Cryptography 101

TOC

Today

- Introduction & Tools
- Classical
- Symmetric
- Hash & MAC

Next

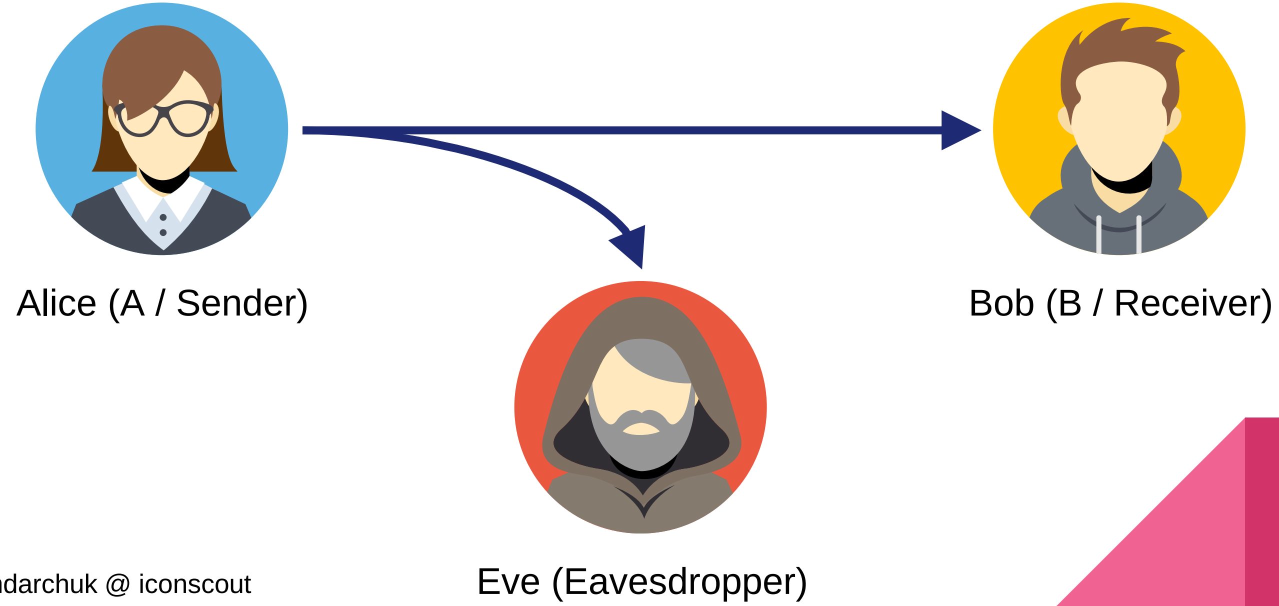
- Abstract Algebra
- Asymmetric
- Elliptic Curve
- Key Exchange
- Signature

- Quantum
- LFSR
- PRNG
- ……

資訊安全 | Infosec

威脅模型 | Threat model

Passive attacker: Monitoring Only



Icon: Dmitriy Bondarchuk @ iconscout

威脅模型 | Threat model

Active attacker: Modify / Substitute / Drop / Replay

Alice (A / Sender)   Oscar (Opponent)   Bob (B / Receiver)

Also known as Man-in-the-middle (MitM)

Icon: Dmitriy Bondarchuk @ iconscout

資訊安全 | Information Security

- Confidentiality (機密性)
- Integrity (完整性)
- Authentication (認證性)

舉一個栗子 | First Example

HelloWorld ⇐ base64 ⇒ SGVsbG9Xb3JsZAo=

✗ Confidentiality (機密性)
✗ Integrity (完整性)
✗ Authentication (認證性)

BTW, base64 is NOT an encryption algorithm

一次性密碼本 | One-time Pad (OTP)

EXAMPLE → 04 23 00 12 15 11 04

+

LDFKJPQ → 11 03 05 10 09 15 16

=

PAFWYAU ← 15 00 05 22 24 00 20

一次性密碼本 | One-time Pad (OTP)

Perfect secrecy: Unbreakable even with infinite resources

Requirements:

Truly random key

len(key) = len(message)

Used once and ONLY once

安全的算法 | Secure Algorithm

Impractical perfect secrecy $\rightarrow$ Computationally infeasible

A problem that can be solved in theory (e.g. given large but finite resources),

but for which in practice any solution takes too many resources to be useful.

