Cryptography

Maojui

Outline

- 針對 RSA 演算法的攻擊
- 雜湊函式 Hash 與 LEA 攻擊

針對 RSA 演算法的攻擊

RSA 安全性 | RSA Security

- 如果質數 p, q 找的不好，將會導致 N 被分解 ...

因式分解 | Factorization

- 一些容易被分解的 N：
  - 由小質數組成
  - 有兩組以上的 N 共用質數 p, q
  - 由相近的兩質數組成
  - 組成 N 的質數 p，(p-1) 是個 K-smooth 數

因式分解資料庫 | Factordb

- Factordb : http://factordb.com/
- 如果遇到一些由小的質數構成的 N 可以直接被分解
- 或是對題目沒有頭緒，也可以丟進來
- 有時候可以撿到一些別人分解好的質數

費馬因式分解 | Fermat Factorization

- 如果合數 N 剛好是由相近的兩質數組成
- Fermat Factorization 可以以 O(√N) 將其分解

[LAB] RSA Factorization

波拉德 | Pollard's p – 1

- 如果組成 N 的質數 p，(p-1) 是個 K-smooth 數
- 且 K 不大，是個可以走完的數

威廉斯 | Williams’ p + 1

- 如果組成 N 的質數 p，(p+1) 是個 K-smooth 數
- 且 K 不大，是個可以走完的數

[LAB] Smooth Prime

RSA 安全性 | RSA Security

- 除了找了不當的質數 p,q 來組成 N 造成的問題
- 接下來換 e ....

因式分解 | Factorization

- 一些容易出問題的 e：
  - e 太小 - 沒加密到
  - e 太大 - 導致 d 很小

維納攻擊 | Wiener Attack

- 使用條件：
  - (1) q < p < 2q
  - (2) d < 1/3 N^(1/4) → 特徵：e 很大
- Wikipedia : https://en.wikipedia.org/wiki/Wiener%27s_attack

[LAB] YANG_RSA-2

共模攻擊 | Common Modulus Attack

- 反過來如果是 `N` 連用，但 `e` 卻沒換？

$$
\begin{aligned}
C_1^x * C_2^y &= (M^{e_1})^x * (M^{e_2})^y \\
&= M^{e_1 x} * M^{e_2 y} \\
&= M^{e_1 x + e_2 y} \\
&= M^1 \\
&= M
\end{aligned}
$$

[Lab] YANG_RSA-4

RSA 安全性 | RSA Security

- 了解找出適當的 p, q 的重要性，以及 e 的注意事項
- 接著，還有公鑰使用方式的問題 ...

因式分解 | Factorization

- 公鑰 (e, N) 使用方面造成的問題：
    - e 不換 N 換了
    - e 換了 N 不換

中國餘事定理 | Chinese Remainder Theorem

- m ≡ c1 mod n1
- m ≡ c2 mod n2
- m ≡ c3 mod n3
- …
- CRT 能夠找出一個數符合上述條件的 C (mod n1*n2*n3* …)

中國餘事定理 | Chinese Remainder Theorem

- 舉個例子：
  - $x \equiv 2 \mod 11$
  - $x \equiv 1 \mod 19$
  - $x \equiv 2 \mod 37$
- CRT → $2851 \pmod{11 \times 19 \times 37}$

