東　海　大　學
應　用　數　學　研　究　所
碩　士　論　文

ElGamal數位簽署的推廣與改良

指導教授：沈淵源

研究生：張草薰

中華民國八十九年六月

ElGamal數位簽署的推廣與改良

研 究 生 ：張草薰　　Student：Tsau-Shiun Jang

指導教授：沈淵源　　Advisor：Yuan-Yuan Shen

東　海　大　學

應　用　數　學　研　究　所

碩　士　論　文

A Thesis

Submitted to the Institute of Applied Mathematics

College of Science

Tunghai University

in

Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Applied Mathematics

June 2000

Taichung, Taiwan, Republic of China.

中華民國八十九年六月

誌　謝

本篇論文得以順利地完成，首先必須感謝我的指導教授沈淵源老師兩年來辛勤的教誨與指導。使我在疑惑的同時，能即時的找到方向，有如事半功倍、漸入佳境的效果。

此外，還要感謝這兩年來朝夕相處的研究所同學們，謝謝你們的鼓勵及關懷，讓我一路走來雖有艱辛，但所結的果實是甜美的。最後，我必須感謝我的父母，辛苦的栽培我長大，為的就是讓我能無憂的唸書和成功的做人。因此，今後希望能在數學的領域裏能貢獻更多的心力以回饋大家。

張 草 薰　　謹識於

東海大學應用數學研究所

民國八十九年六月

摘　要

本論文是由ElGamal的數位簽署所發展出來的,並仿傚RSA系統,能同時的達到秘密通訊和數位簽署的功能;更把ElGamal的數位簽署推廣,使其簽署文有不同的驗證方式,並討論其安全性；另一方面也提出了數位簽署(DSA)的改良。如此一來,數位簽署不再是那麼的單調,可依需要作不同的變化,更能提供給大家一個新的思考方向。

目　　錄

第一章

前言

密碼學 (Cryptology) 一字原自希臘文 “kryptós” 及 “lógos” 兩字, 直譯即為 “隱藏” 及 “科學” 的意思。1949 年, Shannon 提出第一篇討論密碼通訊理論之論文, 近代密碼學可說是濫觴於斯。直至西元 1975 年, Diffie 與 Hellman 提出公開密碼系統之觀念, 近代密碼學之研究方向, 正式脫離傳統秘密金匙密碼系統之巢臼, 蓬勃發展, 至今已二十多載。在此二十多年間, 舉凡各種公開金匙密碼系統、數位簽署技術、金匙交換管理、秘密分享等等各種協定與應用紛紛傾囊而出, 令人有目不暇給之感。

本文先介紹一般的密碼系統及目前最受歡迎的 RSA 密碼系統。接著介紹 ElGamal 密碼系統。在此我們也把

ElGamal數位簽署加以推廣，使其有不同的面貌。最後則介紹單向赫序函數和數位簽署演算法(Digital Signature Algorithm)，並提出一個新的改良方法。

第二章

密碼系統簡介

2.1 密碼系統

密碼術(Cryptography)乃是一門以偽裝的形式來傳送信息之方法的學問,為的是只讓那些指定的收信者能將此偽裝除去並閱讀該信息,以達成秘密通訊的目的。我們將欲傳送的信息稱之為明文(Plaintext),而偽裝之信息稱之為密文(Ciphertext)。兩者皆以某種含N個數目的符號(通常是同一種,但不一定要一樣)來表達。這裡所謂的符號,不僅包含那些我們所熟悉的英文字母,還涵蓋數字、空白、標點符號及其它我們所允許使用的符號。不管明文或密文,其組成的最小單位我們稱之為信息單元(Message unit)。將

明文的信息單元轉換為密文的信息單元我們稱之為加密(Enciphering),而其逆過程則稱之為解密(Deciphering)。我們可將整個架構表示如下圖：

$$\begin{aligned} \mathcal{P} &= \{\text{所有可能的明文信息單元}\} \\ \mathcal{C} &= \{\text{所有可能的密文信息單元}\} \\ f &= \text{加密函數,此函數為一對一函數} \end{aligned}$$

$$\mathcal{P} \xrightarrow{f} \mathcal{C} \xrightarrow{f^{-1}} \mathcal{P}$$

這樣的系統我們稱為一個密碼系統(Cryptosystem)。

2.2 公開金匙密碼系統

甲欲傳送信息給乙,他們沒有事前的接觸,也不希望花時間交專差遞送秘密金匙。因此甲送給乙的信息有可能被第三者丙給攔截,是否有方法能讓甲在公開的管道上傳送信息給乙,而此信息只有乙能閱讀但丙卻不行呢？

在傳統的密碼系統中,想要達到上述的事是不可能的。因為甲必需將秘密金匙遞送給乙(在公開的管道),因此丙可將之攔截,那丙就可以把密文還原成明文並閱讀之。為了滿足上述的要求, Diffie和Hellman[1]在1976年首先提出公

開金匙密碼系統(Public Key Cryptosystem)的觀念,然而他們並沒有一套實際可行的密碼系統。之後幾年,有好幾種可行的公開金匙密碼系統被提出,其中最有名的就是RSA公開金匙密碼系統。

