Module 3:金鑰管理

學習目的

1. 討論私密金鑰分送及管理的問題，並說明公開金鑰產生、分配、交換、托管與回復的原理，建立學生對金鑰管理的整體概念。
2. 本模組利用四個小節介紹以下四個主題
   (1)金鑰分配(key distribution)
   (2)Diffie-Hellman金鑰交換(key exchange)
   (3)共同產生金鑰協定(key agreement protocol)機制安全需求與技術
   (4)金鑰託管與金鑰回復

   透過原理解說及實作，提供專門研習密碼學或資訊安全同學對金鑰管理方法論有一完整性的了解。
3. 建議3-1~3-2使用三個鐘點教授， 3-3~3-4使用三個鐘點教授。

Module 3:大綱

Module 3-1: 金鑰分配(*)

Module 3-2: Diffie-Hellman金鑰交換技術(*)

Module 3-3: 共同產生金鑰協定(key agreement protocol)機制安全需求與技術(**)

Module 3-4: 金鑰託管與金鑰回復(***)

* 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 3-1: 金鑰分配(Key Distribution)

金鑰管理

- 隨著資訊科技的日新月異，資訊作業需要在開放性網路(open network)上安全傳送(secure communication)，資訊的傳送雙方使用金鑰協商(key agreement)或金鑰分配(key distribution) 協定以進行資訊加密/解密;而資訊加密/解密背後均和金鑰管理(key management)有關。
- 金鑰管理可參考ANSI X9.17標準，管理項目包括系統角色(role)、金鑰產生及檢測、金鑰符記(token)格式、金鑰註銷(cancellation)、金鑰回復(recovery)及金鑰封存機制。實務上金鑰管理還包括金鑰分配、金鑰託管與金鑰回復等技術。
- 因篇幅限制，本教材只探討共同金鑰協商的安全需求與技術、金鑰產生後的分配方法、金鑰託管(Key Escrow)與金鑰回復等技術。

金鑰分配

- 運用對稱式加密法傳送資訊，雙方須使用共用金鑰對，為保護不讓別人知道，故需提供金鑰分配的方法，故秘密金鑰分配一直是金鑰管理的重要的議題。
- 金鑰分配是指將共用金鑰送到要交換資訊的雙方，而不會讓別人知道。
- 需要有一安全方法(秘密管道)提供金鑰的分送。

金鑰分配

- 金鑰分配的四種方法
  - 手動方式
    - 由A選定一把金鑰，然後親自交給B
    - 由第三者選定一把金鑰，然後親自交給A和B
  - 自動化方式的金鑰分配
    - 假設A和B已經利用某金鑰在傳送，其中一方可利用舊的金鑰將新的金鑰加密再傳送給另一方
    - 如果A和B都和第三者C有安全連線，那麼C可利用安全連線將金鑰以加密方式傳送給A和B
  - 四種方法中以最後一種為最符合現有作業方式，其中通訊金鑰可由私密金鑰或公開金鑰來產生，首先我們介紹前者的運作方式

連線導向方式的金鑰分配

1. Host sends packet requesting connection
2.Front end buffers packet; asks KDC for session key
3.KDC distributes session key to both front ends
4.Buffered packet transmitted

FEP = front end processor
KDC = key distribution center

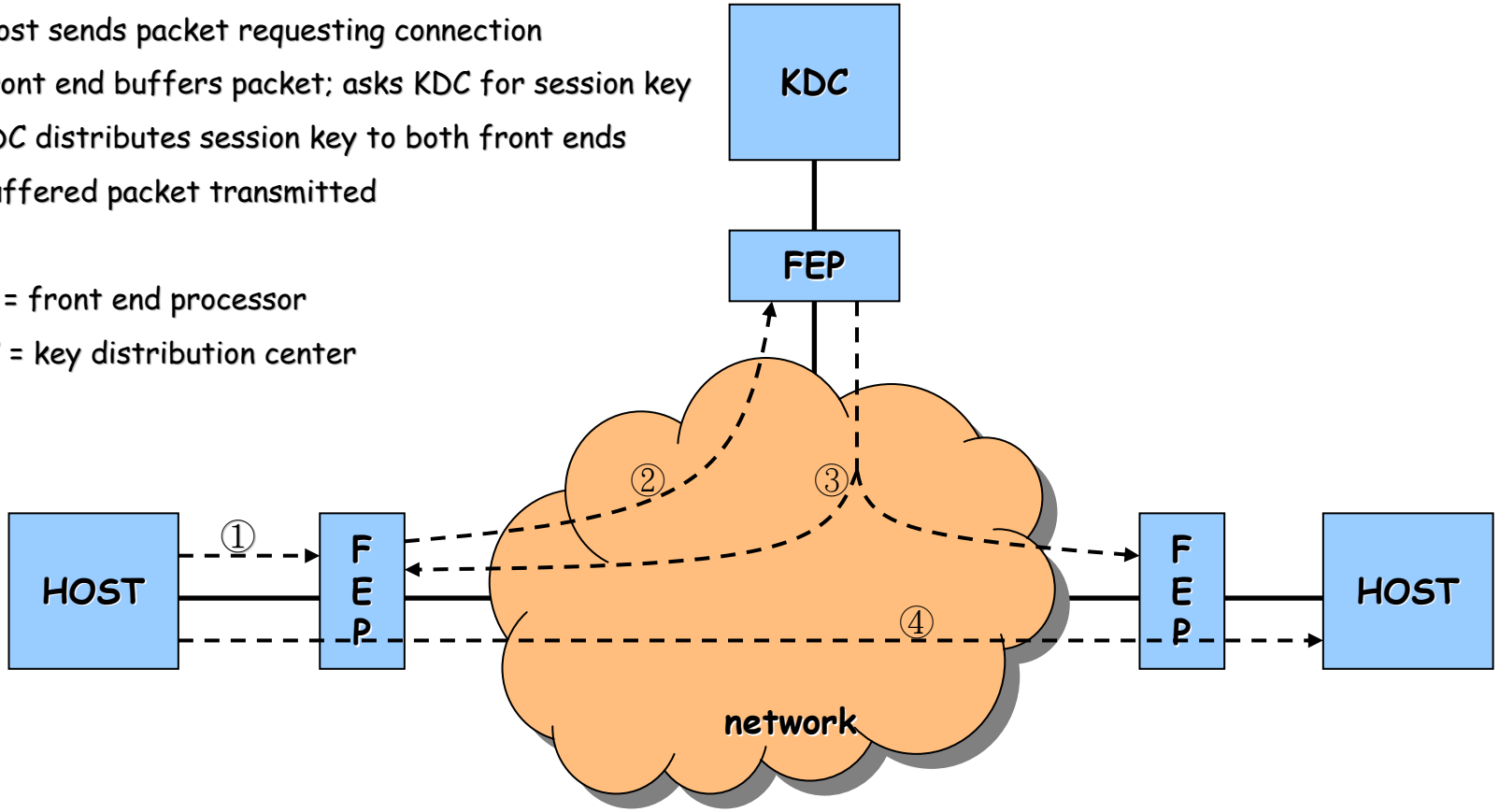


圖3-1 以連線導向協定的自動化金鑰分配

連線導向方式的金鑰分配

- 適用於分散式網路自動化方式的金鑰分配。
- 執行步驟:
  - 某一主機送出要求連線封包
  - 前端處理器 (Front-end processor, FEP)將封包暫存起來，並向金鑰分發中心(Key Distribution center, KDC)要求連線金鑰
  - KDC將運用永久的(私密)金鑰 (permanent key) 產生通訊金鑰，並分送至兩端的FEP
  - FEP運用通訊金鑰將之前暫存的連線封包予以加密後傳送出去

連線導向方式的金鑰分配

- 上述方法採用私密金鑰產生通訊金鑰，但會隨著通訊點增加而須保管過多私密金鑰 (n點通訊須保管n(n-1)/2個私密金鑰)，易造成安全漏洞;

- 以下我們探討利用公開金鑰加密式來分配金鑰(個人只需保管自己私密金鑰，別人的公開金鑰可於公開目錄中取得)。

金鑰分配—公開金鑰加密法

- 公開金鑰加密法有助於解決私密金鑰分配時的過多金鑰對及保管不易問題。
- 有兩項研究的議題敘述如下：
  - 公開金鑰分送
  - 利用公開金鑰加密法來傳送私密金鑰