中國餘事定理 | Chinese Remainder Theorem

1. 令 $N_i = \frac{N}{n_i}$

2. 令 $t_i \equiv N^{-1} \mod n_i$

3. 則 $x = \sum_{i=1}^{n} (a_i \times N_i \times t_i) + k \times N$

中國餘事定理 | Chinese Remainder Theorem

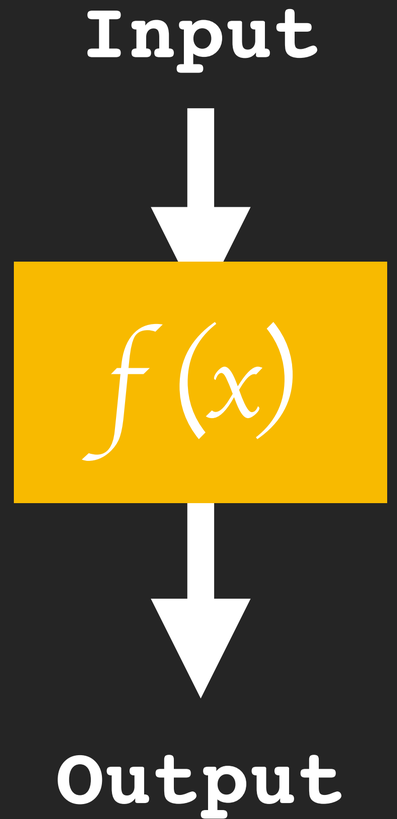
- 同一個 `e` 加密好幾次，卻每次都用了不同的 `n` ？

[LAB] Chinese Remainder Theorem

Hash

單向函數 | One-way compression function

- 可以很容易的算出結果
- 但是很難從結果回推給定的 `Input`



雜湊函數 | Hash function

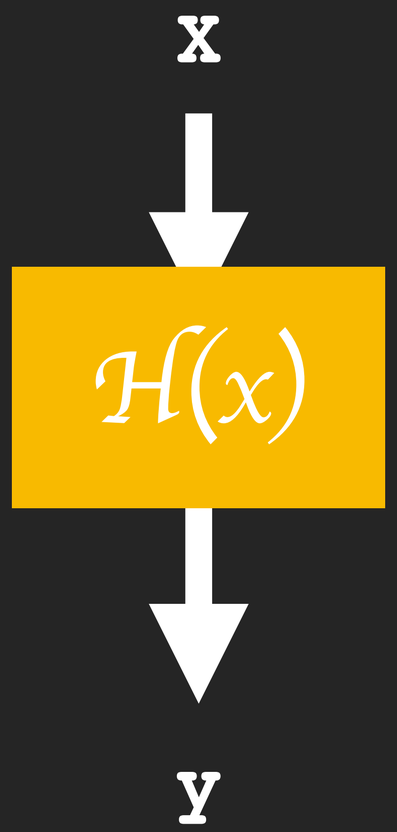
- 功能：將資料壓縮成摘要、保護資料、確保傳遞真實的資訊
- 特徵：固定長度的輸出
- 期望：符合單向函數、$\forall y, \exists! x : h(x) = y$

雜湊函數 | Hash function

- 字串總和 (len)
- Bytes 總和
- Bit 有幾個 1
- LCG

密碼雜湊函數 | Cryptographic Hash function

- Pre-image resistance
  - 已知 y 也無法找到 x
- Second pre-image resistance
  - 已知 $y = H(x_1)$ 無法找到 x_2 使得 $y == H(x_2)$
- Collision resistance
  - 無法找到一組 x_1、x_2 使得 $H(x_1)$ 和 $H(x_2)$ 有相同的雜湊值
- Avalanche effect
  - x_1、x_2 只有微小的差別，但 y_1、y_2 卻差很多



密碼雜湊函數 | Cryptographic Hash function

Kaibro → `8c45dd86e3659040ecdc36b007a99a6e907d2fcd4d2d80142f424912`

KaiBro → `1df7b804e1af7caa5d9959a81727a905658d19ba6719cc6b761d5c0c`

Kaibr0 → `de18feec5e15a615203941bab08edbbedcbdbc91dd8c3885cbdf43dd`

??? → `8d93f6f29f0cd61bd6f7dbc975a7e7ee096841ab1dfcc90f082f6e53`

碰撞 | Collision

- H(M1) = H(M2)

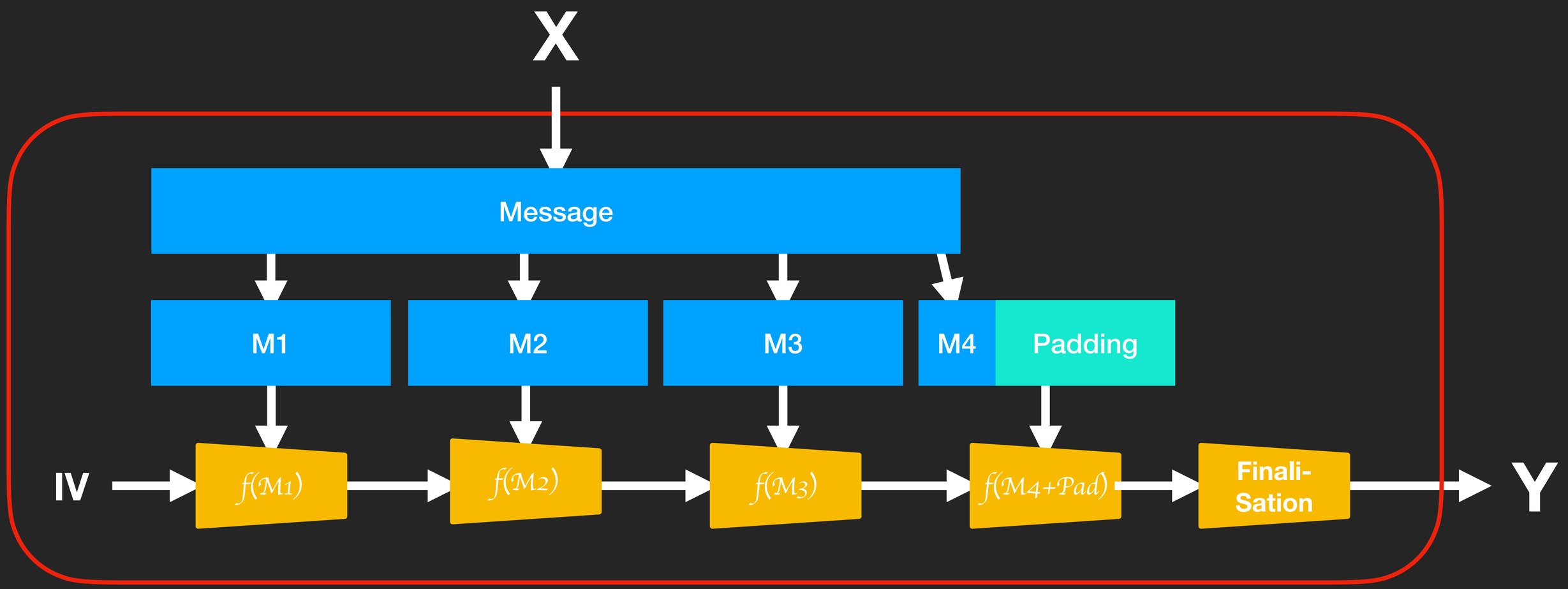
  → H(M1 || M3) = H(M2 || M3)

