第 7 章 數位簽章技術

有鑑於數位簽章技術在文件交換、電子商務等各方面的用途日益增加，需求也愈來愈迫切。爲保障數位簽章的安全性及互通性，各國政府紛紛訂定相關的國家標準。其中以美國訂定的國家數位簽章標準 DSS 較爲著名；其他還有俄羅斯國家數位簽章標準 GOST 以及日本國家數位簽章標準 ESIGN 等等。此外在民間的電子商務應用中，RSA 行之有年，也已經成爲業界認同的標準，還有許多國際標準已納入 RSA 數位簽章[Blaze99]。

7.1 數位簽章方法之演進及相關標準

自 *Diffie* 及 *Hellman* 發表了公開金匙密碼系統的觀念後，許多沿襲此觀念的數位簽章方法陸續提出，最爲著名的兩種方法就是 RSA 及 ElGamal。RSA 是以大質數分解的困難度來保障簽章的安全性；而 ElGamal 則是使用解離散對數的困難度來保障其安全性。其餘各種數位簽章演算法則多以上述二種方法爲藍本所演化而來。

目前明訂以 RSA 爲數位簽章演算法的標準有 ISO/ICE 9796、美國國家標準 X9.30 -199X，而法國銀行界亦採用 RSA 爲標準的數位簽章方法。著名的美國數位簽章標準 DSS 則以 ElGamal 及 *Schnorr* 爲藍本演化而來。1994 年公開的俄羅斯國家數位簽章標準 GOST 也採用與 DSS 類似的架構，是一個以 ElGamal 爲基礎的數位簽章系統。在美國國家標準 X9.30-199X 中，除了 RSA 之外，ElGamal 亦爲其所認可的數位簽章標準[Akl83，Mitchell92，Simmons92]。底下將分別介紹 DSS 與 GOST 的演算法以及相關的正反面參考意見。DSS（Digital Signature Standard）數位簽章標準爲*美國國家技術及標準局（National Institute of Standard*

and Technology，NIST）於 1994 年所制訂的[NBS94]。爲了保護此一標準的安全性，美國國家技術及標準局僅承認以標準文件所載的 DSA 執行方式，並且每隔五年會對此標準作一回顧檢討，以評定其適用性。

7.2 ElGamal 公開金鑰密碼技術

和 RSA 公開金鑰密碼技術相同的，ElGamal 公開金鑰密碼技術也是適用於資料加密及數位簽章的演算法，而破解 ElGamal 的困難點，則和 Diffie-Hellman 演算法相同，都是植基於解離散對數這種問題的困難性。在 RSA 公開金鑰密碼技術的數位簽章中，相同的訊息必對應一個特定的簽章，這種簽章方式稱爲確定式的簽章。然而在 ElGamal 系統中，對某一特定明文，卻可能有許多合法的簽章。這種方式稱爲機率式的簽章。[Stallings99，Stinson95]

7.2.1 ElGamal 數學基礎

本小節將介紹 ElGamal 公開密碼技術所運用到的一項數學原理。本項數學原理的探討，有助於了解在利用 ElGamal 公開密碼技術進行數位簽章的過程中，接收方收到資訊時，檢驗其中的數位簽章是否合法這項動作。

定理 7-1

假設 a、n 爲整數，而且 $GCD(a,n)=1$，則 $a^x(\bmod n)=a^{x(\bmod \varphi(n))}(\bmod n)$ 。

證明：

令 $y=x(\bmod \varphi(n)) \Rightarrow x=k\times\varphi(n)+y$ 對於某個整數 k。

$$a^x (\mathrm{mod}\, n) = a^{k\times\varphi(n)+y} (\mathrm{mod}\, n)$$
$$\because a^{\varphi(n)} (\mathrm{mod}\, n) = 1 \Rightarrow a^{k\times\varphi(n)} (\mathrm{mod}\, n) = 1$$
$$\therefore a^x (\mathrm{mod}\, n) = a^x \times 1 (\mathrm{mod}\, n) = a^x \times a^{k\times\varphi(n)} (\mathrm{mod}\, n)$$
$$= a^y (\mathrm{mod}\, n) = a^{x(\mathrm{mod}\,\varphi(n))} (\mathrm{mod}\, n)$$