議題一、公開金鑰的分送

- 可用下列方法之一來達成：
  - 公開聲明
  - 公用目錄
  - 公開金鑰管理中心
  - 公開金鑰憑證

公開聲明

- 使用者將其公開金鑰送給其他接收者、或是將其金鑰廣播給全部的人知道
  - 例如PGP 金鑰附在訊息之後，或是發佈在新群組或郵遞清單上
- 最大的弱點是偽造
  - 任何人都可以製造一把金鑰，然後宣稱是其他人的公開金鑰並散播這把金鑰
  - 這樣可以偽裝成某使用者，直到被發現是偽裝為止

公用目錄

- 在公用目錄中註冊金鑰，可以達到更高的安全性。
- 可靠的公用目錄必須具備以下特性：
  - 保存{姓名, 公開金鑰}的項目
  - 使用者可以安全地向目錄註冊
  - 使用者可在任何時間更換金鑰
  - 週期性地公布整個目錄
  - 可以電子方式存取此目錄
- 仍然受到竊聽或偽裝的威脅。

公開金鑰管理中心

- 加強對目錄的控管來強化金鑰分送的安全性。
- 具有公用目錄的特性。
- 使用者必須知道目錄的公開金鑰。
- 然後使用者就可以透過目錄來安全地取得所需的公開金鑰
  - 當需要金鑰的時候，可以即時的存取目錄

公開金鑰管理中心

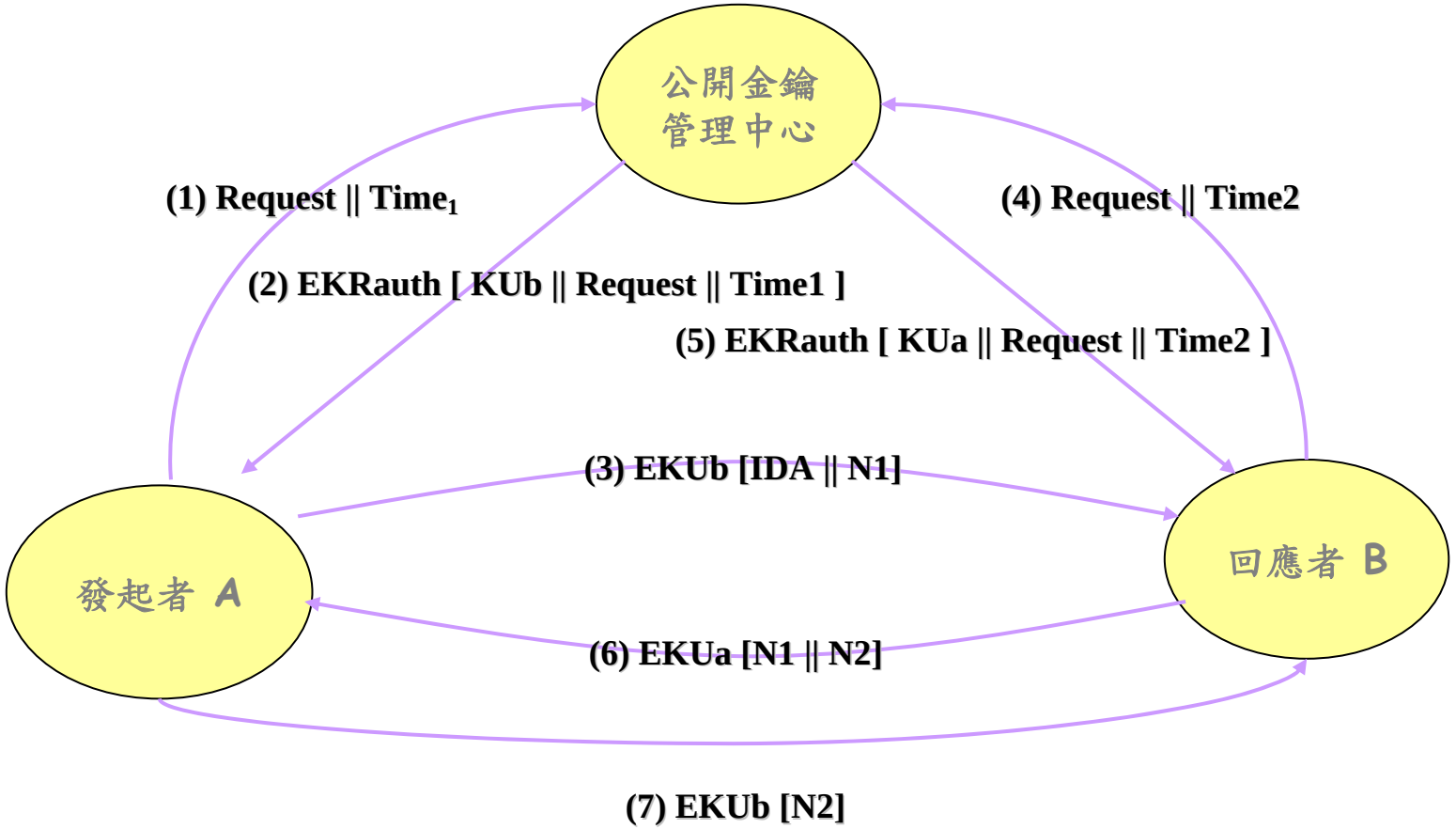


圖3-2 公開金鑰的分送實例

公開金鑰憑證

- 藉由憑證，我們可以在不需即時存取公開金鑰管理中心的情況下來交換金鑰。
- 憑證將使用者與公開金鑰連結在一起
  - 通常還包括其他資訊，像是有效期限、使用者的權力等等
- 憑證的內容必須由可靠的公開金鑰或憑證管理中心 (CA) 來簽署。
- 任何知道公開金鑰管理中心公開金鑰的人，都可以驗證此憑證。