- Shattered: SHA1 collision blocks in PDF

Merkle–Damgård construction

- Fixed Input → Variable Input
- 該結構可以讓碰撞降低 ... 等

Merkle–Damgård construction



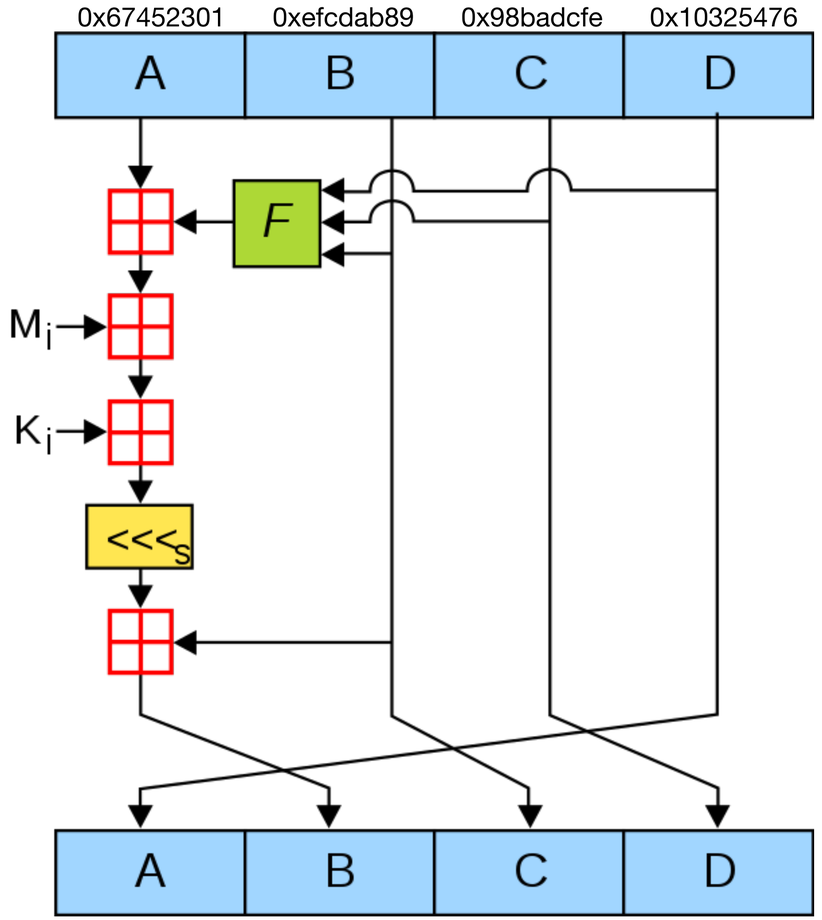
MD5, SHA1, SHA2

MD5

長度：128 bits

特徵：用到 Sin(x)

- ✓ Pre-image resistance
- ✓ Second Pre-image resistance
- ✗ Collision resistance : 2^{18} 太小



