Module 5:鑑別技術
Authentication

學習目的

1. 探討鑑別技術的需求及功能、訊息鑑別與身份鑑別技術；尤其對訊息鑑別碼(MAC)的演算法(例如DAA、SHA-1及HMAC法等)作詳細的說明，以確保資訊傳輸的完整性。

2. 討論通行識別碼 (PAS)原理、數位簽章及目前發展中的多維身份鑑別技術(整合生物特徵、智慧卡及通行識別碼等身份鑑別方法)，讓學生熟悉身份鑑別原理與最新技術的發展。

3. 本模組利用三個鐘點，介紹以下四個小節 (1)鑑別性的需求與功能(2)訊息鑑別技術(3)身份鑑別技術(4)專案實作，增進同學對鑑別技術的了解，專題實作可作為學生homework。

Module 5:大綱

Module 5-1: 鑑別性的需求與功能(*)

Module 5-2:訊息鑑別技術 (**)

Module 5-3:身份鑑別技術(**)

Module 5-4:專案實作(**)

ul

* 初級(basic):基礎性教材內容
**中級(moderate):教師依據學生的吸收情況，選擇性介紹本節的內容
***高級(advanced):適用於深入研究的內容

u1

* 初級(basic):基礎性教材

**中級(moderate):教師依據學生的吸收情況，選擇性介紹本節的內容

***高級(moderate):適用於深入研究的內容

user, 2007/2/14

Module 5-1:鑑別性的需求與功能

鑑別協定

- 在現實的世界裡，要驗證一個人的身分有許多種，如根據長相特徵、看證件、比對指紋或者是特殊持有物如金鑰等。

- 但是這些方法在網路世界裡通常是行不通的，因為現在的電腦還無法做到如同人類一般的判斷，因此在電腦網路裡必須借由溝通雙方所共同信賴的驗證程序來驗證對方身份，而在網路裡這種方式稱作為鑑別協定（Authentication Protocol）。

鑑別的需求

- 在日常生活中，有不法之徒會盜用信用卡、偽造身份證等。

- 而在網路環境裡，懂得TCP/IP協定的人可能可以經由一些協定的弱點來竊取你的帳號通行碼進而盜用你的帳號，偷取你的機密資料，或是偽裝成你的身份去跟別人做線上交易等等。

鑑別的需求

- 鑑別的目的就是(1) 在進行網路連線時，鑑別對方的身份。(2)鑑別傳遞的訊息是否遭到修改。

- 在此，以下列舉了一些網路上的可能發生的情況以說明為什麼在網路需要作鑑別的需求。

鑑別需求的理由

1. **洩露(Disclosure)：**

   讓未授權的人員知道訊息內容。

2. **流量分析(Traffic analysis)：**

   分析雙方之間的訊息樣式。在以連結(connect-oriented)為導向的應用程式中，入侵者可預測連結建立的頻率與使用的時間；在以非連結(connectionless)為導向的應用程式中，入侵者可判斷訊息的數目與長度。

3. **假冒（Masquerade）：**

   第三者可以假造出某人的電腦IP位址所發的封包傳送給欲攻擊的電腦，來擾亂正常連線。

鑑別需求的理由

4. **修改內容（Content modification）：**

   第三者可以對網路上截取到的資料做加入、刪除、置換或修改，使資料失去正確性。

5. **竄改順序（Sequence modification）：**

   第三者也可以截取到資料後，修改資料的先後次序以造成接收端的混亂。

6. **竄改時序（Timing modification）：**

   第三者也可以截取到資料後，以延遲或重送資料，欺騙接收端第三者為合法的使用者。

鑑別需求的理由

7. **來源端否認（Source Repudiation）：**

   對於傳送一些關鍵敏感資料，如不做來源端資訊鑑別，則來源端將來可能會發生否認曾送出或接收到該資料的情況。

8. **目的端否認（Destination Repudiation）：**

   對於傳送一些關鍵敏感資料，如不做目的端資訊鑑別，則目的端將來可能會發生否認曾送出或接收到該資料的情況。

鑑別需求的技術

- 預防前兩種攻擊可使用加密技術。
- 預防第三~第六種攻擊可使用訊息鑑別技術。
- 預防第七種攻擊需使用數位簽章技術。
- 一般使用數位簽章技術也可預防第三~第六種攻擊。
- 預防第八種攻擊需專為此攻擊設計的數位簽章與協定技術。

鑑別需求的技術

- 訊息鑑別技術可鑑別訊息的來源，主要功能包括:
  - 可鑑別訊息在傳輸過程是否遭到修改
  - 鑑別訊息的順序與時序在傳輸過程是否遭到修改
  - 防止來源端及目的端的否認行為

- 數位簽章是訊息鑑別技術中的一種，用以防止來源端及目的端的否認行為。

鑑別性的功能

- 訊息鑑別與數位簽章技術一般來說可區分成兩個層次
  - 產生訊息鑑別碼: 利用某一種函式產出訊息鑑別碼
  - 驗證訊息鑑別碼: 驗證訊息鑑別碼的正確性

Module 5-2:訊息鑑別技術

前言

- 本節我們將介紹產生訊息鑑別碼的三類方法：
  - 訊息加密(Encrypted Message)
  - 訊息鑑別碼(Message Authentication Code, MAC) (*)
  - 雜湊函式(Hashing Function) (*)

u2

u2

* 選擇性(optional)介紹的章節:教師依據學生的吸收情況，選擇性介紹本節的內容

** 先進(advanced)的章節:適用於深入研究的內容

user, 2007/1/24

(一) 訊息加密

- 訊息加密(Encrypted Message)，密文本身可被作為訊息鑑別碼。
- 加密方法包括第二章介紹的對稱式加密與非對稱式加密(公開金鑰)。
- 對稱式加密缺點是接受者很難自動判斷所收到的密文經解密即為正確的明文。
- 解決的方式是將明文強迫設計成容易識別的結構(block structure)，並在每個訊息之後加上錯誤偵測碼(error-detecting code)以確認明文是否遭受修改。

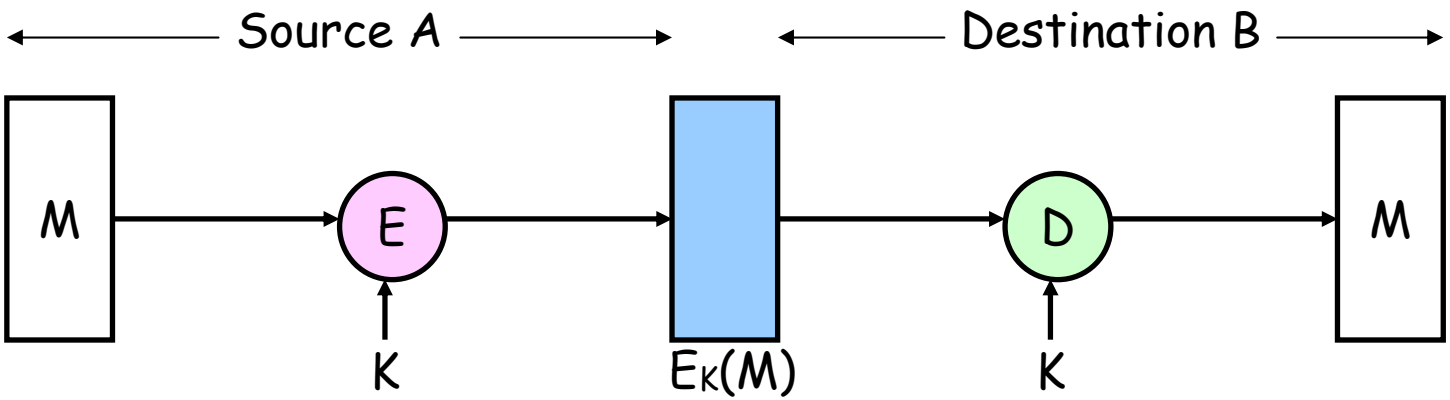
(一) 訊息加密

- 訊息加密也可以提供某種程度的確認性，可使用對稱式加密法與公開金鑰加密法，其分別具有不同效果。
- 如果採用對稱式加密法(如圖5-1(a))：
  - 接收者知道訊息必由傳送者產生
  - 因為只有傳送者與接收者知道所用的金鑰
  - 知道訊息無法被更動
  - 如果訊息具有適當結構的話，可用副本或總和檢查來偵測任何變更
  - 缺點為，接收者無法得知所收到的密文經解密即為正確的明文

(一) 訊息加密
—基本運用

- 如果採用公開金鑰加密(如圖5-1(b)(c)(d))：
  - (b)圖，加密程序無法確認傳送者，因為任何人都知道其公開金鑰。
  - (c)圖，無法確保資料的保密性，因為任何人都知道其公開金鑰，也就是說只要能攔截到就能把它解開。
  - (d)圖，合併使用以下兩種作法，使其兼具安全性與確認性
    - 傳送者用其私密金鑰來簽署訊息
    - 用接收者的公開金鑰來加密
  - 同樣的，必須能鑑別遭受竄改的訊息
  - 但其代價是必須對訊息作兩次公開金鑰加密

(一) 訊息加密
—基本運用

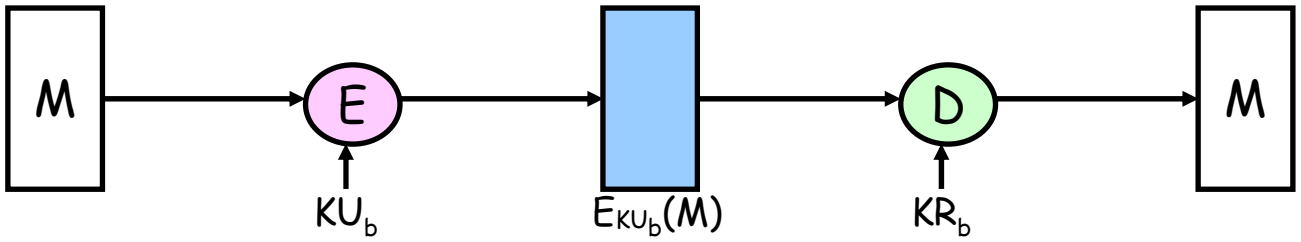


(a)傳統加密法：保密性與確認性

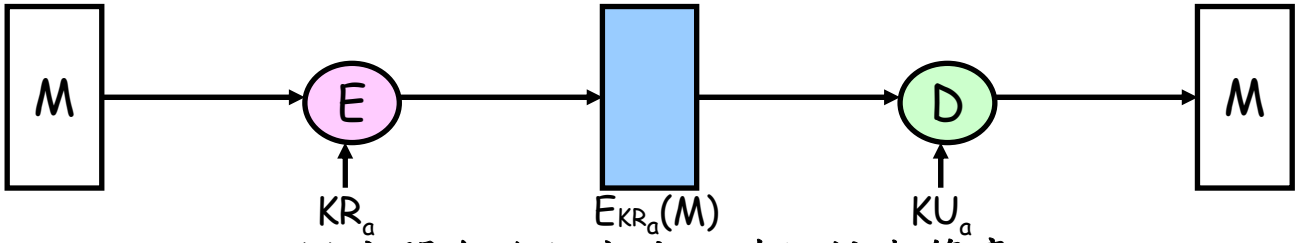
圖5-1 訊息加密的基本運用—對稱式加密法

返回

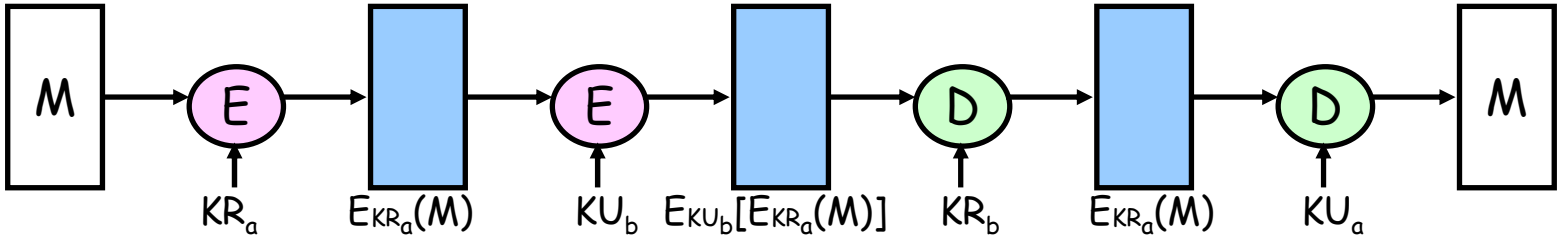
(一) 訊息加密
—基本運用



(b)公開金鑰加密法：保密性



(c)公開金鑰加密法：確認性與簽章



(d)公開金鑰加密法：保密性、確認性與簽章

圖5-1 訊息加密的基本運用—公開金鑰加密法

返回

(二) 訊息鑑別碼

- 訊息鑑別碼: 使用私密金鑰加密可產生固定長度的資料區段，稱為訊息鑑別碼(Message Authentication Code, MAC) 。

  MAC=C$_k$(M)，其中M為訊息， C為MAC函式， k為私密金鑰，MAC為訊息鑑別碼。

- MAC加在每個訊息之後一起傳送給接收端，接收端使用同一把私密金鑰來執行相同的運算，比對兩個MAC以確認明文是否遭受修改。

(二) 訊息鑑別碼
—特性

- 由演算法所產生的一小段固定長度的數字區段
  - 數字區段的內容取決於訊息與金鑰
  - 類似加密，但其為不可逆轉的加密機制
- 數字區段附加在訊息後端，當作簽章然後傳送給對方。
- 接收端以同樣方式產生訊息的鑑別碼，然後與原碼互相比對，若相同代表此訊息未遭竄改，並且的確來自傳送端，詳如圖5-2 (a)訊息的確認性。