得證。

7.2.2 ElGamal 數位簽署演算法

本小節將對 ElGamal 公開密碼技術應用在數位簽署方面的演算法。如同 RSA 公開密碼技術一般，ElGamal 數位簽署演算法亦相當普及，許多數位簽章系統都是經由 ElGamal 改良而來的，其中包括著名的 DSS、Schnorr 等等方法 [ElGamal85]。

ElGamal 數位簽署演算法

取一個足夠大的質數 p 以及隨機取得 Z_p^* 中之原根 g，使得解離散對數很困難。

明文 M：$M \in Z_p^*$。

簽章 S：$S \in Z_p^* \times Z_{p-1}$。

金鑰 K：$\{p, g, x, y \mid y = g^x (\mathrm{mod}\, p)\}$，其中 p、g、y 公開；x 爲祕密金鑰。

簽署過程：

簽署者在對明文 M 進行簽章時，可依照如下步驟進行：

1. 簽章者任選一整數 $k \in Z_{p-1}$。
2. 計算 $r = g^k (\mathrm{mod}\, p)$。

3. 計算 $s = k^{-1} \times (M - x \times r) (\text{mod } p-1)$ 。

4. $S = (r, s)$ 即爲明文 M 之合法簽章。

驗證過程：

收文者欲對簽章 S 進行驗證時，可依照如下步驟進行：

1. 計算 $g^M = y^r \times r^s (\text{mod } p)$ 是否成立。

2. 若是，則爲合法簽章。反之則爲非法簽章。

現在將驗證過程的工作原理說明如下：

證明：

$$s = k^{-1} \times (M - x \times r) (\text{mod } p-1) = k^{-1} \times (M - x \times r) (\text{mod } \varphi(p))$$

$$\Rightarrow y^r \times r^s \text{ mod } p = (g^x)^r \times r^{k^{-1} \times (M - xr) \text{mod} \varphi(p)} \text{ mod } p$$

$$= (g^x)^r \times r^{k^{-1} \times (M - xr)} \text{ mod } p$$

$$= (g^x)^r \times (g^k)^{k^{-1} \times (M - xr)} \text{ mod } p$$

$$= g^{xr} \times g^{M - xr} \text{ mod } p$$

$$= g^M \text{ mod } p$$

7.2.3 ElGamal 訊息加密演算法

如果我們將 ElGamal 數位簽章演算法稍加改變，即可用於訊息加密。和簽章的情況相同的，ElGamal 加密算法亦爲機率式加密演算法；也就是說，相同的訊

息明文可能產生不同的密文。然而這些密文仍能還原成相同的訊息明文。

ElGamal 訊息加密演算法

質數 p：收發文雙方協議之大質數。

原根 g：隨機取得 Z_p^* 中之原根。

明文 P： $P \in Z_p^*$ 。

密文 C： $C \in Z_p^* \times Z_{p-1}$ 。

金鑰 K：$\{p, q, x, y \mid y = g^x \pmod p\}$ ，其中 p、g、y 公開；y 爲加密金鑰，

x 為解密金鑰。

加密過程：

傳送方在對明文 M 進行加密時，可依照如下步驟進行：

1. 傳送方任選一整數 $k \in Z_{p-1}$ 。
2. 計算 $r = g^k \pmod p$ 。
3. 計算 $s = y^k \times M \pmod p$ 。
4. 密文 $C = (r, s)$ 。

