第九章　安全性電子郵件

無論『公文交換』或『伊媚兒』都是安全性電子郵件的範疇，是電子化政府、電子化辦公室不可或缺的工具。

9-1 安全性電子郵件簡介

『安全性電子郵件』(Secure Electronic Mail, Secure E-Mail 或 S/E-Mail) 無論在電子商務或辦公室自動化環境裡，皆扮演非常重要的角色，從網路上的『伊媚兒』到公文交換，都屬於這個範疇，惟本章較偏重於電子郵件方面的介紹，其相關技術多半也能運用於公文交換上。

只要提到安全性就離不開兩大主題：認證與加密，建構 S/E-Mail 也不例外，但其運作程序與一般應用系統存在很大的差異。一般電子商務 (如 Secure Web) 多半在建立連線後，再協議出雙方可以接受的安全機制與會議鑰匙，最後就依協議的結果建立安全連線。至於安全性電子郵件則不然，它的運作方式比較接近於離線 (Off-line) 處理，也就是發信者與收信者不一定同時在線上，甚至收信者多半不知道已有信件到達，更不用說同時在線上協議安全機制與會議鑰匙。為克服此困難，發信者通常先利用鑰匙加密訊息之後，並將加密信息與相關安全機制一併傳送給收信者；待收信者收到信件之後，會先由信件內的訊息了解對方採用何種安全機制，再利用自己擁有的鑰匙解開信件。

為了達到『離線』的安全機制協議，大多需仰賴公開鑰匙系統機制。即是參與通訊者必須擁有公開鑰匙，而且利用數位憑證散佈，如此較容易達成。但許多企業內的公文交換系統，也有可能採用 Kerberos 系統認證身分與分配鑰匙。無論採用何種機制，皆有下列問題有待考慮並解決：

(1) 加密/認證演算法問題：各種加密或認證演算法都有其專屬鑰匙，不同演算法之間的鑰匙也無法交替使用。譬如，一把使用於 3DES 演算法的鑰匙，就不能使用於 RSA 或其他演算法。縱然發信者告知對方採用何種演算法加密或認證，對方若無該演算法的鑰匙，亦無法解開訊息。因此，S/E-Mail 系統有其必要制定標準的加密或認證

演算法，亦即任一套 S/E-mail 系統皆需指定使用何種加密演算法（如 3DES）或數位簽章演算法（如 DSS）。

(2) 鑰匙分配問題：倘若收信人擁有多把鑰匙，每一把鑰匙也針對不同的通訊對象，發信者又如何通知收信者應該使用那一把鑰匙來解密。因此，所有採用 S/E-Mail 系統者，都必須維護一個『鑰匙鏈』的資料庫，其中記錄每一個通訊對象的鑰匙配對。

(3) 鑰匙認證問題：如何確認對方的鑰匙是正確的。一般對於公開鑰匙認證問題都是歸類數位憑證的問題，這方面除了 X.509 憑證之外，PGP 憑證也已廣泛被使用於 S/E-Mail 系統上。

(4) 訊息亂碼問題：加密後的訊息必定產生雜亂無章的亂碼。任何一封郵件多半經由許多郵件伺服器（如 SMTP 伺服器）的轉送，才會到達目的地。另外，Internet 網路上各種伺服器大多採用 NVT ASCII 碼作為控制字元(請參考 [3] 第十二章說明)，所以郵件必須避開這些控制字元。然而加密後的亂碼極有可能和這些字元相衝突，因此，加密後的訊息不可以直接傳送，需經特殊處理始可；目前 S/E-Mail 大多將密文經過 Base-64 編碼後，再附加於信件上傳送。

(5) 郵件格式：一般郵件除了訊息內容之外，通常還包含一些控制訊息，如認證訊息、加密後密文等等。這些都必須制定標準格式，收信者才可以明瞭信件內的含意；目前 S/E-Mail 有 S/MIME 與 OpenPGP 兩大系統，他們分別制定自己的標準郵件格式。

上述的問題分別有其解決方案，本章首先介紹 S/E-Mail 的安全性功能，與其相關技術，接著再介紹 S/E-Mail 的標準格式，最後以 PGP 為例，介紹鑰匙分配及認證的技術。