(二) 訊息鑑別碼
—特性

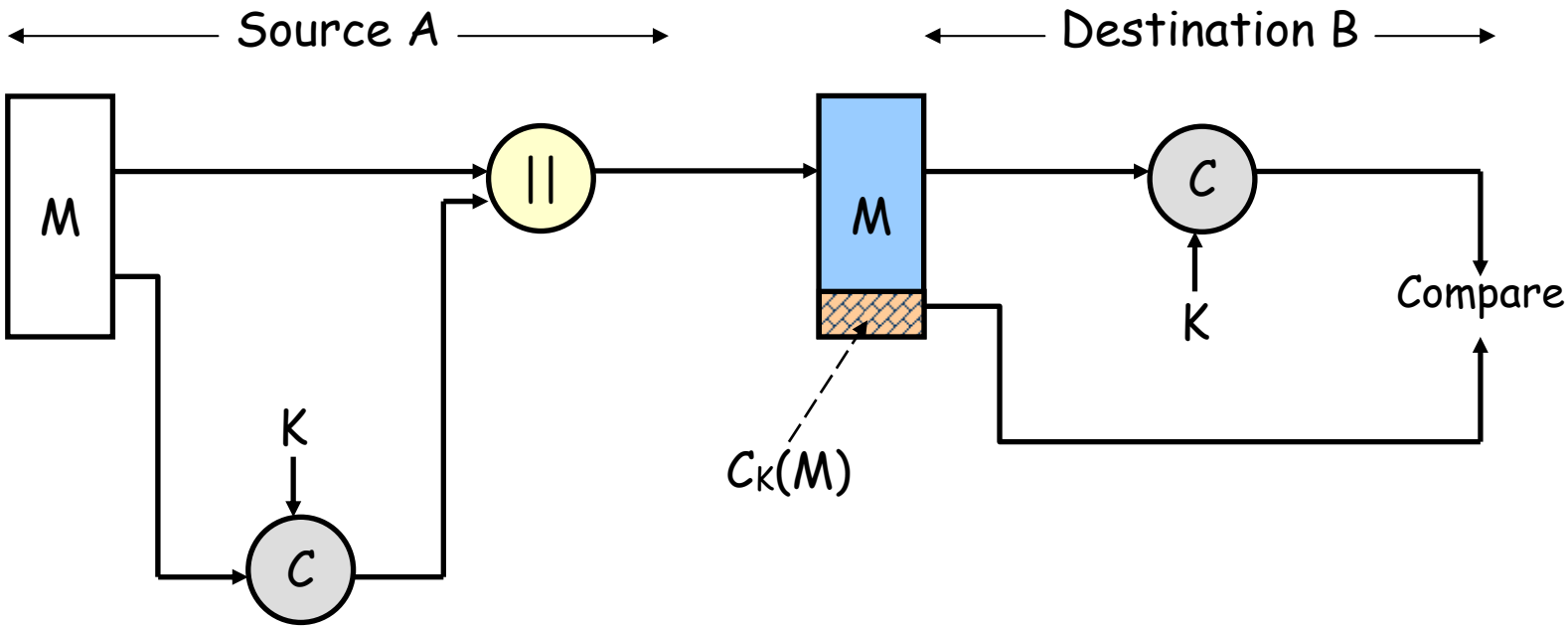


圖5-2 (a)訊息的確認性

(二) 訊息鑑別碼
—特性

- 圖5-2 (a)方法提供訊息的完整性，但無法保護訊息的機密性。

- 我們可於MAC運算之後加密，如圖5-2 (b)以保護訊息的機密性。

- 或於MAC運算之前加密，如圖5-2 (c)以保護訊息的機密性。

(二) 訊息鑑別碼
—特性

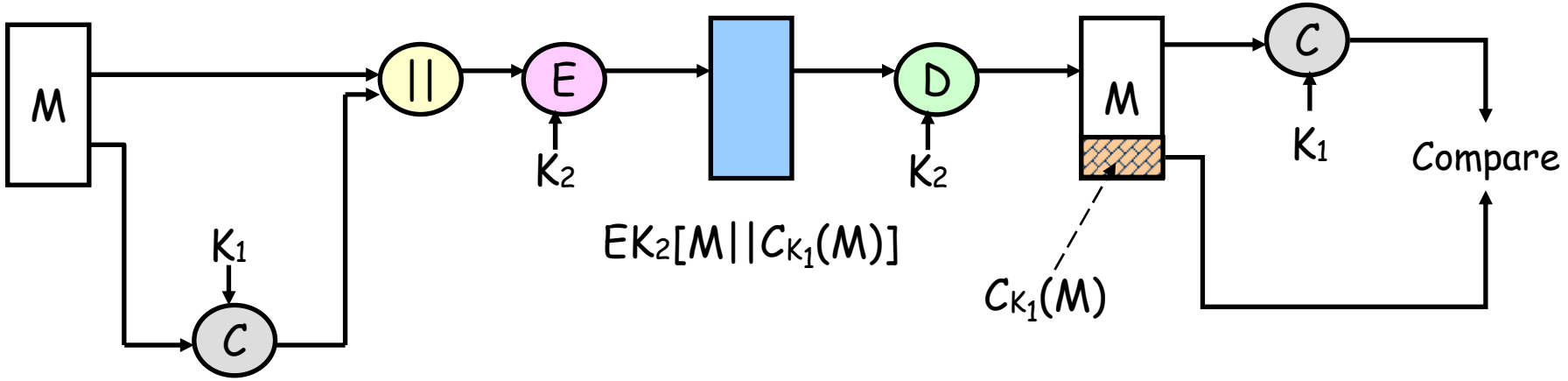


圖5-2 (b)訊息的鑑別性與機密性:藉由明文來確認

返回

(二) 訊息鑑別碼
—特性

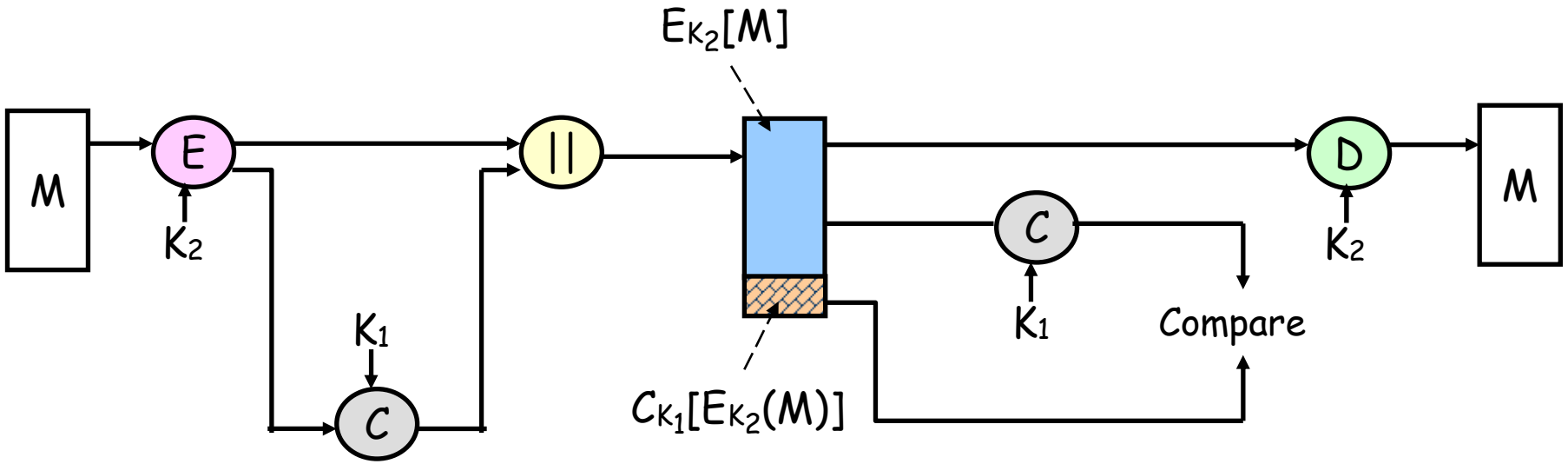


圖5-2 (c)訊息的鑑別性與機密性:藉由密文來確認

(二) 訊息鑑別碼
—特性

- MAC 提供資訊的鑑別性(authentication) ，但不保證資訊的機密性。
- 須配合私密與公開金鑰的加密來提供資訊的機密性。
- MAC 可在加密前或後產生，但因加密後的密文與原本明文已不同，而為達成原明文鑑別功能，所以基本上在加密前產生比較好。
- MAC的用途
  - MAC可針對經由傳輸過後的訊息完整性做驗證
  - 在某些應用層面，訊息完整性的重要超過訊息的保密性(例如：公司內部的一般性公文傳輸，其訊息的完整性比起保密性來得重要。)
- MAC 並不是數位簽章，數位簽章是為達成資訊的不可否認性， MAC 則是提供資訊的鑑別。

(二) 訊息鑑別碼
—特性

- MAC 是一種通行碼學上的總和(cryptographic checksum)檢查

  $MAC = C_K(M)$

  – 壓縮一個不定長度的訊息 M
  – 需要一把秘密金鑰 K_R
  – 產生一個固定長度(fixed block size)的鑑別碼

- 函式C是一個多對一的函式
  – 可能有多個訊息對應到同一個 MAC
  – 但是找出這些巧合是非常困難的

(二) 訊息鑑別碼
—安全考量

考慮可能攻擊的安全性

MAC 必須符合以下條件:

1. 給定訊息與 MAC，很難找出另一個具有相同 MAC 的訊息。
2. MAC 函式必須滿足統計學上的平均分佈以避免不同訊息產生相同的MAC碼。
3. MAC 內容必須均勻地取決於訊息內的所有位元。

(二) 訊息鑑別碼
—訊息鑑別演算法

- 目前知名的訊息鑑別演算法為
  - 使用 DES 的資料鑑別演算法(Data Authentication Algorithm, DAA)
  - 雜湊訊息鑑定碼 (Hash Message Authentication Code, HMAC)等

1.資料鑑別演算法 (DAA)

- 為一常用的MAC演算法，且成為美國國家及聯邦標準**(FIPS PUB 113, ANSI X9.17)**

(二) 訊息鑑別碼
—訊息鑑別演算法

- 是一個以 DES 為基礎的訊息鑑定演算法
- IV=0且在最後一個區段填入零，形成完整的64位元區段
- 使用 DES 的 CBC(Cipher Block Chaining Mode,連鎖式密文區塊模式) 模式對訊息加密

(二) 訊息鑑別碼
—訊息鑑別演算法

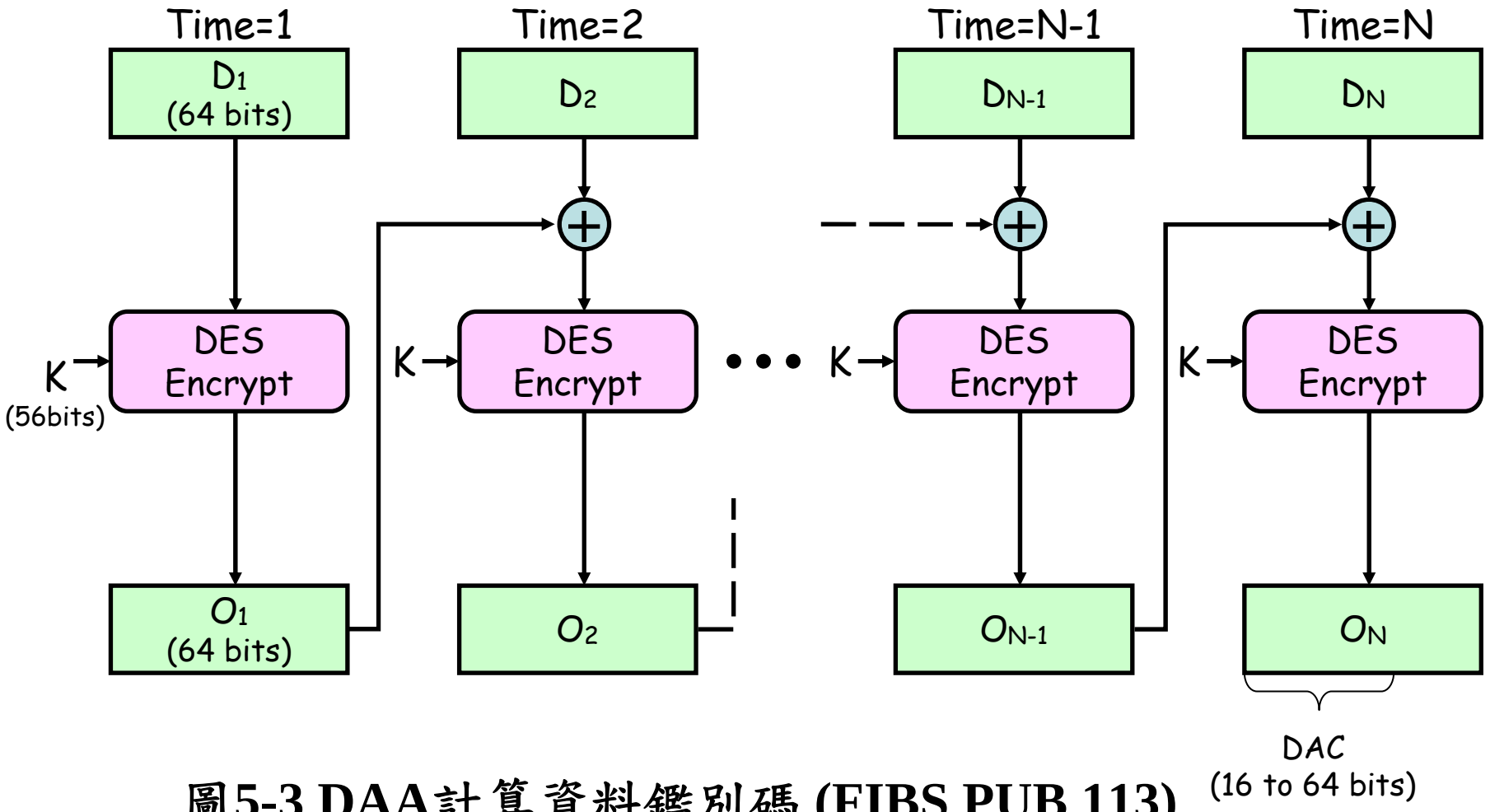


圖5-3 DAA計算資料鑑別碼 (FIBS PUB 113)

