行政院國家科學委員會補助專題研究計畫成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※※
※ 數位產權管理系統 ※
※※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：☑個別型計畫　□整合型計畫

計畫編號：NSC91-2213-E-002-098

執行期間：2002 年 8 月 1 日至 2003 年 7 月 31 日

計畫主持人：賴飛羆 教授

共同主持人：

計畫參與人員：林振群（博士班研究生）
董君樸（碩士班研究生）
曾建華（碩士班研究生）
胡裕華（碩士班研究生）

本成果報告包括以下應繳交之附件：

□赴國外出差或研習心得報告一份

□赴大陸地區出差或研習心得報告一份

☑出席國際學術會議心得報告及發表之論文各一份

□國際合作研究計畫國外研究報告書一份

執行單位：國立台灣大學資訊工程系

中 華 民 國 2003 年 8 月 27 日

行政院國家科學委員會專題研究計畫結案報告

數位產權管理系統

Digital Right Management System

計畫編號：NSC91-2213-E-002-098
執行期限：91 年 8 月 1 日至 92 年 7 月 31 日

主持人：賴飛羆　國立台灣大學資訊工程系
計畫參與人員：林振群　國立台灣大學電機工程系

一、中文摘要

在數位產權管理系統中，有一個重要的問題是如何把被保護的資料，有效，安全地送給多個訂購者。在這篇報告中，我們提出了一套分散式的金鑰管理協定，讓群組成員(訂購者)能夠共用一把秘密金鑰，以保護成員間共享的資訊。這個協定管理動態群組中的群組金鑰，群組成員可以自由地加入或是離開，每一次的金鑰更新需要花費兩個廣播訊息。這個協定的演算法達到了群組金鑰的保密，未來的群組金鑰保密，以及過去的群組金鑰保密三種要求。在產生群組金鑰的計算時間，每一個群組成員所需的儲存空間，還有更新一次金鑰所花的通訊頻寬的複雜度上都是接近群組大小的對數函數的大小，使這個協定適合應用在來將受保護的資料，安全地送至多個訂購者的手中。

Abstract

It is an important issue for digital rights management systems that how protected data can be securely and efficiently delivered to all subscribers. In this report, we propose a distributed key management protocol for group members (subscribers) to share a secret group key, which can be used to protect shared information. The protocol manages the group key of a dynamic group, where members can freely join or leave, and each time the key is updated using two broadcast messages. The protocol algorithms provide group key secrecy, forward group key secrecy, and backward group key secrecy. The complexities of the group key computation time, the storage space for every member, and the total communication bandwidth to update the group key are approximately of logarithmic order of the group size, which make the protocol suitable for sending protected data securely to multiple subscribers.

二、緣由與目的

資訊的保密是在通訊上的重要課題。讓在多個訂購者間共享的資訊受到保護，便成為一個重要的課題，這類的應用包括電子佈告板，群組軟體，電子期刊等。通常的解決方法，必須藉由共享金鑰或是將群組分成更小的群組來達到減少效能負擔的目的。目前的解決之道有兩種類別，第一種是集中式的方法，第二種是分散式的方法，各有其優缺點。而集中式的伺服器需要極大的頻寬、計算能力和儲存空間，容易成為系統效能的瓶頸，如果伺服器也是群組成員，更會出現責任分擔失衡的公平性問題。

由於上述的理由，同時我們將訂購者視為一群組，並著眼於提供一個群組金鑰管理的解決方案。我們提出了一個安全有效率的分散式金鑰管理協定給廣播網路上的動態群組。我們的目標是讓群組成員能夠安全有效率地共享一個相同的群組金鑰，以保護共享資訊。我們的協定支持動態群組的運作，包含了群組金鑰的更新，成員的加入和離開。我們的演算法奠基於兩方的 Diffie-Hellman 的金鑰交換的延伸，使用了一個金鑰樹的變型來加快金鑰的計算。每一個群組成員只保留部分的群

組資訊，使得通訊頻寬和儲存空間的需求非常低。

三、研究方法與演算法

我們的協定使用單元而可靠的網路廣播來傳送普通訊息和更新訊息，意思是廣播的訊息能夠被全部的個體收到，否則這個動作就算失敗。使用廣播的原因是通常一個群組成員或是新進成員必須在不知道完全的群組資訊的情況下傳送訊息給全部的群組成員，而且不能倚賴伺服器的幫忙。