Security through obscurity

“ Security experts have rejected this view as far back as 1851, and advise that obscurity should never be the only security mechanism. ”

Never write your own encryption algorithm

工具 | Tools

Tools

- pyCrypto / cryptography: Crypto algorithms
- gmpy2: Multiple-precision arithmetic and some number theory
- libnum: Number theory
- SageMath / CoCalc: Computer algebra system
- RsaCtfTool: Various attack and utils of RSA
- Factordb: A large database of factor
- yafu: A factorization tool

- Pwntools: Python's Wonderful Networking Tools

古典密碼學 | Classic Cryptography

異或 | Exclusive or (XOR)

1001001010111010101001

$\oplus$

0110100110100101011110

$=$

1111101100011111110111

Input		Output
A	B	
0	0	0
0	1	1
1	0	1
1	1	0

- XOR is an involutory function, i.e. $(A \oplus B) \oplus B = A$
- Addition in GF(2)

替換式密碼 | Substitution cipher

Caesar Cipher:

key 13 (ROT13): HelloWorld ↔ UryybJbeyq

Single/Multi byte XOR:

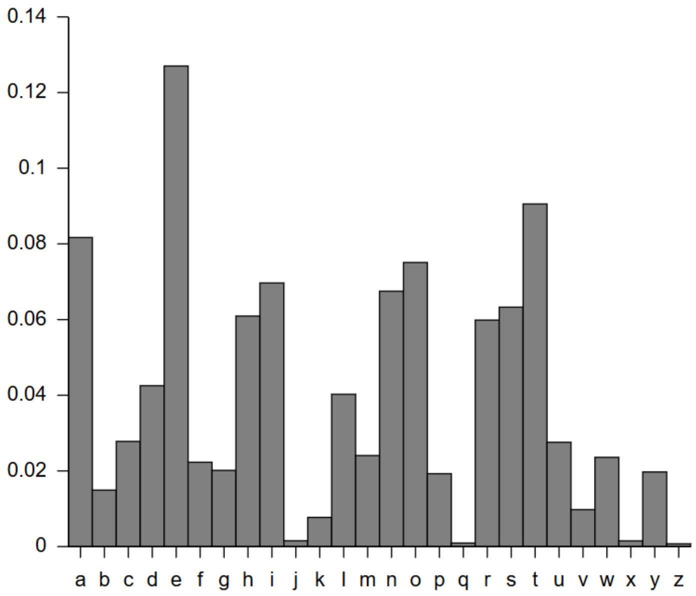
key 42: HelloWorld ↔ bOFFE}EXFN

xortool

詞頻分析 | Frequency analysis

Frequency distribution doesn’t change

after encryption.



quipquip: Automated cryptogram solver

https://upload.wikimedia.org/wikipedia/commons/d/d5/English_letter_frequency_%28alphabetic%29.svg

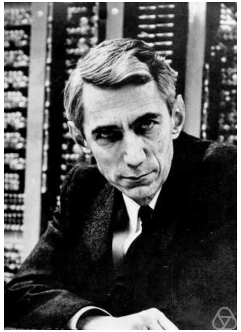
基因演算法 | Genetic Algorithm

Similar key → Similar plaintext

```
ceaowrd  →  cello world (0.6)
helowry  →  hello worly (0.5)
       Cross Over
helowrd  →  hello world (1.0)
celowry  →  cello worly (0.1)
```

現代密碼學 | Modern Cryptography

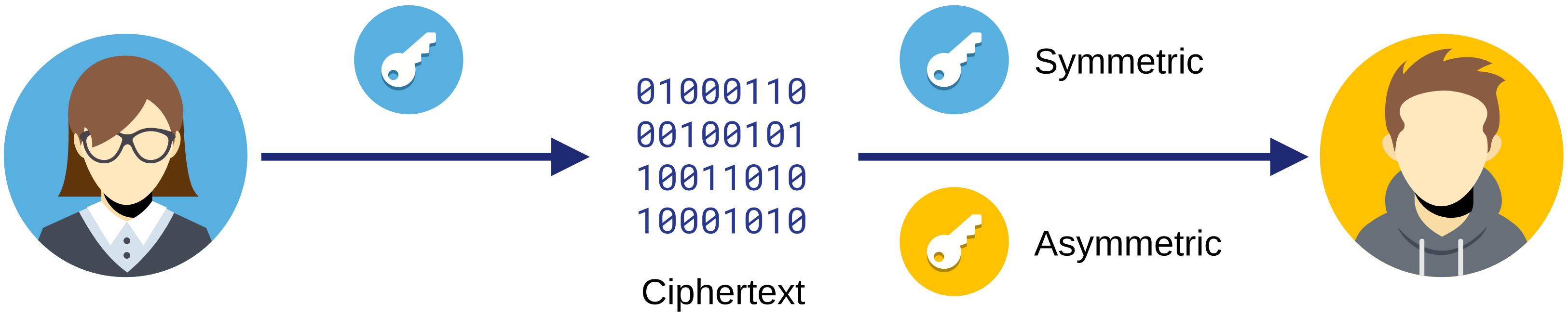
混淆 & 擴散 | Confusion & Diffusion



- Proposed by Claude Shannon
- Diffusion:
  - Change single bit of the plaintext → Half of the bits in the ciphertext changed
  - Change single bit of the ciphertext → Half of the bits in the plaintext changed
- Confusion:
  - Each bit of the ciphertext depend on several parts of the key

