# 加密

- 什麼是資料**加密**？
  - 為使資料僅被合法的人得知， unexpected 的人無法看出端昵
- 資料為什麼要加密？
  - 電腦、網路、科技設備發達，資料都在公開的網路上傳遞，資料需經過加密才能確保安全
- 密碼學是一種藝術
  - 加密 ＋ 解密
  - 加密→ 容易
  - 解密→ 難如登天

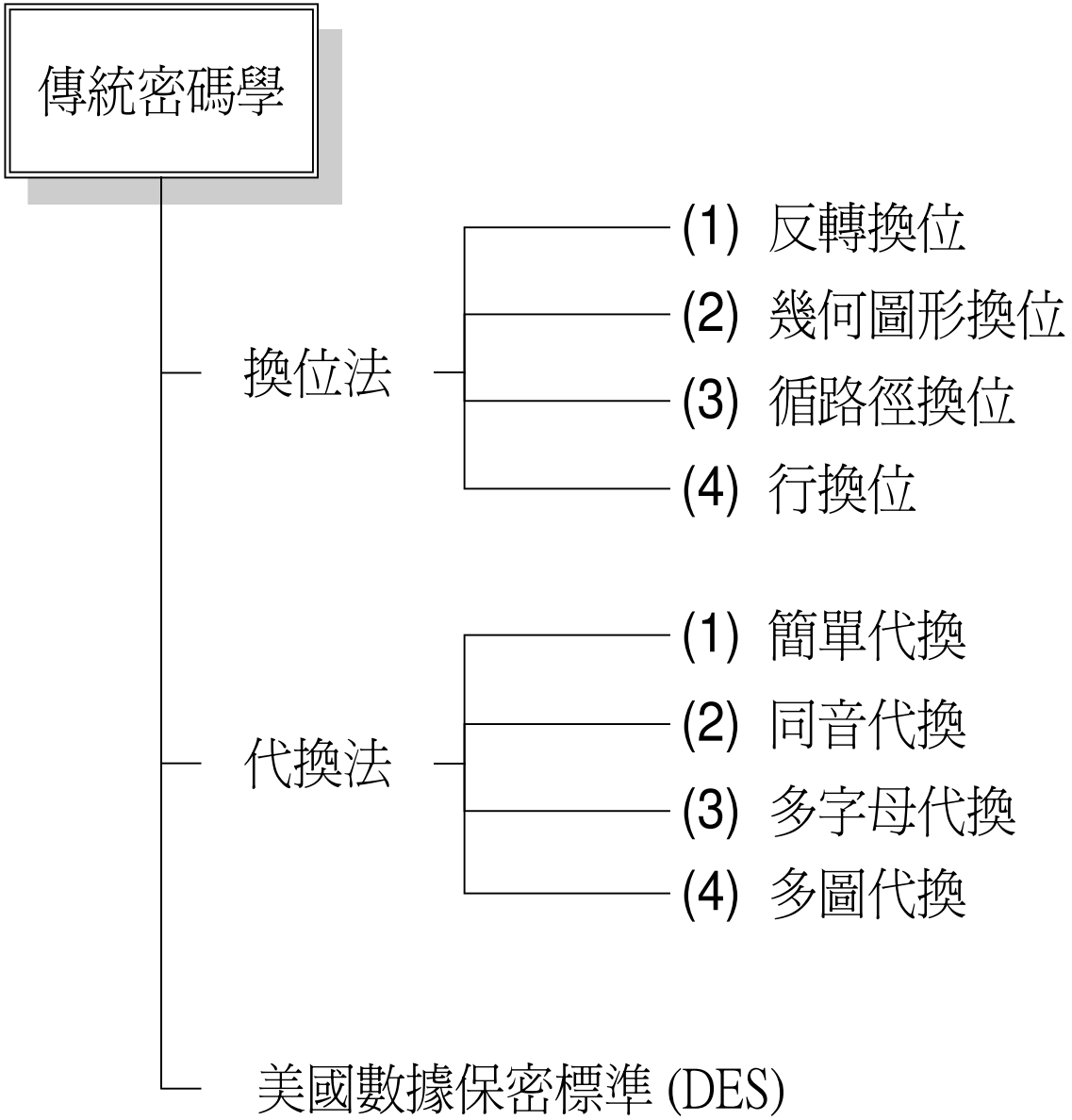
# 加密 (Cont.)