公開金鑰憑證

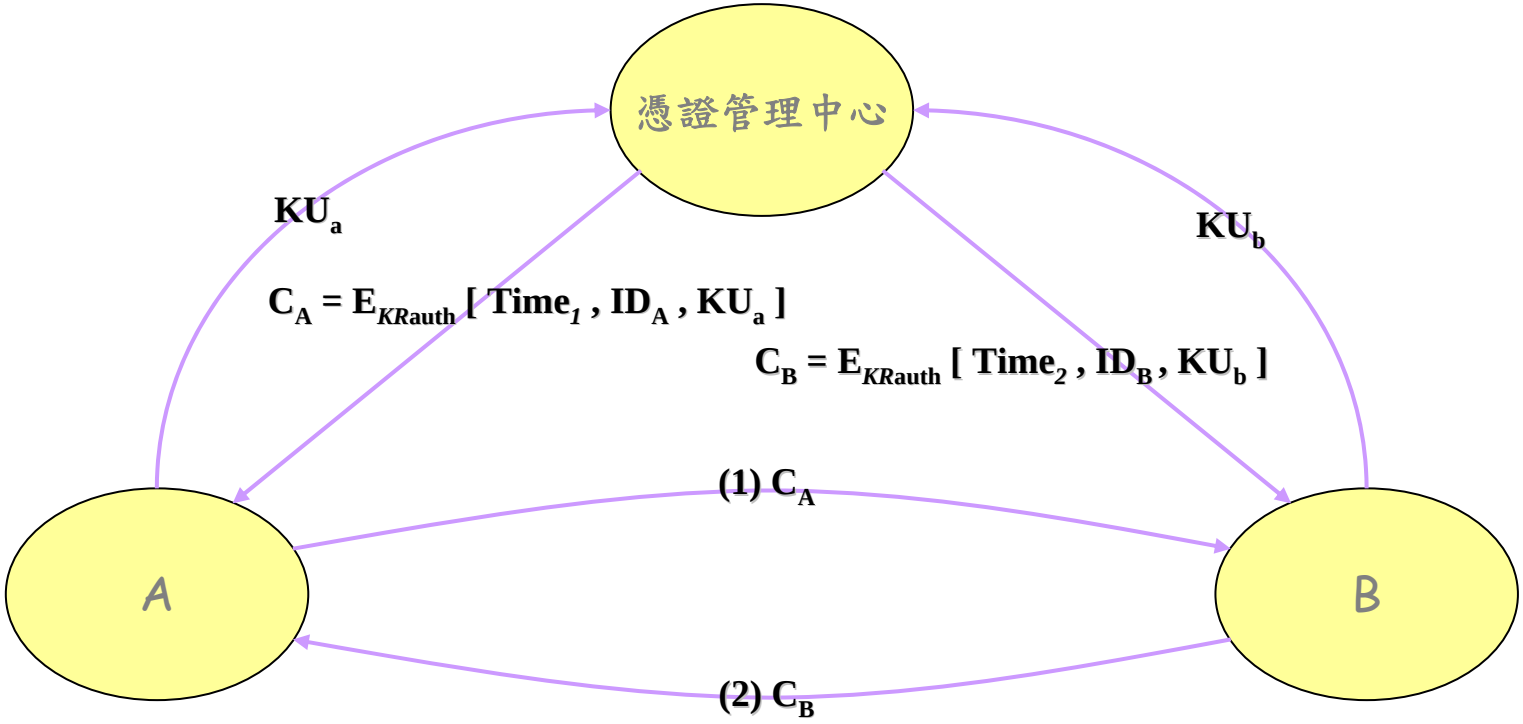


圖3-3 交換公開金鑰憑證

議題二、使用公開金鑰來分送秘密金鑰

- 可用來確保秘密金鑰傳輸時的安全性與確認性
- 但是公開金鑰演算法都很慢
- 所以我們希望用祕密金鑰加密法來保護訊息內容，通常不會直接將祕密金鑰作加密，因為可能遭破解，故需要一把通訊金鑰(session key)
- 有許多可行方案來協調出一把通訊金鑰，例如
  - 簡易秘密金鑰分送法
  - Diffie-Hellman金鑰交換

1.簡易秘密金鑰分送法

- 由 Merkle 於 1979 年所提出
  - A 產生一組臨時的公開/私密金鑰
  - A 將其身份與公開金鑰傳給 B
  - B 產生一把通訊金鑰 K，然後用 A 的公開金鑰加密之後送給 A
  - A 解密取得通訊金鑰，雙方以後的資訊傳輸即可使用這把通訊金鑰K作加密
- 問題在於，對手可以攔截並偽造部分訊息。

2.使用公開金鑰來分送秘密金鑰

- 假設雙方如果已經安全地持有彼此的公開金鑰

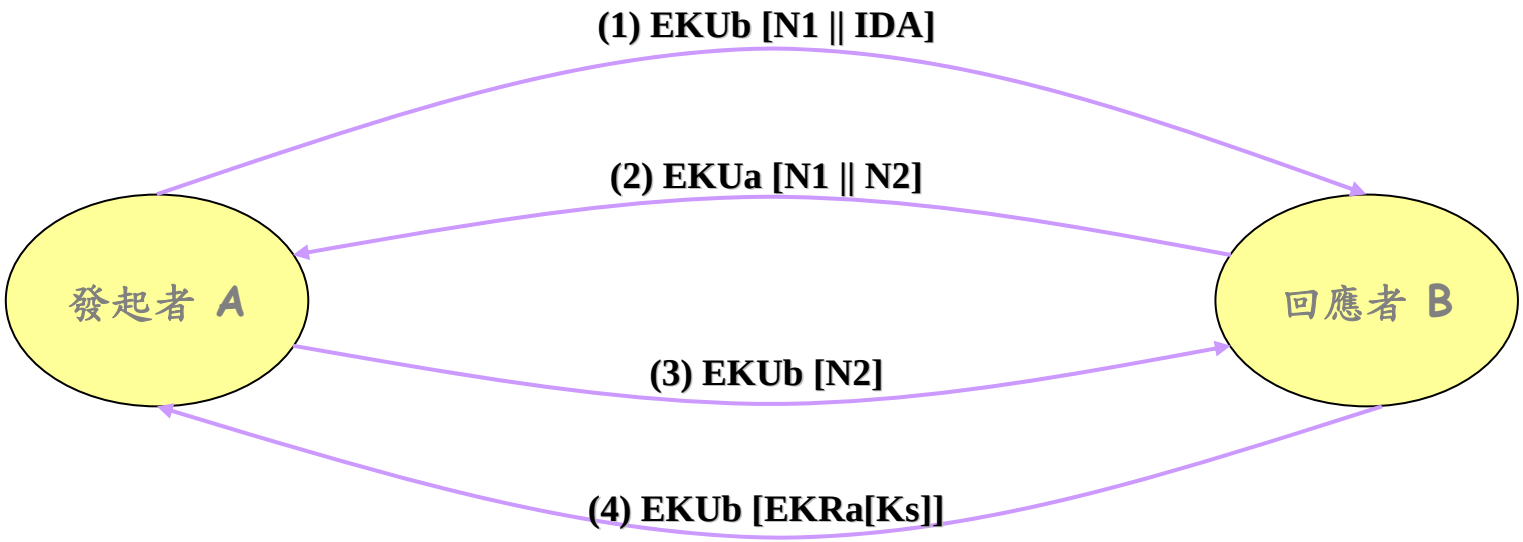


圖3-4 利用公開金鑰來分送祕密金鑰

Module 3-2: Diffie-Hellman金鑰交換技術

Diffie-Hellman金鑰交換技術

- 金鑰交換是指兩個或多個成員在開放網路環境下，透過相互間訊息的交換，彼此能夠共享一個秘密的資訊。

- 由 Diffie 與 Hellman 於 1976 年提出，也引出了公開金鑰的概念來建立一個秘密交換資訊的管道
  - 最早提出的公開金鑰架構
  - 但James Ellis (UK CESG) 已於1970年時不為人知地已提出此概念
- 主要目的在於讓網路上未曾見面的雙方，可以透過計算模指數（modulo）運算，而使得雙方可以獲得相同的交談金鑰（session key）。

- 是一個可以實際用於公開交換秘密金鑰的方法及商用產品。

Diffie-Hellman金鑰交換技術

- 一個利用公開金鑰的分送機制，其特色為
  - 可以用來產生一把通訊雙方共用的金鑰
  - 只有通訊雙方知道此把共用金鑰
  - 金鑰的內容由通訊雙方來決定
  - 以有限體(Galois)的指數運算為基礎(取某質數或多項式的同餘)
- 此金鑰交換系統安全性一般認為是植基於解離散對數(discrete logarithm)的困難度，以目前演算法的安全分析來說，到目前為止還是很安全的。

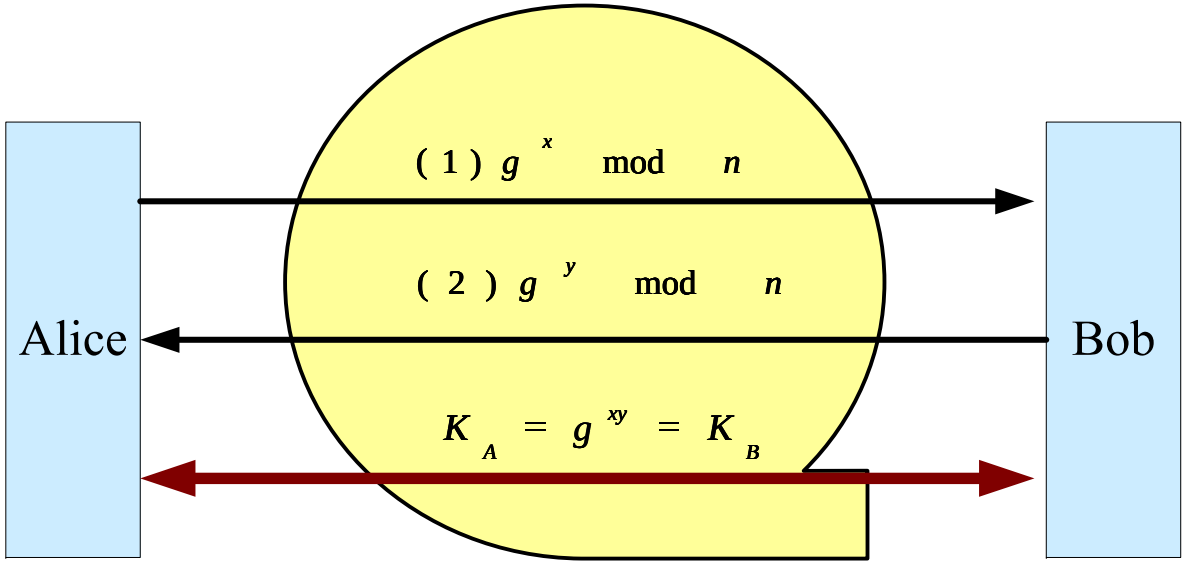
各種相關數學難題 (**)