解密過程：

收文者欲對密文 C 進行解密時，可計算 $P = \frac{s}{r^x} \pmod p$ 。

值得注意的是，在 ElGamal 加密過程中，密文的長度將爲明文的兩倍。另外，與簽署過程相同的，由於每次訊息加密都必須產生一個隨機數 k，因此相同的明

文可產生許多不同的密文。這個良好的性質使得*明文攻擊法（Plaintext attack）*對 ElGamal 演算法無效。

7.3 Schnorr 數位簽署

1989 年，*Claus Schnorr* 將 ElGamal 數位簽署演算法加以改進，增進其在計算上的速度。由於此一新的數位簽署演算法在計算上較簡易，因此特別適用於計算能力較差的 IC 卡系統 [Stinson95] 。

7.3.1 Schnorr 數位簽署演算法

Schnorr 數位簽署演算法是由 ElGamal 數位簽署演算法而來，因此兩者可謂大同小異，然而相較於 ElGamal 數位簽署演算法，Schnorr 數位簽署演算法對於其運算使用的資料有著更多的限制。本段將先介紹 Schnorr 數位簽署演算法，下一段將對 Schnorr 數位簽署演算法的特性進行簡要的說明 [Ong85]。

Schnorr 數位簽署演算法

質數p：質數$p \geq 2^{512}$（即一 512 位元的質數）。

質數q：質數q滿足$q \mid (p-1)$及$q \geq 2^{160}$（即一 160 位元的質數）。

階數爲 q 之原根g：隨機取得Z_p^*中階數爲q之原根g，即g滿足$g^q = 1 (\mathrm{mod}\ p)$

且$g \neq 1$。

函數 h：單向雜湊函數。

明文 M： $M \in Z_p$ 。

簽章 S。

金鑰 K：$\{p,q,g,x,y \mid y = g^x (\mathrm{mod}\, p)\}$，其中 p、q、g、y 公開；x 爲祕密金鑰。

簽署過程：

簽署者在對明文 M 進行簽署時，可依照如下步驟進行：

1. 傳送方任選一整數 $k \in Z_{p-1}$。
2. 計算 $r = g^k (\mathrm{mod}\, p)$。
3. 計算 $e = h(r, M)$。
4. 計算 $s = k - xe (\mathrm{mod}\, q)$。
5. 簽章 $S = (e, s)$。

驗證過程：

收文者欲對簽章 S 進行驗證時，可依照如下步驟進行：

1. 計算 $r' = g^s \times y^e (\mathrm{mod}\, p)$。
2. 驗證 $h(r', M) = e$ 是否成立。
3. 若是，則爲合法簽章。反之則爲非法簽章。

現在將驗證過程的工作原理說明如下：

證明：

$$r' = g^s \times y^e (\mathrm{mod}\, p)$$

$$= (g^{k-xe(\bmod q)}) \times (g^{x} (\bmod p))^{e} (\bmod p)$$
$$= (g^{k-xe}) \times (g^{xe}) (\bmod p)$$
$$= g^{k} (\bmod p)$$
$$= r$$

由上式可知，如果簽章過程確實合法，則 $h(r, M)$ 與 $h(r', M)$ 必然相等。

7.3.2 Schnorr 數位簽署之特性

由上述對 Schnorr 數位簽署的介紹，不難發現 Schnorr 數位簽署演算法與 ElGamal 數位簽署演算法相當類似。事實上，Schnorr 數位簽署的安全性和 ElGamal 相同，都是植基於解離散對數的困難度上。然而這兩個方法仍有些許的不同，現在一一討論如下：

1. 在 Schnorr 的簽署過程中，由於參數 r 與文件 M 無直接關係，因此可事先計算大量的 r 並加以儲存；而在實際進行簽署的時候，則僅需要一次乘法及一次加法的計算。這種做法明顯地較其它數位簽章方法節省計算成本。因此 Schnorr 數位簽章演算法特別適合應用於計算能力較弱的 IC 卡系統中。

2. 在 Schnorr 數位簽章系統中，簽章長度爲 $|e| + |s|$；因此，如果雜湊函數的輸出長度爲 160 位元，則 Schnorr 的簽章長度爲 160 + 160 = 320 位元。與前述 RSA 系統之簽章長度：$|n|$ = 512 ~ 2048 Bits；或是 ElGamal 系統之簽章長度：$2\times|p|$ = 1024 位元，Schnorr 數位署演算法所產生的簽章爲較短的一種。