SHA1

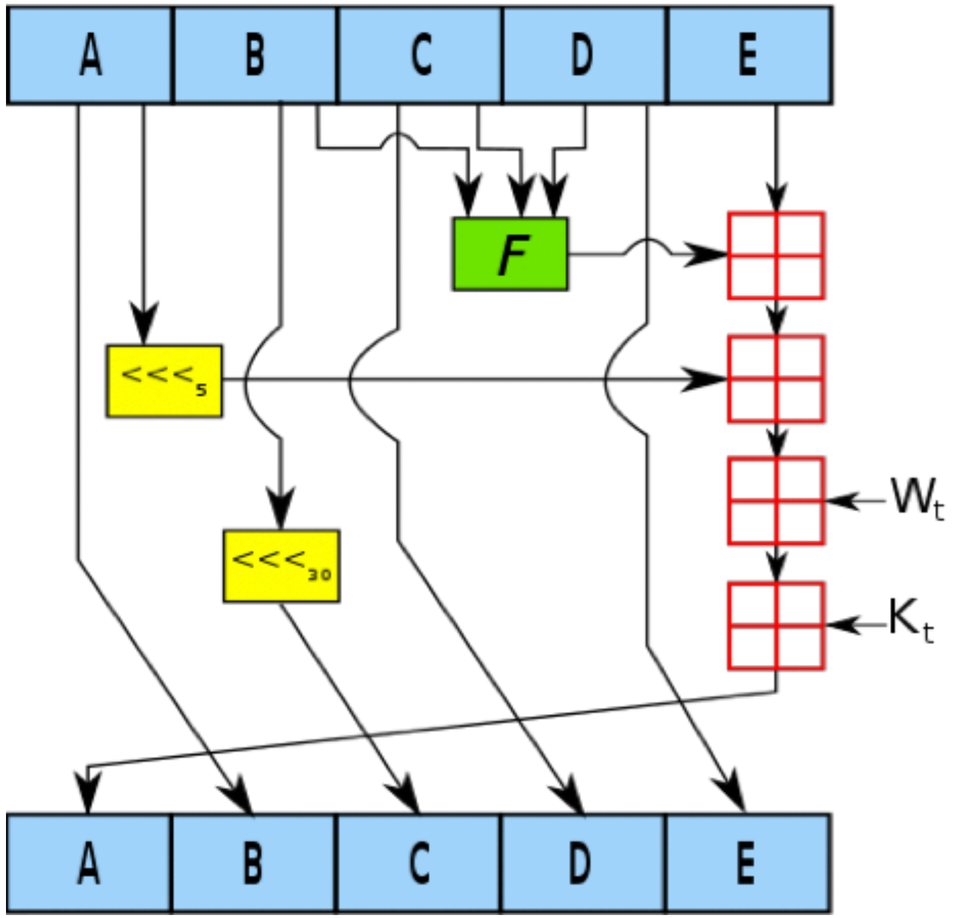
長度：160 bits

特徵：0x428a2f98

✓ Pre-image resistance

✓ Second Pre-image resistance

✗ Collision resistance : 2^{60} 不夠大

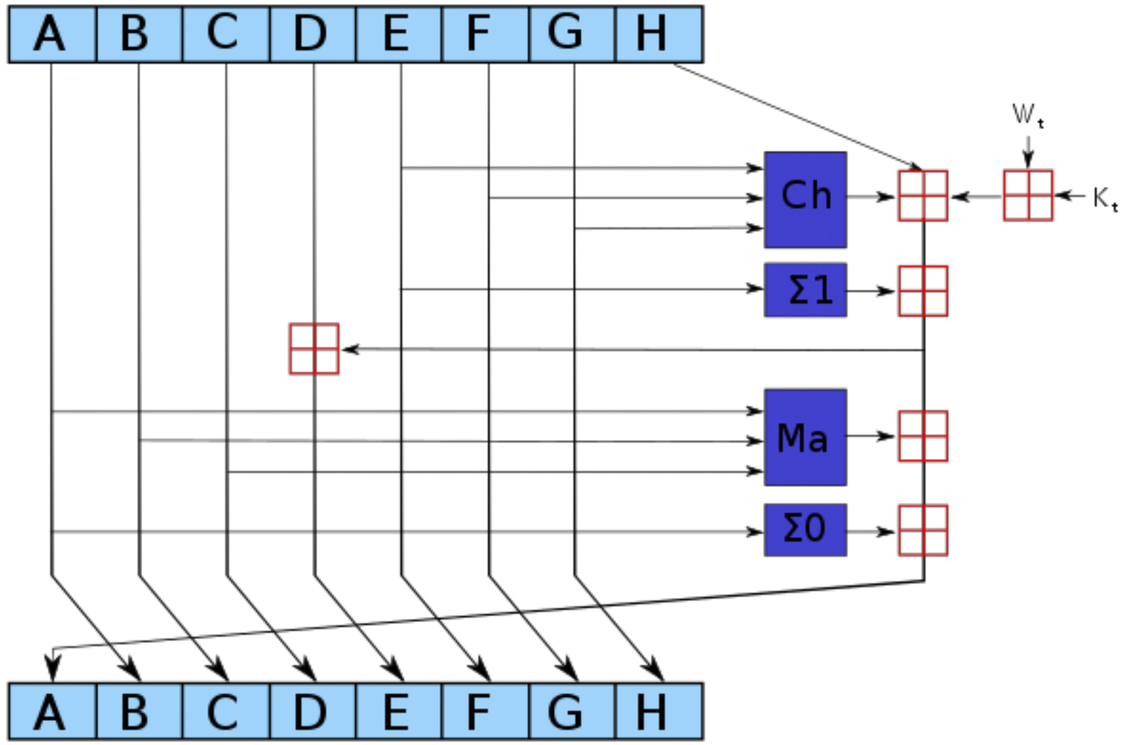


SHA2

長度：224, 256, 384, 512 bit

特徵：Constant 0x428a2f98

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

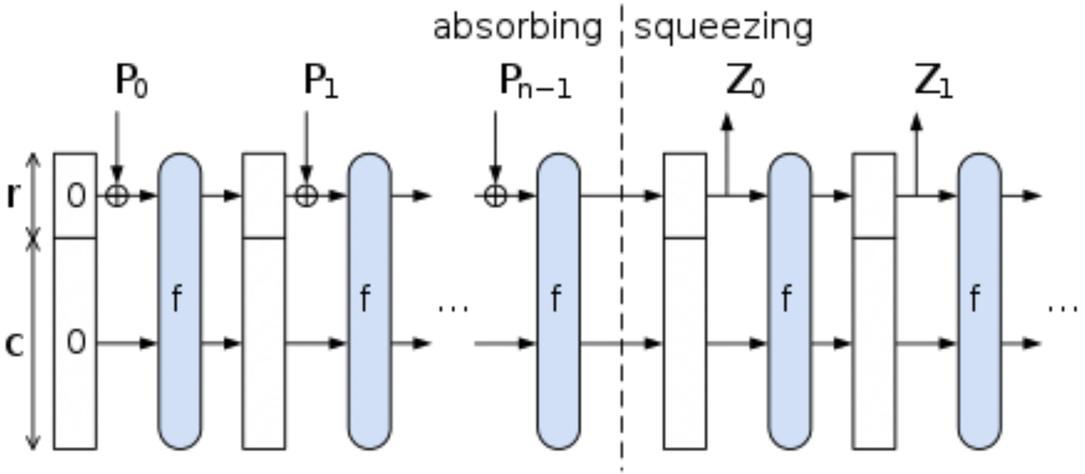


SHA3

長度：224, 256, 384, 512 bit

特徵：0x8000000080008081

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



[LAB] Hash Encrypt

長度擴充攻擊 (LEA Attack)