第三章

RSA 公開金匙密碼系統

RSA 公開金匙密碼系統在 1978 年由美國麻省理工學院的三位教授 Rivest、Shamir 及 Adleman(RSA) [2] 所研發出來的一套密碼系統。

3.1 金匙產生

(1) 每位使用者 A，任意選擇大的質數 p_A 及 q_A，並求出其乘積 $N_A = p_A q_A$ 與 $T_A = \phi(N_A) = (p_A - 1)(q_A - 1)$，此處大質數 p_A 及 q_A 需適當的選擇，以保證分解因數在計算上為不可能。

(2) A 任意選擇一整數 e_A，使得 e_A 與 T_A 互質，並求出 e_A

在模T_A中之乘法反元素d_A，即$e_A d_A = 1 \bmod T_A$。

(3) A將(e_A, N_A)公佈為其公開金匙，並將d_A秘密保存為其私有秘密金匙，而p_A與q_A可以毀去不用，以增加安全性。

3.2 秘密通訊

若A欲秘密傳送明文$m(0 < m < N_B)$給B，其步驟如下:

步驟一　A首先找出B之公開金匙(e_B, N_B), 然後執行加密得密文C.

$$\text{加密：} \qquad C = E_B(m) = m^{e_B} \pmod{N_B}$$

步驟二　A將密文C傳送給B，在B收到密文後，利用其私有秘密金匙d_B，執行解密的動作。

$$\text{解密：} \quad D_B(C) = C^{d_B} = (m^{e_B})^{d_B} = m^{KT+1} = m \pmod{N_B}$$

不論m是否與N_B互質，在RSA系統中執行加密與解密後均可還原m。事實上若m不與N_B互質，則m與N_B之最大公因數不是p_B就是q_B。因此，我們求m與N_B之最大公因數，就得出p_B或q_B進而可分解N_B，則此系統就不安全。但若我們令p_B與q_B均為很大之質數，則在0與

N_B之間任選一數而與N_B不互質的機率為$(p_B+q_B)/N_B = (p_B+q_B)/p_Bq_B$。當$p_B$與$q_B$很相近時，其機率約$2/\sqrt{N_B}$。若$N_B$為512位元，其機率約$2^{-255}$，因此，一般我們均假設$m$與$N_B$互質。

3.3 數位簽署

若A欲將明文m簽署，其進行步驟如下：

步驟一　A利用其密匙d_A，對m加以簽署得簽署文S。

$$\text{簽署：} \qquad S = D_A(m) = m^{d_A} (\text{mod } N_A)$$

步驟二　A將m與簽署文S傳送給B，等B收到m與S後，利用A之公開金匙(e_A, N_A)執行驗證。

$$\text{驗證：} \qquad E_A(S) = S^{e_A} = m' \ (\text{mod } N_A)$$

若$m' = m$則驗證正確，由此可知明文m和簽署文S是由A傳送過來的；若驗證失敗，則可知不是A傳送的。由於任何人均可利用A之公開金匙進行驗證，但只有A知道用d_A來產生S。因此，這是相當保密，無法假造的。

3.4 同時達到秘密通訊與數位簽署

若A欲秘密傳送明文m，並同時對m作簽署，然後傳送給B，其進行步驟如下：

步驟一　A首先對m以d_A簽署，得簽署文S，然後對m與S以B之公開金匙進行加密，得密文C_m與C_S。即

$$\text{先簽署：} \quad S = D_A(m) = m^{d_A} \pmod{N_A} \tag{3.1}$$

$$\text{後加密：} \quad C_m = E_B(m) = m^{e_B} \pmod{N_B}$$

$$C_S = E_B(S) = S^{e_B} \pmod{N_B} \tag{3.2}$$

步驟二　B收到密文C_m與C_S後，先以秘密金匙d_B，對C_m與C_S解密得m與S。接著再利用A之公開金匙，進行驗證。即

$$\text{先解密：} \quad D_B(C_m) = C_m^{d_B} = m \pmod{N_B}$$

$$D_B(C_S) = C_S^{d_B} = S' \pmod{N_B} \tag{3.3}$$

$$\text{再驗證：} \quad E_A(S') = S'^{e_A} = m' \pmod{N_A} \tag{3.4}$$

若$m' = m$則驗證正確，由此可知文件m和簽署文S是由A傳送過來的；若驗證失敗，則可知不是A傳送的。上

述過程中，由於(3.1)式及(3.2)式需要連續使用不同的模N_A和N_B，因此會有一點問題存在：即當$N_A > N_B$時，S有可能大於N_B。但在(3.3)式求得的S'卻小於N_B。因此在(3.4)式所求得的m'並不等於原來的明文m。此問題稱為“Reblocking Problem”，是在連續使用不同的模(如N_A,N_B)時所必須碰到的問題。其有兩種方法解決，如下：