- 保密技術的價值
  - 保密程度 → 越高越好
  - 金鑰大小 → 越小越好
  - 加解密運算的複雜度 → 越簡單越好
  - 錯誤傳播 → 越少越好
  - 明文擴充 → 越少越好
- 測試加密系統的方式
  - 密文攻擊 (Chiper text attack)
  - 明文攻擊
  - 選擇明文攻擊

# 加密 (Cont.)

- 解密
  - 將加密後的訊息反打亂變成可閱讀的訊息
- 加解密系統
  - 收送雙方須先溝通好加解密方式
  - 使用者即使知道加密協定，在其不知道金鑰的情況下在有限時間內無法解得正確訊息
- 加密的基本型態
  - 換位 (Transposition)
  - 替換 (Substitution)

# 加密 (Cont.)


傳統密碼學
換位法
(1) 反轉換位
(2) 幾何圖形換位
(3) 循路徑換位
(4) 行換位
代換法
(1) 簡單代換
(2) 同音代換
(3) 多字母代換
(4) 多圖代換
美國數據保密標準 (DES)

# 加密 (Cont.)

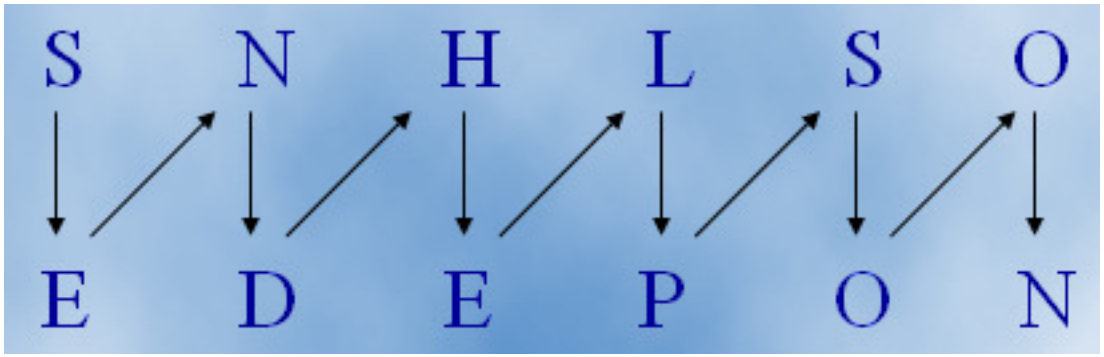
- 反轉換位法
  - 明文： MEET ME MONDAY MORNING
  - 密文： GNINROM YADNOM EM TEEM
- 幾合圖形換位
  - 明文： CONCEAL ALL MESSAGES
  - 密文： CLOMNECSESAALGAELS

明文：CONCEAL ALL MESSAGES

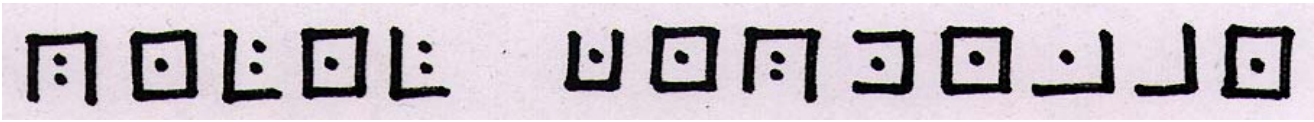
| | |
|---|---|
| CL | CON |
| OM | CEA |
| NE | LAL |
| CS | LME |
| ES | SSA |
| AA | GES |
| LG | |
| AE | |
| LS | |

# 加密 (Cont.)

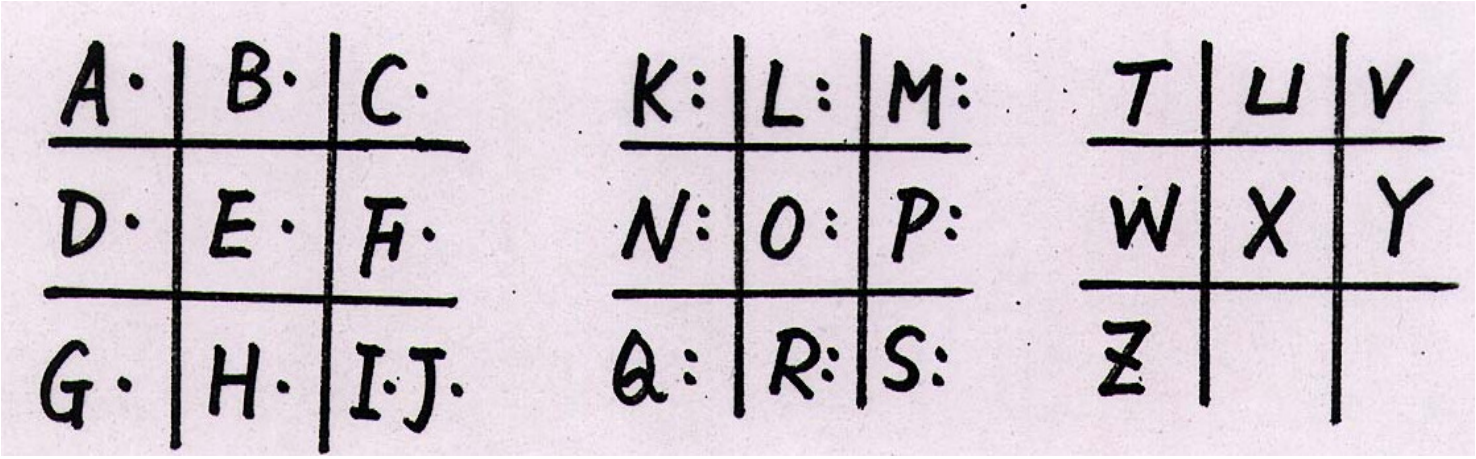
- 循路徑換位法
  - 明文： SEND HELP SOON
  - 密文： SNHLSOEDEPON



