m^e)^d = m^ed = m^(ed mod phi) = m^1 = m

mod (%)

如何找到一個 d，使得 ed mod Φ(n) = 1 ？ Extended Euclidean algorithm

好工具:

pip2 install git+https://github.com/hellman/libnum

pip3 install git+https://github.com/maojui/libnum

>> d = invmod(e, phi) # e 跟 phi 要互質

Public-Private key Encryption

RSA 還有一堆怪招, CTF 最愛出的

DSA … (特色 : 給 p,q,g 三個質數, 一個亂數 r)

Rabin Cryptosystem … (m^2 % p)

Paillier cryptosystem … (g^m * r^n mod n^2)

Public-Private key Encryption

ECC … 橢圓曲線密碼學

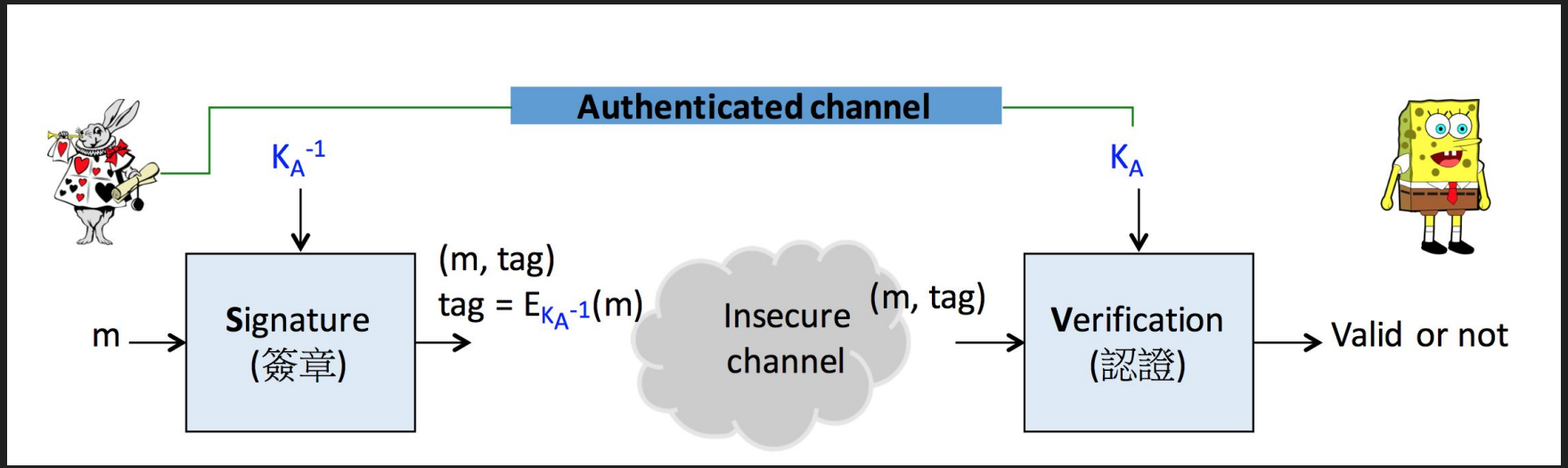
http://andrea.corbellini.name/2015/05/17/elliptic-curve-cryptography-a-gentle-introduction/