對稱/非對稱加密 | Symmetric vs Asymmetric

Same / Different key for encryption and decryption



Symmetric example: AES, Asymmetric example: RSA

Icon: Dmitriy Bondarchuk @ iconscout

區塊加密法 | Block cipher

區塊加密法 | Block Cipher

$E_K(P_1) = C_1$

Input:

Fixed-length key K

Fixed-length plaintext P

Output:

Fixed-length ciphertext C

費斯妥結構 | Feistel structure

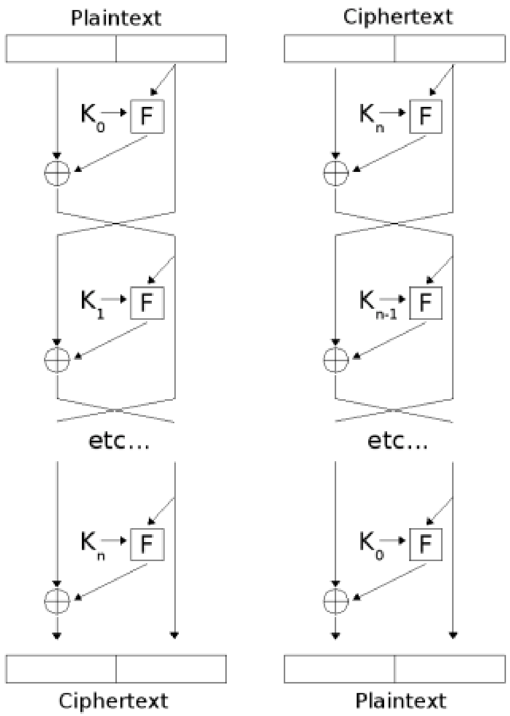
Encryption and decryption operations are very similar.

F does not have to be invertible,

It can be any kind of pure function,

e.g. SP-network, hash, or even neural network.

Example: DES, Blowfish, TEA …



https://zh.wikipedia.org/wiki/File:Feistel_cipher_diagram.png

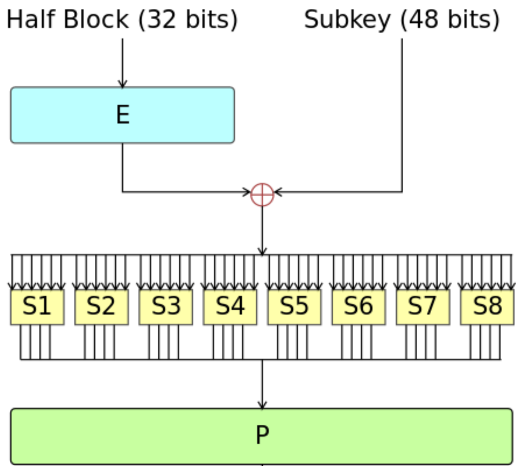
資料加密標準 | DES

Feistel cipher

64bits block size

Small keyspace (56bits) → 3DES: up to 168bits key

Ciphertext = $E_{K3}(D_{K2}(E_{K1}(\text{message})))$



Task: Google 2018 CTF Quals - DM Collision

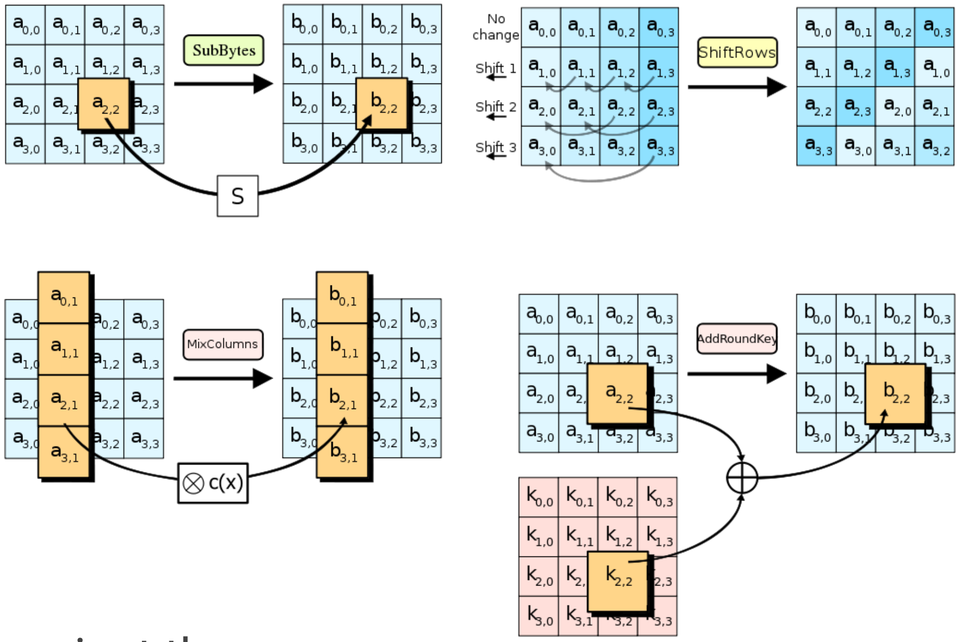
https://en.wikipedia.org/wiki/Data_Encryption_Standard

進階加密標準 | AES

128 bits block size

Large keyspace (128, 192, or 256 bits)

Reversing tips: S-box



Currently no practical published attacks against the full AES algorithm.