(二) 訊息鑑別碼

—訊息鑑別演算法(**)

1.資料鑑別演算法(DAA)

- 將資料分割成64位元區段 $D_1,D_2,...,D_N$，使用DES演算法與私密金鑰K ，運用圖5-3計算資料鑑別碼(DAC)
- 其中 $O_1=E_K(D_1)$; $O_2=E_K(D_2 \oplus O_1)$ .. $O_N=E_K(D_N \oplus O_{N-1})$
- 只送出最後一個區段O_N當作 DAC
- 或是最後一個區段的最左 M (16≤M≤64) 個位元當作資料鑑別碼(DAC)

- 以目前的趨勢看來， 64位元key長度已經太短，不夠安全。

(二) 訊息鑑別碼
—訊息鑑別演算法(**)

2.HMAC法

- 目前有很多將金鑰加入雜湊函式的方法，其中最受歡迎的就是HMAC
- HMAC被發佈成為RFC 2104
- 在IP Security規格中就要求用HMAC來實作MAC
- SSL網際網路協定也是採用HMAC
- HMAC也被發佈成為FIPS的初稿

(二) 訊息鑑別碼
—訊息鑑別演算法(**)

- HMAC的設計目標
  - –在不需要修改的情況下就可以使用現有的雜湊函式
  - –可輕易地更新內嵌的雜湊函式
  - –要能保持原來雜湊函式的效能
  - –要能夠以簡單的方式使用、掌控金鑰
  - –要能夠以內嵌雜湊函式所能提供的合理假設為基礎，進而充分分析此認證機制之安全強度

(二) 訊息鑑別碼
—訊息鑑別演算法(**)

- HMAC表示如下：
  $HMAC_K(M) = H[(K^+ \oplus opad) \| H[(K^+ \oplus ipad) \| M]]$
  - 其中 ipad=00110110，重複b/8次；opad=01011100，重複b/8次
- HMAC演算法的過程(圖5-4)：
  1. 在K的左邊加入0產生出長度為b的K+。
  2. K+ 和ipad，逐位元做XOR運算。
  3. 在S_i加入M。
  4. 將第3步驟的結果輸入H。
  5. 將 K^+和opad做XOR運算，產生b位元的So。
  6. 把第4步驟的雜湊結果加入So。
  7. 把第6步驟產生的串流輸入H，輸出結果。

(二) 訊息鑑別碼

—訊息鑑別演算法(**)

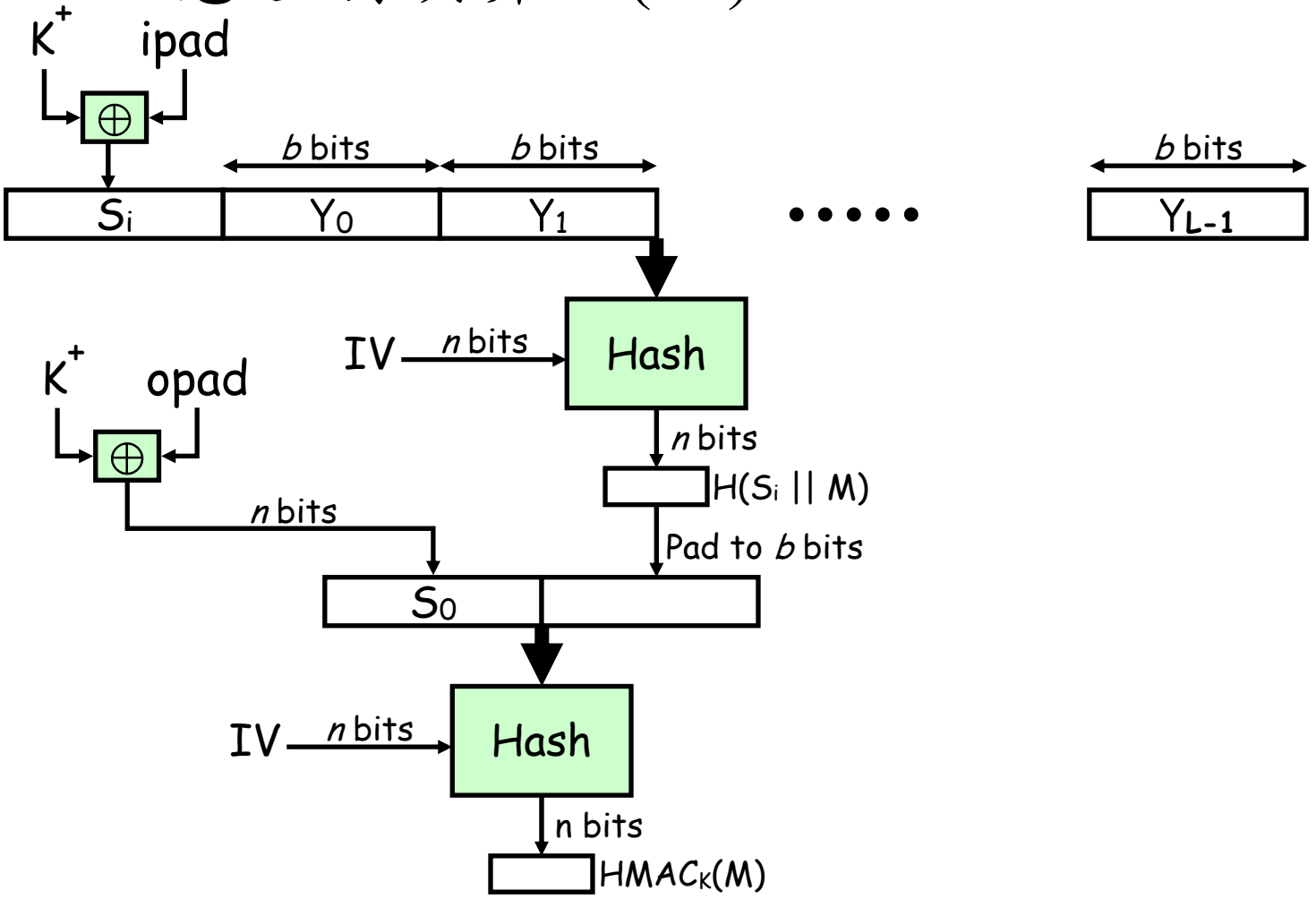


圖5-4 HMAC的架構

(三) 雜湊函式(**)

- 雜湊函式: 可將一個可變長度的訊息產生固定長度的雜湊碼，功能具備錯誤偵測。
- 與訊息鑑別碼不同處，雜湊函式不使用私密金鑰加密。
- 雜湊碼亦稱訊息摘要(message digest)或雜湊值(hashing value)。

(三) 雜湊函式

- 通常會假設雜湊函式是已知的，且不需要金鑰。
- 雜湊碼常用來偵測訊息的變動。
- 具有各種處理訊息的方式，最常用來產生數位簽章。

(三) 雜湊函式
—特性

- 雜湊函式用來產生檔案/訊息/資料的「指紋(fingerprint)」。

$$h = H(M)$$

  - 壓縮不定長度的訊息 M
  - 產生固定長度的指紋 h

- 雜湊函式是公開的，故須保護資訊的雜湊值。
- 以下說明雜湊函式的基本運作。

(三) 雜湊函式
—基本運作

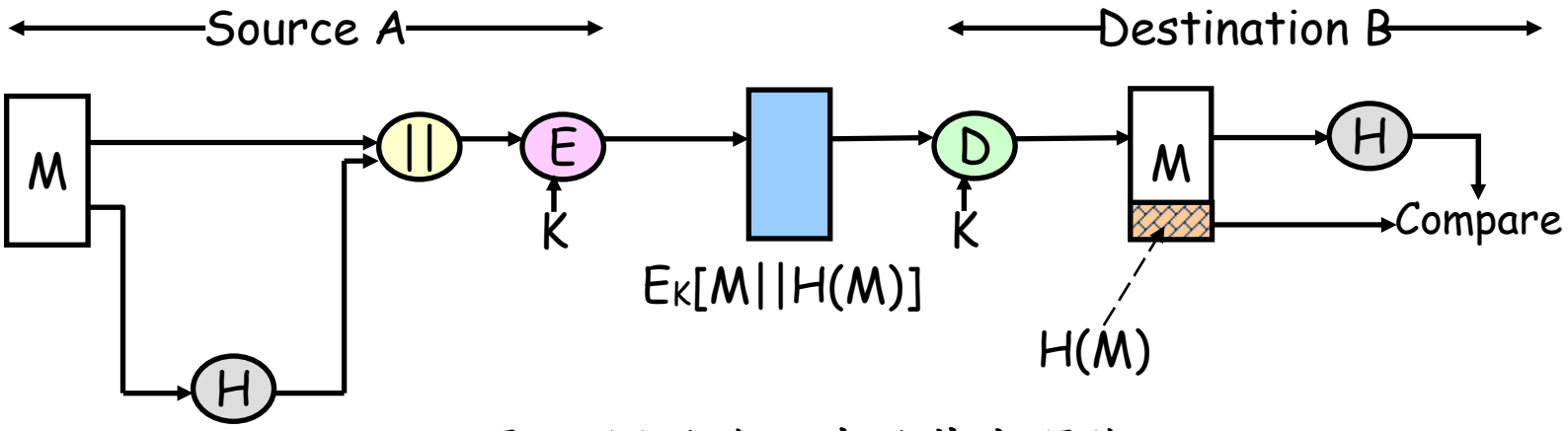


圖5-5(a)雜湊函式的基本運作

- 此方法為將訊息接上雜湊碼後再以對稱式加密法加密，因只有A與B共享祕密金鑰，所以可確認訊息一定來自於A，以及經由祕密金鑰加密提供訊息保密性，雜湊碼則是提供對訊息的確認性。

(三) 雜湊函式
—基本運作

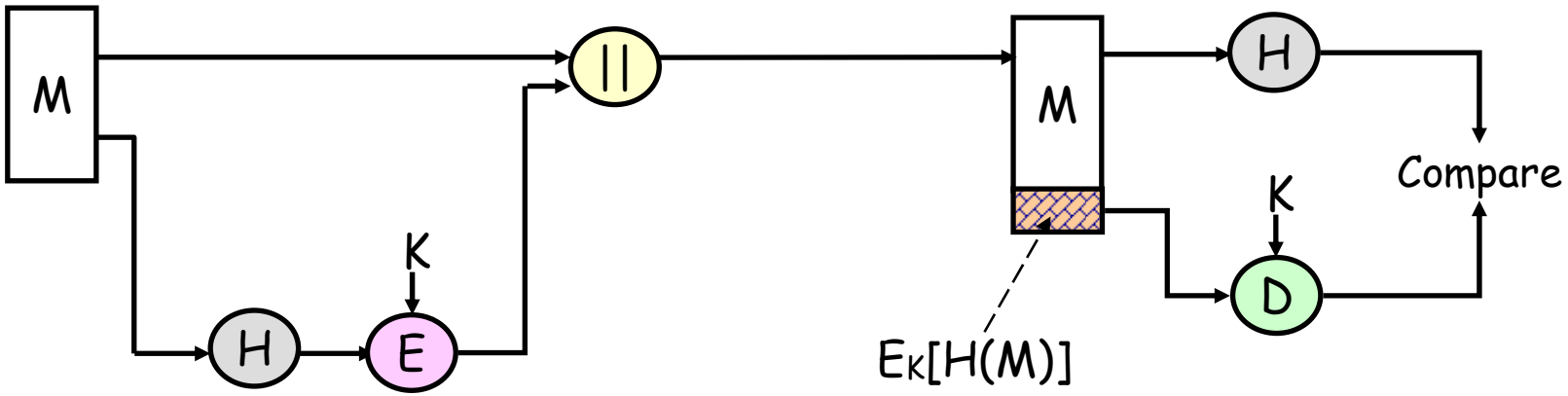


圖5-5(b)雜湊函式的基本運作

- 此方法對不需要保密的應用來說可以減輕運算的負擔。將雜湊碼以對稱式加密法加密後與明文一同傳輸，因為以明文傳輸所以不提供訊息的保密性，但可經由雜湊碼提供對訊息的確認性。