強度：

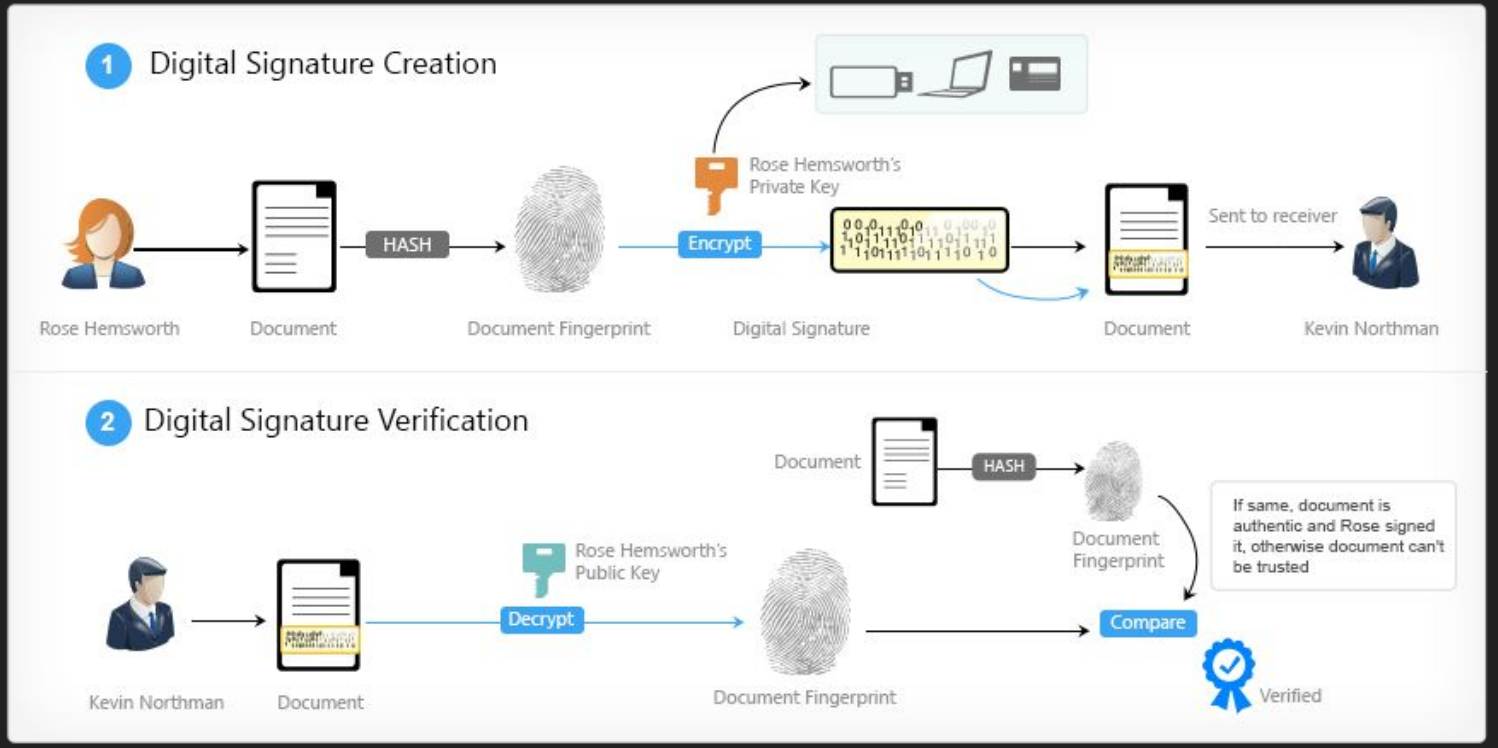
RSA ： 1024 bit | 15360 bit

ECC ： 160 bit | 512 bit

Digital Signature



Digital Signature



Digital Signature

數位簽章就是 ...

把非對稱式加密 和 Hash 喇來喇去在一起 @@

工具

- SageMath：數學運算軟體
- factordb：因式分解資料庫
- Yafu：幫助你因式分解

工具

- pycipher：古典密碼學加密演算法
- pycrypto：現代密碼學加解密演算法
- cryptography：同上
- sympy：符號運算

CTF 試刀

Classical

picoCTF 2018, Caesar Cipher 1 (150)

picoCTF 2018, Caesar Cipher 2 (150)

Seccon CTF 2016, Vigenere (100)

Seccon CTF 2017, Vigenere3d (100)

Qiwi-Infosec CTF-2016, Crypto 100_1 (100)

CTF 試刀

One Time Pad - crib drag

EasyCTF 2018 - Not OTP (100)

Pragyan CTF 2018, Improper encryption (100)

AlexCTF 2017 - CR2 Many time secrets (100)

CTF 試刀

Linear Cryptanalysis

TUMCTF Teaser 2015 (350)

0CTF Qual 2018 - zer0SPN (550)

CTF 試刀

ECB Cut & paste

CSAW Quals 2017, BabyCrypt (350)

PicoCTF 2018, SpyFi (300)

Nuit du Hack CTF Quals 2016, Toil33t (400)

CTF 試刀

CBC Padding Oracle

BambooFox CTF 2018, mini-padding (200)

Hitcon CTF 2017, Secret Server (221)

CSAW Qual 2016, Neo (200)

Hack.lu CTF 2016, Cryptolocker (200)

ASIS Qual 2018, Yunny It (500)

picoCTF 2018, Magic Padding Oracle (450)

CTF 試刀

Length Extend Attack (LEA)

TW.edu CTF 2015, LEA (200)

RCTF 2018, cpushop (176)

Viettel Mates CTF 2018, Viettel Store (100)

BambooFox CTF 2018, baby-lea (100)

Hackover CTF 2015, securelogin (250)

CTF 試刀

Sha1 collision

Defcon CTF 2018, Easy Pisy (104)

Seccon CTF 2017, SHA-1 is dead (100)

CTF 試刀

RSA

NTUSTISC