方法一：假設有一臨界值h存在，如$h = 10^{109}$。令每位使用者A均有兩組公開金匙：(e_{A_1}, N_{A_1})用於做數位簽署系統，(e_{A_2}, N_{A_2})用於做密碼系統，且$N_{A_1} < h < N_{A_2}$。同樣的，B也有兩組公開金匙：(e_{B_1}, N_{B_1})及(e_{B_2}, N_{B_2})，且$N_{B_1} < h < N_{B_2}$。當A欲密傳m及其簽署文S給B時，A將

$$C = E_{B_2}(D_{A_1}(m)) \tag{3.5}$$

傳送給B，然後B再計算

$$E_{A_1}(D_{B_2}(C)) = E_{A_1}(D_{B_2}(E_{B_2}(D_{A_1}(m)))) = E_{A_1}(D_{A_1}(m)) = m \tag{3.6}$$

上述(3.5)式和(3.6)式均可正確執行，因為滿足$N_{A1} < h < N_{B_2}$。此方法的缺點是每位使用者需要有兩組公開金匙，使得公開金匙的儲存空間加倍。

方法二[3]：每位使用者只有一組公開金匙，當$N_A > N_B$時，則A先簽署再加密時才會產生Reblocking Problem，若$N_A < N_B$則不會。Konfelder觀察到當$N_A > N_B$時，先加密後簽署就不會產生Reblocking Problem。因此，當A欲秘密傳送m及其簽署文S給B時，A可先比較N_A與N_B的大小再進行下列之一：

1. 若$N_A < N_B$，$C = E_B(D_A(m))$ (先簽署後加密)
2. 若$N_A > N_B$，$C = D_A(E_B(m))$ (先加密後簽署)

而後將C傳送給B，等B收到後再執行下列之一：

1. 若$N_A < N_B$，$E_A(D_B(C)) = E_A(D_B(E_B(D_A(m)))) = m$
2. 若$N_A > N_B$，$D_B(E_A(C)) = D_B(E_A(D_A(E_B(m)))) = m$

B即可將m正確的還原。

第四章

ElGamal公開金匙密碼系統

在1985年，ElGamal[4]提出一種基於離散對數的公開金匙密碼系統及簽署設計。

4.1 金匙產生

(1) 系統的參數：設系統中存在一大質數p，及模p之原根(primitive root) g，使得解離散對數為不可能。

(2) 個人密匙：使用者A任選一整數x_A，$1 < x_A < p-1$，為其密匙。

(3) 個人公開金匙：使用者A求出$y_A = g^{x_A} \pmod{p}$，y_A為其公開金匙。

4.2 秘密通訊

若A欲秘密傳送明文$m(1 \le m \le p-1)$給B，其步驟如下：

步驟一　A首先找出B之公開金匙y_B並任選一整數k，使得$(k, p-1) = 1$，然後執行加密得密文$C = (C_1, C_2)$。

$$\text{加密：} \qquad C_1 = g^k \pmod{p}, \quad C_2 = y_B^k m \pmod{p}$$

步驟二　A將密文C傳送給B，等B收到密文後，利用其私有密匙x_B，執行解密的動作。

$$\text{解密：} \qquad \frac{C_2}{C_1^{x_B}} = \frac{y_B^k m}{(g^k)^{x_B}} = \frac{(g^{x_B})^k m}{g^{k x_B}} = m \pmod{p}$$

4.3 數位簽署

若A欲將明文$m(1 \le m \le p-1)$簽署，其進行步驟如下：

步驟一　A先任選一整數k，使得$(k, p-1) = 1$，然後利用其個人密匙x_A對m加以簽署得出簽署文$S = (r, s)$

$$\text{簽署：} \qquad r = g^k \pmod{p}, \quad m = x_A r + k s \pmod{p-1}$$

步驟二　A將m與簽署文S傳送給B，等B收到m與S後，

利用A之公開金匙y_A執行驗證。

$$\text{驗証：} \qquad g^m = y_A^r r^s \pmod{p}$$

若以上正確，則(r, s)為m之合法簽署文，否則為非法簽署文；因為$y_A^r r^s = g^{x_A r + ks} \pmod{p} = g^m$。

4.3.1 安全性分析及討論

1. 本簽署系統的安全性係基於解離散對數之困難上。若能解離散對數，則由y_A及g，可求出A之密匙x_A，本系統就不安全。

2. 若第三者欲偽造一合法簽署文，其任選r（或s），欲求出s（或r）滿足$g^m = y_A^r r^s \pmod{p}$，則面臨解離散對數問題。

3. 第三者已獲得一明文m及簽署文(r, s)，欲由$m = x_A r + ks \pmod{p-1}$求出$x_A$。則因式中有兩個未知數$x_A$及$k$，所以無法求得$x_A$。但若A利用相同的$k$簽署兩次，則會得到$m_1$、$m_2$，其簽署文分別為$(r, s_1)$及$(r, s_2)$，則第三者可利用$m = x_A r + ks \pmod{p-1}$來解聯立方