我們的協定支援三種運作：群組金鑰更新，新成員的加入，以及舊成員的離開。每一個運作都需要兩個廣播訊息，包含一個要求訊息，和一個更新訊息。要求訊息是用來找尋一個群組成員來起始一個群組的運作。更新訊息則是傳送給全部群組成員以計算新的群組金鑰。

由於運作的順序對系統的恆定有重要的影響，所以全部的群組成員必須以同樣的順序接收到要求的訊息。我們的協定倚靠的是下層網路對完全順序化訊息發送的支援。關於這個傳輸模型更詳細的討論可以在[11]裡面找到。

所有的協定運作都必須被簽章和認證過。使用簽章有兩個理由，首先，接收者能夠確認這個訊息確實是訊息裡宣稱的發送者所送出。另一個理由是要保持訊息的完整性，以確保訊息在傳遞的過程中未遭竄改。

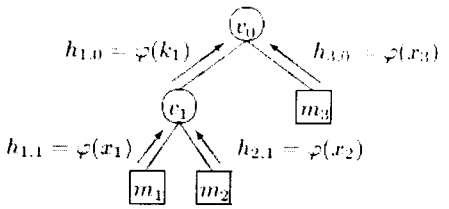
觀念上，每一個群組成員使用兩個執行緒，一個叫做命令執行緒，等待使用者的命令然後用適當的演算法送出要求訊息。另外一個叫做協定執行緒，負責接受要求訊息然後處理。在處理要求的時候，某一個群組成員會廣播一個更新訊息給包括發送者的所有群組成員，然後各自更新他們的成員資訊。

演算法：

群組的金鑰管理協定必須依照金鑰更新的要求或是群組成員身分的改變來更新群組金鑰。我們的協定支援兩種群組身分的運作，加入和離開。一個新的成員起始一個加入的動作來加入這個群組。一個要離開的成員在成功通知全部成員以後離開。我們的協定使用延伸的 Diffie-Hellman 金鑰協商演算法，奠基於兩個個體的 Diffie-Hellman 金鑰交換演算法[8,9] 。每一個群組成員保留部分的群組資訊來計算群組金鑰和保持成員身分的恆定性。

基本觀念：

本金鑰管理機制基本上是金鑰樹的變型，金鑰樹的每一個葉端點代表一個群組成員或者他所擁有的秘密金鑰，每一個內部節點代表和樹根之間包含這個節點的那些群組成員的共享資訊。圖一是一個金鑰樹的範例。



圖一

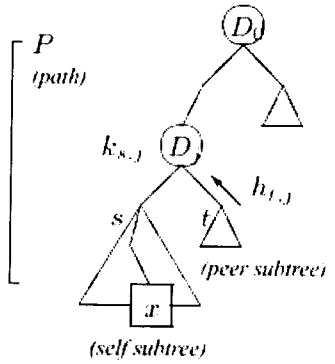
在圖一中，群組中有三個成員，m_1，m_2 和 m_3，每一個 m_i 擁有一個秘密金鑰 x_i。內部節點則被貼上 v_0，v_1 的標籤。首先 m_1 ，m_2 做了一個兩個個體的 Diffie-Hellman 金鑰交換並且產生了一個共享金鑰 $k_1=\varphi(x_1x_2)$ ，它可以被當作 v_1 的秘密金鑰。而 v_1 和 v_2 也做一個 2DH 來產生共享金鑰

$$k_0 = \varphi(k_1 x_3) = g^{g^{x_1 x_2} x_3} \bmod p$$

k_0 可以說是 m_1，m_2 和 m_3 共享的群組金鑰。這個例子暗示，只要我們能把所有群組成員放到一個二元樹的葉端點上，我們就能藉由下而上應用 2DH 很容易地算出群組金鑰。

在沒有集中式的伺服器的情況下，又不適合每個成員都保留整棵的金鑰樹，因為那樣會消耗極大的儲存空間和傳輸頻寬。所以每個群組成員只需要保留從樹根到自己的葉端點的路徑上的資訊即可。每個群組成員保留部分資訊 $M \equiv (I,x,k_G,P)$ ，I 代表他自己的身分代號，x 是他的秘密金鑰，k_G 是群組金鑰，P 是路

徑資訊，如同圖三的內容。每一個節點的資訊都是一個六個單元的項目 $D \equiv (I_s,W_s,k_s,I_t,W_t,h_t)$ 。k_x 和 h_t 用來計算群組金鑰，I_s，W_s，I_t，W_t 則是用來做群組成員身分的管理。這些節點從樹根開始編號，從零開始編。小寫的 s 和 t 代表自己(self)和夥伴(peer) 。我們可以用圖二來解釋自己和夥伴的觀念。