- 離散對數問題（DLP）*

  $b \in Z_q^*$為已知數，給定一大質數n 和生成數g ，計算 $1 \le x \le q-1$且滿足 $g^x \equiv b \pmod n$，其中Z_q^*為基底為q的所有實數集合。

- Diffie-Hellman問題（DHP）*

  $a,b,c \in Z_q^*$為未知數，給定$g^a \bmod n$ 和$g^b \bmod n$，要求得 $g^c \bmod n$ 且滿足$c = ab \bmod q$。



(3) $K_A = (g^y \bmod n)^x$
$= g^{xy} \bmod n$

(4) $K_B = (g^x \bmod n)^y$
$= g^{xy} \bmod n$

圖3-5 Diffie-Hellman金鑰交換

Diffie-Hellman 的步驟

1. 給予所有使用者以下共同參數：
   - 一個大質數n
   - 取 n 的同餘之下的原根(primitive root)*，g
2. 每位使用者 (例如 A) 產生其金鑰
   - 選取一把秘密金鑰 (一個數)：$x_A < n$
   - 計算其公開金鑰： $y_A = g^{x_A} \bmod n$
3. 公布此金鑰 y_A，同理 $y_B = g^{x_B} \bmod n$

Diffie-Hellman 的步驟(續)

4. 使用者 A 與 B 共用的通訊金鑰 K_{AB}：

   $K_{AB} = g^{x_A.x_B} \bmod n$

   $= y_A^{x_B} \bmod n$ (B 可以算出K_B)

   $= y_B^{x_A} \bmod n$ (A 可以算出K_A)

5. K_{AB} 是用於私密金鑰加密架構下的通訊金鑰，由 Alice 與 Bob 共同持有

6. 在 Alice 與 Bob 的後續通訊中，他們會使用與之前相同的通訊金鑰，除非他們選用新的公開金鑰

7. 攻擊者要知道某個 x, 必須計算離散對數

Diffie-Hellman 範例

- Alice 與 Bob 想要交換金鑰:
- 雙方都知道質數 n=353 與 g=3
- 隨機選取秘密金鑰:
  - A 選取 x_A=97, B 選取 x_B=233
- 計算公開金鑰:
  - $y_A = 3^{97} \bmod 353 = 40$ (Alice)
  - $y_B = 3^{233} \bmod 353 = 248$ (Bob)
- 計算共有的通訊金鑰:

$$K_{AB} = y_B^{x_A} \bmod 353 = 248^{97} \bmod 353 = 160 \quad \text{(Alice)}$$

$$K_{AB} = y_A^{x_B} \bmod 353 = 40^{233} \bmod 353 = 160 \quad \text{(Bob)}$$

Diffie-Hellman 範例

- 如果某人監聽線上的流量時，他們或許會得知a、b、I和J，但是i 和j依舊安全無虞。
- 這個系統的安全性，主要是依據即使得知$I = a^i \bmod b$也很難找到 i。
- 這個問題稱為離散對數問題，因而數字越大也就越難算出結果（也就是說以現今的電腦能力也很難算出結果）。
- 必須非常小心選擇a和b(夠大)。

Diffie-Hellman方法的限制

- 金鑰交換過程缺乏提供溝通雙方相互身份認證的功能，任何人都可以假冒成對方跟其他使用者作秘密通訊，無法有效抵擋中間人攻擊(man-in-the-middle attack)

- 因此此協定須增加身份認證管道以交換雙方的識別資訊(如憑證)—認證金鑰協商協定(authenticated key agreement protocols)

- 詳細內容請參考—Efficient authenticated key agreement protocols resistant to a denial-of-service attack 鄭育民(2005)[1]

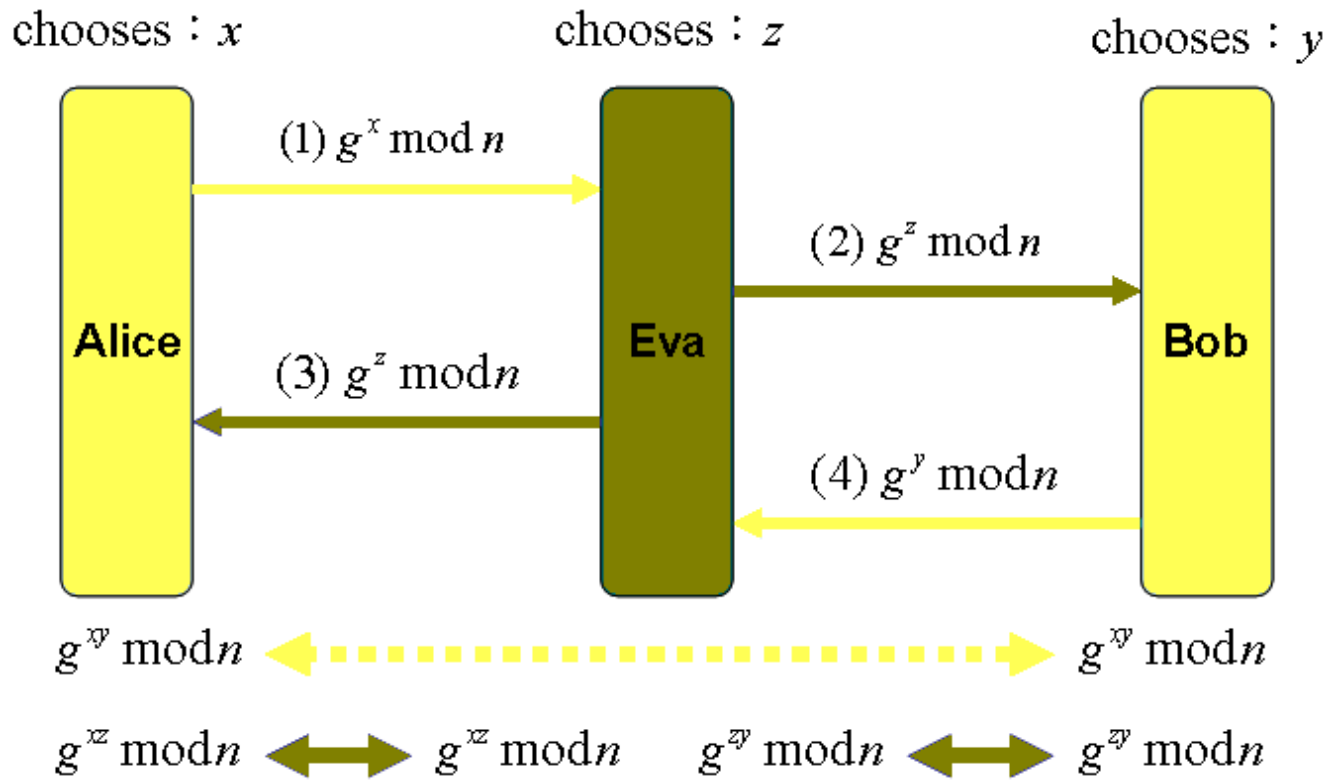


圖3-6 中間人攻擊的運作模式

中間人攻擊

- 假設有一個惡意的攻擊者Eva，她只要輕易地將Alice原本要傳送給Bob的訊息，偷偷換成自己所知道的再傳送給Bob。
- 因為Bob無法驗證訊息的來源是誰，Bob便會誤信自己是和Alice在作秘密通訊，而事實上Bob卻是與Eva建立了共同的交談金鑰。
- 同樣地，Eva也能使Alice誤以為自己是和Bob作了秘密通訊。
- Diffie-Hellman協定沒有身份鑑別（authentication）的能力，此即為此機制最大的安全缺點之一。

Module 3-3: 共同產生金鑰協定機制安全需求與技術(**)

金鑰產生協定的基本概念

- 設計一個優質的資訊系統時，「效率」和「安全」被視為設計時最須考量的兩項關鍵因素。在合乎快速使用需求的同時，也不能對系統安全有些許的退讓和鬆懈。
- 除了要考慮「最少計算成本」、「最佳執行效率」及其「使用便利」的因素之外，系統的安全性更是決定此一設計為人廣泛使用的最大關鍵。