(三) 雜湊函式
—基本運作

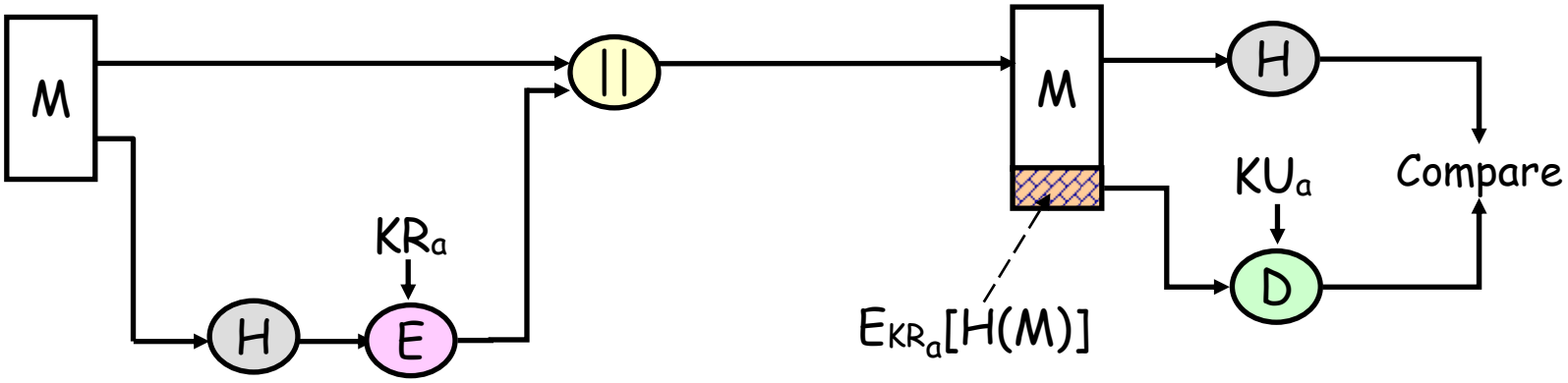


圖5-5(c)雜湊函式的基本運作

- 此方法則是以公開金鑰來對雜湊碼加密，其主要應用與圖5-5(b)相同，這提供了訊息的確認性，也因唯有傳送端擁有自己的私鑰以作為雜湊碼的加密，所以也提供了數位簽章。

(三) 雜湊函式
—要求

1. 可用在任意長度的訊息 M 上
2. 產生固定長度的輸出 h
3. 對任意的訊息 M 來說，很容易計算出 h=H(M)
4. 給定 h，我們很難求得 x，使得 H(x)=h，稱為單向特性
5. 給定 x，很難求得 y，使得 H(y)=H(x)，稱為弱碰撞抵抗力 (weak collision resistance)
6. 很難求出一組 x,y，使得 H(y)=H(x)，稱為強碰撞抵抗力 (strong collision resistance)

- 接下來介紹簡單雜湊函式

(三) 雜湊函式
—簡單的雜湊函式(*)

- 有各種簡單函式的方案
- 主要是對訊息區段的每一位元作 XOR 運算，此法可表示如下:

$C_i = b_{i1} \oplus b_{i2} \oplus \ldots \oplus b_{im}$,

其中ci =雜湊碼的第i的位元

m = 輸入的n位元區段總數

b_{ij} = 第j區段中的第i位元

$\oplus$ = XOR運算

- 以通行碼學的安全考量來說，簡單雜湊函式並不安全，需要更具威力的通行碼學函式 (如SHA-1函式)

(三) 雜湊函式

—威脅—生日攻擊法(**)

- 64 位元的雜湊碼似乎是安全的，但是從生日迷思(birthday paradox)看來並不夠安全。
- **生日攻擊法**以下列方式攻擊：
  - 來源A簽署一個訊息
  - 對手針對此訊息產生$2^{m/2}$種有效訊息的變形，這些變形的訊息本質上意義都相同
  - 對手也產生另一組欺騙訊息的 $2^{m/2}$ 種變形
  - 比較這兩組訊息，藉此找出一對產生相同雜湊碼的訊息(根據生日迷思，找到的機率 > 0.5)
  - 讓使用者簽署有效的訊息，然後在欺騙訊息後附上此有效的簽章再送出
- 如果使用64 位元的雜湊碼，只需要2^{32}次就能為偽造訊息，結論就是需要長度更長的 MAC碼。

(三) 雜湊函式
—利用區段加密的雜湊函式(**)

- 此方向通常利用區段加密來當作雜湊函式，而不是使用金鑰加密，Robin最早提出此一方法，敘述如下:
  - 首先將訊息分割成N個固定大小區段 $M_1,M_2,...,M_N$，使用DES演算法來計算雜湊碼
  - 利用 H_0=初始值、以及在最後一個區段附加零
  - 計算 $H_i = E_{M_i} [H_{i-1}]$
  - 計算雜湊碼 $G = H_N$
  - 把最後一個區段當作雜湊碼
  - 類似 CBC 模式，但是不需要金鑰

(三) 雜湊函式
—利用區段加密的雜湊函式

- 產生的雜湊碼太小了 (64 位元)
  - 易遭受直接的生日攻擊法，以及「中點交會攻擊法」(meet in the middle attack)
  - 所謂「中點交會攻擊法」，攻擊者居中發送訊息與兩端溝通,藉此獲取相關明文及密文的一些關係,再根據一些法則演算出正確的Secure Key。
  - 其他的修正法(如Davies & Price及Meyer法)也有被攻擊的危機

(三) 雜湊函式
—知名雜湊演算法

- 目前知名且常用的三個雜湊函式為
  - MD5 (128位元)
  - SHA-1 (160位元)
  - RIPEEMD-160 (160位元)

  以下我們簡介MD5及SHA-1:

- 其中MD5訊息摘要演算法(RFC 1321)是由Ron Rivest在MIT發展出來，為目前最多人使用的雜湊函式。
- 以通行碼學安全角度的觀點，面對暴力攻擊法或生日破解法 ，MD5已不夠安全。

(三) 雜湊函式
—知名雜湊演算法

- 安全雜湊演算法(SHA-1)
  - SHA (Secure Hash Algorithm)是由國家標準與技術協會(NIST)所發展出來。
  - 1993年發佈成為第180號聯邦資訊處理標準(FIPS PUB 180)；1995年發佈修正版本成為第181號聯邦資訊處理標準(FIPS PUB 180-1)，一般將此版本通稱為SHA-1。

(三) 雜湊函式
—知名雜湊演算法(**)

- SHA-1的原理
  - 此演算法將最大長度不超過2^{64}位元的訊息輸入轉換成160位元的訊息摘要。
  - 輸入的訊息會被分成L個512位元區段來處理
  - 處理的過程包含以下步驟：
    1. 附加填充位元。
    2. 增加原始訊息的長度資訊。
    3. MD暫存區的初始化，使用5個32位元(A、B、C、D、E)。
    4. 處理訊息中所有的512位元區段(此為演算法的核心部分)。
    5. 輸出。
  - 以下針對演算法的核心部分作簡介

(三) 雜湊函式
—知名雜湊演算法(**)

- 處理訊息中所有的512位元區段的簡介(步驟4)
  - 由4個回合所組成
  - 每回合有20個步驟
  - 每個回合使用不同的基本邏輯函式，以f_1、f_2、f_3、f_4標示
  - 以512位元區段(Y_q)及160位元的暫存區為每回合的輸入
  - 每回合還會加上一個常數K
  - 第四回合的輸出會跟第一回合的輸入(CV_q)加在一起，結果就是CV_{q+1}(相加方式為CV_q與暫存區的4個字元相加並取2^{32}的同餘)
- 處理步驟詳見圖5-6

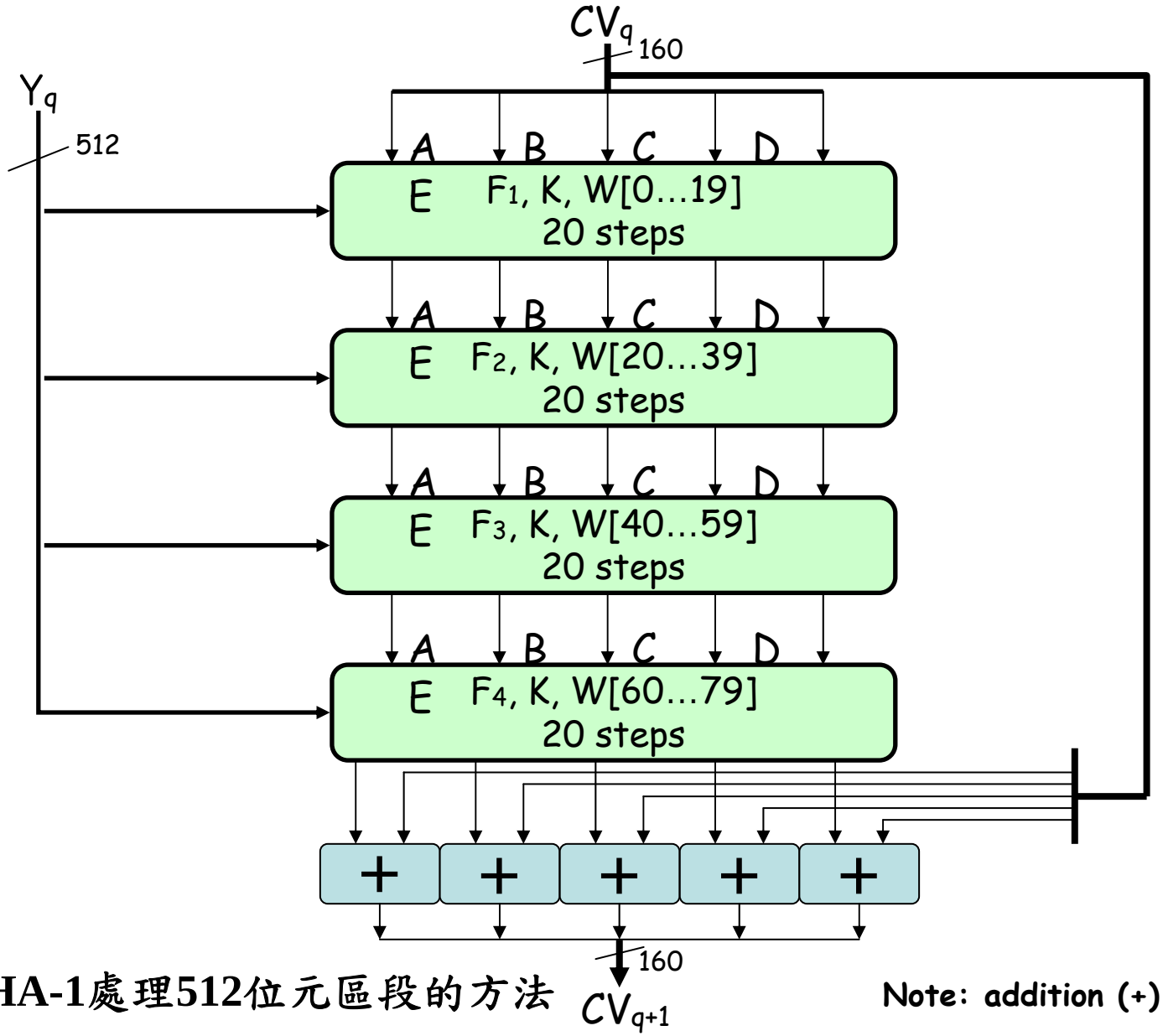


圖5-6 SHA-1處理512位元區段的方法

(三) 雜湊函式
—知名雜湊演算法(**)

- 當所有L個的512位元區段都處理過後，第L個階段所產生的輸出就是160位元的訊息摘要。
- SHA-1歸納為：

$$CV_0=IV$$
$$CV_{q+1}=SUM_{32}(CV_q,ABCDE_q)$$
$$MD=CV_L$$

其中

IV　　=ABCDE暫存區的初始值

$ABCDE_q$ =訊息中的第q個區段在最後一回合的輸出

L　　=訊息中的區段個數(包括已附加的位元及長度欄位)

SUM_{32} =將兩個輸入區段的字元分別獨立相加後再取2^{32}的同餘

MD　　=最後的訊息摘要

(三) 雜湊函式
—知名雜湊演算法

- SHA-1與MD5的比較
  - 對抗暴力攻擊法的安全程度

    SHA-1的摘要比MD5多了32位元，比較起來SHA-1是比較安全的。
  - 對抗通行碼破解的安全程度

    MD5結構是不安全的，但SHA-1的設計策略不太為人所知，所以很難判斷它到底安不安全。
  - 速度

    相同硬體設備下，SHA-1的執行速度比MD5慢。
  - 簡單與簡潔

    這兩個演算法都容易理解，也很容易用程式寫出來。
  - 「以較低位元為結尾」對「以較高位元為結尾」的架構

    前者指MD5，後者指SHA-1，兩者沒有什麼差別。

(三) 雜湊函式
—雜湊函式與 MAC 的安全性(**)

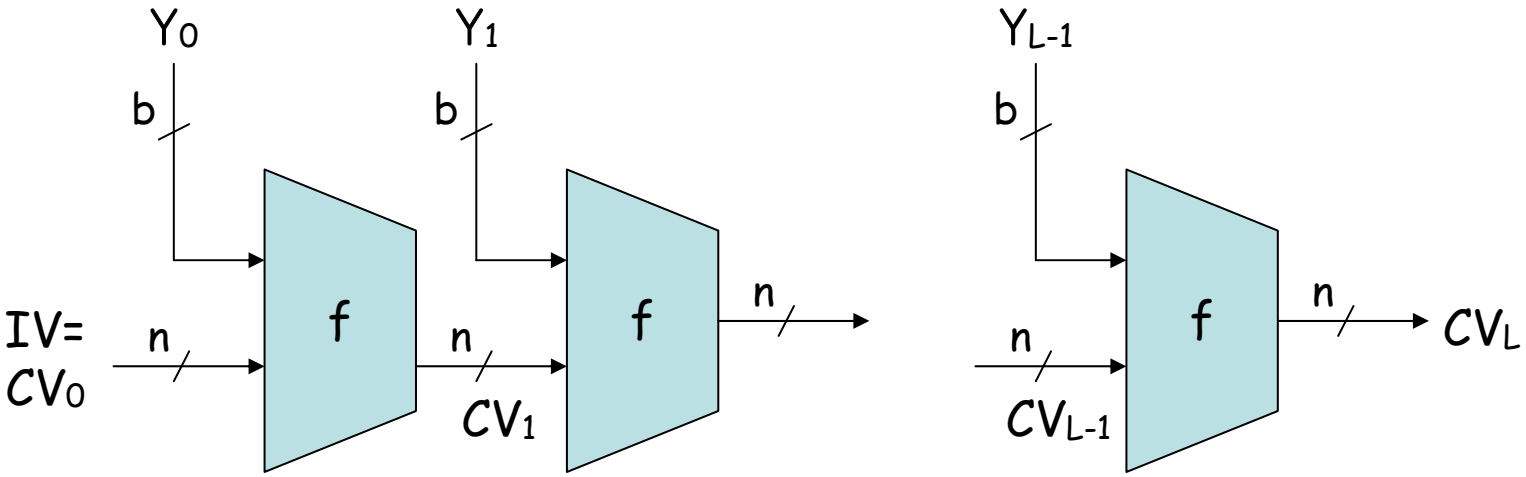
- 雜湊函式與區段加密法一樣受到暴力攻擊法(brute-force)與通行碼破解法(cryptanalysis)的威脅
- **暴力攻擊法:**
  - 具有強碰撞抵抗力的雜湊法的成本為 $2^{m/2}$
    - 已有人提出硬體 MD5 的破解法
    - 128 位元的雜湊值似乎不安全、160 位元比較保險
  - 已知一組或多組的「訊息-MAC」組合
    - 透過金鑰搜尋可以找出一把金鑰以產出正確的 MAC
    - 至少需要128 位元的 MAC 才安全