圖二

對於一個節點，k_S 可以視為它的秘密金鑰或者它自己和夥伴的子樹共享的金鑰，而 2DH 協商金鑰 h_t 是 $\varphi(k_t)=g_{kt} \pmod p$，它可以被用來計算一個共享金鑰 $\varphi(k_sk_t)=g^{kskt} \pmod p = (g^{kt})^{ks} \pmod p = h_t^{ks} \pmod p$。普遍上來說，對於一個 L 個節點的路徑來說，我們可以使用下列的循環關係計算出 $k_{s,0}$，也就是群組金鑰。

$$h_{t,j} = r(k_{t,j+1}) = g^{k_{t,j+1}} \bmod p.$$

$$k_{s,j} = r(k_{s,j+1}k_{t,j+1}) = h_{t,j}^{k_{s,j+1}} \bmod p.$$

其中 $0 \le j \le L-1$。圖三中的演算法計算出群組金鑰並且更新路徑上的重量因子。

```
algorithm ComputeGroupKey()
  L ← |P|; k ← x
  for j ← L − 1 down to 0 do
    k_{s,j} ← h_{t,j}^k mod p
    k ← k_{s,j}
    if j ≠ L − 1 then W_{s,j} ← W_{s,j+1} + W_{t,j+1}
  k_G ← k
```

圖三

每一次有成員加入或是離開，所有成員的路徑資訊都要更新來反應群組成員身分的改變。因此，路徑更新資訊 U 被廣播出去，U 和路徑資訊非常類似，它是由一連串的節點更新資訊 E 組成，E 包含六個項目 $(I_S,W_S,h_S,I_t,W_t,h_t)$。E 和 D 唯一的差別是 h_s，它儲存訊息發送者算出的 2DH 的協商金鑰。圖四顯示了處理 U 和 P 的演算法。

```
algorithm CreateU()
  U' ← P; L ← |P|; k ← x
  for j ← L − 1 down to 0 do
    h'_{s,j} ← g^k mod p
    k ← h'_{t,j}^k mod p
    if j ≠ L − 1 then W'_{s,j} ← W'_{s,j+1} + W'_{t,j+1}
  return U'

algorithm UpdatePath(Ũ)
  L ← |P|
  for j ← 0 to L − 1 do
    if (I_{s,j} = Ĩ_{s,j}) and (I_{t,j} = Ĩ_{t,j}) then
      W_{s,j} ← W̃_{s,j}; W_{t,j} ← W̃_{t,j}
    else if (I_{s,j} = Ĩ_{t,j}) and (I_{t,j} = Ĩ_{s,j}) then
      W_{s,j} ← W̃_{t,j}; W_{t,j} ← W̃_{s,j}; h_{t,j} ← h̃_{s,j}
      break
```

圖四

演算法 CreateU 被用來根據一個成員的路徑 P 和它的秘密金鑰 x 產生更新資訊 U。每一個成員都使用演算法 UpdataPath 並根據路徑更新資訊 U 來更新它的路徑。

群組金鑰更新演算法

當一個成員要更新群組金鑰的時候使用群組金鑰更新的演算法。SendRekeyRequest 演算法起始了這個動作，其他每一個的成員執行 HandleRekeyRequest 來完成這個動作。首先發送者選擇一個新的秘密金鑰，廣播出去路徑更新資訊 U，而其他每一個群組成員根據 U 來修改自己的路徑資訊並產生群組金鑰。圖五列出了這個演算法。

```
algorithm SendRekeyRequest()
  broadcast REKEY_REQUEST[I]

algorithm HandleRekeyRequest(Ĩ)
  if I = Ĩ then
    x ← randomly select a new secret key
    U ← CreateU()
    broadcast REKEY_UPDATE[U]
  receive REKEY_UPDATE[Ũ]
  UpdatePath(Ũ)
  ComputeGroupKey()
```

圖五

成員加入的演算法

當一個新進成員想要加入群組的時候，它必須使用演算法 SendJoinRequest 來產生一個秘密金鑰並且傳送加入要求給所有群組成員，這個加入的要求 JOIN REQUEST 包含一個額外的項目 h，也就是用來協商的金鑰，給其他群組成員用來產生新的群組金鑰。每一個成員都會使用 HandleJoinRequest 這個演算法來處理。群