LEA 情境

- 有一台 Service 接收 data 和 signature 驗證後執行。
- data: count=10&lat=37.351&user_id=1&long=-119.827&waffle=eggo
- Signature: md5(secretkey + data) = 6d5f807e23db210bc254a28be2d6759a0f5f5d99

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

LEA 情境

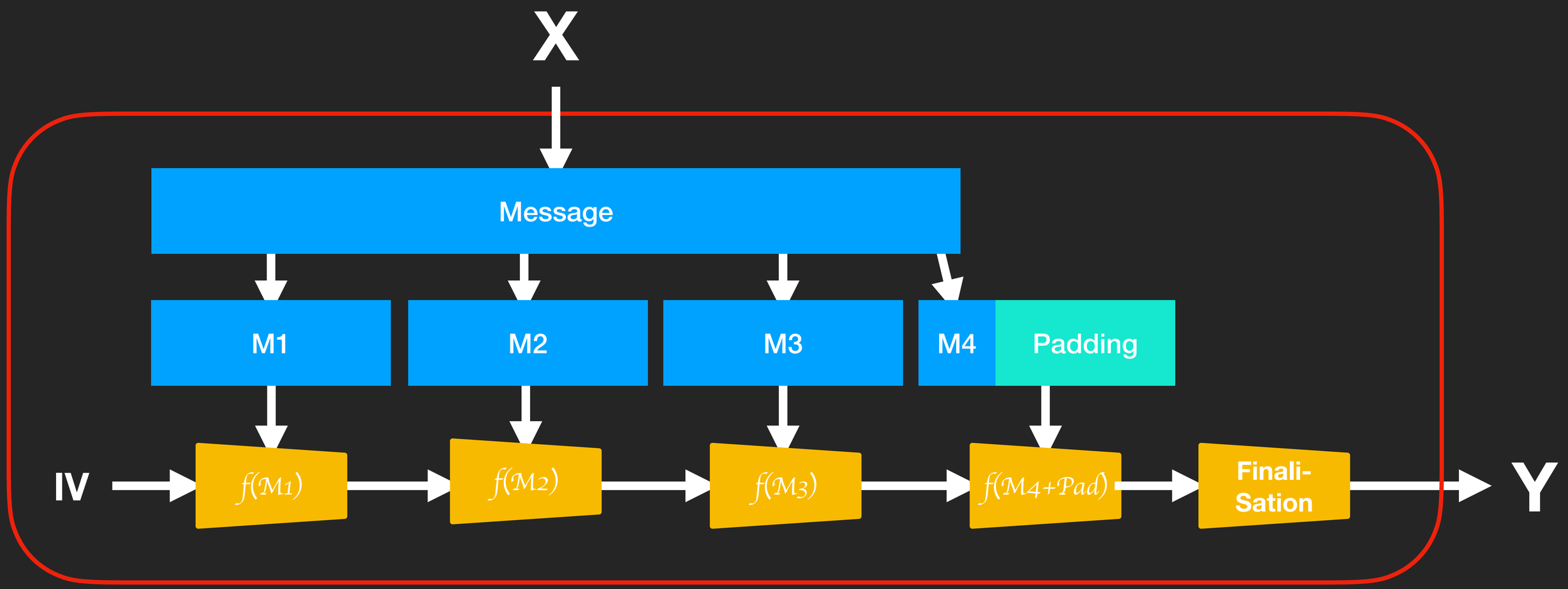
- 攻擊者可以由提供的 data 和 signature 自己生成下面訊息，繞過驗證此系統：

- data: count=10&lat=37.351&user_id=1&long=-119.827&waffle=eggo**\x80\x00\x00**
  \x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
  \x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
  \x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
  \x00\x00\x00\x02\x28&waffle=liege