# 加密 (Cont.)

- 代換法
  - 明文： M=RENAISSANCE
  - 密文：$E_K$(M)=XKNAUGGANSK
- Affine 轉換
  - 密文： 

依照下列取代法則取代而成

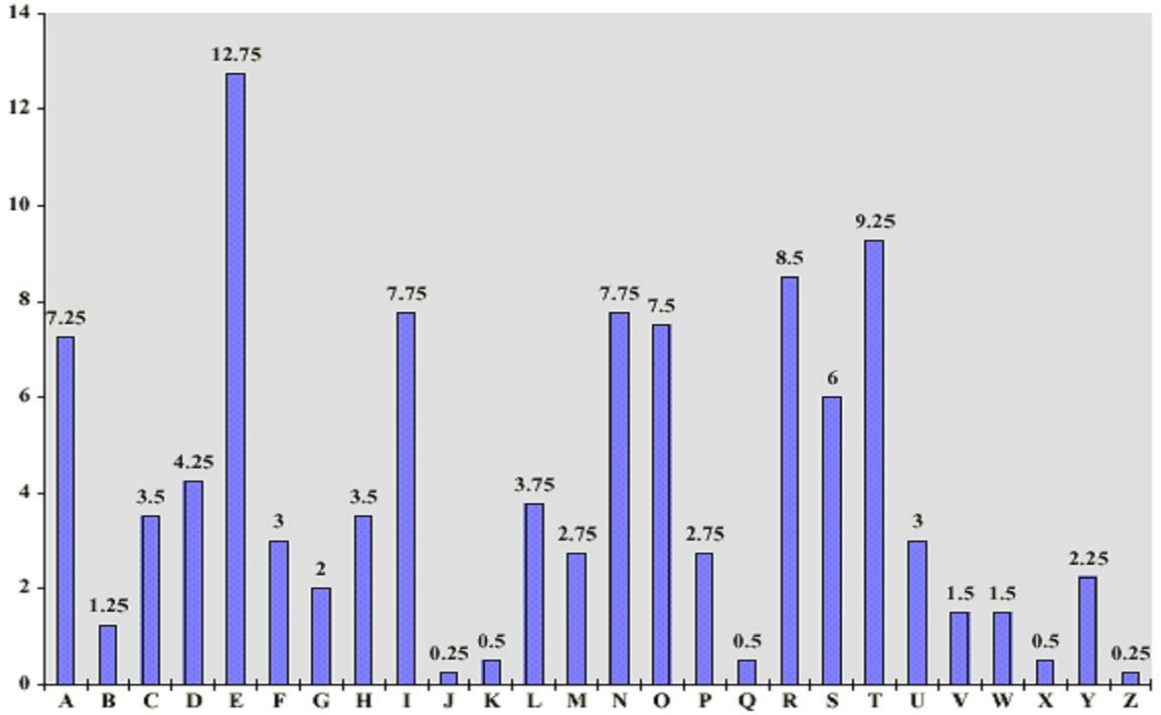


# 加密 (Cont.)

- 凱撒加密法 (Caesa Cipher)
  - 換位加密
    - A DOG
    - 位移 1 位 → B EPH
    - 位移 2 位 → C FQI
    - 位移 -1 位 → Z CNF
  - 容易被破解
    - 利用統計分析的技巧
    - 分析一般文章中最常出現的字母
    - 分析一份密文出現最多的字母
    - 比較兩者即有可能被破解

# 加密 (Cont.)

| | 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 |



# 加密 (Cont.)

- 替換符號系統 (Substitution alphabet)
  - 每一個字母都找另一個字母替代
    - Ex. 獵殺 U-571 裡的密碼機
  - 不同於換位加密
- 多字母替換法 (Multi-alphabetic)
  - 一次可以選多個位移字母
    - Ex: 位移 1, 2, -1
    - A DOG ==> B FNH
- 舊方法已存在多年，如今已不安全