組中有一個成員要負責此次的新成員加入的動作，他選擇一個新的秘密金鑰，計算出路徑更新資訊 U，然後廣播給全部成員。群組成員用 U 來更新他們的路徑資訊，新的成員則用 CreatePath 這各演算法來從 U 產生他的路徑資訊。這時候新進成員就正式成為群組的一員了。他可以根據他的路徑和秘密金鑰計算出群組金鑰。圖六列出了成員加入的演算法。

```
algorithm SendJoinRequest()
  x ← randomly select a new secret key
  h ← g^x mod p
  broadcast JOIN_REQUEST[I, h]

algorithm HandleJoinRequest(Ĩ, h̃)
  if I ≠ Ĩ and IsLeader() then
    x ← randomly select a new secret key
    D ← (I, 1, 0, Ĩ, 1, h̃)
    append D to P
    U ← CreateU()
    broadcast JOIN_UPDATE[U]
  receive JOIN_UPDATE[Ũ]
  if I = Ĩ then CreatePath(Ũ)
  else UpdatePath(Ũ)
  ComputeGroupKey()

algorithm IsLeader()
  L ← |P|
  for j ← 0 to L − 1 do
    if W_{·,j} > W_{I,j} then return false
    if W_{·,j} = W_{I,j} and I_{·,j} < I_{I,j} then return false
  return true

algorithm CreatePath(Ũ)
  P ← Ũ; L ← |P|
  I_{·,L−1} ← Ĩ_{I,L−1}; I_{I,L−1} ← Ĩ_{·,L−1}; h_{I,L−1} ← h̃_{·,L−1}
```

圖六

只有一各成員的 IsLeader 運算結果會是正面的，這各成員要負責產生 JOIN_UPDATE 的訊息並且廣播出去。IsLeader 從樹根開始檢查，凡是存在於比較大的子樹的成員就被排除成為暫時的領導者，所以新近的成員會被安排在比較小的子樹裡面，藉以達到平衡樹的目的。如同 HandleJoinRequest 演算法中顯示的，暫時的領導者只需要把新進的成員加在他自己的路徑的最下端並且更新路徑資訊就可以了。

成員離開的演算法

一個要離開的成員必須使用 SendLeaveRequest 這個演算法來通知其他所有成員他將要離開。它必須等到它的要求被處理完畢以後才能離開。離開者的夥伴(Peer)必須負責產生更新的訊息並且發送給大家，他首先產生一個新的金鑰，並且計算出路徑的更新資訊 U，然後廣播出去。圖七列出了相關的演算法。

```
algorithm SendLeaveRequest()
  broadcast LEAVE_REQUEST[I]

algorithm HandleLeaveRequest(Ĩ)
  if I = Ĩ then leave the group
  if IsPeer(Ĩ) then
    x ← randomly select a new secret key
    ReplaceMember(Ĩ, I)
    U ← CreateU()
    broadcast LEAVE_UPDATE[U]
  receive LEAVE_UPDATE[Ũ]
  L ← |Ũ|; I' ← Ĩ_{I,L−1}
  ReplaceMember(I, I')
  UpdatePath(Ũ)
  ComputeGroupKey()

algorithm IsPeer(Ĩ)
  L ← |P|
  for j ← 0 to L − 1 do
    if I_{·,j} = I and Ĩ_{·,j} = Ĩ and W_{·,j} = 1 then return true
  return false

algorithm ReplaceMember(Ĩ, I')
  L ← |P|
  for j ← 0 to L − 1 do
    if I_{·,j} = Ĩ then I_{·,j} ← I'
    else if I_{I,j} = Ĩ then
      if W_{I,j} = 1 then remove D_j from P
      else I_{I,j} ← I'
```

圖七

四、成果與討論

安全性的分析

我們的協定下層利用的演算法是一個延伸的 Diffie-Hellman (EDH) 金鑰協商演算法，它奠基於重複執行兩個個體的 Diffie-Hellman金鑰交換(2DH) 。EDH中，每一個群組成員對群組金鑰的貢獻是獨立的。這個協定也提供了金鑰完整性，除了群組成員以外，沒有其他個體對群組金鑰有所貢獻。假如EDH能夠提供群組金鑰的保密，未來的金鑰保密和過去的金鑰保密，那麼我們的協定對動態群組來說就是安全的。

群組金鑰保密意思是一個有被動的惡意者試圖計算出群組金鑰在計算上不可行。未來的群組金鑰保密保證有惡意者想由舊的秘密金鑰和群組金鑰推算出新的群組金鑰是計算上不可行的。過去的群組金鑰保密保證惡意者想由新的秘密金鑰和新的群組金鑰推出舊的群組金鑰是計算上不可行的。這些特性確保任何一個群組成員只能知道他在群組裡面的時候，他該有的秘密金鑰以及群組金鑰。