程式

$$m_1 \equiv x_A r + k s_1 \pmod{p-1}$$

$$m_2 \equiv x_A r + k s_2 \pmod{p-1}$$

因有兩方程式和兩變數(x_A 和 k)，則 x_A 可被求出。所以，為避免此情形發生，k 不可重覆使用。

4. 第三者可偽造 m 之合法簽署文 (r, s) ，但 m 無法事先固定。此偽造方法如下：第三者任選亂數 u 及 w ，滿足 $1 < u, w < p-1$ 且 $\gcd(w, p-1) = 1$ 。再計算

$$r \equiv g^u y_A^{-w} \pmod{p}$$

$$s \equiv r w^{-1} \pmod{p-1}$$

$$m \equiv us \pmod{p-1}$$

由此三式可得

$$y_A^r r^s = y_A^r (g^u y_A^{-w})^s = y_A^r g^{us} y_A^{-ws} = y_A^r g^{us} y_A^{-r} = g^{us} = g^m \pmod{p}$$

因此 (r, s) 為 m 之合法簽署文。此偽造過程中，由於對 m 並無控制能力，故 ElGamal 簽署仍為安全。

4.4 同時達到秘密通訊和數位簽署

4.4.1 系統參數相同時

使用者A與B其系統參數(p, g)相同，個人密匙分別為x_A及x_B；個人公開金匙分別為$y_A = g^{x_A} \pmod p$及$y_B = g^{x_B} \pmod p$。

一、先加密後簽署若A欲秘密傳送明文$m(1 \le m \le p-1)$，並同時對m作簽署，然後傳送給B，其進行步驟如下：

步驟一　A首先先對明文m加密，得到密文$C = (C_1, C_2)$。然後再對密文C加以簽署，得到簽署文$S = (S_1, S_2)$。即A先任選一整數k使得$(k, p-1) = 1$。

$$\begin{aligned} \text{先加密：} \quad C_1 &= g^k \pmod p \\ C_2 &= y_B^k m \pmod p \\ \text{後簽署：} \quad S_1 &= C_1 \pmod p \\ m &= x_A S_1 + k S_2 \pmod{p-1} \end{aligned}$$

步驟二　B收到$\{C, S\}$後，先進行解密的動作，以求得明

文m；然後再執行驗証，確定是否為A傳送過來的。

$$\text{先解密：} \qquad m = C_2/C_1^{x_B} \pmod{p}$$

$$\text{再驗証：} \qquad g^m = y_A^{S_1} S_1^{S_2} \pmod{p}$$

二、先簽署後加密

若A想先簽署明文$m(1 \le m \le p-1)$，然後再加密傳送給B，其進行步驟如下：

步驟一　A首先先對明文m簽署，得到簽署文$S = (S_1, S_2)$。然後再對簽署文S加以加密，得到密文$C = (C_1, C_2)$。即A先任選一整數k使得$(k, p-1) = 1$。

$$\begin{aligned} \text{先簽署：} \qquad S_1 &= g^k \pmod{p} \\ m &= x_A S_1 + k S_2 \pmod{p-1} \\ \text{後加密：} \qquad C_1 &= S_1 \pmod{p} \\ C_2 &= y_B^k m \pmod{p} \end{aligned}$$

步驟二　B收到$\{S, C\}$後，先執行解密，求得明文m。然

後再進行驗証的動作，以確定是否為A傳送過來的。

$$\text{先解密：} \qquad m \ = \ C_2/C_1^{x_B} \pmod{p}$$

$$\text{再驗証：} \qquad g^m \ = \ y_A^{S_1} S_1^{S_2} \pmod{p}$$

4.4.2 系統參數不同時

使用者A與B其系統參數分別為(p_A, g_A)及(p_B, g_B)，個人密匙分別為x_A及x_B；個人公開金匙分別為$y_A = g^{x_A} \pmod{p_A}$及$y_B = g^{x_B} \pmod{p_A}$

一、先加密後簽署

若A欲秘密傳送明文$m(1 \le m \le p_B - 1)$，並同時對m作簽署，然後傳送給B，其進行步驟如下：

步驟一　首先A先對明文m加密，得到密文$C = (C_1, C_2)$。然後再對密文C加以簽署，得到簽署文$S = (S_1, S_2)$。即A先任選一整數k使得$(k, p_A - 1) = 1$與$(k, p_B - 1) = 1$成立。

$$\text{先加密：} \quad C_1 = g_B^k, \ C_2 = y_B^k m \pmod{p_B} \tag{4.1}$$

$$\text{後簽署：} \qquad S_1 \ = \ g_A^k \pmod{p_A}$$

$$m \ = \ x_A S_1 + k S_2 \pmod{p_A - 1} \tag{4.2}$$

步驟二　B收到$\{C, S\}$後，先進行解密的動作，以求得明文m。然後再執行驗証，確定是否為A傳送過來的。

$$\text{先解密：} \qquad m = C_2 / C_1^{x_B} \pmod{p_B} \tag{4.3}$$

$$\text{再驗証：} \qquad y_A^{S_1} S_1^{S_2} = g_A^m \pmod{p_A} \tag{4.4}$$

Reblocking **問題的討論**

若先執行加密再簽署時，因為A和B之系統參數不同，分別為p_A和p_B，所以有可能造成Reblocking問題。

(1) 當$p_B < p_A$時，在解密時因由(4.1)式解密出來(4.3)式的m和原來明文(4.1)式中的m相同，所以不會造成Reblocking。且在驗證時因$p_B < p_A$，所以在(4.1)式和(4.2)式中的m相同，因由(4.1)式解密出來(4.3)式的原來明文m和(4.2)式中的m相同，所以在驗証時(4.4)式中的等號相等(因指數$x_A S_1 + k S_2 = m \pmod{p_A - 1}$)，因此不會造成Reblocking。