7.4 DSA 數位簽署

1991 年，*美國國家技術及標準局（National Institute of Standards and Technology，NIST）*公佈了 *DSS 數位簽署標準（Digital Signature Standard）*[NBS94]。此一標準採用 DSA（Digital Signature Algorithm）作爲其數位簽章演算法。和 Schnorr 數位簽署演算法相同，DSA 亦爲 ElGamal 演算法的變形，因此其安全性也是基於解離散對數的複雜度。本節將討論 DSA 數位簽署演算法。

DSA 數位簽署演算法

質數 p：爲一 512 ~ 1024 位元之質數。

質數 q：爲一 160 位元之質數且滿足 $q \mid (p-1)$。

階數爲 q 之原根 g：隨機取得 Z_p^* 中階數爲 q 之原根 g，即 g 滿足 $g^q = 1 (\mathrm{mod}\ p)$

且 $g \neq 1$。

明文 M： $M \in Z_p^*$。

簽章 S： $S \in Z_q \times Z_q$。

金鑰 K：$\{p, q, g, x, y \mid y = g^x (\mathrm{mod}\ p), x \in Z_q\}$，其中 p、q、g、y 公開；x 爲祕密金鑰。

簽署過程：

簽署者在對明文 M 進行簽署時，可依照如下步驟進行：

1. 傳送方任選一整數 $k \in Z_{p-1}$。
2. 計算 $r = g^k (\mathrm{mod}\ p) (\mathrm{mod}\ q)$。
3. 計算 $s = k^{-1} \times (M + xr) (\mathrm{mod}\ q)$。

4. 簽章 $S = (r, s)$ 。

驗證過程：

收文者欲對簽章 S 進行驗證時，可依照如下步驟進行：

1. 檢查 $0 \le r \le q$ 及 $0 \le s \le q$ 是否成立，若否，則 (r, s) 並非 DSA 簽章。
2. 計算 $g = M \times s^{-1} (\mathrm{mod}\, q)$ 。
3. 計算 $u = r \times s^{-1} (\mathrm{mod}\, q)$ 。
4. 驗證 $r = (g^t \times y^u (\mathrm{mod}\, p))(\mathrm{mod}\, q)$ 是否成立。
5. 若是，則爲合法簽章。反之則爲非法簽章。

現在將驗證過程的工作原理說明如下：

證明：

因爲 $(g^t \times y^u (\mathrm{mod}\, p))(\mathrm{mod}\, q) = (g^t \times g^{ux} (\mathrm{mod}\, p))(\mathrm{mod}\, q)$

$$= (g^{(t+ux)} (\mathrm{mod}\, p))(\mathrm{mod}\, q)$$

且 $r = g^k (\mathrm{mod}\, p)(\mathrm{mod}\, q)$ ，

所以 $r = (g^t \times y^u (\mathrm{mod}\, p))(\mathrm{mod}\, q)$ 成立的充分必要條件是：

$k = (t + ux)(\mathrm{mod}\, p - 1)$ 成立。

而 $(t + ux)(\mathrm{mod}\, p - 1) = (M \times s^{-1} + r \times s^{-1} \times x)(\mathrm{mod}\, q)$

$$= s^{-1} \times (M + rx)(\mathrm{mod}\, q)$$
$$= s^{-1} \times s \times k (\mathrm{mod}\, q)$$
$$= k (\mathrm{mod}\, q)$$

故得證。

參數產生過程

而關於 DSA 數位簽章演算法中各參數的產生過程詳述如下：

測試一個數是否爲質數的過程：

1. 令 i = 1，n = 50。
2. 假設 w 爲欲測試之數字，將 w 利用這種形式來表示：

   $w = 1 + 2^a \times m$，其中m爲奇數，a爲大於或等於零的整數。
3. 產生任意整數 b，使得 $1 < b < w$。
4. 令j = 0 及$z = b^m \pmod w$。
5. 若 j = 0 且 z = 1 或 z = w – 1，跳至步驟 9。
6. 若 j < 0 且 z = 1，跳至步驟 8。
7. 讓j = j + 1，若j < a，令$z = z^2 \pmod w$，跳至步驟 6。
8. 如果 w 不是質數則停止。
9. 若i < n，讓i = i + 1，跳至步驟 3；否則w不爲質數的機率將小於 $1/4^n$。