# 加密 (Cont.)

- 二進制的基本運算
  - AND
    $$\begin{array}{r} 1101 \\ 1001 \\ \hline 1001 \end{array}$$
  - OR
    $$\begin{array}{r} 1101 \\ 1001 \\ \hline 1101 \end{array}$$
  - XOR
    $$\begin{array}{r} 1101 \\ 1001 \\ \hline 0100 \end{array}$$
- 這三種運算的差別在哪？

# 加密 (Cont.)

- 以 XOR 加密
  - 將明文轉換成 ASCII 碼
    - A DOG ==> 065 032 068 079 071
  - 將 ASCII 碼轉成二進制
    - 0100 0001, 0100 0100, 0100 1111, 0100 0111

A D O G

  - 以加密金鑰 (1111 0111) 進行 XOR 加密運算

| | | | |
|---|---|---|---|
| 01000001 | 01000100 | 01001111 | 01000111 |
| 11110111 | 11110111 | 11110111 | 11110111 |
| 10110110 | 10110011 | 10111000 | 10110000 |

# 加密 (Cont.)

- 以 XOR 解密
  - 以解密金鑰 (1111 0111) 進行 XOR 解密運算

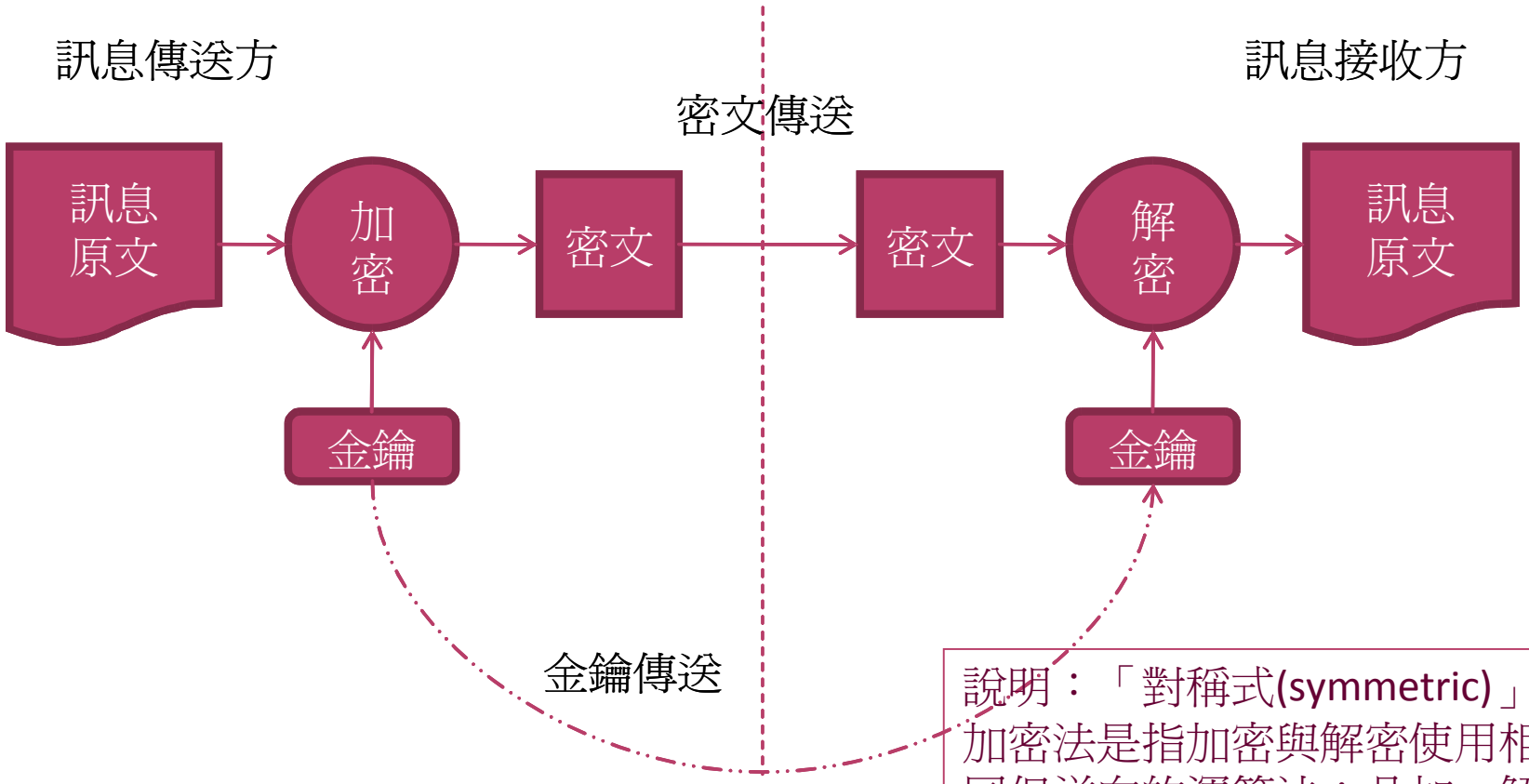
| | | | |
|---|---|---|---|
| 10110110 | 10110011 | 10111000 | 10110000 |
| 11110111 | 11110111 | 11110111 | 11110111 |
| 01000001 | 01000100 | 01001111 | 01000111 |