EDH的安全性是建立在兩個更基本的問題上：解離散對數問題(DL)的困難性和2DH問題。普遍上來說，離散對數問題是很難解的，但是如果p是一個小質樹的乘積，就存在有效率的演算法，像是Silver-Pohlig-Hellman演算法可以解出來。2DH的安全性建立在決定性的Diffie-Hellman假設上(DDH)，意思是說給定GF(p)中任意的a，b，c以及(φ(a), φ(b), φ(ab))和(φ(a), φ(b), φ(c))，沒有任何多項式時間的演算法能夠判斷哪一個有二分隻一以上的機率符合2DH的條件。

由於惡意者所能拿到的資訊都是以指數型態存在，所以要產生指數的難度如同解DL問題。因此任何惡意者想要算出任何群組成員的秘密金鑰都是很困難的。為了鑰證明本協定能夠提供群組金鑰保密，未來群組金鑰保密以及過去群組金鑰保密，我們使用了一個EDH的等效型式，它看起來像是之前提到的被寫成循環模式的金鑰計算演算法。

algorithm EDHS(S, X_n)
  if $n = 1$ **then return** $((\varphi(x_1)), x_1)$
  else if $n = 2$ **then return** $((\varphi(x_1), \varphi(x_2)), \varphi(x_1x_2))$
  else
    partition X_n *to* X_i, X_j *using* S
    $(V_{S,i}, k_{S,i}) \leftarrow$ EDHS(S, X_i)
    $(V_{S,j}, k_{S,j}) \leftarrow$ EDHS(S, X_j)
    return $((V_{S,i}, V_{S,j}), \varphi(k_{S,i}), \varphi(k_{S,j})), \varphi(k_{S,i}k_{S,j}))$

圖八

圖八秀出了EDH的模擬演算法EDHS。EDHS有兩個輸入：$X_n \equiv (x_1,x_2,\ldots,x_n)$ 是一個n元素有序對，其中每個元素都是一個群組成員的秘密金鑰。S是和協定有關的演算法，用來將n元素的有序對X_n切割成I元素的有序對X_i和j元素的有序對X_j。它控制了模擬演算法的輸出結果。為了要模擬EDH，S必須是決定性的而且和X的值互相獨立，因為所有我們的演算法都不是機率性的而且和金鑰本身的值或是算出的金鑰互相獨立。S可以被惡意者取得，因為我們的協定是公開的。注意X_i和X_j的順序並不重要，因為$\varphi(k_ik_j)=\varphi(k_jk_i)$。交換$X_i$和$X_j$只會影響$V_n$有序對的順序。演算法的輸出一共有兩部分：

$k_{S,n}(X_n)$ 模擬所有群組成員算出的金鑰值。

$V_{S,n}(X_n)$是有順序的2(n-1)個元素的有序對，用來模擬需要傳送去計算$k_{S,n}(X_n)$的所有值。

舉例來說，在4.1章節的例子裡，n=3，$X_3=(x_1,x_2,x_3)$。演算法S會把X_3分成兩個有序對,(x_1,x_2)和(x_3)。所以EDHS(S,X_3)的結果會是：

$$k_{S,3}(X_3) = \varphi(\varphi(x_1x_2)x_3).$$
$$V_{S,3}(X_3) = (\varphi(x_1), \varphi(x_2), \varphi(\varphi(x_1x_2)), \varphi(x_3)).$$

我們利用一個類似[4]裡面所使用到的方法來證明EDH確實提供了群組金鑰的保密，我們只需要證明，在本協定傳送的所有值都是已知的情況下，EDHS產生的金鑰和GF(p)的亂數能夠在多項式時間之內被分辨出來。對一個亂數y∈GF(p)，考慮

$$A_{S,n}(X_n) = (V_{S,n}(X_n), y).$$
$$F_{S,n}(X_n) = (V_{S,n}(X_n), k_{S,n}(X_n)).$$