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

區塊加密工作模式 | Mode of Operation

填充 | PKCS#7 Padding

The value of each added byte is the number of bytes that are added.

For example:

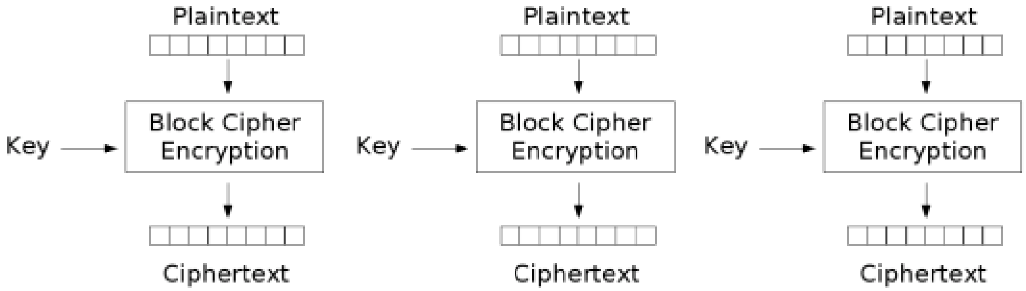
```
... | DD DD DD DD DD DD DD DD | DD DD DD DD 04 04 04 04 |
```

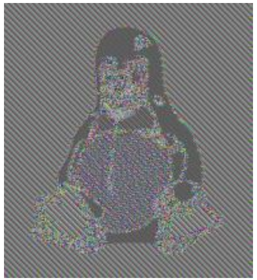
To avoid padding, you could use “Ciphertext stealing (CTS)”.

電子密碼本 | ECB

Each block is encrypted separately

Lack of diffusion





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

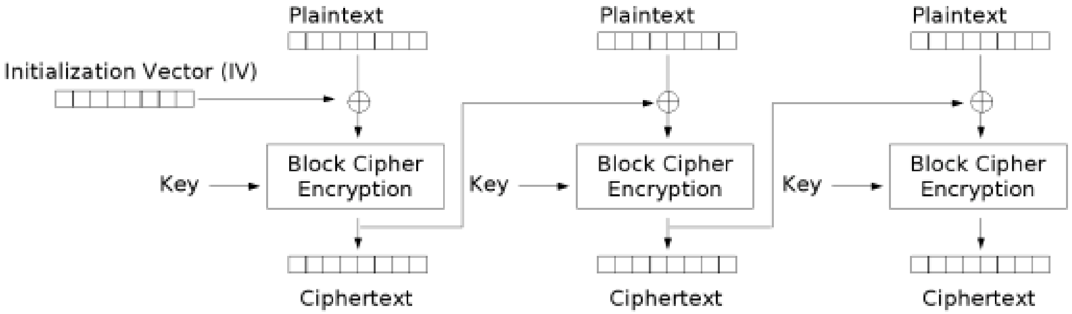
Ctrl-X & Ctrl-V | Cut and Paste

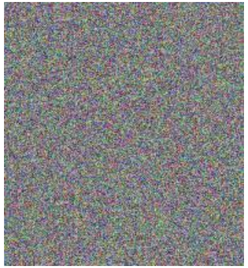
```
|usr=a&pw|=a&root=|N.......|  →  |A|B|C|
|usr=abcd|Y&pw=aaa|&root=N.|  →  |D|E|F|

|usr=a&pw|=a&root=|Y&pw=aaa|N.......|  →  |A|B|E|C|
```

密碼塊連結 | CBC

Most commonly used

Decryption depends on two adjacent blocks





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

Padding Oracle

```
| 12 34 56 78 | 04 04 04 04 | ↔ | 9e 42 7b a0 | f9 08 2c d5 |  :  OK
| 04 cd 72 b9 | 04 04 04 05 | ↔ | 9e 42 7b a1 | f9 08 2c d5 |  :  Invalid padding
| 11 cf e6 95 | 04 04 04 01 | ↔ | 9e 42 7b a5 | f9 08 2c d5 |  :  Corrupted data
| 25 64 b6 f9 | 04 04 05 02 | ↔ | 9e 42 7a a6 | f9 08 2c d5 |  :  Invalid padding
| 70 72 df bc | 04 04 02 02 | ↔ | 9e 42 7d a6 | f9 08 2c d5 |  :  Corrupted data
```

Real world example: POODLE (SSL 3.0 / TLS 1.0)