- 將二進制轉成 ASCII 碼
  - 0100 0001, 0100 0100, 0100 1111, 0100 0111

A D O G

# 現代的加密技術

- 對稱式加密




說明：「對稱式(symmetric)」加密法是指加密與解密使用相同但逆向的運算法；且加、解密使用相同的金鑰。

# 現代的加密技術 (Cont.)

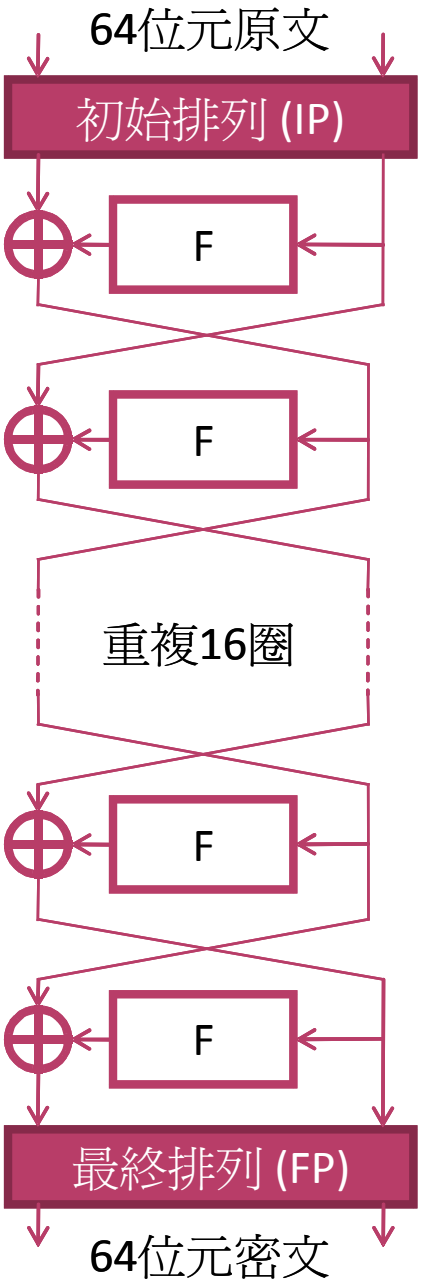
- Blowfish
  - Bruce Schneier (1993 年提出 )
  - 將 64 位元明文區段加密成 64 位元的密文區段，屬於區塊加密法
  - 以訊息的『區塊』做為加解密單位
  - 金鑰長度可變，其長度範圍為 32~448 位元
  - 特性 :
    - 快速：以 32 位元的電腦加密一個位元只需 18 時脈
    - 小巧：記憶體需求為 5K 以下
    - 架構簡單易於實作
    - 可變的安全性

# 現代的加密技術 (Cont.)

- 資料加密標準 (Data Encryption Standard, DES)
  - 1973 年美國國家標準局公開徵求加密系統 , IBM 的 Lucifer 被選中
  - 對稱式加密 , 每次加密 64 位元資料
  - 作法：
    - 將資料打亂
    - 把一半的資料 (i.e. 32 位元 ) 經由 F-function 加密，加密後的結果再與另外一半的 32 位元資料進行 XOR 運算
    - 進行 16 回合類似的運作，再將最後結果打亂
    - 輸出加密後的 64 位元密文

# 現代的加密技術 (Cont.)

- DES 使用 56 位元的 Key
  - $2^{56} \approx 10^{17}$
  - 若查對速率為 $10^6$ key/sec $\approx 3 \times 10^{13}$ key/yr
  - 使用暴力攻擊法要 3000 年
- 優點：
  - 運算速度相對於非對稱式加密快
  - 演算法設計得宜，加密強度很高
  - 相關工具容易取得，且大多免費
  - 軟、硬體建置容易