(三) 雜湊函式

—雜湊函式與 MAC 的安全性(**)

- **通行碼破解法**
  - 首先須了解由Merkle所提出反覆性雜湊函式架構，目前MD5, SHA-1及RIPEMD-160均使用此架構如圖5-7
  - 攻擊方法使用數種針對反覆性雜湊函式的架構，以執行攻擊如下:
    - CV =IV=初始值的n位元值；$CV_i = f[CV_{i-1}, y_{i-1}]$; $H(M)=CV_L$
    - 其中輸入訊息M包括區段$Y_0, Y_1, \ldots, Y_{L-1}$
    - 基本上關心的是函式 **f** 的碰撞(collision)的發生重覆情形
    - 與區段加密法一樣，雜湊值由多個回合運算所組成，攻擊方法使用根據回合函式的特性來有效率的產生雜湊函式的碰撞以破解

(三) 雜湊函式
—雜湊函式與 MAC 的安全性(**)



IV = 啟始值
CV = 串接變數
Y_i = 第i個輸入區段
f = 壓縮函式
L = 輸入區段的個數
n = 雜湊碼的長度
b = 輸入區段的長度

圖5-7安全雜湊碼的一般架構

(四) 三種方法的整合應用(**)

- 結合單向雜湊函式及加密技術，一個提供資料完整性、資料隱密性以及資料鑑別的基本方式如下：

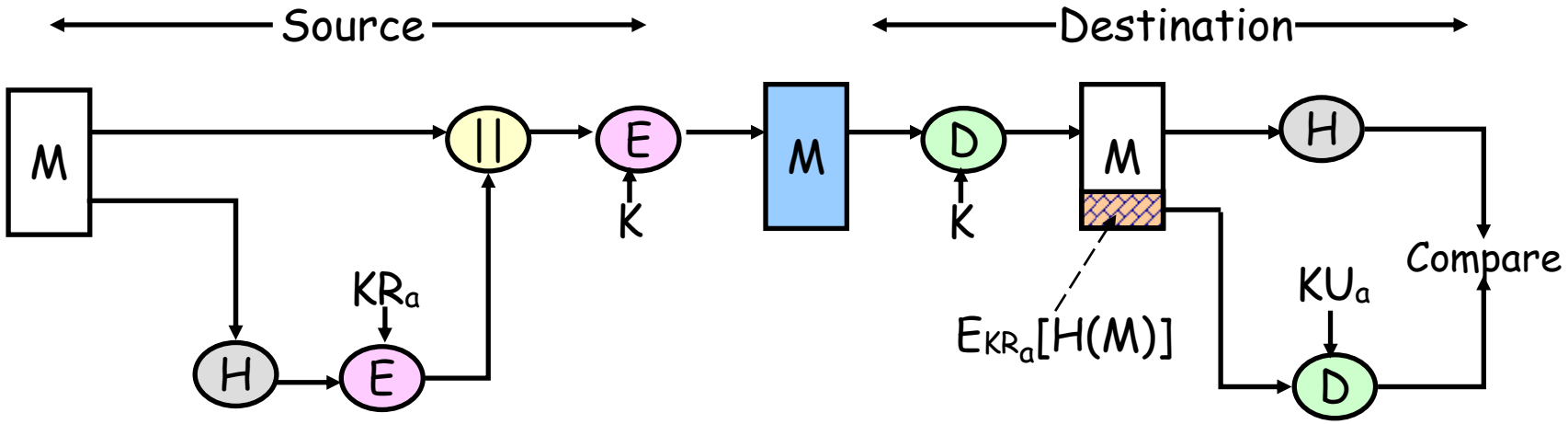


圖 5-8 結合單向雜湊函式以及加密技術的認證方式

(四) 三種方法的整合應用(**)

- 傳送端利用雜湊函式對資料產生Hash Code，然後利用傳送端的私有金鑰加密Hash Code ，再與原有資料用對稱金鑰加密。

- 接收端收到加密的資料後，利用對稱金鑰解密，再利用傳送端的公開金鑰解密加密過的Hash Code ，最後再利用雜湊函式對原資料產生Hash Code ，比對自己產生的Hash Code是否相同於收到的Hash Code 。

- 這過程中， Hash Code可確保資料的完整性，而利用私有金鑰加密該Hash Code則可確保資料傳送端的不可否認性，利用對稱金鑰加密所有資料可確保傳輸過程中的保密性。

Module 5-3:身份鑑別技術

身份鑑別概述

- 本節利用以下四小節介紹身份鑑別的方法與協定

(一) 身份鑑別方法:介紹使用者識別碼及數位簽章兩種常用的方法

(二) 身份鑑別協定:介紹三種網路鑑別型態

(三) KDC (Key Distribution Center,金鑰分配中心)鑑別協定:說明使用秘密金鑰認證協定如Needham-Schroeder認證協定與公開金鑰認證協定

(四)數位簽章標準:介紹數位簽章標準 (Digital Signature Standard, DSS)與DSA演算法

身份鑑別概述

- 在現實的世界裡，要驗證一個人的身分，一般其鑑別方法有下列三種方式：
  - 自己才知道的某些事情(something you know）
    例如通行碼。
  - 自己擁有的某些事物(something you have)
    例如通行證。
  - 自己才有的特徵(something you are)
    例如指紋及視網膜。

身份鑑別概述

- 但是這些方法在網路世界裡部份是行不通的，因為現在的電腦還無法做到如同人類一般的判斷。

- 在電腦網路裡必須藉由溝通雙方所共同信賴的驗證程序來驗證對方身份，而在電腦網路裡這種方式稱作為認證協定（Authentication Protocol）。

身份鑑別概述

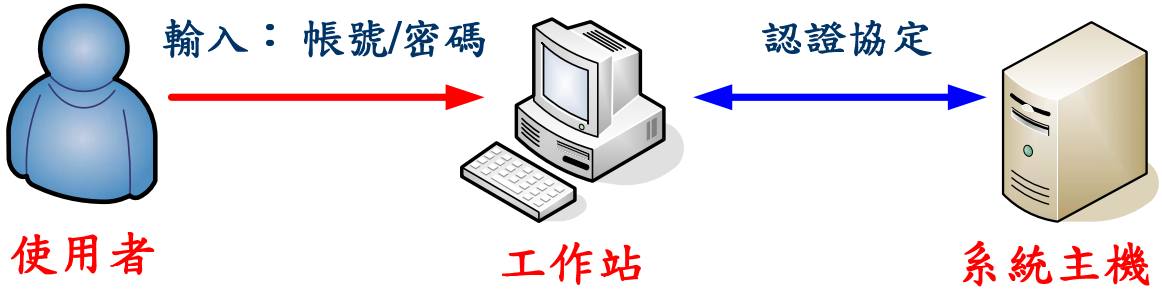
- 從網路使用的觀點來看，身份鑑別有下列幾種方法：
  - 使用者的帳號/通行碼
  - 網路位址/網域名稱
  - 共享秘密金鑰
  - 公開金鑰
  - 數位簽章
  - 數位憑證
  - 智慧卡等

(一) 身份鑑別方法

- 因限於篇幅，以下介紹常用的兩種身份鑑別方法:
  - 使用者識別碼 (password)
  - 數位簽章(digital signature)

1. 使用者識別碼

- 『帳號/通行碼』(Account/Password)
- 身份認證的運作



- 處理通行碼的重點：
  - 不可明文傳送。
  - 不可明文儲存。
  - 加密後的通行碼也不可放在網路空間當中。
- 處理通行碼的目的是產生共享秘密金鑰

1. 使用者識別碼

- **共享秘密金鑰 (Shared Key) 建立：**
  工作站與系統主機之間的共享秘密金鑰。

- **『醃製法』**(Salt Value) ：
  加入某一數值(個人的鹽)與通行碼混合，這種處理過的通行碼稱為共享秘密金鑰。
  – **處理過程：**
    - 通行碼（Password）：P
    - 個人的鹽：S_m，m = 1, 2, …, n
    - 演算法：f()，DES、MD5、RC4
    - 個人的共享秘密金鑰：KT = f(P, S_m)