- Signature: md5(secretkey + data) = 0e41270260895979317fff3898ab85668953aaa2

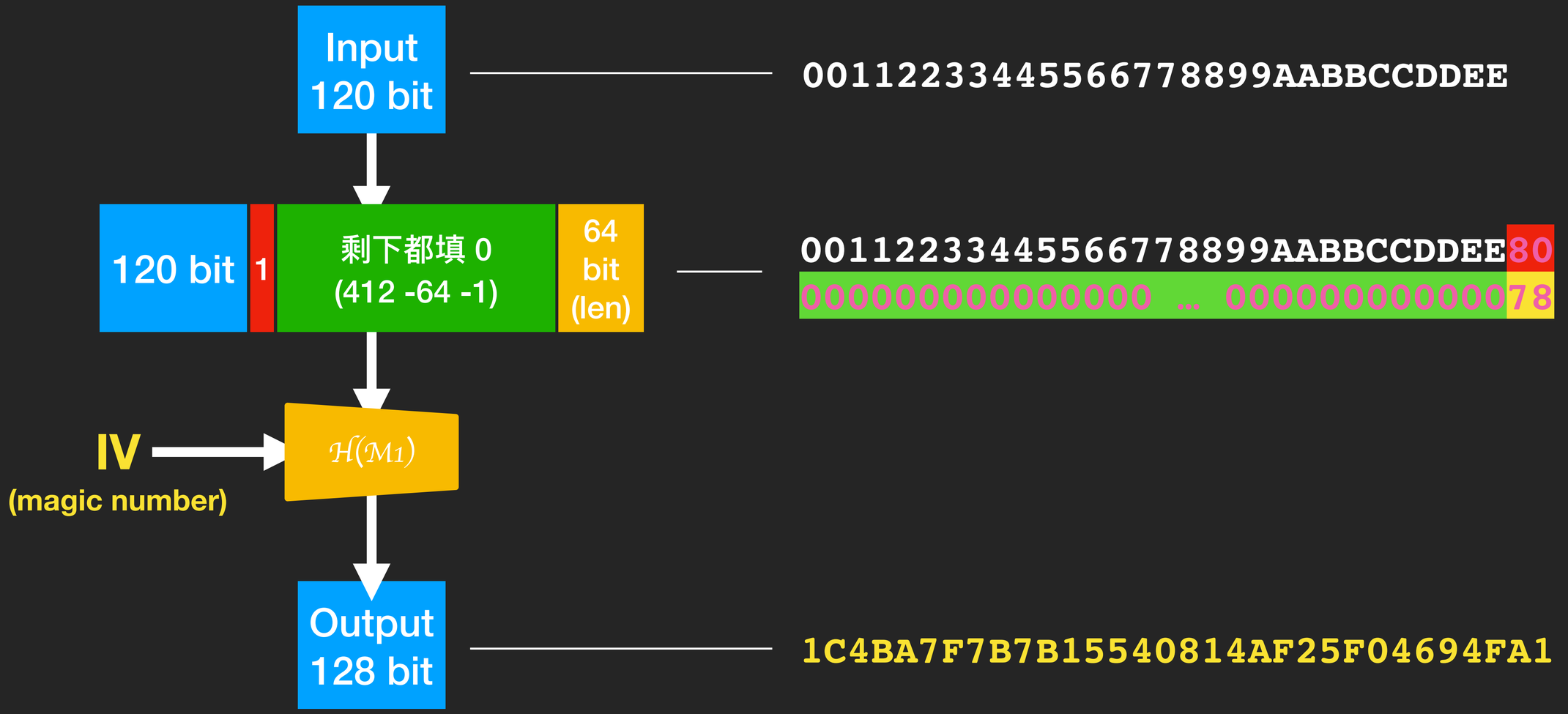
https://en.wikipedia.org/wiki/Length_extension_attack