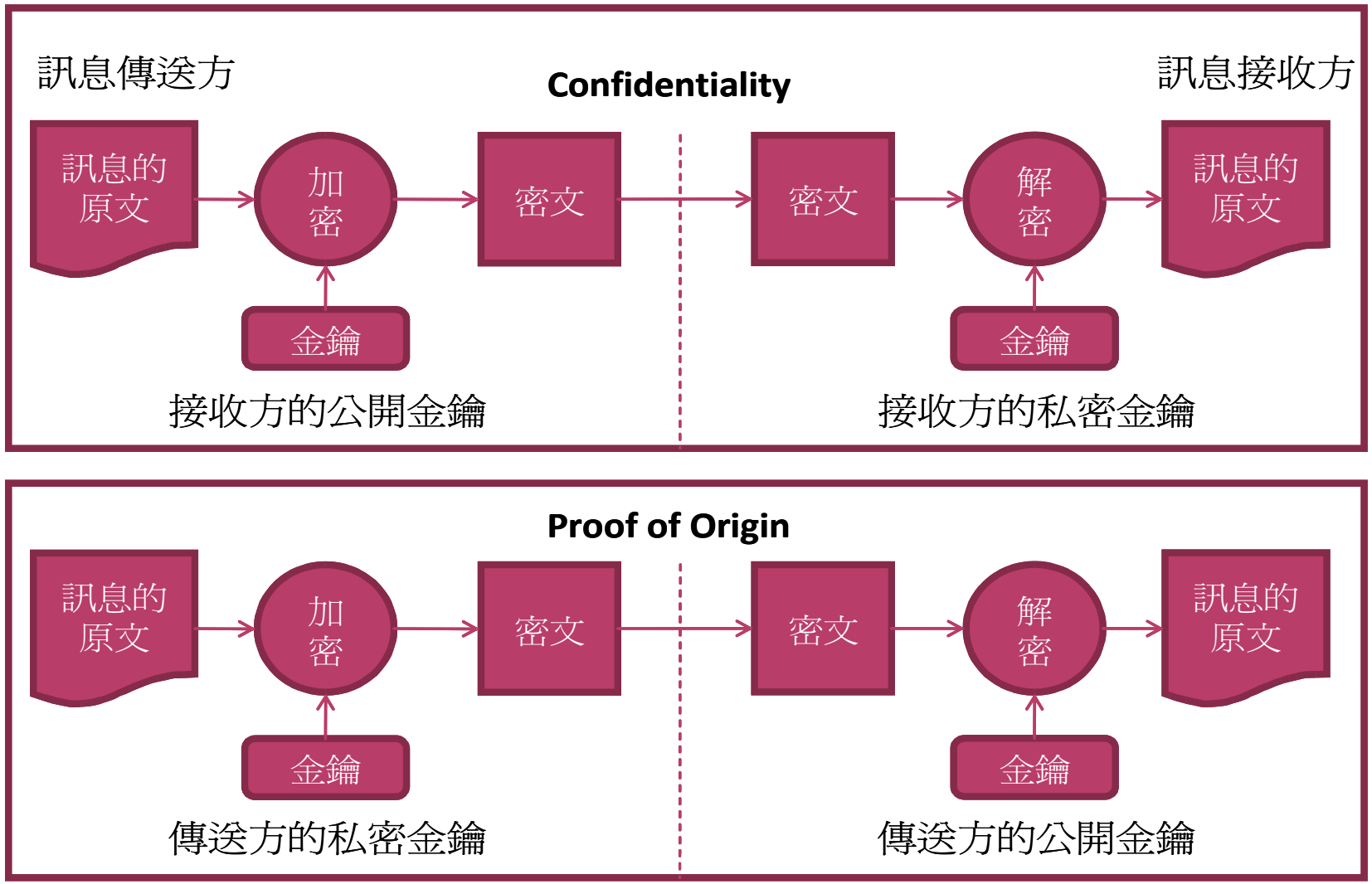


# 現代的加密技術 (Cont.)

- 缺點：
  - 加解密使用同一把 key ，傳送 Key 有風險
  - 金鑰管理的複雜性
    - 每兩個人通訊便要用一把 key ，當有 n 個人時便要管理 n(n-1)/2 把 key
  - 對稱式加密可對文件加密，但文件的『來源證明』無法同時保證，必須靠非對稱式加密
- 對稱式金鑰加密法的最大問題
  - 如何傳送 Key 給解密端？

# 現代的加密技術 (Cont.)

- 公開金鑰加密法（非對稱式加密）解決了傳送 Key 的問題



# 現代的加密技術 (Cont.)

- 公開金鑰加密法
  - 與大質數、因數分解、數論 ... 等有關
  - 是目前最被廣泛討論與使用的技術
  - 特性：
    - 改進傳統加密系統的缺點
    - 公開加密金鑰也不會造成洩密問題
    - 加密金鑰以 (e, m) 表示，解密金鑰表示為 d
    - d 控制的運算必須可從 (e, m) 加密的密語中解出明文

# 現代的加密技術 (Cont.)

- RSA (Rivest, Shamir, Adleman, 1978)
  - 選擇兩個質數分別為 p 跟 q
  - 找一個 e 滿足 (e, (p-1)(q-1))=1，即 e 與 (p-1)*(q-1) 互質
  - 令 m = p * q 且 0<M<m, 其中 M 為明文
  - 則對應於 M 的密文 C 可由 $E(M) = M^e \bmod m$
  - 令 d 為解密金鑰且滿足 $e \times d \equiv 1 \pmod{(p-1)(q-1)}$
  - 解密時可經由計算 $M = D(C) = C^d \bmod m$