產生質數q且 $2^{159} < q < 2^{160}$：

1. 令n爲任意數，滿足 $2^{158} < n < 2^{159-1}$。
2. 令 $t = 2 \times n + 1$。
3. 若$t < 2^{160}$，測試t是否爲質數。

4. 若 t 爲質數，令 q = t；否則令 t=t+2，回到步驟 3.

產生一質數p，使得q滿足 $2^{511} < p < 2^{512}$且q能整除p–1：

1. 依前述方法產生 q。
2. 產生任意整數n，滿足（2^{511}-1）/（2q）< n <（2^{512}-1）/（2q）。
3. 令 t = 2×n×q + 1。
4. 測試 t 是否爲質數。
5. 若 t 爲質數，則令 p = t；否則回到步驟 2。

產生參數 g：

1. p、q 如先前所述之方法產生。
2. 令 h 爲滿足 1 < h < p–1 的任意數。
3. 令$t = h^{(p-1)/q} \pmod p$。
4. 若 t = 1，則回到步驟 2；否則讓 g = t。

7.5 DSS 數位簽章標準

DSS 所採用的數位簽章演算法即爲前一節中所描述的 DSA 數位簽章演算法，在此不再贅述。在 DSS 公佈之後，立刻引起廣泛的討論。因爲 DSS 爲美國官方所公佈之數位簽章技術，一旦爲各方採用，影響甚鉅，也因此特別受到各方重視。這些來自各方的意見，正反兩面的評價都有。現將這些意見整理、簡述如下：

對 DSS 的正面評價：

1. 利用 DSA 所產生的數位簽章長度較短。
2. 在簽署過程中，如對 r 採用預處理程序，可減少簽署時間。
3. 金匙產生速度相當快。
4. 經過政府的確認，可保證一定程度的安全性。

對 DSS 的負面評價：

1. 因爲 DSA 與 RSA 並不相容，業界多已接受 RSA，兩套標準勢必造成困擾。
2. DSA 可能侵犯 Schnorr 簽署方法的專利權。
3. 在驗證過程中，若 s=0，將造成除零錯誤。此一情況在標準文件中並未提及。其解決方法爲在簽署時若發現 s=0，則另選一個 k 重新簽署。
4. DSS 的驗證程序比 RSA 約慢了 100 倍。
5. DSA 之安全並不全基於離散對數，是否可能影響其安全性值得研究。
6. DSA 採用了與 ElGamal 類似的技術，然而卻未採用 ElGamal 原根的機制，這個作法是否會影響其安全性，值得研究。
7. k 值的產生過程對 DSA 的安全性有極大的影響。
8. 有人懷疑美國政府公佈此一標準是否有其它的動機。換言之，美國政府是否已知 DSA 潛在的漏洞，以便於執法。

7.6 GOST 數位簽章標準

GOST 34.10 爲俄羅斯的數位簽章國家標準。除了 GOST 34.10 外，相關的標準還有 GOST 34.11 及 GOST 28147。其中 GOST 28147 *爲區塊加密法（Block Cipher）*標準；GOST 34.11 則是一個由 GOST 28147 衍生出來的單向雜湊函數標準。圖 7-1 簡略的表示了 GOST 的演進及各種相關的標準：