金鑰產生協定的基本概念

- 尤其是以一個金鑰交換系統的設計而言，健全的系統安全更是重要

- 因此，在我們接著探討金鑰協議機制之前，必須先討論金鑰協議機制可能會遭受什麼型態的攻擊方法和一些必備的安全性質。

基本概念

首先先定義金鑰交換時可能面臨的二種攻擊類型：

被動式攻擊（passive attacks）

- 被動式攻擊方法是指攻擊者能夠竊取網路上所傳送訊息的內容，但並不對訊息及系統本身進行破壞。攻擊者常用的方法有：竊聽（eavesdropping）、窺探（Snooping）、攔截（Interception）…等。

主動式攻擊（active attacks）

- 主動式攻擊方法是指攻擊者對網路上所傳送的訊息進行破壞。常見的攻擊方式有：竄改（modify）、偽造（masquerade）、延遲（delay）、重送（replay）、阻斷服務攻擊（DDOS）、否認攻擊…等。

3-3共同產生金鑰協定機制

- 因為Diffie-Hellman方法缺乏提供溝通雙方相互身份認證，已有多個身份認證金鑰協商協定(authenticated key agreement protocols)被提出，依據認證技術分類可將協定區分成兩大類：
  - 公開金鑰為基礎的金鑰協商協定(public-key-based key agreement protocols)
  - 密碼為基礎的金鑰協商協定(password-based key agreement protocols)
- 前者整合數位簽章以提供雙方身份認證；而後者事先交換一組安全密碼以提供雙方身份認證。
- 本節探討共同(協商式)產生金鑰協定(key agreement protocol)機制的安全需求與技術。

3-3共同產生金鑰協定的安全需求

- 依據詹進科，曾育民等(民91)，一個身份認證金鑰協商協定(authenticated key agreement protocol, AKAP) 必須達成下列兩種安全目標(security goals)的其中一種：

1. 內隱式金鑰確認(Implicit key confirmation)：

   通訊的單方須證明給對方知道它能計算出通訊金鑰

2. 外顯式金鑰確認(Explicit key confirmation)：

   通訊的單方須證明給對方知道它已經計算出通訊金鑰

3-3共同產生金鑰協定的安全需求

此外，一般在設計AKAP金鑰協議機制時必須符合五大安全特性以抵抗惡意的攻擊

1. 已知金鑰安全(Known-key security)：

- 在這裡所謂的金鑰指的是共享的交談金鑰，已知金鑰安全是指當某一次的交談金鑰不小心洩漏了，其它次的交談金鑰也不會因此而跟著洩漏。
- 已知金鑰安全主要是將交談金鑰洩漏所產生的安全危害侷限在當次的秘密通訊過程內。

3-3共同產生金鑰協定的安全需求

2. 前推安全(Forward secrecy) :

- 前推安全是指即使一個或多個使用者生命週期較長的秘密金鑰（long-term private key）不小心洩漏了，在此秘密金鑰洩漏之前所產生的交談金鑰也不會因此而連帶洩漏。前推安全主要是對於過去被加密的資料提供完善的機密性保護。

- 若在所有的使用者的秘密金鑰全部洩漏的情形下，系統還能保護所有過去已被加密的資料的話，我們稱此為完美前推安全（Perfect Forward Secrecy, PFS）。

3-3 共同產生金鑰協定的安全需求

3.金鑰洩漏模仿(Key-compromise personation)：

- 金鑰洩漏模仿是指當A的秘密金鑰不小心洩漏給了攻擊者B。
- 在攻擊者握有此秘密金鑰的情形之下，攻擊者只能模仿自己是A來欺騙其他人，並無法模仿B或其他成員回過頭來蒙騙A。

3-3共同產生金鑰協定的安全需求

4. 未知金鑰分享 (unknown key-share resilience):

- 在金鑰交換協議中，任何人皆不能在使用者A不知情的狀況下，強迫與使用者A分享一把共同交談金鑰。
- 而未知金鑰分享是在指，當完成金鑰協議機制後，使用者A相信自己與使用者B共同分享一交談金鑰，但使用者B卻認為自己是和另一攻擊者C建立一交談金鑰。
- 最後造成在使用者A不知情的狀況下與攻擊者也分享了此共同交談金鑰，因此，一個健全的金鑰協議機制必須能夠抵擋這類型的攻擊。

3-3共同產生金鑰協定的安全需求

5. 金鑰控制／支配（Key Control, KC）

- 金鑰控制安全指的是，任何一次的交談金鑰建立過程中，交談金鑰的產生方式必須由使用者A和B共同合作建立。
- 金鑰控制安全主要是保護交談金鑰的產生方式不能為單獨一方所決定，以達到安全及公平的原則。

3-3共同產生金鑰協定的安全需求

- 唯有同時能滿足上述的「已知金鑰安全」、「完美前推安全」、「金鑰洩露模仿安全」、「未知金鑰分享安全」以及「金鑰支配安全」等五種安全特性的金鑰協議機制，才能符合金鑰協議機制的安全需求。

3-3共同產生金鑰協定技術(**)

- 金鑰交換的觀念最早是在1976年，由Diffie和Hellman兩位學者所提出。但是，原始的方法卻無法有效抵擋中間人的攻擊，原因在於，Diffie和Hellman的方法中並無法鑑別參與通訊成員的身分。
- Seo和Sweneey在1999年提出了共享密碼的概念來驗證通訊雙方的身份，並利用冪次方運算產生通訊金鑰 (周世峯, 王有禮,民94)
  - Seo-Sweeney 協定仍舊有遭受反向重送攻擊的弱點

3-3共同產生金鑰協定技術(**)

- 在2000年，Joux首先提出一套僅需一次通訊量的三方式金鑰協議協定。
  - Joux利用Weil Pairing的特性提出了三方的Diffie-Hellman金鑰交換協定，在Joux的協定中每個人只需廣播一次公開的訊息就可協議出一把共同的通訊金鑰
  - 如同原始Diffie和Hellman方法中沒有驗證參與者的身分一樣，Joux的協定仍然面臨了中間人攻擊的問題

3-3共同產生金鑰協定技術(**)

- 在2003年K. Shim為了解決Joux協定的問題提出了具有身份驗證的三方金鑰交換協定
  - K. Shim的協定主要的概念是利用數位憑證來作身份的驗證，並將Weil Pairing運用在冪次方的運算。
- 此外，Al-Riyami等人亦針對身分鑑別問題加以研究，並提出了多個具有鑑別能力的三方式金鑰協議協定。

3-3共同產生金鑰協定技術(**)

- 在2003年，Liu等人則是以身分碼密碼系統為考量基礎，設計了一套能夠防止中間人攻擊的三方式金鑰協議協定（LZC金鑰協議協定）。

- Liu等人聲稱此方法合乎所有的安全需求，並且在一次訊息交換後，能同時產生八把共享的交談金鑰。

- 但是，Shim等人在2005年發表的論文中，指出LZC金鑰協議協定仍然無法有效防範未知金鑰分享的攻擊。並且發表了一套改進的金鑰協議協定（Shim-Woo金鑰協議協定）。

3-3共同產生金鑰協定技術

- Shim-Woo金鑰協議於簽章的設計概念採用SCDHP的計算難題假設，用以解決LZC金鑰協議機制遭受未知金鑰分享的攻擊。

- 此機制中除了使用者本身之外，其他人皆無法同時擁有兩個秘密參數和秘密金鑰，因此任何人皆無法經由偽造合法的簽章進行攻擊。

- 此金鑰協定因不採用單向雜湊函式的運算，所以有較好的執行效率。

3-3共同產生金鑰協定技術(**)

- 密碼學上研究金鑰交換議題範疇仍然相當廣泛，陸陸續續的還有學者提出雙方式金鑰協議、群體式金鑰協議，甚至是樹狀結構群體式金鑰協議...等各種方法。
- 唯因網路攻擊手段不段翻新，在開放網路環境下，金鑰協商協定需不斷被檢視；以目前金鑰協議機制的發展狀況來看，仍然存在許多尚待繼續深入研究及解決的問題(如計算的複雜度及使用不易等)。

Module 3-4:金鑰託管與金鑰回復(***)