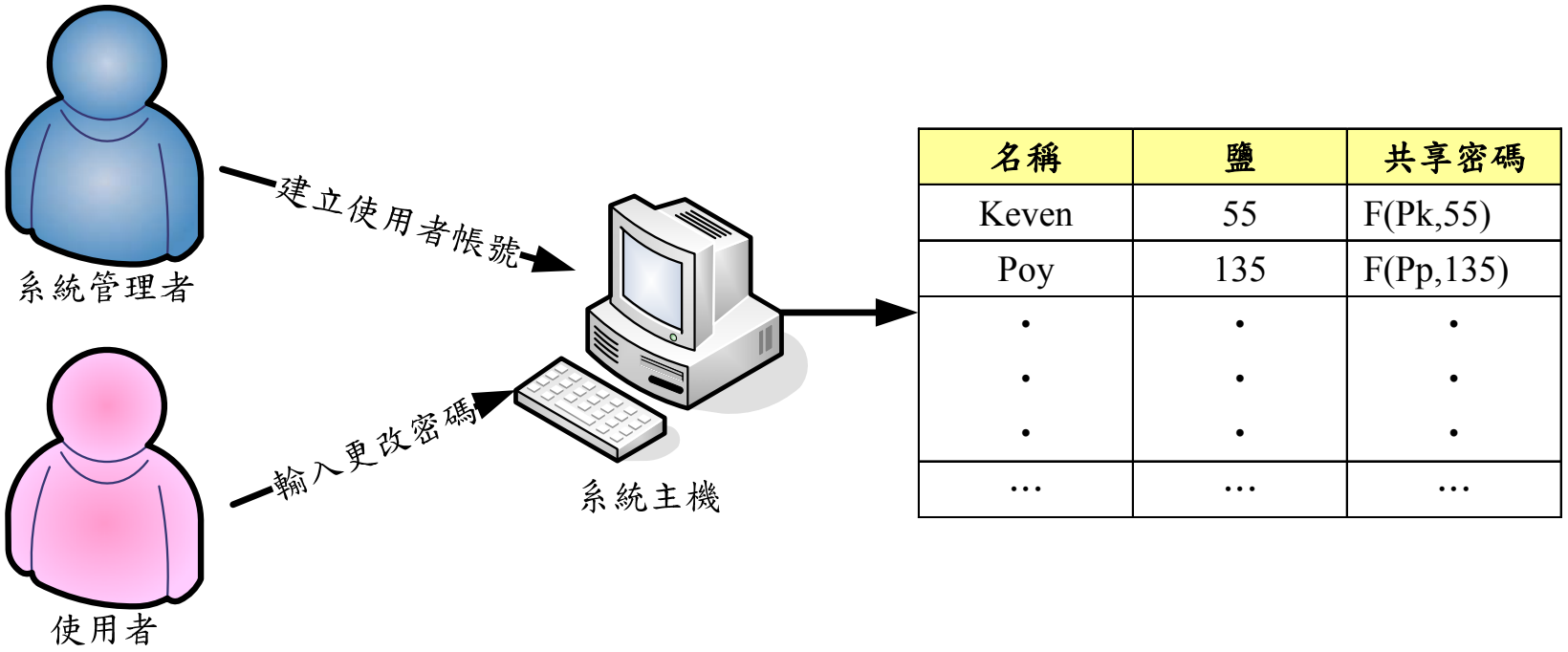


圖5-9 共享秘密金鑰 (Shared Key) 建立

1. 使用者識別碼
—共享秘密金鑰 (Shared Key) 建立

- 簡單的通行碼確認方式

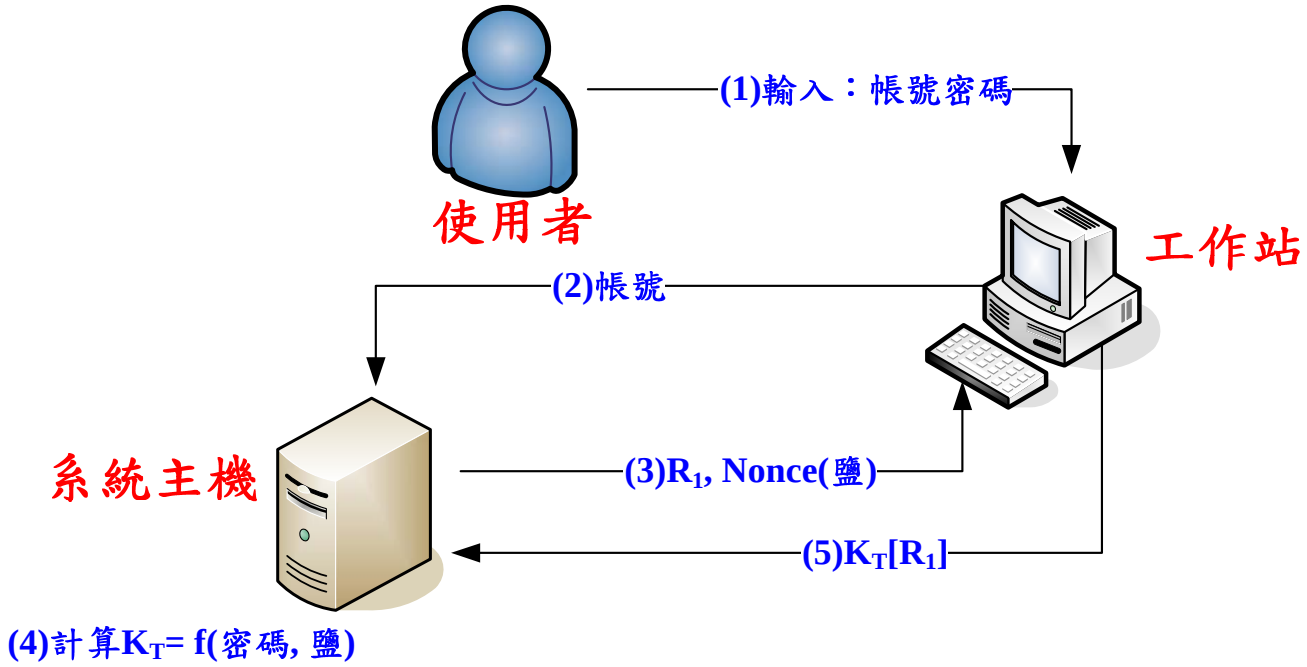


圖5-10盤問與回應(request & response) 的通行碼確認

1. 使用者識別碼
—共享秘密金鑰 (Shared Key) 建立

- 主密鑰(Master Secret)
  - 一套應用系統會有許多個使用者，每一使用者與系統共分享一把秘密金鑰，稱為個人的主密鑰

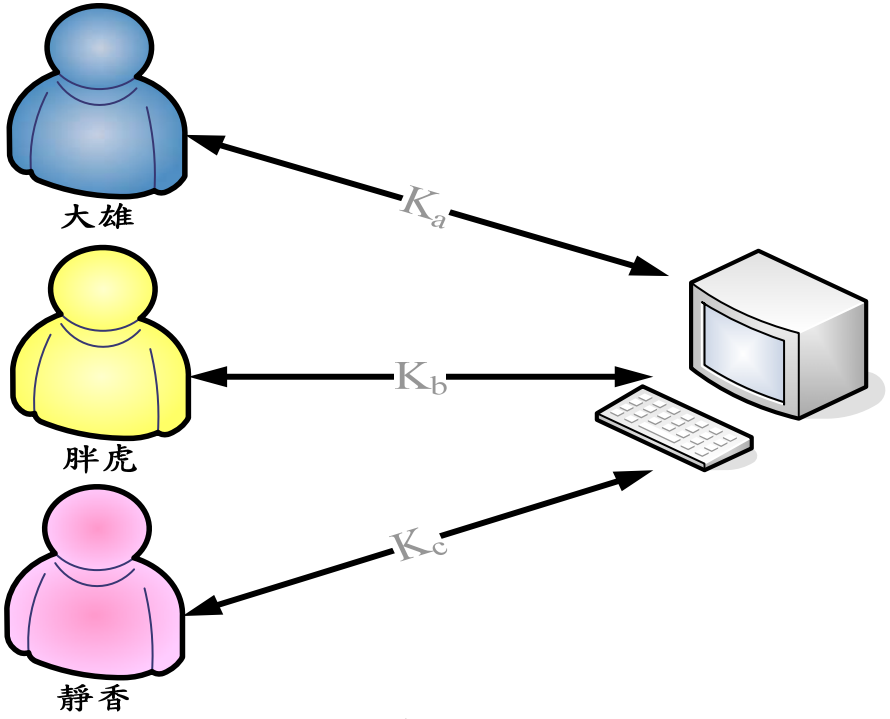


圖5-11 系統內對應每一使用者的主密鑰

1. 使用者識別碼
—通行碼破解

- **使用者針對通行碼處理常有下列疏忽：**
  - 輸入通行碼被窺視
  - 通行碼儲存於某一檔案內
  - 通行碼容易被猜出
  - 被詐騙輸入通行碼
  - 將通行碼寫在紙上
  - 長久未更改通行碼
- 常見的通行碼攻擊方法包括線上及離線通行碼猜測兩種方式。

1. 使用者識別碼—通行碼破解

- **線上通行碼猜測**
  - 字典攻擊法 (Dictionary Attack)
    - 生日日期、結婚日期、子女生日等
  - 防範方法—固定時間內限制通行碼輸入次數(例如3次)。
- **離線通行碼猜測**
  - 取得破解加密後的通行碼嘗試解開通行碼
  - 與社交攻擊法(social attack)交叉使用

1. 使用者識別碼
—通行碼處理技巧

- **通行碼處理技巧**
  - 一般將通行碼處理後稱為『共享金鑰』或『主密鑰』
- **如何建立『共享金鑰』或『主密鑰』**
  - 一般運用共享金鑰或主密鑰對通行碼進行加密，常見的方法包括：

    (1) 單向加密

    (2) 單向雜湊

    (3) 一次通行碼

1. 使用者識別碼
—通行碼處理技巧(**)

(1)單向通行碼加密法(One-way Encryption)

a.通行碼雜湊函式

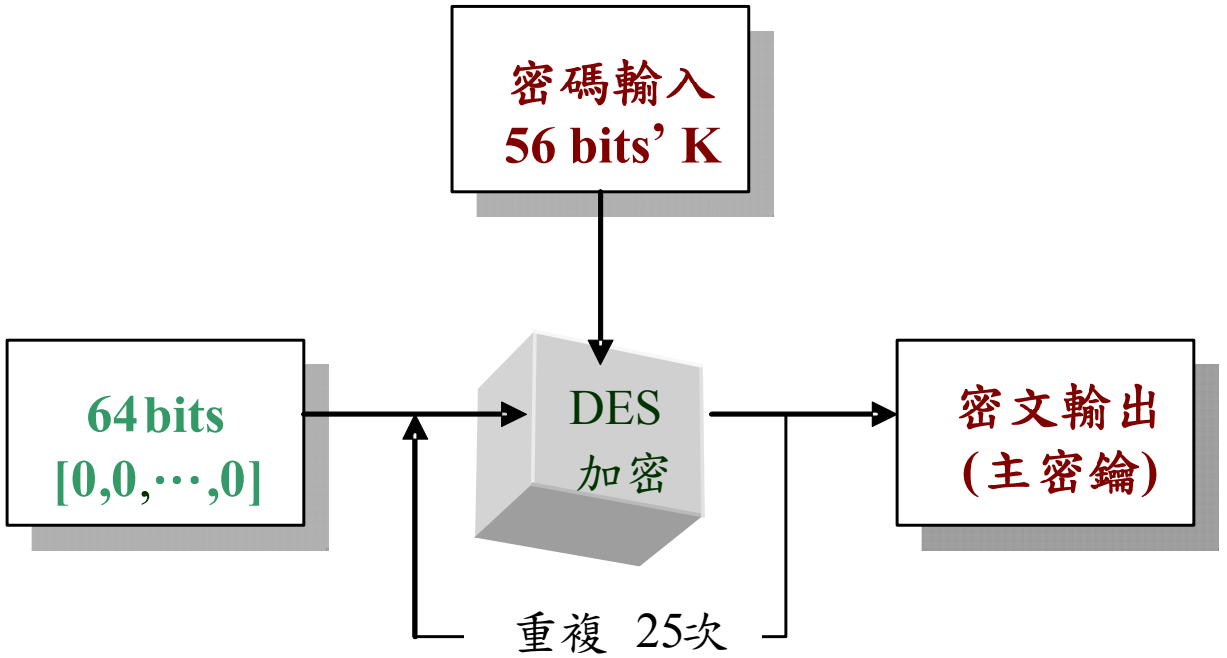


圖5-12 a.通行碼雜湊函式

1. 使用者識別碼
—通行碼處理技巧(**)

(1)單向通行碼加密法(One-way Encryption)

b. 醃製法通行碼加密

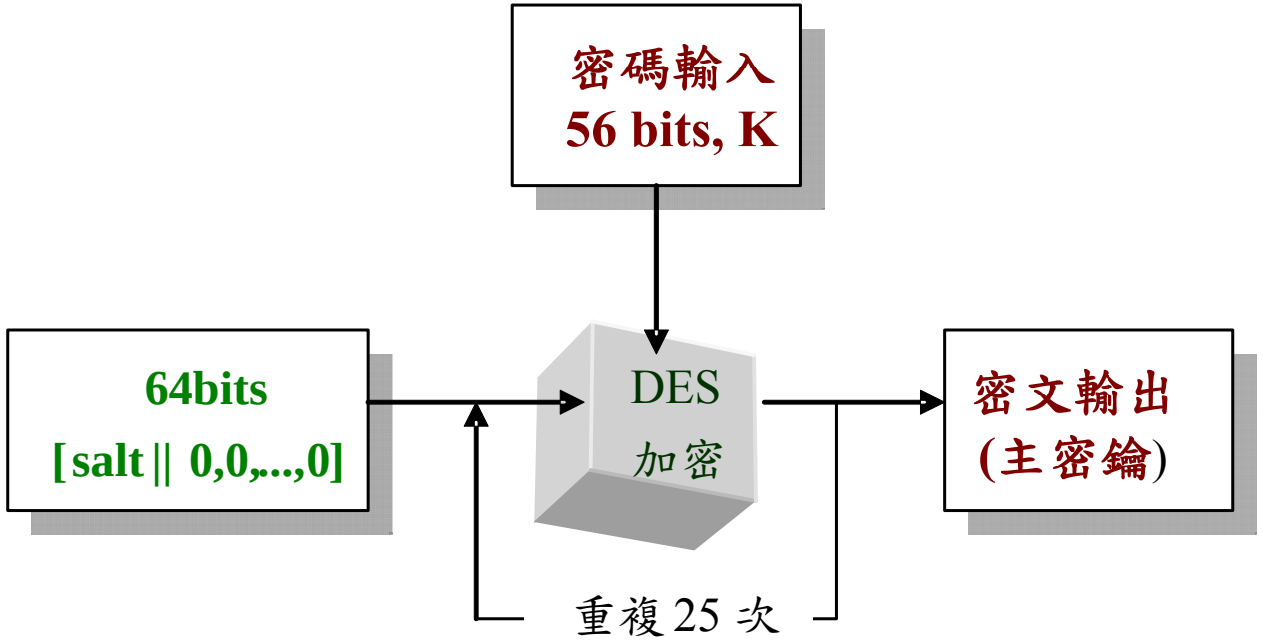


圖5-12 b.醃製法通行碼雜湊函式

1. 使用者識別碼
—通行碼處理技巧(**)
(2) 單向雜湊函式 (One-way Hash Function)

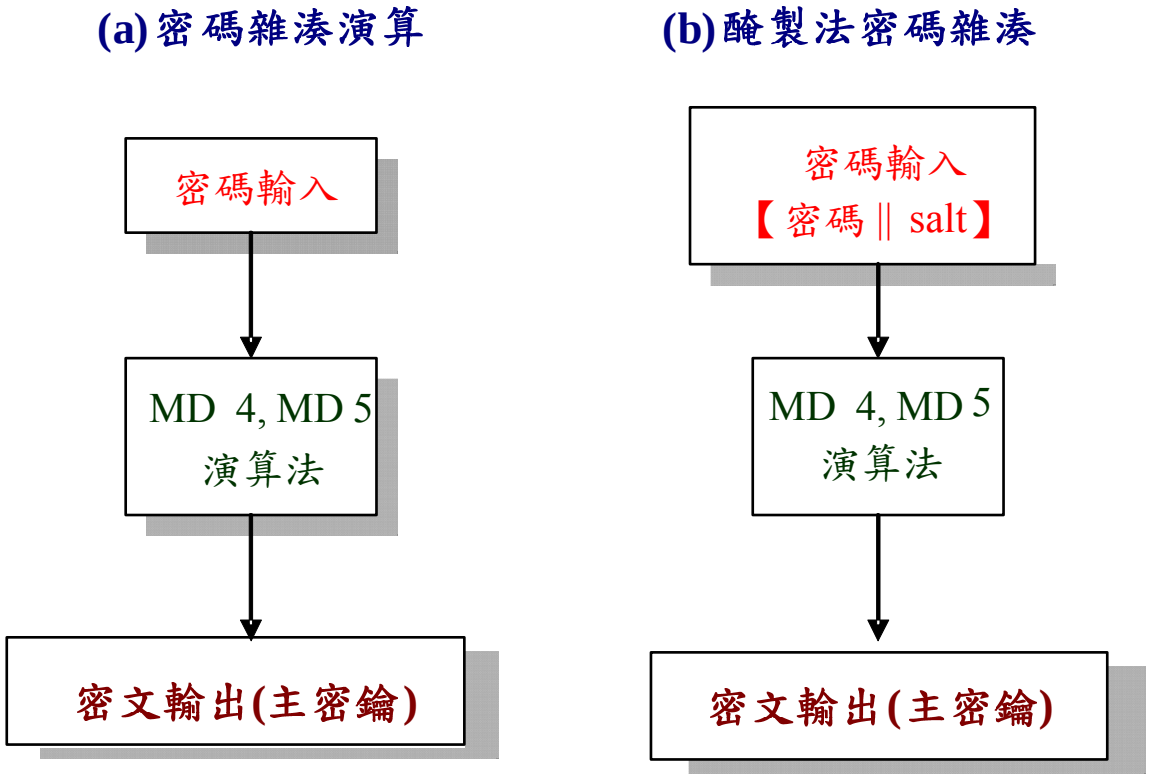


圖5-13 單向雜湊的通行碼處理

1. 使用者識別碼
—通行碼處理技巧(**)