當我們寫出$A_n \approx_{poly} F_n$的時候，代表A和F是多項式時間可分辨的，不管是什麼樣和協定有關的演算法S。

定理一 解 DDH 的計算上不可行性暗示了，當 n>2，$A_n \approx_{poly} F_n$

證明： 我們使用了另一種形式的歸納法來證明。在給定 n=2，屬於 GF(p)的 x_1，x_2 和協定相關的演算法 S 的情況下，我們可以推得 $A_{S,2}(X_2)=(V_{S,2}(X_2),y) = (\varphi(x_1), \varphi(x_2),y)$,以及 $F_{S,2}(X_2) = (V_{S,2}(X_2), K_{S,2}(X_2)) = (\varphi(x_1), \varphi(x_2), \varphi(x_1x_2))$。在演算法 EDHS 裡，當 n=2 的時候 S 是被忽略的，所以解 DDH 在計算上不可行的假設暗示了 A_2 和 F_2 在多項式時間下是不可分辨的。假設在 n=2,3,...,m-1,m>2 的情況下，$A_{m-1} \approx_{poly} F_{m-1}$ 是對的。那麼我們只需要推論出當 n=m,m>2 的時候 $A_m \approx_{poly} F_m$ 就能夠完成這個證明。

為了達到這個目的，我們把 A_m 和 F_m 重寫並且定義了一些有序對：

$$A_{S,m}(X_m) = (V_{S,i}(X_i), V_{S,j}(X_j), \varphi(k_{S,i}(X_i)), \varphi(k_{S,j}(X_j)), y).$$
$$B_{S,m}(X_m) = (V_{S,i}(X_i), V_{S,j}(X_j), \varphi(a), \varphi(k_{S,j}(X_j, S)), y).$$
$$C_{S,m}(X_m) = (V_{S,i}(X_i), V_{S,j}(X_j), \varphi(a), \varphi(b), y).$$
$$D_{S,m}(X_m) = (V_{S,i}(X_i), V_{S,j}(X_j), \varphi(a), \varphi(b), \varphi(a \cdot b)).$$
$$E_{S,m}(X_m) = (V_{S,i}(X_i), V_{S,j}(X_j), \varphi(a), \varphi(k_{S,j}(X_j)), \varphi(a \cdot k_{S,j}(X_j))).$$
$$F_{S,m}(X_m) = (V_{S,i}(X_i), V_{S,j}(X_j), \varphi(k_{S,i}(X_i)), \varphi(k_{S,j}(X_j)), \varphi(k_{S,i}(X_i) \cdot k_{S,j}(X_j))).$$

其中 $a \neq k_{S,I}(X_i)$ 並且 $b \neq k_{S,j}(X_j)$ 是從 GF(p) 中用亂數選出來的。由於 $A_m \approx_{poly} B_m$，$B_m \approx_{poly} C_m$，$C_m \approx_{poly} D_m$，$D_m \approx_{poly} E_m$，$E_m \approx_{poly} F_m$ 合起來暗示了 $A_m \approx_{poly} F_m$，只需要證明每一個小項目都是對的就能完成這個證明。

命題一：只要 m>2 則 $A_m \approx_{poly} B_m$。
假設對一些協定的演算法 S，存在一個多項式時間的演算法 M，能夠分辨 $A_{S,m}$ 和 $B_{S,m}$。對某些例子 $\chi=(V_{S,i}(X_i), V_{S,j}(X_j), \varphi(z), \varphi(k_{S,j}(X_j)), y)$，M 能夠分辨 χ 是否屬於 $A_{S,m}(X_m)$ 或是 $B_{S,m}(X_m)$。當 m>2，演匱法 EDHS 使用 S 來把 X_m 分成 X_i 和 X_j 兩個有序對，而且我們選定 i 和 j 讓 i≥j(記住 X_i 和 X_j 的順序並不重要)。現在我們要來決定 $\lambda=(V_{S,I}(X_i), z)$ 屬於 $A_{S,I}(X_i)$ 或 $F_{S,I}(X_i)$。我們選擇了一個亂數值 y'，和 j 個屬於 GF(p) 的亂數值來組成 X_j'。然後執行 $EDHS(S, X'_j)$ 來產生 $V'_{S,j}(X'_j)$，以及 $k'_{S,j}(X'_j)$。這個新的 $\chi'=(V_{S,i}(X_i), V'_{S,j}(X'_j), \varphi(z), \varphi(k'_{S,j}(X_j)), y')$ 被餵給 M。假如 M 告訴我們 χ' 屬於 $B_{S,m}$，則 λ 屬於 $A_{S,I}$(因為這個狀況下 z=a)。這顯示了 M 是一個多項式時間能夠分辨 $A_{S,I}$ 和 $F_{S,I}$ 的演算法，其中 $i \geq j \geq 1$ 並且 I+j=m>2。這和 $2 \leq i \leq m-1$ 時 $A_i \approx_{poly} F_i$ 矛盾，所以我們推出當 m>2 時 $A_m \approx_{poly} B_m$ 的結論。