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

金鑰託管與金鑰回復

- 近年來，金鑰回復 (Key recovery) 在密碼相關研究領域裡逐漸成為一個熱絡討論的議題。此議題起源於1992 年Micali 所提出公正之公開金鑰密碼系統 (Fair public key cryptosystem)。
- 但是引起廣泛的注意與探討則是在1993 年4 月16 日，美國政府提出一種金鑰託管系統(Key Escrow System, KES)稱為託管加密標準(Escrow Encryption Standard, EES)，是一個用於在法院授權下進行合法監聽電話通訊的系統。

金鑰託管與金鑰回復

- 該系統讓使用者通訊設備之間先經由金鑰交換協定取得通訊金鑰，再經由內部的Clipper 晶片另外產生一執法存取欄位(Law Enforcement Access Field, 簡寫LEAF)，將先前協議所得的通訊金鑰用設備的識別金鑰加密存放在該欄位中，再與經由通訊金鑰加密過之內容，一同送至對方完成通訊。

- 當使用者有嫌疑時，在法院的授權下，執法單位可解密LEAF 欄位以取得使用者間的通訊金鑰進行解密監聽。

金鑰託管

- 金鑰託管依金鑰加解密方式可區分為
  - 私鑰託管的金鑰加密系統 (Private-Key-Escrow Cryptosystem)： KES 採用對稱式加解密運算模式來保護通訊金鑰於LEAF欄位中
  - 公鑰基礎的金鑰託管系統(Public-Key-Based Escrow System)： 許多其他研究則是用Micali 的構想，採用非對稱式加解密運算模式來保護通訊金鑰

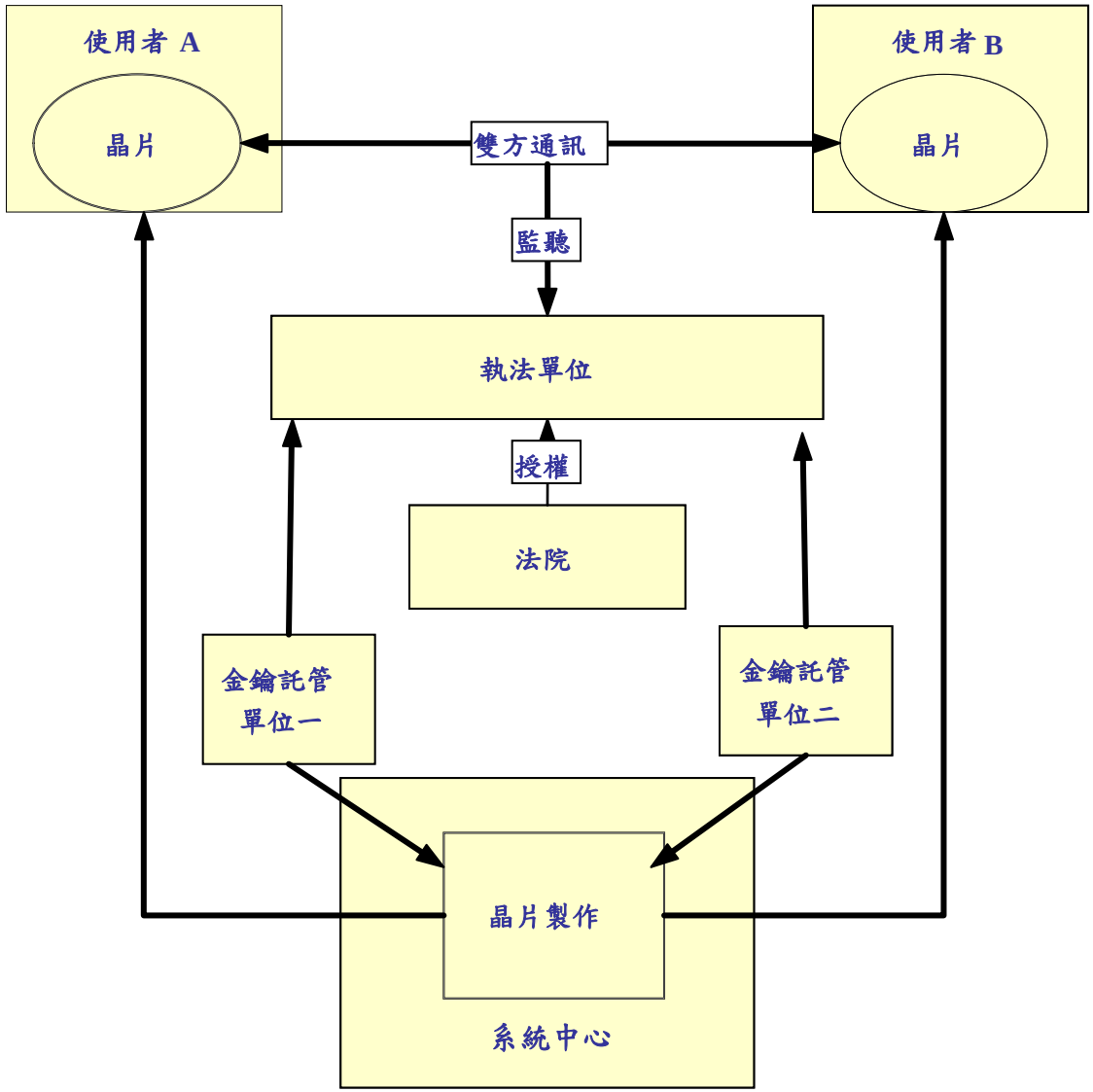


圖3-7 金鑰託管系統架構

資料來源:摘自林幸君, 曹偉駿, 適用電子商務環境之金鑰恢復與託管機制研究

金鑰託管系統(KES)

金鑰託管系統(Key Escrow System, KES)介紹

- 一九九二年，Micali首先提出用非對稱式加解密運算模式來實現金鑰託管的概念，稱為公平密碼系統(Fair Cryptosystem)。
- 其想法是，使用者的私鑰(Private Key)若經過分割而受到託管，則使用者可用其已被託管的私鑰所對應之公鑰(Public Key) 進行金鑰交換(Fair Diffie-Hellman system)或加解密運算(Fair RSA system)。
- 而執法單位在法院授權下，可透過私鑰託管單位的配合重組回通訊金鑰，以對有嫌疑的使用者進行監聽，同時也兼顧到合法通信者的通信安全。

金鑰託管系統(KES) (***)

以下簡單介紹此套系統的運作模式。運作模式分為註冊、通訊及監聽三個階段 。

1. 註冊階段：

- 當使用者要使用此套系統時，首先需向系統中心申請，系統中心利用亂數種子(Random Seed)之技術產生相關的金鑰，並將其儲存於Clipper 晶片中。
- 晶片中之唯一金鑰(Device Unique Key, KU)會被分成兩個部分金鑰KU1 與KU2，且滿足KU=KU1⊕KU2之條件，再將部分金鑰KU1 與KU2 分別託管於兩個可信賴且獨立之金鑰託管單位。

金鑰託管系統(KES) (***)

2. 通訊階段：

- 使用者收到系統中心給的具有防竄改能力之晶片(Clipper Chip)後，便可開始使用此套系統。此晶片中包含有晶片的辨識碼(Device Unique Identifier, UID)，唯一金鑰(Device Unique Key, KU)以及團體金鑰(Family Key, KF)。
- 每一個晶片內的團體金鑰都是一樣的。使用者雙方欲進行通訊時，Clipper 晶片會先利用像Diffie-Hellman 公開金鑰交換協定進行金鑰交換，讓通訊雙方各自獲得一把相同的通訊金鑰(Session Key, KS)。

金鑰託管系統(KES) (***)

- 通訊雙方傳送之訊息則需經過此通訊金鑰的加密過後才傳送出去。
- 訊息傳送方的Clipper 晶片會利用其內部處理器與通訊金鑰(KS)對訊息做加密，得到密文。再利用唯一金鑰將通訊金鑰加密後儲存於執法存取欄位(Law Enforcement Access Field, LEAF)中，最後將密文與LEAF 一同傳送給接收端。
- 訊息接收方收到訊息傳送方傳來的訊息後，先對訊息中的LEAF 內容作驗證動作，確認後再利用KS 解開密文，還原訊息傳送方傳送之訊息。

金鑰託管系統(KES) (***)

3.監聽步驟：

- 當執法單位發現可疑之通訊時，需先向法律授權單位(Court)提出監聽之要求，法律授權單位允許後，執法單位截取可疑之訊息後，可由擁有之團體金鑰解密LEAF，便可取得發送端之晶片識別碼。