# 現代的加密技術 (Cont.)

- 例子
  - 設取 $p = 5, q = 11$ 則 $(p-1)(q-1) = 4*10 = 40$
  - 另 $m = p*q = 5 * 11 = 55$
  - 取 $e = 3$, 且明文 $M = 7$
  - 則明文經加密運算 $M^e \bmod m = 7^3 \bmod 55 = 13$
  - 又 d 必須滿足 $e*d \equiv 1 \pmod{40}$, 即 $3*d \equiv 1 \pmod{40}$
  - 於是 d=27
  - 解密時 $C^d \bmod m = 13^{27} \bmod 55 = 7$

# 現代的加密技術 (Cont.)

- 優點：
  - 保護機密性隱私性
    - 文件需要接收方的私密金鑰才能解開
  - 可被應用於存取控制
    - 私密金鑰只有一位使用者持有
  - 可做到來源證明
    - 傳送方以其擁有的私密金鑰對文件加密，接收方以傳送方的公開金鑰解密，因此，如果文件可正確解得則傳送方無法否認傳送此文件

# 現代的加密技術 (Cont.)

- 若欲分解 m = p*q
  - 假設 m 為 200 位元
  - 電腦 $10^6$ 指令 /sec
  - 需要約 $10^6$ 年才能求得
- 保密性高，不需傳遞金鑰

- 缺點：
  - 複雜性大，運算速度慢 (DES 比 RSA 快 1000 倍 )
    - 尋找大質數、因式分解、 mod 等非線性運算
  - 易造成嚴重傳播錯誤：分段加密

# 現代的加密技術 (Cont.)

- 雜湊函數 (Hash function)
  - 將任意長度的訊息字串轉化成固定長度的輸出字串，這個輸出字串稱之為『雜湊值』
  - 雜湊值是訊息原文的『濃縮』，任一訊息的雜湊值都要有一定程度的獨特性
  - 雜湊函數應為單向函數 (one-way function)，意即我們應該無法從雜湊值反推求得訊息原文
  - 不同的雜湊值間不該存在任何線性關係
    - 把兩個雜湊值合併 ( 如相加或者 XOR) 後得到的新值，不應該等於訊息原文做同樣處理後雜湊出來的結果

# 現代的加密技術 (Cont.)

- 一個訊息產生雜湊值後，應該無法以數學方法找到另外一個訊息也能產生相同雜湊值 ( 情況並非不存在，但應沒有方法可以找到 )
- 訊息原文只要稍有變動，雜湊值會有巨大改變
- 目前常用的雜湊函數：
  - MD2, MD4, MD5, SHA-1, SHA-256 等

# 現代的加密技術 (Cont.)

- 數位憑證
  - 是個體 ( 如持卡人、企業、銀行等 ) 在網路上進行資訊交流及商務活動之身份證明
  - 憑證是一個經憑證管理中心製作的數位簽章文件，其中包含擁有者資訊及其公開金鑰資訊
  - 依用途區分
    - 簽章憑證：對訊息進行數位簽章，保證認息的不可否認性
    - 加密憑證：對訊息加密，保證認息的真實性及正確性
  - 格式及內容遵循 X.509 標準