命題二：對所有 m>2，$C_m \approx_{poly} D_m$。
由於 $X_{S,m}$ 和 $V_{S,m}$ 和 a，b，y 互相獨立，而這些值是多餘的而且可被忽略，所以這個問題可以轉化成 DDH，它暗示了 $C_m \approx_{poly} D_m$。

命題三：對所有 m>2，$B_m \approx_{poly} C_m$，$D_m \approx_{poly} E_m$，$E_m \approx_{poly} F_m$。
這可以用類似命題一的方法證明，只要選擇適合的 χ 和 λ。

定理一讓我們確認了我們演算法中使用的 EDH 提供群組金鑰保密的特性。對於攻擊力更強大的惡意者，像是離開的群組成員或是合法的成員，我們必須顯示出 EDH 提供未來以及過去的金鑰保密特性。我們使用 X_n 來代表惡意者知道的秘密金鑰的 n 元素有序對，而 $F_{S,I,n}(X_i, V_{S,n}(X_n))$ 是一個可以在多項式時間內從 X_i 和 $V_{S,n}(X_n)$ 計算出來的值的有序對。我們進一步定義了下列的有序對。

$$H_{S,I,n}(\bar{X}_i, X_n) = (\bar{X}_i, V_{S,n}(X_n), F_{S,I,n}(\bar{X}_i, V_{S,n}(X_n)), y).$$
$$K_{S,I,n}(\bar{X}_i, X_n) = (\bar{X}_i, V_{S,n}(X_n), F_{S,I,n}(\bar{X}_i, V_{S,n}(X_n)), k_{S,n}(X_n)).$$

其中 y 是從 GF(p)裡用亂數選出來的。

定理二對所有 $n \geq 2$ 和 $i>0$，$A_n \approx_{poly} F_n$ 暗示了 $H_{i,n} \approx_{poly} K_{i,n}$。

證明：假設對一些協定的演算法 S，存在一個多項式時間的演算法 M，能夠分辨 $H_{S,I,n}$ 和 $K_{S,I,n}$。對某些例子 $\lambda=(V_{S,n}(X_n), z)$，我們選擇了 i 個屬於 GF(p)的亂數值來組成有序對 X_j' 並算出 $F'_{S,I,n}(X'_i, V_{S,n}(X_n))$。這個 $\chi'=(X_i', V_{S,n}(X_n), F'_{S,i,n}(X'_I, V_{S,n}(X_n)), z)$ 被餵給 M。假如 M 告訴我們 χ' 屬於 $H_{S,i,n}$，則 λ 屬於 $A_{S,n}$(因為這個狀況下 z=y)。假如 M 告訴我們 χ' 屬於 $K_{S,i,n}$，則 λ 屬於 $F_{S,n}$(因為這個狀況下 $z=k_{S,n}(X_n)$)。使用這個程序，M 能夠對協定演算法 S 分辨 $A_{S,n}$ 和 $F_{S,n}$，這和定理一矛盾，所以對任何協定的演算法 S 來說，不存在這種 M，而且 $H_{i,n} \approx_{poly} K_{i,n}$。

根據定理二，只要知道 X_i，$V_{S,n}$ 和演算法 S，惡意者就能夠在多項式時間內算書更多的資訊，但是這些資訊對 k_n 來說是沒有任何用處的。因此我們的演算法中使用的 EDH 能夠提供未來以及過去的金鑰保密功能。定理二也告訴我們，在沒有相對的秘密金鑰的情況下想要破解群組金鑰是不可行的，同時暗示出本協定能夠抵抗

好幾個惡意者的合作攻擊。

複雜度的分析

計算的複雜度是以迴圈的動作數目來計算的，所有迴圈都是單層的回圈，而且迴圈的執行次數是由每個成員的路徑長度來決定的。空間的複雜度則是由每個成員紀錄群組資訊 M 所需的儲存空間來量測，這只和路徑資訊的長度有關，因為其他資訊都只需要固定的儲存空間。而通訊的複雜度則是以訊息的數目和所需的頻寬來量測。所有的要求訊息長度都是固定的，而更新訊息的長度和發送者的路徑長度成正比。表一總結了群組動作所需的花費，並且和 GDH.2[15]以及 TGDH[7]做了比較。