(3)一次通行碼 (One-time Password)

- **IC 卡：Secure ID 或 Session Key**

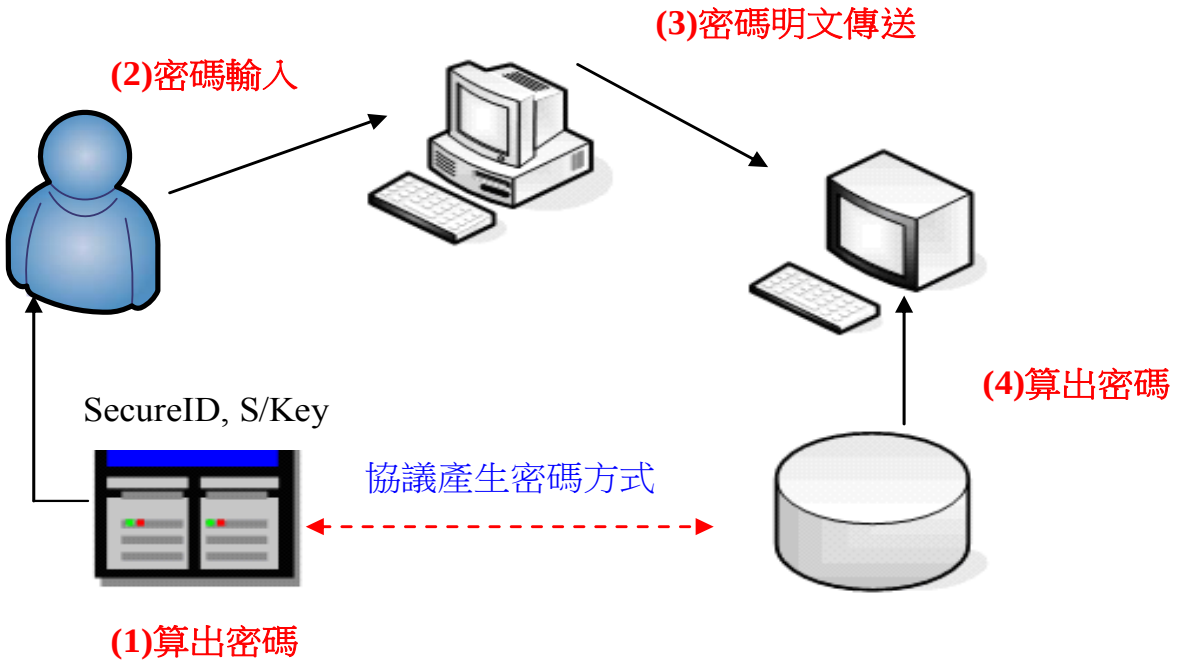


圖5-14 一次通行碼的運作方式

2. 數位簽章

- 數位簽章標準(Digital Signature Standard, DSS)為美國標準與技術協會(NIST)所發佈第186號聯邦資訊處理標準(FIPS PUB 186)。

- 數位簽章標準(DSS)使用安全雜湊演算法(SHA)，並提出數位簽章演算法(DSA)。

2. 數位簽章

- **數位簽章必須具備下列特性**
  - 驗證簽章的作者、日期與時間
  - 在簽章的同時，必須能夠確認訊息的內容
  - 發生爭議時，此簽章可透過第三者驗證來解決爭議

- 因此數位簽章可以提供具有身份鑑定性

2. 數位簽章

- **數位簽章的必備條件**
  - 數位簽章取決於簽署過的訊息
  - 數位簽章必須使用傳送者獨有的資訊，如此可以預防偽造與耍賴
  - 數位簽章必須易於產生
  - 數位簽章必須易於簽章與驗章
  - 「針對既有簽章偽造新訊息」或「針對既有訊息偽造新簽章」在計算上必須無法偽造數位簽章
  - 數位簽章副本必須能夠實際地被存在記憶體內

2. 數位簽章

- 已有數種數位簽章方法被開發出來，這些方法可分成「直接式數位簽章」與「仲裁式數位簽章」兩大類
- **直接式數位簽章**
  - 只有傳送者與接收者參與
  - 假設接收者擁有傳送者的公開金鑰
  - 數位簽章的形式是由傳送者以私密金鑰來簽署整個訊息或將雜湊值加密
  - 若進一步要提供機密性(confidentiality)，傳送者可以利用接收者的公開金鑰來加密

2. 數位簽章

- 先簽署數位簽章再加密才能提供訊息的機密性。
- 發生爭議時，第三者須驗證訊息與簽章，並取得接收者的私密金鑰來解開原始訊息。
- 直接式數位簽章的缺點是此架構的有效性是建立在傳送者的私密金鑰，若傳送者宣稱遺失私密金鑰則可能否認傳送此訊息。

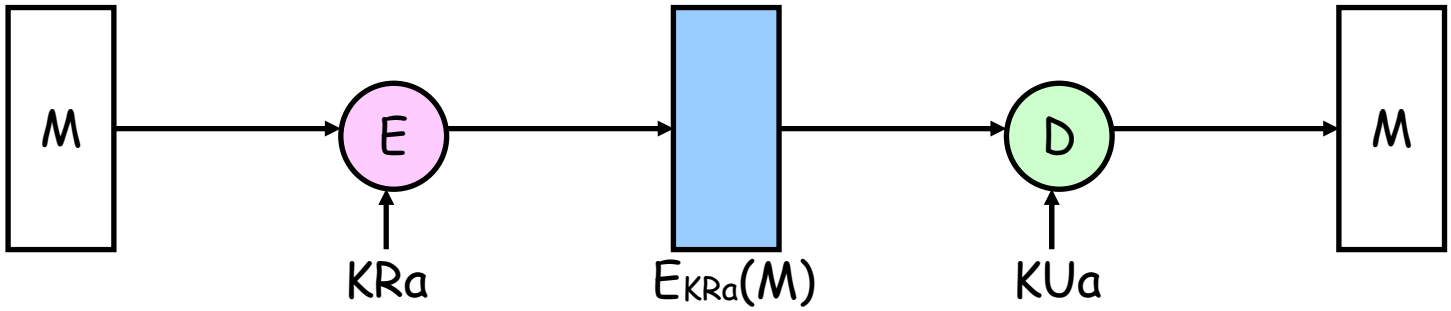


圖5-15 傳送者以私密金鑰進行數位簽章

2. 數位簽章

- **仲裁式數位簽章**
  - 解決直接式數位簽章的問題是在傳送者X傳給接收者Y之前必須加入仲裁者A ， A會執行下列工作:
    - 驗證已簽署的訊息
    - 附上日期並且傳送給接收者
  - 仲裁者可利用信任系統(trusted system)來實作
  - 可以用對稱式或公開金鑰加密法來實作
  - 仲裁者可以看或不看訊息本身

(二) 鑑別技術協定

- 當一個使用者要求登入某台主機或使用其提供的服務時，其通行碼很容易就會被第三者利用監聽軟體(如Ethereal封包分析工具)所竊得。

(二) 鑑別技術協定

- 目前有許多的網路鑑別協定被提出，以型態歸類來看，鑑別方式可能是單向或雙向相互認證，依架構可區分下列三種鑑別型態:
  - 單向認證 (One-way authentication)
  - 雙向認證 (Two-way authentication)
  - 使用公正的第三者的認證 (Trusted Third-party Authentication)

(二) 鑑別技術協定

1.單向認證（One-way authentication）

- 這是最簡單的認證方式，用戶端只需提供ID和通行碼給伺服端作存取確認，伺服端確認後就允許用戶端的登入。

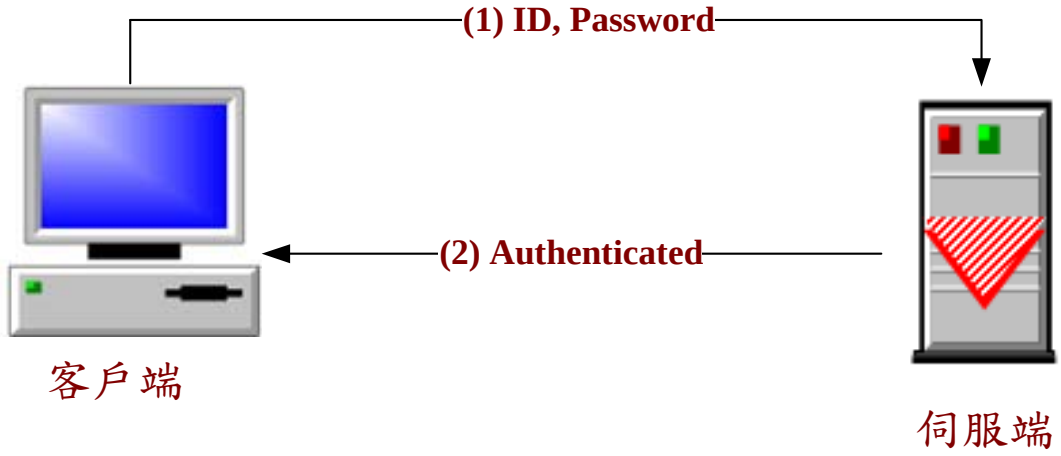


圖 5-16 單向認證

(二) 鑑別技術協定

1.單向認證（One-way authentication）

- 伺服端認證大都採用盤問/回應(challenge/ response)方式。
- 認證內容可區分明文盤問、密文盤問及時間戳記盤問三種方式。

明文盤問

密文盤問

時間戳記盤問-(a)加密時間戳記盤問

時間戳記盤問-(b)雜湊時間戳記盤問

資料來源:摘自
黎添壽, 吳順裕, 資訊與網路安全技術

(二) 鑑別技術協定

2.雙向認證 （Two-way authentication）

- 這是一種雙方相互認證（mutual authentication）的方式，雙方都得提供ID和通行碼給對方，才能通過認證。

- 這種認證方式不同於單向認證在於用戶端還需要認證伺服端的身份，但這樣用戶端必須維護各伺服器所對應的ID和通行碼。

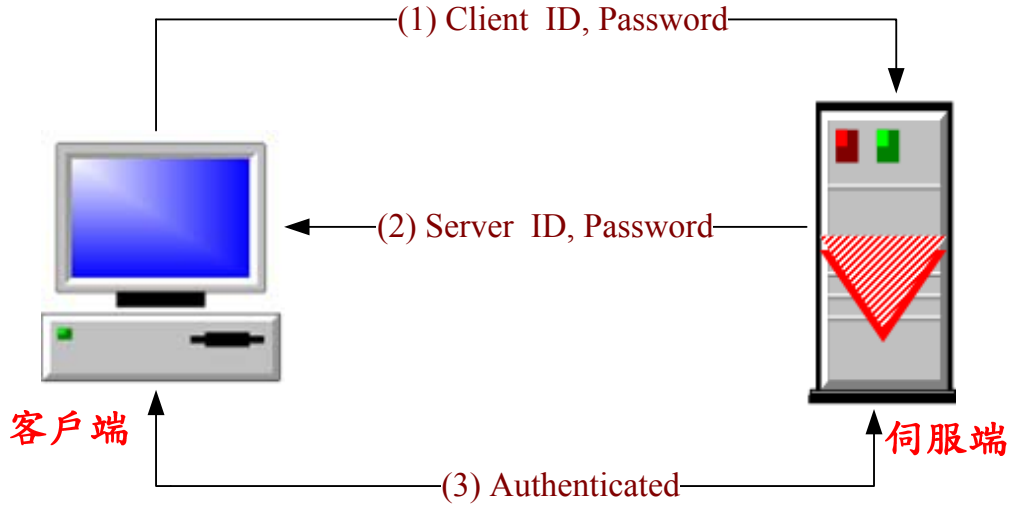


圖 5-18 雙向認證

(二) 鑑別技術協定

2.雙向認證（Two-way authentication）

- 身份鑑別協定透過交換通訊金鑰(session key)用來確認雙方 ID。
- 身份鑑別主要的安全考量是
  - 保密性(confidentiality)–保護通訊金鑰，防止外洩。
  - 時效性(timeliness)–預防重送攻擊(replay attack)。

(二) 鑑別技術協定

3.使用公正的第三者的認證（Trusted Third-party Authentication）

- 這也是一種通訊雙方相互認證的方式，但是認證過程必須借助於一個雙方都能信任的公正第三者（Trusted Third Party）。
- 一般而言可以是政府的法院或是其他可供信賴的機構，當兩端欲進行連線時，彼此必須先通過公正第三者的鑑別，然後才能互相交換金鑰，而後進行連線。
- 由這種借助於公正第三者的認證方式變化而來的鑑別協定相當多，各有各的特色與優缺點，其中一個最著名的例子就是Kerberos認證協定。