- 再將此晶片識別碼及法律授權許可證明送至金鑰託管單位，利用金鑰託管單位傳回之KU1 與KU2可得回KU，接著即可得回KS，最後則可利用KS得知通訊雙方之通訊內容。

金鑰託管系統(KES)的限制(***)

- 系統的缺失—此套密碼系統中仍存在著許多缺失，例如
  - SKIPJACK 演算法未公開，所以使用者無法證實 LEAF 沒有存在任何暗門(Trap-door)，讓未經授權者輕易恢復通訊金鑰，造成民眾的不信賴，也因此學者們紛紛提出相關的改進辦法，例如利用公開金鑰加解密技術來代替原先的技術，或將硬體上的操作轉為軟體.....等 。
  - 由於KU 被建構與出現在金鑰產生與託管階段，使用者不能確認KU 不會遭到不法的保留或複製，因為一旦系統中之晶片金鑰KU 被知道，則加解密金鑰KS 便可獲得，通訊內容即被得知。

金鑰託管系統(KES)的限制(***)

- 隱私權被侵犯— EES 提供具敏感性但卻非機密性之通訊安全，包括電話、傳真及資料傳輸。 使民眾保有通訊之隱密性，同時也提供政府監聽違法行為之技術。但是這卻造成了民眾不願意去使用，因為民眾認為個人的隱私權被侵犯了。
- 單向監聽— KES金鑰恢復與託管機制，只能由發送方監聽，使得當嫌疑犯是單向通訊的接收方時，執法單位將無法監聽嫌疑犯的接收訊息。

金鑰託管系統(KES)的改進(***)

1.Shamir提出新的金鑰託管機制，稱為部分金鑰託管(Partial Key Escrow)，此新機制之目的在於防止政府無限制的監聽。

2.Mao提出公開驗證部分金鑰託管，此機制對Shamir 部分金鑰託管機制提出改進辦法，利用公開驗證能力讓系統在建立部分託管金鑰時改善其效率與確保正確性。

金鑰託管系統(KES)的改進(***)

3. Shaoquan 與Yufeng 也針對部分金鑰託管機制中的缺失提出探討，其缺失為一旦LEA 監聽一位使用者後，則LEA 將永遠知道使用者的秘密金鑰c，所以很容易造成使用者的權利受損，這樣的缺失一般稱為監聽問題(Monitoring Problem)。

4. 針對此問題，Jiang 與Zhang 提出了一個新的監聽機制，當LEA 執行監聽時，KEA 不提供shares給LEA，而是傳一些有用的資訊，這個有用的資訊只有在此次的監聽中才能有效的被使用，若LEA 下一次要再監聽相同的使用者時，則必須重新向KEA 提出申請。

金鑰託管系統(KES)的改進(***)

5. Micali提出的公平的密碼系統讓基於秘密金鑰的金鑰託管系統有了替代性的選擇。

- Micali開啟以公開金鑰密碼系統支援加密與增加產生和託管金鑰構成要素的方法。
- 由於使用公開金鑰密碼系統，所以金鑰會被分割成秘密金鑰與公開金鑰，被託管的是使用者的秘密金鑰。

金鑰回復需求(***)

- 在金鑰管理中，通常我們會使用一把主金鑰(master key)來產生短暫使用的密碼金鑰(例如session key)，因此需對這把主金鑰作防護措施以以確保系統的安全性及可運作性(availability)。
- 在主金鑰管理中，可能產生以下的問題：
  - 若主金鑰單獨由一為管理者保管，而管理者將主金鑰洩露給他人，此系統產生嚴重的安全性問題

金鑰回復需求(***)

- 若主金鑰單獨由一為管理者保管，而管理者不慎將主金鑰遺失，影響此系統正常操作
- 若保管主金鑰的管理者離職，為了系統管理通常須更換此把主金鑰，但系統的密碼金鑰均由主金鑰來產生，需大廢周章使用舊主金鑰解密，再用新主金鑰對密碼金鑰加密
- 主金鑰通常儲存於某種型式的媒體上，增加了失竊與被盜拷的風險

金鑰回復用途(***)

- 現今商業經濟掛師的社會中，金鑰恢復機制被廣泛地應用，金鑰恢復機制能提供組織和企業處理金鑰遺失、損害與無效的情形。
- 因為在企業中常存在一些商業上的機密文件或資訊，當利用密碼系統將這些機密文件或資訊加密後，倘若解密金鑰遺失或者毀損，則加密的資料便無法解密還原。
- 但若利用金鑰恢復機制，則可重新取得解密金鑰，避免企業的損失。

金鑰回復

一般的金鑰恢復方法可分為下列四種類型

1.金鑰託管(Key Escrow)

- 傳送密文的一方將其秘密金鑰託管到一個或一個以上的金鑰託管單位(Key Escrow Agent, KEA)。
- 經認證授權之使用者需向金鑰託管單位提出請求，才能恢復金鑰。
- 儲存在金鑰託管單位中的金鑰通常是使用者的秘密金鑰或每次通訊之通訊金鑰，由使用者直接託管。例如，Clipper 晶片 。
- 此方法之缺點為在初始與儲存階段時，與第三方傳輸的通訊量非常大。

金鑰回復

2.公正第三者(Trusted Third Party, TTP)

- TTP 就像是金鑰分配中心(Key Distribution Center, KDC)，負責提供使用者金鑰。經認證授權之使用者需向TTP 提出請求，才能恢復金鑰。例如，Yaksha 系統。
- 其優點為由獨立公正第三者集中金鑰管理，此方法之缺點為需大量儲存空間及成本高。

金鑰回復

3.商業的金鑰備份(Commercial Key Backup)

- 傳送方使用通訊金鑰(Session Key)作對稱式加密資訊，再利用其公鑰非對稱加密此通訊金鑰。此外，傳送方的私鑰必須要儲存在一個備份單位(Backup Agent)中。
- 經認證授權之使用者需向此備份單位提出請求，才能恢復金鑰。(例如 AT&T Crypto Backup)
- 其缺點為備份空間的需求大及備份儲存位置有潛在的危險性。

金鑰回復

4.金鑰封裝(Key Encapsulation)

- 使用者利用產生的金鑰恢復資訊(Key Recovery Information, KRI)替代其通訊金鑰或密鑰被儲存到金鑰恢復單位(Key Recovery Agents)中。KRI中包含了在恢復解密金鑰時所需要的資訊。
- 經認證授權之使用者需向金鑰恢復單位要求恢復KRI後，便可由其中得出使用者的通訊金鑰或密鑰。
- 此方法之缺點為如何標準化KRI及在傳送與儲存階段KRI不能遺失。

金鑰回復

- 由上述四種金鑰恢復方法，我們發現其實主要可將金鑰恢復的方法分為兩種：
  - 金鑰託管
  - 金鑰封裝
- 兩者間最大的差異處在於，前者為直接將使用者金鑰託管，託管單位可能知道使用者金鑰；後者則是託管關聯使用者金鑰的資訊，託管單位較不容易知道使用者金鑰。

金鑰回復需求(***)

- 針對金鑰託管(Key Escrow)的金鑰回復方法，依據主金鑰託管的對象(內部託管或外部託管) 可細分成以下兩種方式：

  (1) 主金鑰內部分享回復

  (2) 主金鑰外部協助回復

金鑰回復功能(***)

- 主金鑰內部分享回復
  - 運用shamir所提的(3,3)門檻式秘密分享技術來設計，透過系統伺服端的程式運用內部次金鑰回復產生主金鑰。
- 主金鑰外部協助回復
  - 採用結合公開金鑰與(t,n)門檻式秘密分享技術，產生次金鑰並由外部使用者協助下加密儲存，若要回復產生主金鑰須經t個以上外部使用者解密次金鑰回復所需主金鑰。

金鑰回復程序(***)

- 主金鑰內部分享回復程序
  - 分享祕密階段

    (1)分配者任意選擇a_1,a_2，套入以下2次多項式

    $q(x)=a_2x^2 + a_1x + a_0 \bmod p$

    其中$ai \in [1,p-1],1 \leqq i \leqq 2$，p為一質數，$a_0$為主金鑰，主金鑰須要小於p

    (2)三位參與者都有一唯一的公開的識別名字w_1,w_2,w_3，分配者依各人的公開代號產生各別次金鑰$v_i=q(w_i)$，$i=1,2,3$，其中w_1為管理者,w_2為伺服器檔案,w_3為網路卡卡號，w_1的次金鑰v_1會被儲存至IC卡