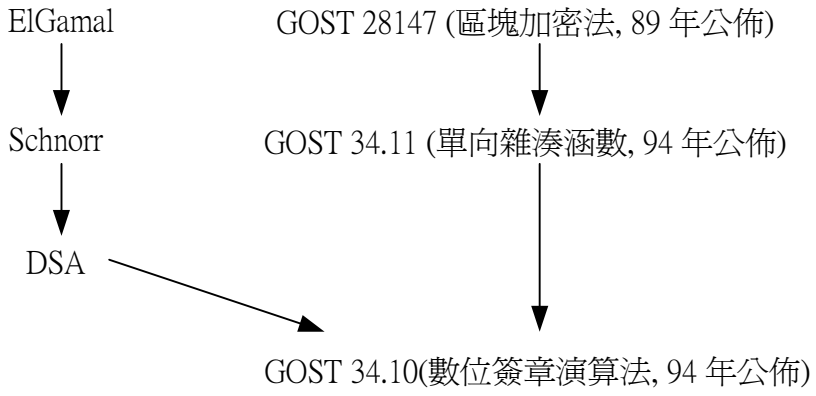

圖 7-1 GOST 的演進及各種相關的標準

和 DSA 類似，GOST34.10 數位簽章演算法也採用和 ElGamal 類似的形式，搭配 GOST 28147 區塊加密法及 GOST 34.11 單向雜湊函數標準，已足以建構一套俄羅斯國家中專有的密碼系統。雖然 GOST34.10 數位簽章演算法採用 ElGamal 的精神，與 ElGamal 數位簽章演算法大同小異，然而其演算法經由俄羅斯官方認可公佈，相信其中亦有其可觀之處。以下將介紹 GOST34.10 數位簽章演算法。

參數及定義：

質數p： 爲一滿足滿足 $2^{509} \le p \le 2^{512}$或 $2^{1020} \le p \le 2^{1024}$之質數。

質數q： 爲一滿足 $2^{254} \le p \le 2^{256}$且滿足q|p-1 之質數。

階數爲q 之原根g： 隨機取得Z_p^*中階數爲q之原根g，即g滿足$g^q = 1$（mod p），且$g \neq 1$。

明文M：$M \in Z_p^*$。

簽章S：$S \in Z_q \times Z_q$。

金匙K：{p，q，g，x，y| $y = g^x$（mod p），$x \in Z_q$ }，其中p、q、g、y公開；

x 爲私密金匙。

簽署過程：

簽署者在對明文 M 進行簽署時，可依照如下步驟進行：

1. 對文件 M 產生文件彙記 h（M），若 h（M）= 0，令 h（M）= 1。
2. 發文方任選一整數$k \in Z_q$。
3. 計算$r = g^k$（mod p）（mod q）。
4. 若 r = 0 則另取一個 k，重新計算 r。
5. 計算 s =（k × h（M）+ x×r）（mod q）。
6. 若 s = 0 則另取一個 k，重新計算 r、s。
7. 簽章 S =（r, s）。

驗證過程：

收文者欲對簽章 S 進行驗證時，可依照如下步驟進行：

1. 檢查 0 < r < q 及 0 < s < q 是否成立，若否，則（r, s）並非 DSA 簽章。
2. 計算$t = s \times h(M)^{(q-2)}$（mod q）。
3. 計算$u =（q–r）\times h(M)^{(q-2)}$（mod q）。

4. 驗證r =（$g^t \times y^u$（mod p））（mod q）是否成立。

5. 若是，則爲合法簽章。反之則爲非法簽章。

與 DSS 數位簽章標準相仿，GOST 也定義了參數產生、簽署步驟、驗證步驟等詳細過程，以供使用者遵循。以下將敘述 GOST 數位簽章之演算法細節。

計算初始階段：

在系統初始階段，由公正當局產生 p、q、g 等參數。GOST 標準提供四種產生大質數的演算法，本段將擇一描述（其餘方法皆相當類似，故從略）。以下程序產生依線性同餘方式產生一長度爲 t 的質數，$t \geq 17$，及另一質數 q，滿足 q|（p-1）且質數 q 之長度爲 |t/2|。線性同餘方程式如下：