(二) 鑑別技術協定

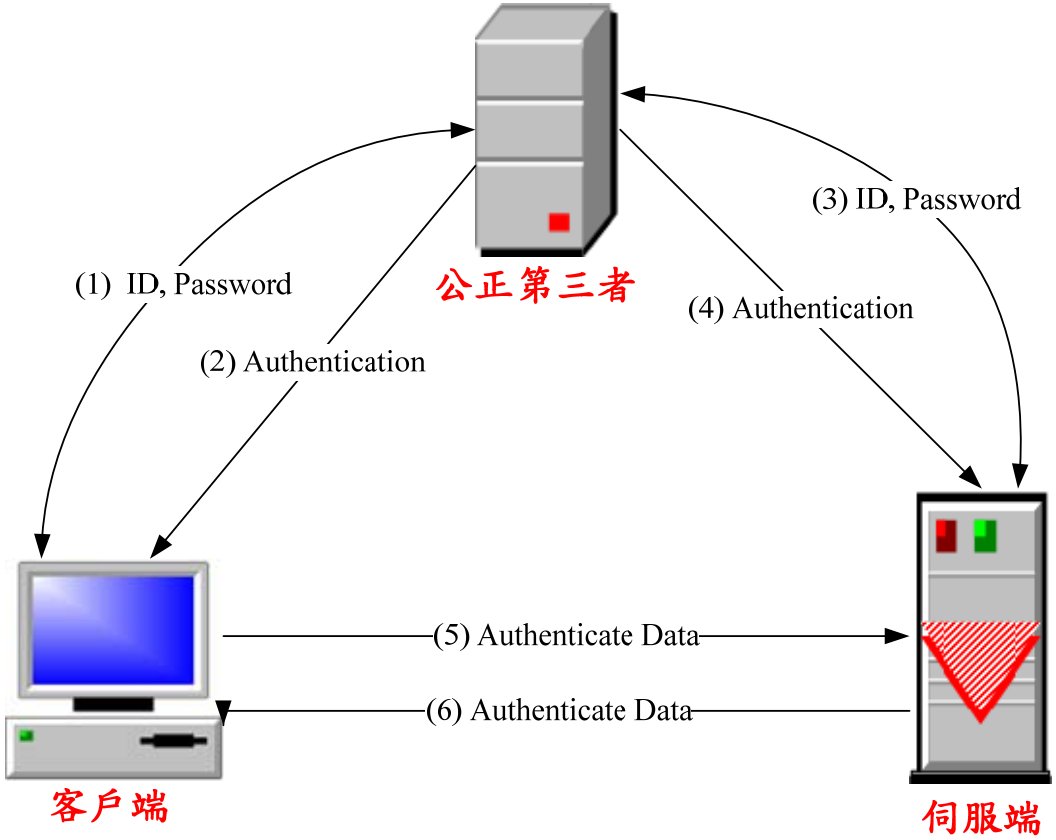


圖 5-19 使用公正的第三者的認證

(二) 鑑別技術協定

- 安全威脅—重送攻擊
  - 複製有效的訊息並於稍後重送
    - 單純的重送
    - 可以被記錄下來的重送
    - 無法被偵測的重送
    - 將未修改的訊息往回重送
  - 防禦之道包括
    - 採用流水號 (一般來說不實際)
    - 時戳 (必須進行時脈同步)
    - 盤查/回應 (採用唯一的臨時亂數)

(三) KDC鑑別協定

基本概念

- 金鑰分配中心(Key Distribution Center ，KDC)大都應用於封閉的網路環境，負責分配使用者的私密金鑰，使用者利用此私密金鑰來進行身份鑑別及存取網路內部資源。
- 透過KDC進行的身份鑑別的協定，我們稱為「KDC鑑別協定」。

- KDC應用於封閉的網路環境的主要功能如下
  - 主機使用者的身份鑑別
  - 工作群組使用者的身份鑑別
  - 網路使用者的身份鑑別

(三) KDC鑑別協定

- **KDC鑑別協定的特色—**
  - 使用對稱式加密
  - 使用兩層的金鑰
  - 在KDC管轄範圍內，所有人都與 KDC 共享一把相同的共享密鑰
  - KDC利用使用者的共享密鑰(主金鑰)來產生通訊雙方所需的通訊金鑰，以供暫時的通訊使用
  - KDC利用共享密鑰安全的將通訊金鑰送給雙方

(三) KDC鑑別協定
—使用對稱式加密(**)

Needham-Schroeder 協定

- 運用KDC來分送A 與 B 之間的金鑰傳送協定與通訊，協定具有身份鑑別功能。
- 協定的運作程序如下:

1. A→KDC: $ID_A \| ID_B \| N_1$

2. KDC→A: $E_{Ka}[Ks \| ID_B \| N_1 \| E_{Kb}[Ks \| ID_A]]$

3. A→B: $E_{Kb}[Ks \| ID_A]$

4. B→A: $E_{Ks}[N_2]$

5. A→B: $E_{Ks}[f(N_2)]$

其中Ka與Kb為共享密鑰， ID_A與ID_B識別碼， N_1與N_2為A 與B產生的臨時訊息, Ks為通訊金鑰，f為對N_2做一些轉換的函數。

(三) KDC鑑別協定
—使用對稱式加密(**)

Needham-Schroeder 協定運作程序說明：

- 步驟1：A將(ID_A , ID_B , N_1)一起送給KDC
- 步驟2：KDC 以E_{Ka}加密回覆 Ks及 $E_{Kb}[Ks||ID_A]$給A
- 步驟3：A將$E_{Kb}[Ks||ID_A]$ 傳送給B
- 步驟4：B以E_{Kb}解密取得Ks，並將資訊N_2以通訊Ks加密傳送給A
- 步驟5：A以E_{Ks}解密取得N_2，並將資訊N_2取函式值(如計算雜湊值)以通訊Ks加密回傳送給B

(三) KDC鑑別協定
—使用對稱式加密(**)

Needham-Schroeder 協定的修訂

- 原協定可安全地傳送 A 與 B 所需的通訊金鑰
- 但是舊的通訊金鑰Ks被破解，可能遭受重送攻擊(replay attack)的威脅
- 因此，步驟3的訊息可能會被用來重送，以致讓 B 認為正與 A 進行通訊
- Denning 以加入時戳來修訂協定防止訊息被重送

(三) KDC鑑別協定
—使用對稱式加密(**)

- 時戳修訂協定: Denning (1981):於步驟2及3增加時戳(timestamp) T，加上KDC與A或B局部時脈差及傳輸延遲即可計算出合理的訊息時間，可避免訊息被重送，完整的運作程序如下:

1. A→KDC： $ID_A \| ID_B$

2. KDC→A： $E_{Ka}[Ks \| ID_B \| T \| E_{Kb}[Ks \| ID_A \| T]]$

3. A→B：$E_{Kb}[Ks \| ID_A \| T]$

4. B→A：$E_{Ks}[N_!]$

5. A→B：$E_{Ks}[f(N_!)]$

其中T表是Ks產生的時間，f為對N_1做一些轉換的函數。

(三) KDC鑑別協定
—使用對稱式加密(**)

- 時戳修訂協定存在著網路主機時脈同步問題，若時脈同步的機制產生錯誤或被破壞，則無法有效計算出正確合理的訊息時間，以致無法避免訊息重送攻擊。
- Neuman 及Stubblebine (1993)利用額外的臨時訊息(亂數) 以解決主機A或B與KDC時脈同步問題。

(三) KDC鑑別協定
—使用公開金鑰加密(**)

- 以下介紹以公開金鑰加密為基礎以進行通訊金鑰的分送，必須具備下列特性
  - 必須正確地擁有對方的公開金鑰
  - 使用一個認證伺服器 (AS)
  - 有許多協定利用時戳或臨時訊息來避免重送攻擊

(三) KDC鑑別協定
—使用公開金鑰加密(**)

- Denning 於1981 提出利用時戳的公開金鑰加密鑑別協定, AS Protocol, 其運作程序如下:

1. A→AS： $ID_A \| ID_B$

2. AS→A：$E_{KRas}[ID_A||KU_a||T] \| E_{KRas}[ID_B||KU_b||T]$

3. A→B：$E_{KRas}[ID_A||KU_a||T] \| E_{KRas}[ID_B||KU_b||T] \| E_{KUb}[E_{KRas}[K_s||T]]$

其中KU_a與KU_b為A及B的公開金鑰, KR_a與KR_b為A及B的私密金鑰, ID_A與ID_B 識別碼,E_{KRas}為以AS的私密金鑰進行加密, Ks為通訊金鑰。

(三) KDC鑑別協定
—使用公開金鑰加密(**)

- Denning AS Protocol的特色
  - AS 提供公開金鑰KU_a與KU_b的鑑別
  - 通訊金鑰由 A 來選定，因此 AS 不負保護通訊金鑰之責，不用擔心AS 會洩露通訊金鑰
  - 時戳可以預防重送攻擊，但是必須進行時脈同步

(四) 數位簽章標準 (DSS)

- 數位簽章標準 (Digital Signature Standard)為一個經美國政府確認的簽章機制 FIPS 186
- 採用 SHA 雜湊演算法
- 由 NIST 與 NSA 於九零年代初期設計
- DSS是標準，DSA是演算法
- 為 ElGamal 與 Schnorr 架構的變形
- 產生 320 位元的簽章, 但是等同於 512-1024 位元的安全性
- 安全性取決於計算離散對數的困難度

(四) 數位簽章標準 (DSS)(**)

DSA 金鑰產生方法

- 公有的公開金鑰值 (p,q,g):
  - 一個大質數 p (L位元長度)
    - L= 512 到 1024 位元，並且是 64 的倍數
  - 選取 q, 一個 160 位元的質數，且為 p-1 的因數
  - 取 $g = h(p-1)/q$
    - $h<p-1$, $h(p-1)/q \pmod p > 1$
- 使用者選取私密金鑰，並計算公開金鑰:
  - 選取 $x<q$
  - 計算 $y = g^x \pmod p$

(四) 數位簽章標準(**)

DSA 簽章產生方式

- 傳送者以下列方式簽署訊息:
  - 產生一個隨機簽署金鑰 k，k<q
  - k 必須是亂數，用完必須丟棄，不可重複使用
- 計算一組簽章:

  $r = (g^k \bmod p) \bmod q$

  $s = (k^{-1}.SHA(M)+ x.r) \bmod q$
- 將簽章 (r,s) 與訊息 M 一併傳送

(四) 數位簽章標準(**)

DSA 簽章驗證方式

- 收到 M 與簽章 (r,s)
- 接收者以下列方式驗證簽章:

  $w = (s')^{-1} \bmod q$

  $u1 = [SHA(M').w] \bmod q$

  $u2 = (r'.w) \bmod q$

  $v = [(g^{u1}.y^{u2}) \bmod p] \bmod q$
- 如果 v=r 則簽章無誤

  預知證明細節，請看書或網站

•經由本章上述說明，我們可將各演算法的功能性統整於表5-1 G1

演算法	訊息加密	訊息鑑別	身份鑑別
對稱式加密法	V		V
公開金鑰加密法	V		V
MAC		V	V
DAA		V	V
HMAC		V	
MD5		V	
SHA-1		V	
RIPEEMD-160		V	
使用者識別碼(Password)			V
DSS		V	V
DSA		V	V
KDC	V		V

表5-1 各演算法的功能性

G1 GUUBA, 2006/8/20

※未來發展趨勢與研發議題

- 生物特徵(Biometrics)身份認證
- IC卡結合Biometrics認證機制
- Three factor authentication
- Multi factor authentication

習題

習題一

- 何謂訊息鑑別碼？

習題二

- 雜湊函式的目的為何？其與訊息鑑別碼又有何不同？

習題三

- 何謂數位簽章？

習題四

- 直接式數位簽章與仲裁式數位簽章有何不同？

習題五

- 直接式數位簽章有何弱點？

習題六

- 主金鑰(master key)與通訊金鑰(session key)有何不同？

Module 5-4: 專案實作

專案目的

- 部署數位簽章的應用環境。
- 實作數位憑證的申請、公告、加密、解密過程。
- 藉由實際操作，進一步了解數位簽章的運作流程，加深同學對此課程的概念。

數位簽章的應用

- 班級成員中每位選修的課程都不同，以致於班代無法有效的宣導班級、系務訊息，為此，班代決定以E-mail方式將訊息內容檔寄給同學。
- 有些情況之下，班代會要求同學填寫個人資料（姓名、生日、身份證號碼.....）以完成班級事務。
- 同學該怎麼鑑別訊息中的內容確實是班代所傳達，而沒有遭到有心人士的竄改？
- 班代可於傳送前將訊息檔以自己的私密金鑰簽章，同學只要有班代的公開金鑰即可驗證簽章。

實作步驟

- 班代先至暨南國際大學憑證管理中心(http://163.22.20.99/NTP/CAMain.htm)申請一組屬於自己的憑證(包含公開金鑰及私密金鑰)。
- 取得憑證後將自己的公開金鑰公告或是面交給同學，若是確實有至暨南國際大學憑證管理中心申請憑證，則可請同學自行至該網站下載班代的憑證。
- 班代以私密金鑰為訊息檔做簽章後傳送給同學，同學以班代的憑證(公開金鑰)對此訊息檔的簽章做驗章，透過此過程則可鑑別訊息檔的傳送者及訊息檔的完整性。

專案摘要

- 唯有班代的私密金鑰才能簽出以班代身份所簽署的簽章，所以班代的私密金鑰務必妥善保管，以防失竊遭冒用。
- 相對的，若是班代希望同學回傳的訊息能確保其完整性，班代也可要求同學申請憑證，並以上述反向的方式運作。
- 驗章所需的是發訊者的憑證(公開金鑰)，若無法取得憑證即無法執行驗章，所以應該考量如何有效地將自己的憑證予以公告。

參考文獻

1. S. William, Cryptography and Network Security: Principles and Practice. Prentice Hall, Second Edition, 1999.
2. E. Maiwald, Fundamentals of Network Security, McGraw-Hill Technology Education, 2004, pp. 404 -411.
3. D. E. G.Denning, M. Sacco, "Timestamps in Key Distribution Protocols." Commun. ACM 24(8), PP.533-536,1981.
4. Neuman BC, Stubblebine SG, "A note on the use of timestamps as nonces." Operating Systems Review, 27(2):1014,1993.
5. 周伯錕,利用智慧卡之遠端身份認證之研究, 國立中興大學碩士論文, 民國九十二年。
6. 謝續平, 網路安全概要2005教材, 交通大學, 民國九十四年, http://dsns.csie.nctu.edu.tw/course/intro~securit/2005/。
7. 黏添壽、吳順裕,資訊與網路安全技術,旗標,民國九十三年十二月, http://www.tsnien.idv.tw/book-3/book3.html。