(2) 當$p_B > p_A$時，在解密時因由(4.1)式所解出來(4.3)式的m和原來明文(4.1)式中的m相同，所以不會造成Reblocking。但在驗證時因$p_B > p_A$，若$m > p_A$則在(4.2)式中的m和(4.1)式中的m不相同，相差了

$p_A - 1$ 的整數倍，由費馬定理知若 $\gcd(g_A, p_A) = 1$，則 $g_A^{p_A-1} \pmod{p_A} = 1$，因此，在 (4.4) 式中左式和右式的指數只差了 $p_A - 1$ 的整數倍，所以，由費馬定理知其 (4.4) 式的等號相等，不會有 Reblocking 的問題。

由上可知雖然 p_A, p_B 不同,但不會影響解密和驗証的過程。

二、先簽署後加密

若 A 欲秘密傳送明文 $m(1 \le m \le p_A - 1)$，並同時對 m 作簽署，然後傳送給 B，其進行步驟如下：

步驟一　首先 A 對明文 m 簽署，得到簽署文 $S = (S_1, S_2)$，而後再對簽署文 S 加密，得到密文 $C = (C_1, C_2)$。即 A 先任選一整數 k，使得 $(k, p_A - 1) = 1$ 與 $(k, p_B - 1) = 1$ 成立。

先簽署:
$$S_1 = g_A^k \pmod{p_A}$$
$$m = x_A S_1 + k S_2 \pmod{p_A - 1} \quad (4.5)$$

後加密:
$$C_1 = g_B^k \pmod{p_B} \quad (4.6)$$
$$C_2 = y_B^k m \pmod{p_B} \quad (4.7)$$

步驟二　B 收到 $\{S, C\}$ 後，先執行解密，求得明文 m。然

後再進行驗証的動作，以確定是否為A傳送過來的。

$$\text{先解密:} \qquad m \;=\; C_2/C_1^{x_B} \pmod{p_B} \tag{4.8}$$

$$\text{再驗証:} \qquad g_A^m \;=\; y_A^{S_1} S_1^{S_2} \pmod{p_A} \tag{4.9}$$

Reblocking **問題的討論**

若先執行簽署再加密時，因為A和B之系統參數不同，分別為p_A和p_B，所以有可能造成Reblocking問題。

(1) 當$p_B > p_A$時，因$p_B > p_A$，所以在(4.5)式和(4.7)式中的m相同，因此由(4.6,4.7)式解密出來(4.8)式的m和原來明文(4.5)式中的m相同，又因解密出來(4.8)式的m和原來的明文(4.5)式的m相同，所以在驗証時(4.9)式中的等號相等(因指數$x_A S_1 + kS_2 = m \pmod{p_A - 1}$)，因此不會造成Reblocking。

(2) 當$p_B < p_A$時，因$p_B < p_A$，若$m > p_B$則在(4.7)式中的m和(4.5)式中的m不相同，因此由(4.6,4.7)式解密出來(4.8)式的m和原來明文(4.5)式中的m不相同，如此解密出來的明文m不是原來(4.5)式的m，又因解密出來(4.8)式的m和原來的明文(4.5)式的m不相同，所以在驗証時(4.9)式中的等號不相等(因指數

$x_A S_1 + kS_2 \neq m \pmod{p_A - 1}$)，驗証無法成立，會產生Reblocking的問題。

由上可知此法適用於$p_B > p_A$時，才不會產生Reblocking。

4.5 ElGamal數位簽署的推廣

4.5.1 介紹

設系統存在一大質數p，及模p之原根g。簽署者A任選一整數x_A，$1 < x_A < p-1$為其密匙，並求出$y_A = g^{x_A} \pmod{p}$為其公開金匙。若A欲將明文m $(1 \leq m \leq p-1)$簽署，其進行步驟如下：

<u>步驟一</u>　A先任選整數k_1, k_2，使得$(k_1 k_2, p-1) = 1$，然後利用其個人密匙x_A對m加以簽署得簽署文$S = (r, s, t)$。

簽署：$r = g^{k_1},\ s = g^{k_2} \pmod{p};\ m = (r+s)x + k_1 s + k_2 t \pmod{p-1}$

<u>步驟二</u>　A將m與簽署文S傳送給B，等B收到m與S之後，再利用A之公開金匙y_A執行驗證。

$$\text{驗証：} \qquad g^m = y_A^{r+s} r^s s^t \pmod{p}$$

因為$y^{r+s} r^s s^t = g^{x(r+s)} g^{k_1 s} g^{k_2 t} = g^{(r+s)x + k_1 s + k_2 t} = g^m \pmod{p}$。

4.5.2 安全性分析及討論

此法優於一般的ElGamal數位簽署，原因如下:

(1) 在原始的ElGamal數位簽署中，其簽署文為(r, s)，明文m，若A利用相同的k簽署兩次，即m_1, m_2之簽署文為(r, s_1)及(r, s_2)，其中的$r = g^k \pmod{p}$。

$$m_1 = x_A r + k s_1 \pmod{p-1}$$

$$m_2 = x_A r + k s_2 \pmod{p-1}$$

由上二變數(x_A, k)和二個方程式，則x_A可被求出，因此k不可重覆使用，否則不安全。

(2) 在目前的方法中，若A利用相同的k_1, k_2簽署須要三次，即m_1, m_2, m_3之簽署文$(r, s, t_1), (r, s, t_2), (r, s, t_3)$，其中的$r = g^{k_1} \pmod{p}, s = g^{k_2} \pmod{p}$。

$$m_1 = (r+s)x_A + k_1 s + k_2 t_1 \pmod{p-1}$$

$$m_2 = (r+s)x_A + k_1 s + k_2 t_2 \pmod{p-1}$$

$$m_3 = (r+s)x_A + k_1 s + k_2 t_3 \pmod{p-1}$$

要求出x_A較前困難，且簽署者必需要簽署相同的k_1, k_2三次才可求出x_A。因此，此方法較安全。

第五章

數位簽署演算法(DSA)

5.1 單向赫序函數

對任意長度的明文m，經由赫序函數h可產生固定長度的赫序值，用$h(m)$來表示。赫序函數值在對明文鑑定(Authentication)或是數位簽名(Digital Signature)上都是非常必要的工具。赫序函數值可以說是對明文的一種“指紋”(Fingerprint)或是“摘要”(Digest)，所以，對赫序值的數位簽名，就可以視為對此明文的數位簽名。因此，使用赫序函數可以提高數位簽名的效率。使用在數位簽名上的赫序函數必需滿足下面條件[5]:

1. 赫序函數必需對任意長度的明文，產生固定長度的赫

序函數值。

2. 對任意的明文m，赫序函數值$h(m)$可借由軟體或硬體很容易得到。

3. 對任意的赫序函數值x，要找到一個明文m與之對應即$x = h(m)$，在計算上是不可行的。

4. 對一個明文m_1，要找到另一個不同的明文m_2，而且具有相同的赫序函數值$h(m_1) = h(m_2)$，在計算上是不可行的。

5. 要找到任一對不同的明文(m_1, m_2)，而且具有相同的赫序值$h(m_1) = h(m_2)$，計算上是不可行的。

條件1和2是所謂的“單向”(One-Way)特性，條件3和4是對使用赫序值的數位簽名方法所做的安全保障，否則攻擊者可以由已知的明文及相關的數位簽名，來任意偽造對其他明文的數位簽名。通常滿足條件$1 \sim 4$，我們稱為“弱赫序函數”(Weak Hash Function)。若能滿足條件5的，我們稱為“強赫序函數”(Strong Hash Function)。由此可見應用在數位簽名上的，必需是強赫序函數。

簡單的赫序函數:

Rabin[6]在1978年利用DES，使用密文塊串連(Cipher Block Chaining ,簡寫為CBC)的方法，提出一種簡單且快速的赫序函數，方法如下:

將明文m分成固定長度64bit的明文塊(Block)，$m_1, m_2, ..., m_N$，使用DES的CBC操作方法，對每一明文塊陸續加密:

$$令 h_0 = 初始值, h_i = E_{m_i}[h_{i-1}] 及 G = h_N 。$$

唯一不同的是這種加密沒有使用任何密匙，而G就是64bit的赫序函數值。還有二種常見的赫序函數:一個是MD5[7]，一個是安全赫序函數[8] (SHA , Secure Hash Function)。在設計上的觀念都十分類似上述方法。同樣的，都先將明文分成固定長度的明文塊，再對每一塊的明文做相同的處理。

5.2 數位簽署演算法(Digital Signature Algorithm)

1991年8月，美國國家標準局(National Institute of Standard and Technology NIST)公佈了數位簽署標準(Digital Signature Standard DSS)[9]，此標準採用的演算法稱為DSA(Digital Signature Algorithm)，為ElGamal系統之變型，

並且採用了Schnorr系統中g為非原根的做法，以降低其簽署文的長度。

DSA **之參數如下**:

p :512位元之質數。

q :160位元之質數，且$q \mid p-1$。

g : 滿足$g = h^{p-1/q}$ 。

h : 單向赫序函數。

x : $0 < x < q$ 為秘密金匙。

y : $y = g^x \pmod p$ 為公開金匙。

p、q、g及h，為系統公佈之共同參數與公開金匙y均要公開，x為簽署者之密匙。

若A欲將明文m $(1 \le m \le p-1)$簽署，其進行步驟如下：

步驟一　A先任選一整數$k(0 < k < q)$，使得$(k, q) = 1$。然後求出簽署文(r, s)。

簽署: $r = g^k \pmod p \pmod q,\ s = k^{-1}(h(m) + xr) \pmod q$

步驟二　A將m與簽署文(r, s)傳送給B，等B收到m與

(r, s) 後，再利用 A 之公開金匙 y 執行驗證。

驗証：

(1) 先檢查 r, s 是否都在 $[0, q]$，若否則 (r, s) 不是簽署文。

(2) 計算 $t = s^{-1} \pmod{q}$ 及 $r' = g^{h(m)t} y^{rt} \pmod{p} \pmod{q}$

若 $r' = r$，則 (r, s) 為 m 之合法簽署文。換言之若 (r, s) 為合法簽署文則

$$\begin{aligned} r' &= g^{h(m)t} y^{rt} \pmod{p} \pmod{q}, \\ &= (g^{h(m)} y^{r})^{t} \pmod{p} \pmod{q}, \\ &= \left(g^{h(m)+xr}\right)^{k(h(m)+xr)^{-1}} \pmod{p} \pmod{q}, \\ &= g^{k} \pmod{p} \pmod{q} = r. \end{aligned}$$