$$x_n = 19381 \times x_{n-1} + c$$

其中 $1 \leq x_0 \leq 2^{16}-1$，$1 \leq c \leq 2^{16}-1$，且c爲奇數。

計算步驟：

1. $y_0 = x_0$。

2. 產生t_0、t_1、……、t_s：

   $t_0=t$。

   如果$t_i \geq 17$，則t_{i+1} = floor（t_i/2），否則s = i。

3. 求長度爲t_s之最小質數p_s。

4. m = s–1。

5. r_m = ceiling（t_m+1/16）。

6. 產生y_1、y_2、……、y_{rm}：

   $y_{i+1} =$（19381・y_i + c）mod 2^{16}。

7. $Y_m = \sum_{i=0}^{r_m-1} y_i \cdot 2^{16 \cdot i}$ 。

8. $y_0 = y_{r_m}$ 。

9. N = ceiling（$2^{t\times m-1}/p_m$+1）+ floor（$2^{t\times m-1}\times Y_m$/（$p_{m+1}\times 2^{16\times r\times m}$））；

   如果 N 是奇數，則 N = N + 1。

10. k = 0。

11. $p_m = p_{m+1}\times$（N + k）+ 1。

12. 如果$p_m \geq 2^{tm}$跳到步驟 6。

13. 如果（$2^{pm+1(N+k)}$（mod P_m）= 1）且（$2^{(N+k)}$ mod $P_m \neq 1$），則m = m–1;

    否則讓k = k–2 並跳至步驟 11。

14. 如果 m ≥ 0 則跳至步驟 5；

    否則讓p = p_0且q = p_1。

GOST 標準中亦定義一與上述類似的方法產生長度大於 33-Bit 的質數 p，其所採用的線性同餘方程式如下：

$$y_{i+1} =（997781173\times y_i + c）\bmod 2^{32}。$$

產生 p、q 後，以下列步驟產生 g：

1. 產生隨機數 d，使得 d < p 且與 p 互質。

2. f = $d^{(p-1)/q}$（mod p）。

3. 如果 f = 1 則跳至步驟 1；否則讓 g = f。

金匙產生階段：

1. 每一位使用者自行產生其密匙 0 < x < q。
2. 每一位使用者自行計算對應的公匙y= α^{x}（mod p）。
3. 雖然在標準文件中，並未明確提及公匙認證當局（CA）的架構，但爲保障系統的安全性，仍須設一公正當局，對每個使用者的公匙作認證。

簽署過程：

1. 對於文件 M，依照 GOST 34.11 計算雜湊值 h（M）。
2. 如果h（M）= 0（mod q），則讓h（M）= $0^{255}1$。
3. 隨機產生 k，使得 k 滿足 0 ≤ k < q 且 GCD（k, q）= 1。
4. 計算r’ = g^{k}（mod p）及r = r’（mod q）。
5. 如果 r = 0 則跳至步驟 2。
6. 文件 M 之數位簽章即爲（r, s）。

驗證過程：

1. 如果這兩個條件 0 < s < q 及 0 < r < q 其中有一不成立，此簽章即爲不合法之簽章。
2. 對於收到的訊息 M’，依照 GOST 34.11 計算雜湊值 h（M’）。
3. 如果h（M’）= 0（mod q），則讓h（M’）= $0^{255}1$。
4. 計算v = h（M’）$^{(q-2)}$（mod q）。

5. 計算$z_1 = s \times v \pmod q$及$z_2 = (q-r) \times v \pmod q$。

6. 計算$u = (g^{z1} \times y^{z2} \pmod p) \pmod q$。

7. 如果 r = u 則（r, s）確實爲 M’之簽章。

7.7 本章摘要

對於電子文件的交換等等應用而言，統一的數位簽章標準是相當重要的。有鑑於此，現代化各國莫不致力於此，除了以基本的數位簽章理論爲基礎外，更強調在現實環境的實用性。本章介紹了包括 ElGamal、Schnorr、DSS 與 GOST 的演算法，其中美國的 DSS 與俄羅斯的 GOST 是其中兩項相當知名的標準，我們也比較了兩者之間的優點與缺點。