Plaintext Truncate

```
def unpad(pad_msg):
        unpad_msg = pad_msg[:-pad_msg[-1]]
        return unpad_msg
```

```
| 12 34 56 78 | 90 ab cd ef | 04 04 04 04 | → unpad → | 12 34 56 78 | 90 ab cd ef |
| 12 34 56 78 | 0c d4 4c e9 | 04 04 04 09 | → unpad → | 12 34 56 |
```

Other Oracles

- Add & Xor oracles:
  - $a(m) = \mathrm{Enc}(flag + m)$
  - $x(m) = \mathrm{Enc}(flag \oplus m)$
  - Task: Hack.lu 2018 – Relations
- ……

串流加密法 | Stream cipher

串流加密法 | Stream Cipher

Keystream = $PRG_K(IV)$

$E_K(IV, M) = M \oplus Keystream[:len(M)]$

Input:

- Fixed-length key K
- Fixed-length IV
- Variable-length plaintext M

Output:

- Variable-length ciphertext C

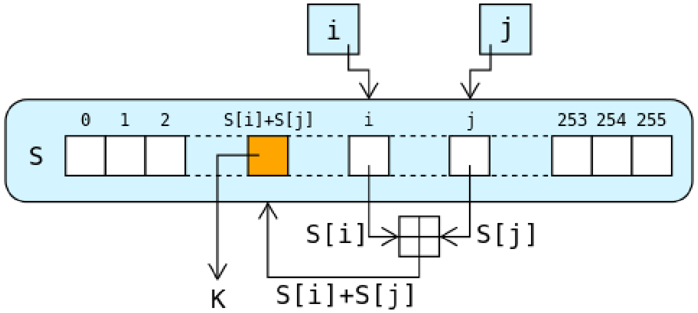
位元翻轉 | Bit Flip

```
| 12 34 56 78 | 90 ab cd 00 | ↔ | 9e 42 7b a0 | d7 6b f6 88 |

| 12 34 56 78 | 90 ab cd 01 | ↔ | 9e 42 7b a0 | d7 6b f6 89 |

| 12 34 56 78 | 91 ab cd 00 | ↔ | 9e 42 7b a0 | d6 6b f6 88 |
```

$$E_K(M_1) \oplus M_2 = E_K(M_1 \oplus M_2)$$

RC4 | Rivest Cipher 4

Simple and fast



Broken: Statistical bias in keystream

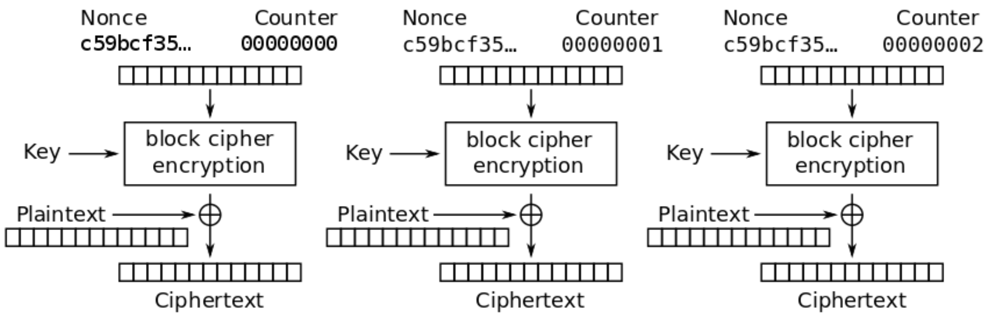
Example: WEP, WPA-TKIP, SSL/TLS

https://upload.wikimedia.org/wikipedia/commons/thumb/e/e9/RC4.svg/480px-RC4.svg.png

計數器模式 | CTR

Turn a block cipher to stream cipher

Nonce should never be reused



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

Nonce Reuse / Counter Reset

```
Given          | 12 34 56 78 | 04 04 04 04 | ↔ | 46 b3 bc cb | d9 b2 00 17 |

keystream =    | 54 87 ea b3 | dd b6 04 13 |

ciphertext =   | 54 96 c8 80 | 99 e3 62 12 |

plaintext =    | 00 11 22 33 | 44 55 66 01 | = ciphertext ^ keystream
```

Task: *CTF 2018 - ssss

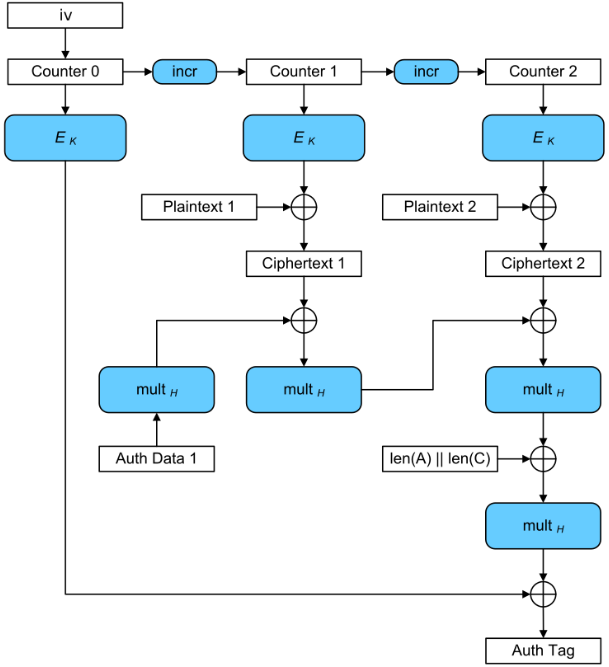
Real world example: KRACK (WPA2)