金鑰回復程序(***)

(3)程式取得網路卡卡號以及系統內某檔案之一段內容值s，先計算sk(加密次金鑰時所使用的金鑰)

$sk=h(w_3||s)$

其中h()為雜湊函式

再個別計算$u_1=E_{sk}(v_2)$，$u_2=E_{sk}(v_3)$，並將之儲存於 系統中某檔案內

金鑰回復程序(***)

– 回復祕密階段

(1)系統讀取系統管理者IC卡中之次金鑰v_1，並將u_1,u_2從檔案中抽取出來

(2)程式取得網路卡卡號w_3及在分享秘密階段(3)之位址值s，計算$sk=h(w_3||s)$，再個別計算$v_2=D_{sk}(u_1)$,$v_3=D_{sk}(u_2)$，以便取得3組(w_i,v_i)，i=1,2,3之多項式二維平面上的座標點

(3)得知3個座標點後可使用Lagrange多項式插入法回復二次多項式

(4)多項式回復後，常數項即為主金鑰

金鑰回復程序(***)

- 主金鑰外部協助回復程序
  - 分享祕密階段

    (1)在mod p 之下任意選擇n個整數 $y_1,y_2,\ldots,y_n$

    (2)系統伺服器端程式利用二維座標系中n+1組座標值(x_i,y_i),i=1,2,…,n與(0,mk)，建構一n次多項式f(x)後將f(i),i=1,2,…,n-t+1算出並暫存於記體憶中

    (3)利用n位外部協助者的公鑰分別將y_i,i=1,2,…,n經加密運算後可得z_i,i=1,2,…,n。如下式

    $$z_i=E_{k_{pub\_i}}[y_i]\ ,\ i=1,2,\ldots,n$$

金鑰回復程序(***)

(4)將f(i),i=1,2,…,n-t+1與z_i,i=1,2,…,n備份到另一資料儲存媒體中，以提高回復機制的可用性

(5)以系統的私鑰K_{priv_s}對雜湊值h(f(1),(2),…,f(n-t+1),z_1,z_2,…,z_n)簽章，計算結果為

$sig=Sign_{k_{priv\_s}}[h(f(1),f(2),\ldots,f(n-t+1),z_1,z_2,\ldots,z_n)]$

(6)公佈f(1),f(2),…,f(n-t+1),z_1,z_2,…,z_n及sig

金鑰回復程序(***)

– 回復祕密階段

- 假設有u位外部協助者$x_{i_1},x_{i_2},\ldots,x_{i_u}$($t \leqq u \leqq n$)要回復主金鑰$m_k$

(1)計算$h(f(1),(2),\ldots,f(n-t+1),z_1,z_2,\ldots,z_n)$後以系統的公鑰$k_{pub_s}$透過下列式子驗證sig，若驗證無誤則進行下一步驟，否則停止回復祕密，因為其值非當初分享祕密所使用的

$$h(f(1),(2),\ldots,f(n-t+1),z_1,z_2,\ldots,z_n)=Verify_{k_{pub\_s}}[sig]$$

(2)接著依序讀出u位協助者的IC卡，並個別計算下式中各值，計算完後再將$(x_{i_1},y'_{i_1}),(x_{i_2},y'_{i_2}),\ldots,(x_{i_u},y'_{i_u})$傳送給系統

$$y'_a=D_{k_{priv\_i_a}}[Z_{ia}],a=1,2,\ldots,u$$

金鑰回復程序(***)

(3)系統驗證下式各值是否成立，若全部驗證無誤，則表示

$y'_{i_a}=y_{i_a},a=1,2,\ldots,u$，接著進行下一步驟；否則即表示協助者$x_{i_a}$之IC卡有問題，需馬上停止回復祕密

$$Z_{i_a}=E_{k_{pub\_i_a}}[y'_{i_a}],a=1,2,\ldots,u$$

(4)最後，系統將$y_{i_1},y_{i_2},\ldots,y_{i_u}$及$f(1),f(2),\ldots,f(n-u+1)$以Lagrange多項式插入法建構多項$f(x)$，即可回復

$m_k=f(0)$

結語

- 我們已學到內容:
  - 公開金鑰的分送方式
  - 使用公開金鑰來分送秘密金鑰
  - Diffie-Hellman金鑰交換法
  - 共同產生金鑰協定機制安全需求與技術
  - 金鑰託管與金鑰回復方法

※未來發展趨勢與研發議題

- 數位版權管理(Digital Rights Management , DRM)金鑰應用
- IPv6 Multicast 群體金鑰管理(group key management)
- 無線感測器網路(Wireless Sensor Networks)金鑰管理

習題

習題一

- 請舉出4種分送公開金鑰的一般架構。

習題二

- 試問以公用目錄系統分送公開金鑰，此機制下有何弱點？

習題三

- 試簡述Diffie-Hellman金鑰協議機制易遭受中間人攻擊之原因。

習題四

- 請簡要的介紹Diffie-Hellman金鑰交換法的功用。

習題五

- 何為金鑰託管，請簡單描述

習題六

- 何為金鑰回復，請簡單描述之。

參考文獻

1. Yuh-Min Tseng, "Efficient authenticated key agreement protocols resistant to a denial-of-service attack", International Journal of Network Management, 2005; 15: 193–202.
2. M. A. Strangio, Efficient Diffie-Hellmann Two-Party Key Agreement Protocols based on Elliptic Curves, 2005 ACM Symposium on Applied Computing, SAC'05, March 1317,2005, Santa Fe, New Mexico, USA
3. W. Stallings Cryptography and Network Security， Third Edition ，Pentice Hall.
4. W. Stallings Network Security Essential — Applications and Standards ，Pentice Hall.
5. D. H. Seo, P. Seweeney, "Simple authenticated key agreement algorithm". Electronic Letter, 35, (13)., pp. 1073-1074,1999.
6. W. Diffie, and M. E. Hellman, "New directions in cryptography", IEEE Trans., IT-22, (6), pp. 644-654, 1976
7. A. Joux, "A one-round protocol for tripartite Diffie-Hellman", Proceedings of the 4th International Algorithmic Number Theory Symposium(ANTS-IV), LNCS 1838, July 2000, pp. 385-394.
8. J. Shaoquan and Z. Yufeng: Partial Key Escrow Monitoring Scheme, Cryptology ePrint archive, record 2002.

參考文獻(續)

9. S. Micali, "Fair Public-Key Cryptosystems," CRYPTO, pp.113-138, 1992.
10. K. SHIM, "Efficient one-round tripartite authenticated key agreement protocol from Weil pairing", Electronics Letters, Vol. 39, No. 2, January 2003, pp. 208-209.
11. A. Shamir, “Partial key escrow: A new approach to software key escrow.” Key Escrow Conference, Washington, 1995.
12. S. Liu, F. Zhang, K. Chen, “ID-based tripartite key agreement protocol with pairing”, 2003 IEEE International Symposium on Information Theory, 2003, pp. 136–143.
13. K. Shim and S. Woo. “Weakness in ID-based One Round Authenticated Tripartite Multiple-key Agreement Protocol with Pairings”, Applied Mathematics and Computation 2005, Vol. 166, pp. 523-530.
14. S.S. Al-Riyami and K. G. Paterson. “Tripartite Authenticated Key Agreement Protocols from Pairings”, In Proceedings of IMA Conference of Cryptography and Coding 2003, LNCS 2898.

參考文獻(續)

15. 林幸君、曹偉駿, 適用電子商務環境之金鑰恢復與託管機制研究, 大葉大學碩士論文, 民國九十一年。

16. 顏嵩銘、張明聖,商業性金鑰恢復與金鑰託管機制之研究,國立中央大學資訊工程研究所碩士論文,民90

17. 周世峯、王有禮, 植基於雙線性配對運算的鑑別式金鑰協議協定, 國立台灣科技大學碩士論文,民國九十四年。

18. 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。

19. 詹進科、曾育民、陳育毅、王景弘(民91),可抵擋阻絕服務式攻擊之可認證金鑰交換協定之設計與實作,國立中興大學應用數學系,碩士論文。