Merkle–Damgård construction

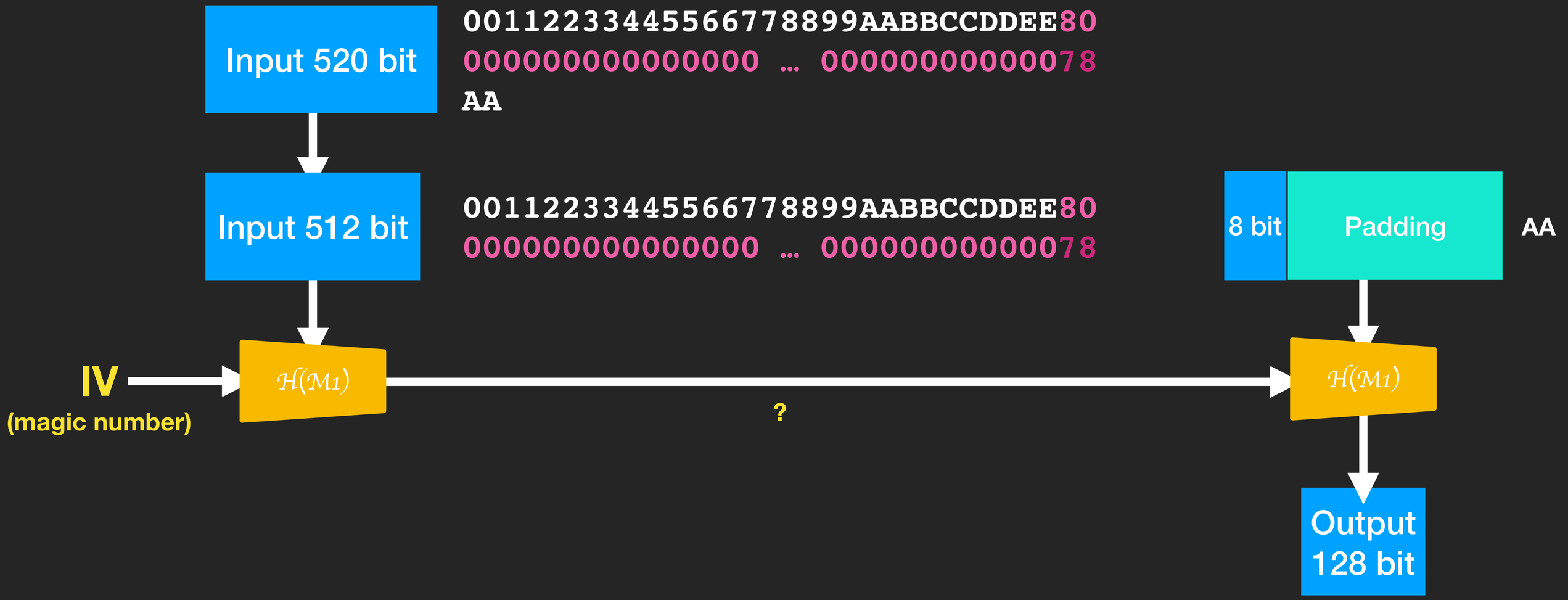


MD5, SHA1, SHA2

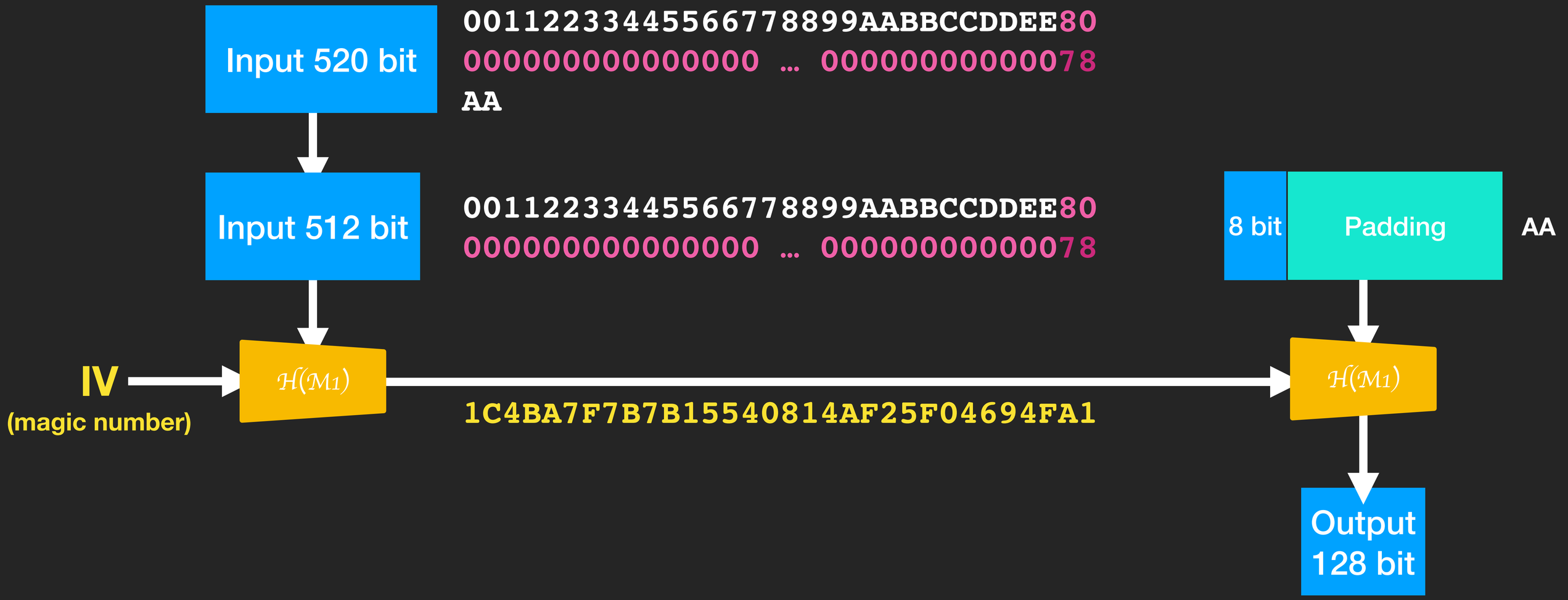
長度擴充攻擊 | Length extension attack



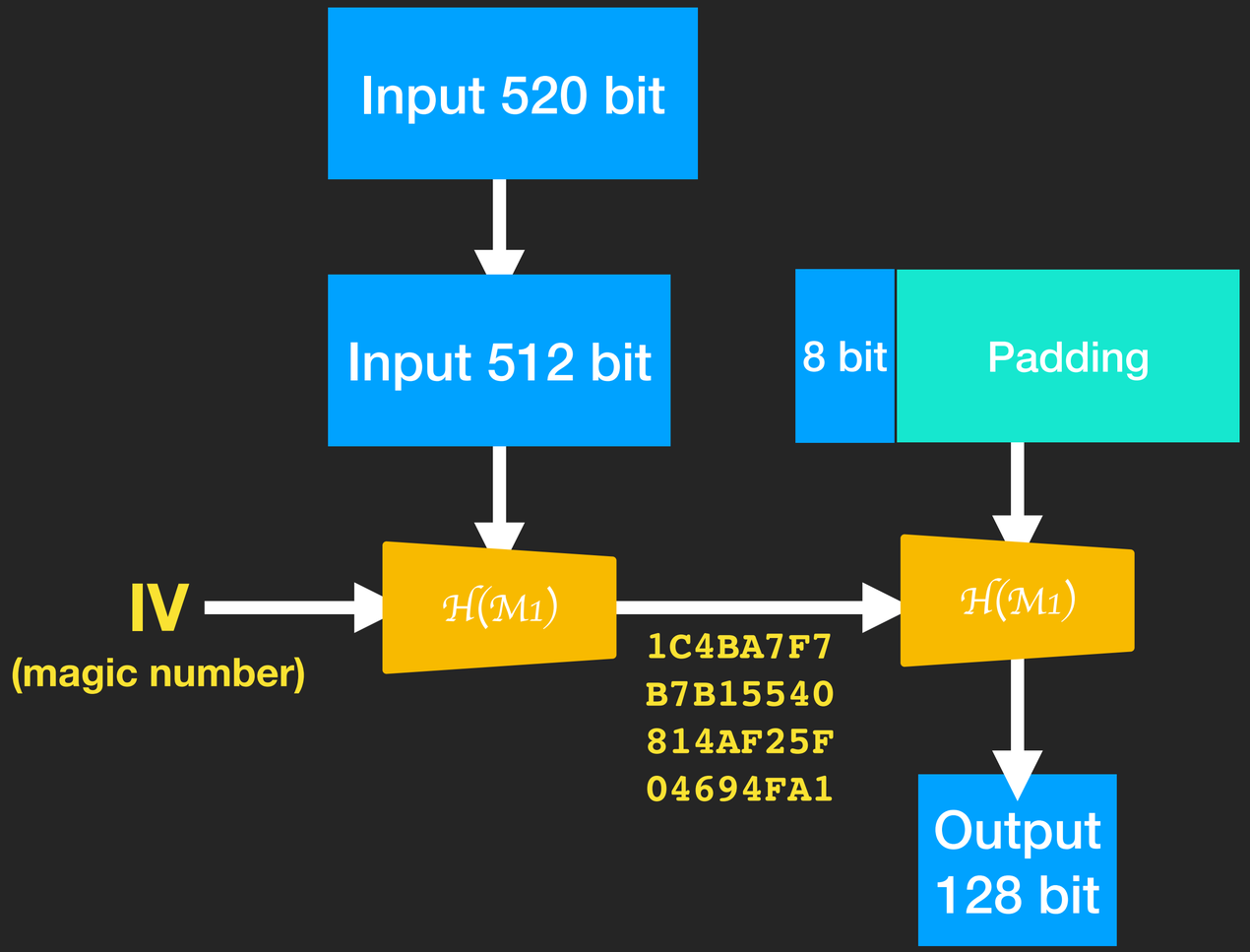
長度擴充攻擊 | Length extension attack



長度擴充攻擊 | Length extension attack