GCM | Galois/Counter Mode

High throughput, parallel

Confidentiality & Integrity

Combine CTR Mode & GMAC



IV reuse → Authentication key (H) recover

https://en.wikipedia.org/wiki/Galois/Counter_Mode

GCM Tag Truncate

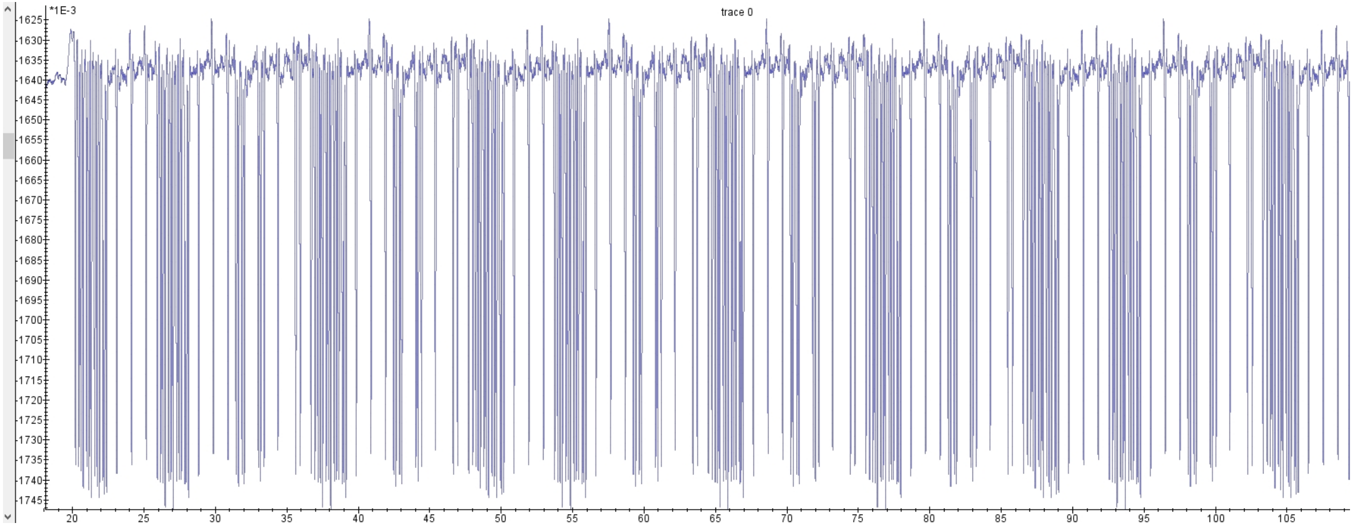
```
Msg = | 12 34 56 78 |, Enc = | 6d 0f 87 64 |, Tag = | 23 f4 d4 ea |   :  OK

Msg = | 12 34 56 79 |, Enc = | 6d 0f 87 65 |, Tag = | 23 f4 d4 ea |   :  Invalid

Msg = | 12 34 56 78 |, Enc = | 6d 0f 87 64 |, Tag = | 23 f4 d4 eb |   :  Invalid

Msg = | 12 34 56 78 |, Enc = | 6d 0f 87 64 |, Tag = | 23 |            :  OK (OA O )???

Msg = | 12 34 56 79 |, Enc = | 6d 0f 87 65 |, Tag = | c9 |            :  OK (BOOOOM!!!)
```

CVE-2018-10903: python-cryptography GCM tag forgery

Task: Pwn2Win CTF 2018 - GCM

旁路攻擊 | Side channel attack

Power leakage
Radio leakage
Sound leakage
...



Task: SCTF 2018 - 側信道初探

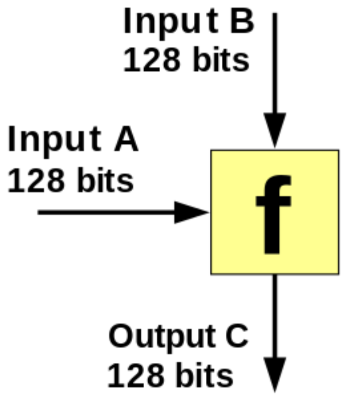
雜湊函式 | Hash

單向函數 | One-way Compression Function

Easy to compute output with given inputs.

Difficult to compute inputs which compress to a given output.