# 現代的加密技術 (Cont.)

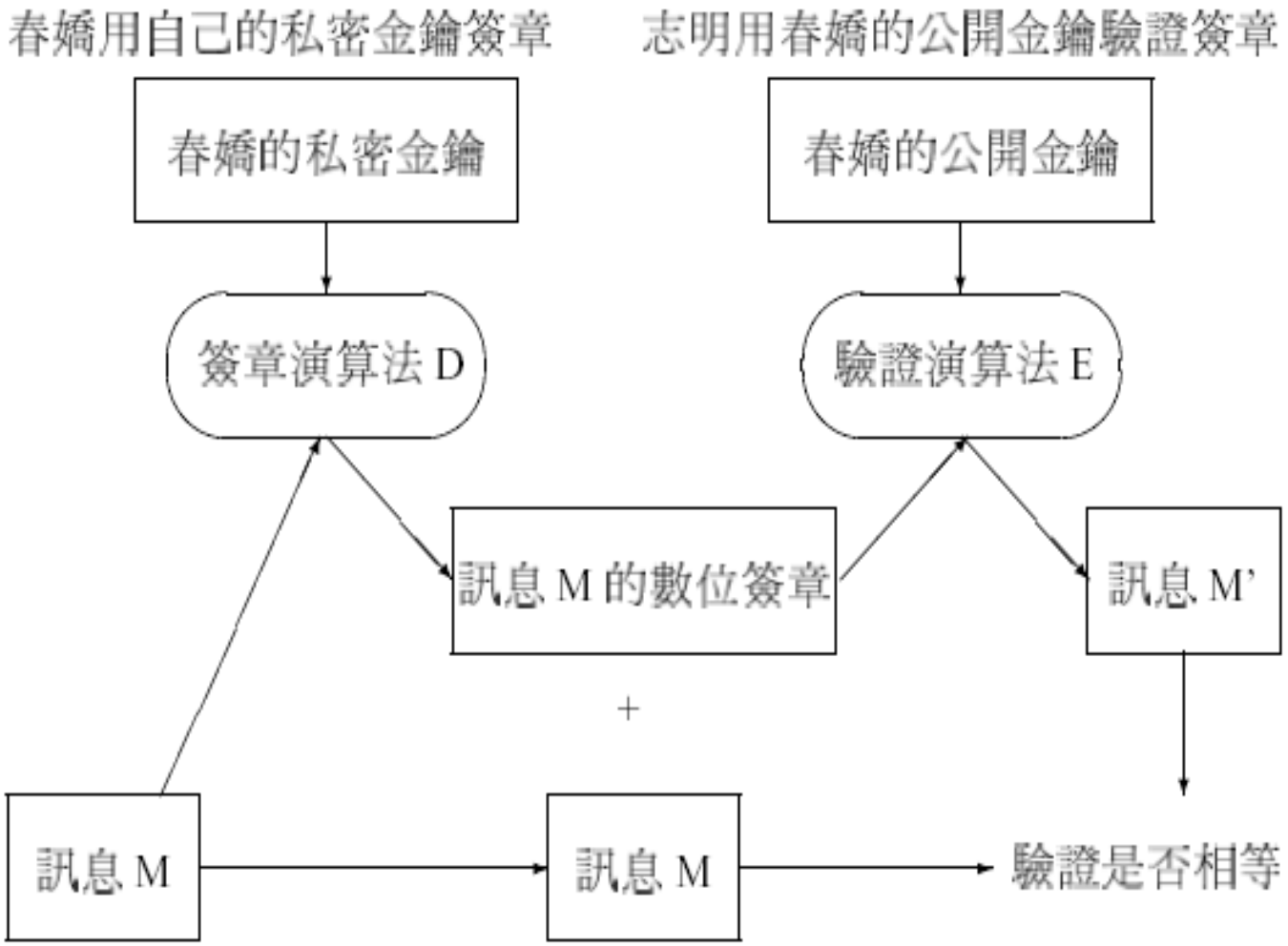
- X.509

| | | |
|---|---|---|
| 版本 (version) | V3 | X.509 版本編號 |
| 序號 (serial number) | 18 da de 91 … | CA指派給憑證的唯一序號 |
| 簽章算法(signature algorithm) | sha1RSA | CA用來數位簽署憑證的雜湊演算法 |
| 發行者 (issuer) | VeriSign Class 3 Public Primary CA | 關於CA的資訊 |
| 有效期自 (valid from) | 2006/11/8 | 憑證有效期間的開始日期 |
| 有效期至 (valid to) | 2036/7/1 | 憑證有效期間的最後日期 |
| 主體 (subject) | Bank of ABC | 發給憑證的目標個人、電腦、裝置或憑證授權單位名稱 |
| 公開金鑰 (public key) | RSA (2048)<br>2a 14 5c 70 … | 與憑證相關的公開金鑰類型及長度，與金鑰數據 |
| 延展資訊 (extension) | V3 定義的諸多延伸欄位 | |
| CA 簽章 (CA signature) | 使用CA私密金鑰，透過憑證演算法識別項欄位中所指定的演算法，所做出的實際數位簽章 | |

# 現代的加密技術 (Cont.)

- 數位簽章 (Digital signature)
  - 效力等同於親筆簽名
  - 數位簽章無法被偽造
  - 數位簽章由簽章者之**私密金鑰**產生
  - 驗證者以簽章者之**公開金鑰**驗證
  - 簽章上有時間戳記，以維其有效期間
  - 受簽章保護之文件內容只要一經修改，則簽章驗證就會失敗

# 現代的加密技術 (Cont.)



# 金鑰管理

- 金鑰管理 (Key management)
  - 指處理金鑰的流程
    - 金鑰的產生 (Key generation)
    - 金鑰的儲存及配送 (Key storage and distribution)
    - 金鑰託管 (Key escrow)
    - 金鑰過期 (Key expiration)
    - 金鑰收回 (Key revocation)
    - 金鑰中止 (Key suspension)
    - 金鑰復原與歸檔 (Key recovery and achival)
    - 金鑰更新 (Key renewal)
    - 金鑰銷毀 (Key destruction)