在 DSA 中，簽署者與驗証者均需各求一次模 q 之乘法反元素，由於求反元素幾乎相當於指數運算相當耗時，Yen 及 Laih[10] 提出兩種改良方法可以消除簽署者 (或驗証者) 之求反元素運算，方法在下兩節。

5.2.1 DSA 改良方法 1

步驟一

簽署: $r = g^k \pmod p \pmod q,\ s = (rk - h(m))x^{-1} \pmod q$

步驟二

驗証: $t = r^{-1} \pmod q,\ r' = g^{h(m)t}y^{st} \pmod p \pmod q$

因為 $r' = g^{h(m)t}g^{xst} = (g^{h(m)+xs})^t = (g^{h(m)+xs})^{k(xs+h(m))^{-1}} = g^k \pmod p \pmod q = r$。在此方法中 x^{-1} 為密匙 x 之乘法反元素，由於密匙 x 為固定，故 x^{-1} 亦為固定，x^{-1} 可以事先計算並存在記憶體，在每次簽署時無需計算，故可以減少一次乘法反元素之計算。非常適合於簽署者為計算能力較小者(如 IC 卡)之應用。

5.2.2 DSA 改良方法 2

步驟一

簽署: $r = g^k \pmod p \pmod q,\ s = k(h(m) + xr)^{-1} \pmod q$

步驟二

驗証: $t = sh(m) \pmod q,\ r' = g^t y^{sr} \pmod p \pmod q$

因為$r' = g^t g^{xsr} = g^{sh(m)} g^{xsr} = \left(g^{h(m)+xr}\right)^s = \left(g^{h(m)+xr}\right)^{k(h(m)+xr)^{-1}}$ $= g^k \pmod p \pmod q = r$。在此法中，驗証者無需求反元素，因此，適合於驗証者計算能力較小者之應用。因此Yen[11]建議在IC卡運用上可以混合使用兩種DSA的改良方法，即當IC卡與主電腦連線時，若IC卡為簽署者，則使用改良方法1，若IC卡為驗証者時，則採用改良方法2。且Yen及Laih証明他們改良方法之安全性與DSA完全一樣，均是基於解離散對數問題。

5.3 DSA之正面與負面評價[12]

DSA之正面評價

1. 標準的公布已顯示美國政府終於確認公開金匙密碼系統的使用。
2. 此標準所產生的簽署文長度較短(僅有320位元)。
3. 在簽署程序中對於r的計算可在事先已做好，可減少產生簽署文的時間。
4. DSS標準在金融服務業特別有用。
5. 假若DSS被實現的話，只需要最小的成本。

6. 此系統的金匙產生相當快速。

7. 這是世界上唯一由政府所公布的簽署演算法。

8. 這個演算法應該被採納為聯邦資訊處理標準(FIPS)。

DSA 之負面評價

1. DSA 可能侵犯了兩個現存的美國專利(Schnorr 簽署方法及Public Key Partners，其中PKP 為Diffie 及Hellman 之公開金匙分配方法)。這是NIST 急需解決的首要問題。

2. DSS 並不和現存的國際標準相容，國際標準組織如ISO、CCITT 及SWIFT 已接受RSA 為標準。但DSS 並不和它相容，這個問題將導致美國工業界必須採用兩種不同標準的困擾。

3. DSS驗証程序比RSA 大約慢100 倍。

4. 根據Rivest 教授(RSA 發明人之一) 的說法，DSS 的驗証程序中有點小毛病，就是當s剛好為0 時，會產生1 除以0 的情況，但NIST 似乎沒有注意這件事，這問題的解決方法只要在簽署程序時規定只要$s = 0$出現的

時候，就再另選一個k值產生另一組s不為0的簽署文就可以了。這問題若疏忽掉的話將會危及安全性，因為若$s = 0$就很容易可算出使用者個人的秘密金匙為$x = -h(m)/r \bmod q$。