P =?= NP



https://en.wikipedia.org/wiki/One-way_compression_function

密碼雜湊函數 | Cryptographic Hash Function

It should be difficult to solve following problems:

Pre-image: Find M such that $h = H(M)$

Second pre-image: Find M_2 such that $H(M_1) = H(M_2)$

Collision: Find M_1 and M_2 such that $H(M_1) = H(M_2)$

舉一個栗子 | Example

H = summation of all bytes

✗ Pre-image resistance: xx = H(xx)
✗ Second pre-image resistance: H(xx) = H(xx 00)
✗ Collision resistance: H(xx 00) = H(00 xx)

How about the following hash function?

```
h = 1
for c in m:
    h = (h * p + c) % q
```

Merkle–Damgård construction

Fixed-length to variable-length

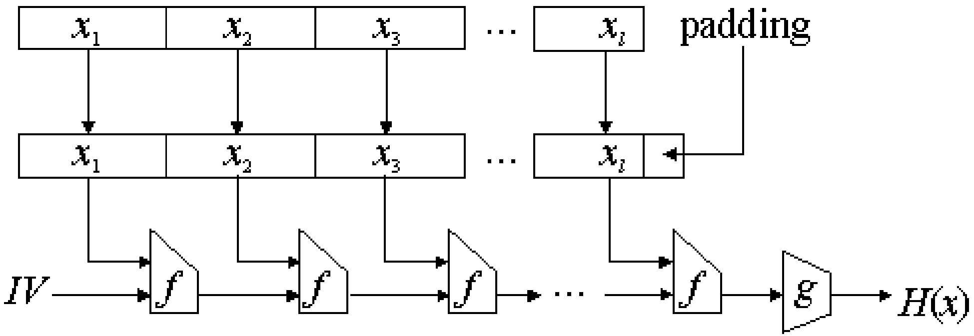
x: input message

f: one-way compression function

g: finalization function

Usually, g is an identity function.

Example: MD5, SHA1, SHA256



https://upload.wikimedia.org/wikipedia/en/8/89/Merkle-Damgard_diagram.png

長度擴充攻擊 | Length Extension Attack

```
IV = | 00 00 00 00 |, X = | 12 34 |,       H(x) = | 27 0a 19 4e |
IV = | 00 00 00 00 |, X = | 12 34 56 78 |, H(x) = | 51 b4 c0 ad |
IV = | 27 0a 19 4e |, X = | 56 78 |,       H(x) = | 51 b4 c0 ad |
```

Tools: Hashpumpy

填充 | Padding

```
Pad_msg = ... | 32 10 00 00 ..... 00 80 xx xx xx xx xx xx xx xx |
```

512 bits

```
xx xx xx ... = len(msg) in 64 bits integer
```

Some non-printable bytes in message when using length extension attack.

MD5

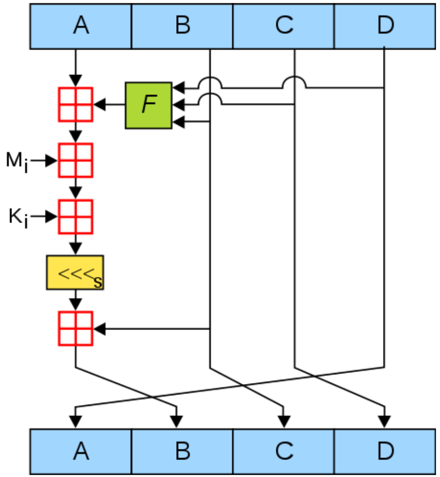
128 bits (16 bytes) digest size
Much faster than SHA-family

Reversing tips:

Sine function / Constant 0xd76aa478

Security properties:

✓ Pre-image resistance
✓ Second pre-image resistance
✗ Collision resistance: 2^18



https://en.wikipedia.org/wiki/MD5

SHA1 | Secure Hash Algorithm 1

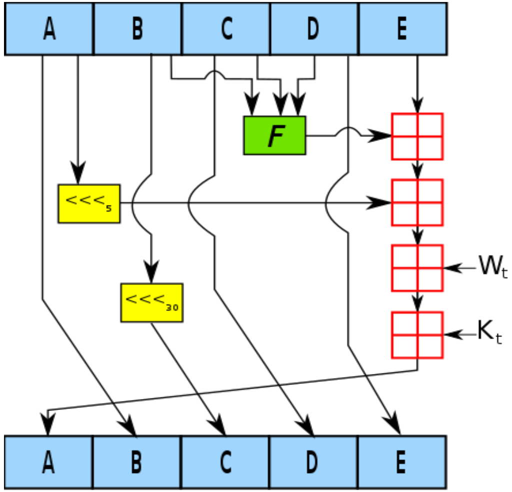
160 bits (20 bytes) digest size

Reversing tips:

Constant 0xc3d2e1f0

Security properties:

- ✓ Pre-image resistance
- ✓ Second pre-image resistance
- ✗ Collision resistance: 2^60



https://en.wikipedia.org/wiki/SHA1

SHA2 | Secure Hash Algorithm 2

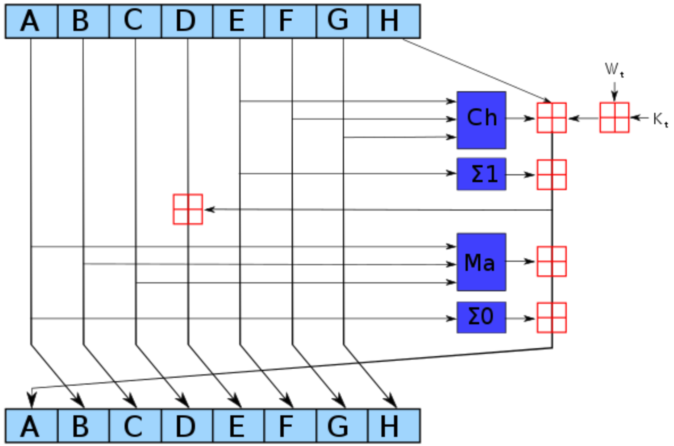
224, 256, 384, 512 bits digest size

Reversing tips:

Constant 0x428a2f98

Security properties:

- ✓ Pre-image resistance
- ✓ Second pre-image resistance
- ✓ Collision resistance



https://en.wikipedia.org/wiki/SHA2

碰撞 | Collisions

$H(M_1) = H(M_2) \rightarrow H(M_1 \mid\mid M_3) = H(M_2 \mid\mid M_3)$

md5-tunneling: Identical prefix collision within several seconds
hashclash: MD5 chosen prefix collision
Shattered: SHA1 collision blocks in PDF

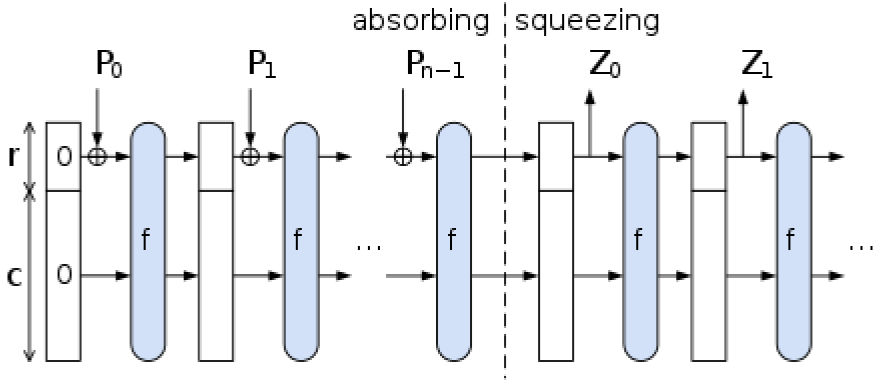
Task: DEFCON 2018 Quals - Easy Pisy
Task: ebCTF 2013 - MD5 Colliding

SHA3 | Secure Hash Algorithm 3

Sponge construction
Arbitary digest size

Reversing tips:

Const 0x8000000080008081



Security properties:

- ✓ Pre-image resistance
- ✓ Second pre-image resistance
- ✓ Collision resistance

https://en.wikipedia.org/wiki/SHA3

訊息鑑別碼 | MAC

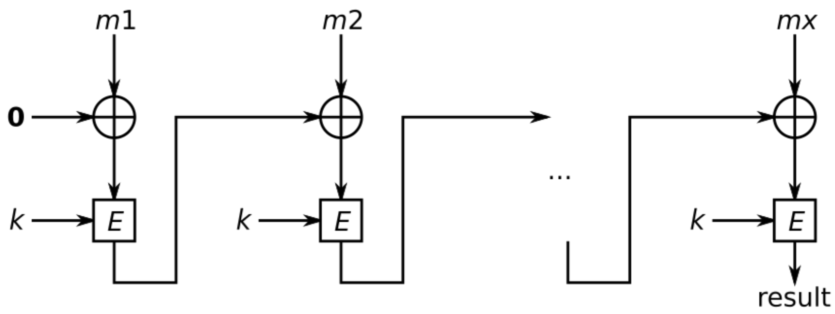
CBC-MAC

Last ciphertext block of CBC mode

Secure only for fixed-length messages

Solution for variable-length messages:

- Length prepending
- Encrypt last block with another key



Simple MAC

- $MAC(M) = H(M \| K)$: Internal collision
  - $H(M_1) = H(M_2) \rightarrow H(M_1 \| K) = H(M_2 \| K)$
- $MAC(M) = H(K \| M)$: Length extension attack
  - $M(M_1 \| P \| M_2) = H(K \| M_1 \| P \| M_2 , IV=0) = H(M_2 , IV=H(K \| M_1 , IV=0))$

HMAC | Keyed-hash message authentication code

- $HMAC(K,m) = H\Big((K' \oplus opad) \,||\, H\big((K' \oplus ipad) \,||\, m\big)\Big)$

- HMAC-MD5 does not suffer from the same weaknesses that have been found in MD5, but it shouldn’t included in new protocol.