The group key updating operation			
protocol	GDH.2	TGDH	our scheme
computation	$O(N^2)$	$O(\log N)$	$O(L)$
unicasts	$N-1$	0	0
broadcasts	1	1	2
total bandwidth	$O(N^2)$	$O(N)$	$O(L)$

(a)

The member joining operation			
protocol	GDH.2	TGDH	our scheme
computation	$O(N)$	$O(\log N)$	$O(L)$
unicasts	1	0	0
broadcasts	1	2	2
total bandwidth	$O(N)$	$O(N)$	$O(L)$

(b)

The member leaving operation			
protocol	GDH.2	TGDH	our scheme
computation	$O(N)$	$O(\log N)$	$O(L)$
unicasts	0	0	0
broadcasts	1	1	2
total bandwidth	$O(N)$	$O(N)$	$O(L)$

(c)

The member storage requirement			
protocol	GDH.2	TGDH	our scheme
storage	$O(N)$	$O(N)$	$O(L)$

(d)

表一

我們可以粗略的估計每個成員加入花費的代價和[logN]成正比，而 N 是它加入時群組成員的總數。

結論

這篇論文中，我們提出了一個給動態群組使用的分散式金鑰管理協定，而且每個動作只需要兩個廣播訊息。這協定的演算法在機率上是安全的。產生群組金鑰的時間，傳送訊息的頻寬以及每個成員需要的儲存空間，和群組成員的數目來比，都是非常小的。我們計畫改進我們的協定，使它能夠支援子群組。假定每個成員都只影響到同一個子群組的成員，可以使規模擴展的能力和效率都能夠有所改善。我們也在設計非廣播網路環境下的協定，並且發展容錯的演算法。

參考資料

[1] G. Ateniese, M. Steiner, and G. Tsudik. Authenticated group key agreement and friends. In *5th ACM CCS*, pages 17–26, Nov. 1998.

[2] G. Ateniese, M. Steiner, and G. Tsudik. Key agreement in dynamic peer groups. *IEEE Transactions on PDS*, 11(8), Aug. 2000.

[3] G. Ateniese, M. Steiner, and G. Tsudik. New multiparty authentication services and key agreement protocols. *IEEE JSAC*, Apr. 2000.

[4] K. Becker and U. Wille. Communication complexity of group key distribution. In *Proceedings of the 5th ACM CCS*, Nov. 1998.

[5] D. Boneh. The decision Diffie-Hellman problem. In *Third Algorithmic Number Theory Symposium*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer-Verlag, 1998.

[6] M. Burmester and Y. Desmedt. A secure and efficient con-ference key distribution system. In *Eurocrypt '94*, volume 950 of *Lecture Notes in Computer Science*, pages 275–286. Springer-Verlag, 1995.

[7] Y. Kim, A. Perrig, and G. Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative group. In *Proceed-ings of ACM CCS (CCS-7)*, Nov. 2000.

[8] N. Koblitz. *A Course in Number Theory and Cryptography, 2nd edition*. Springer-Verlag, 1994.

[9] U. M. Maurer and S. Wolf. The Diffie-Hellman protocol. *Designs, Codes and Cryptography*, 19(2/3):147–171, 2000.

[10] S. Mittra. Iolus: A framework for scalable secure multicast-ing. In *ACM SIGCOMM*, pages 277–288, Sept. 1997.

[11] L. E. Moser, Y. Amir, P. M. Melliar-Smith, and D. A. Agar-wal. Extended virtual synchrony. In *International Conference on Distributed Computing Systems*, pages 56–65, 1994.

[12] B. Schneier. *Applied Cryptography, 2nd edition*. John Wiley & Sons, Inc., 1996.

[13] W. Stallings. *Cryptography and Network Security: Principles and Practice, 2nd edition*. Prentice-Hall Inc., 1999.

[14] D. Steer, L. Strawczynski, W. Diffie, and M. Wiener. A secure audio teleconference system. In *Crypto '88*, volume 403 of *Lecture Notes in Computer Science*, pages 520–528. Springer-Verlag, 1988.

[15] M. Steiner, G. Taudik, and M. Waidner. CLIQUES: A new approach to group key agreement. In *IEEE ICDCS*, pages 380–387, 1998.

[16] C. Wong, M. Gouda, and S. Lam. Secure group communications using key graphs. *Proceedings of the ACM SIG-COMM'98*, pages 68–79, Sept. 1998.

[17] P. R. Zimmermann. *The Official PGP User's Guide*. MIT Press, 1995.