5. DSS的安全性所倚靠的數學難題和長久被研究的解離散對數問題並不完全相同，其安全性還需要一段時間來徹底審驗才能確定。

6. DSS並未明確的指明k值的選取，其實只要一個k值就可危害到系統的安全性，所以k值的產生方式對系統有致命的影響。

7. DSS並未明確指出數位簽署系統所需的"赫序函數"及"金匙交換"的做法準則(但最近美國政府已公布一套赫序函數標準，而且NIST說明DSS並不是要用來做金匙交換系統的功用，另有更符合聯邦密碼準則的金匙交換技術出現)。

5.4 新的改良方法

由前兩種改良方法，我們提出一個新的改良方法，並討論其優缺點。

參數:

p : 512 位元之質數。

q : 160 位元之質數且 $q \mid p-1$ 。

g : 滿足 $g = h^{p-1/q}$ 。

h : 單向赫序函數。

x : $0 < x < q$ 為秘密金匙。

y : $y = g^x \bmod p$ 為公開金匙。

p 、 q 、 g 及 h ，為系統公佈之共同參數與公開金匙 y 均要公開， x 為簽署者之密匙。

若 A 欲將明文 m $(1 \le m \le p-1)$ 簽署，其進行步驟如下：

步驟一　A 先任選一整數 $k(0 < k < q)$ ，使得 $(k, q) = 1$, 然後求出簽署文 (r, s) 。

簽署 : $r = g^k \pmod p \pmod q$, $s = x^{-1}(kr^{-1} - h(m)) \pmod q$

步驟二　A將m與簽署文(r,s)傳送給B，等B收到m與(r,s)後，再利用A之公開金匙y執行驗證。

驗証:

(1) 先檢查r,s是否都在$[0,q]$，若否則(r,s)不是簽署文。

(2) 計算$t = rh(m) \pmod q$及$r' = g^t y^{sr} \pmod p \pmod q$

若$r' = r$成立，則(r,s)為m之合法簽署文。換言之若(r,s)為合法簽署文，則

$$\begin{aligned} r' &= g^t y^{sr} \pmod p \pmod q, \\ &= (g^{rh(m)} g^{xsr}) \pmod p \pmod q, \\ &= (g^{(h(m)+xs)r}) \pmod p \pmod q, \\ &= (g^{(h(m)+xs)k(h(m)+xs)^{-1}}) \pmod p \pmod q, \\ &= g^k \pmod p \pmod q \\ &= r. \end{aligned}$$

優點:

1. 在簽署時，因為x固定，所以x^{-1}可事先求出，並儲存於記憶體，且簽署者可事先求出r，並計算其乘法反元素r^{-1}，同樣的儲存於記憶體。因此，在每次簽署時無需計算乘法反元素，故可減少二次乘法反元素的計算。

2. 在驗証時無需求乘法反元素。

3. IC卡的持有者適用於簽署者或驗証者皆可。

4. 若$s = 0$，則不會產生DSA中1除以0的情況，且個人秘密金匙x也無法被求出。

缺點:

其缺點是若$r = 0$時，在簽署時會產生1除以0的情況。而在驗証時，因為$r = 0$，則$t = 0$。若(r, s)為合法簽署文，則

$$r' = g^t y^{sr} \Rightarrow 0 = g^0 y^{s \cdot 0} \Rightarrow 0 = 1(\text{矛盾}).$$

由上可知，若$r = 0$時則會使驗証產生不合的情況。所以當r為0時，則需再選擇另一k值使r不為0。

參考文獻

[1] W. Diffie and M.E. Hellman, "*New Directions in Cryptography*," IEEE Transactions on Information Theory, Vol. IT-22, No. 6, pp. 644-654, 1976.

[2] R. Rivest, A. Shamir,and L. Adleman, "*A Method for Obtaining Digital Signature and Public-key Cryptosystem*," Comm. ACM, Vol. 21, No. 2, pp. 120-126, 1978.

[3] L.M. Konfelder, "*On the Signature Reblocking Problem in Public-key Cryptosystem*," Comm. ACM, Vol. 21, p. 179, 1978.

[4] T. ElGamal, "*A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms*," IEEE Trans. on Information Theory, Vol. IT-31, No. 4, pp. 469-472, 1985.

[5] J. Nechvatal, "*Public-Key Cryptography*," in Contempo-

rary Cryptology: The Science of Information Integrity, G. J. Simmons, ed. , Piscatoway, N. J. : IEEE Press, pp. 177-288, 1992.

[6] M. Rabin, "*Digitalized Signature,*" in Foundations of Secure Computation, R. DeMillo, D. Dobkin, A. Jones and R. Lipton, eds. , Piscatoway, N. J. : IEEE Press, pp. 177-288, 1992.

[7] R. Rivest, "*The MD5 Message Digest Algorithm*," RFC 1321, 1992.

[8] NIST FIPS PUB 180, "*Secure Hash Standard*," National Institute of Standards and Technology , U.S' . Department of Commerce, DRAFT, 1993.

[9] "*Proposed Federal Information Processing Standard for Digital Signature Standard(DSS)*," Federal Register, Vol .56 , No.169, pp. 42980-42982, Aug. 30, 1991.

[10] S.M. Yen and C.S. Laih, "*Improved Digital Signature Algorithm*," IEEE Trans. on Computers, Vol. 44, No. 5, pp. 729-730, May 1995.

[11] 顏嵩銘, "公開金匙密碼系統之設計與運用法," 國立成功大學電機工程研究所博士論文,1994.

[12] "*Responses to NIST's Proposed by NIST*," Commun. ACM, Vol. 35, No.7, pp. 36-40, July 1992.