長度擴充攻擊 | Length extension attack

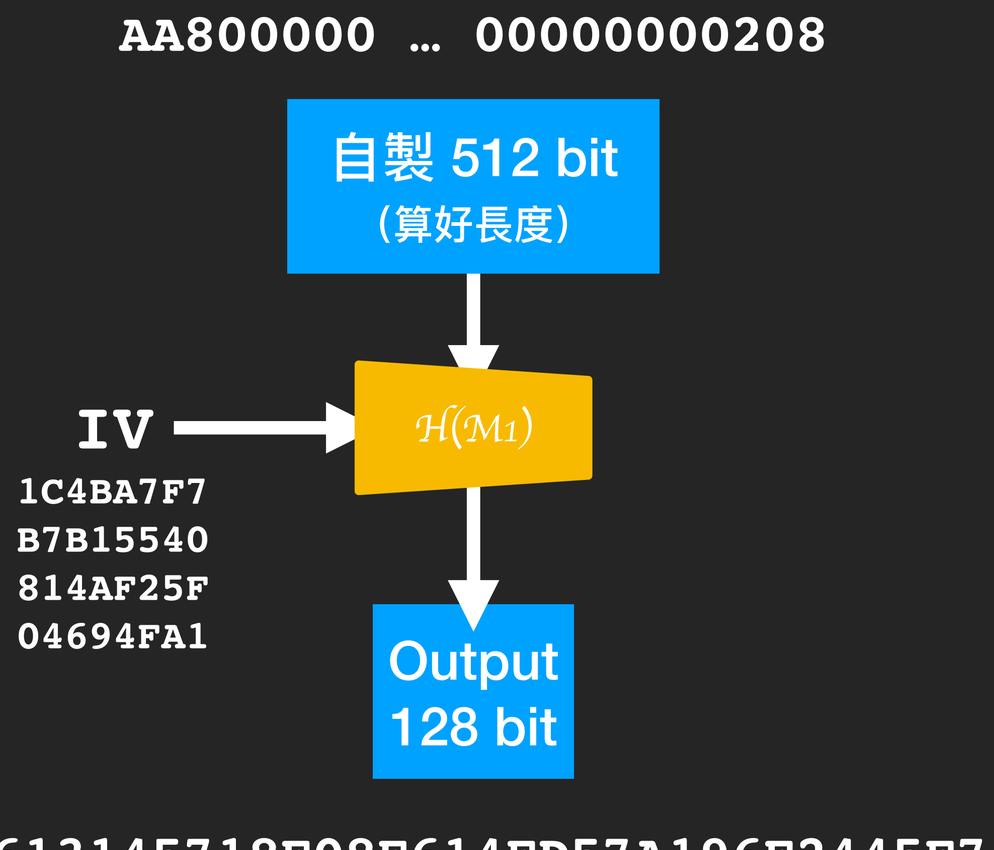


AA800000 … 00000000208

E6131457 18E08F61 4ED57A19 6E2445E7

長度擴充攻擊 | Length extension attack

- 若已知 Hash 值，且可輸入 bytes
- 即可任意加字在後方
- 自行算出其 hash value



[LAB] OWO_Apple Shop