9-2 Secure E-mail 安全性郵件

9-2-1 Secure E-mail 郵件結構

在網路通訊上，S/E-Mail 最能表現出資訊安全技術的特性，因為一般電子郵件就安全性的考量包含有：

- ◈ 隱密性（Privacy）：加密處理過的郵件內容，盜取者無法窺視，所以可達到隱密性功能。
- ◈ 確認性（Authenticity）：收件者接收到一封屬名的信件，必須能確定該信件確實是發信者，而不是他人冒名頂替或偽造的。
- ◈ 完整性（Integrity）：確定信件或公文沒有被他人竄改。
- ◈ 不可否認性（Non-repudiation）：信件發送之後，收件人持有憑據，讓發送者無法否認發送該信件。

由上述可以看出，S/E-Mail 幾乎包含所有資訊安全的基本技術。其實建構安全性郵件並非想像中那麼很複雜，其基本概念是將處理後的訊息與安全機制資料一併植入於郵件內，植入的方法如圖 9-1 所示。圖 9-1 為一般安全郵件的典型格式，它將郵件劃分若干個區段，每個區段承載著各種安全訊息，且每一區段都有一個區段『標頭』，記載著該區段內所攜帶訊息的特性，稱為『內文型態』（Content Type）。至於將郵件分割為若干個區段是 MIME 郵件的主要功能，因此，安全郵件多半都建立在 MIME 郵件上。接下來，我們就依照圖 9-1 的模式，來討論建構 S/E-Mail 安全性功能的方法，之後再來探討如何將這些安全功能嵌入於 MIME 郵件上。

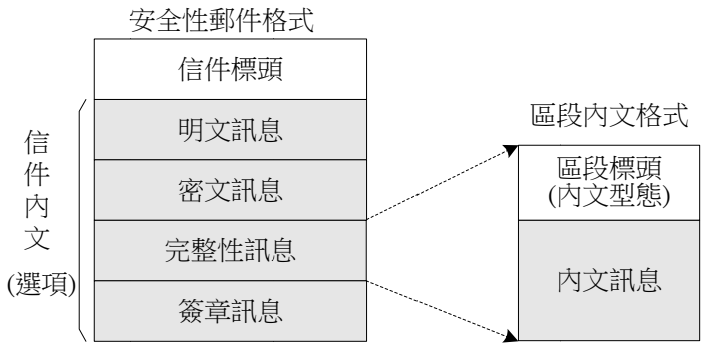

圖 9-1 安全性郵件之典型格式

9-2-2 隱密性功能

一封被發送的郵件，通常經過多個 SMTP 伺服器轉送才能到達收信人的郵件伺服器內，且需經由收信人下載或開啟信件後，才算真正完成郵件的傳送。在傳送過程當中，郵件每經過一個轉送器，多半以『儲存後再前送』（Store-and-Forward）的方式處理，因此，有心人士欲窺視郵件內容並不困難；甚至，郵件到達收件人的郵件伺服器之後，

在收件者還未開啟或下載之前，也很容易被他人盜取或窺視，有此可見，隱密性功能對安全郵件而言是很重要的。簡單的方法是訊息經過加密後再傳輸，收件者再利用鑰匙解密，只要能解決鑰匙分配的問題，即能達到隱密性功能；但鑰匙分配方法牽涉到加密系統的機制，基本上，S/E-Mail 有公開鑰匙與秘密鑰匙兩種加密方法，如下所述。

【(A)公開鑰匙加密】

如果收信者有公開鑰匙配對的話，則發信者可利用收信者的公開鑰匙對郵件加密，收信者再利用自己的私有鑰匙解密，如此便能達到隱密性的功能。譬如，Bob 傳送一封信件給 Alice，假設 Alice 的公開鑰匙配對是 $\{K_{Ua}, K_{Ra}\}$，其中前者為公開鑰匙、後者為私有鑰匙，則 Bob 的運作程序如下：

傳送訊息：M

加密系統：E（如 3DES）

訊息加密：$E_{KUa}[M]$

最後 Bob 所建構的郵件內容為：

信件標頭
內文型態：encrypted; public-key
$E_{KUa}[M]$

收件者由『內文型態』欄位中了解所傳送的訊息是經過公開鑰匙加密，因此，只要使用自己的私有鑰匙就可以解開該加密訊息。

【(B)會議鑰匙加密】

因為公開鑰匙加密的傳輸效益遠比秘密鑰匙來得低，尤其對大量傳輸訊息更為明顯。就整體而言，並不鼓勵常常使用公開鑰匙配對處理訊息加密，過度增加鑰匙的曝光率，易導致遭破解的危機。最好還是利用會議鑰匙加密較妥當，只要發信者選擇一個亂數作為加密鑰匙，先對訊息加密後，再利用對方的公開鑰匙對會議鑰匙加密，一併傳送給對方。譬如，Bob 傳送一封信件給 Alice（$\{K_{Ua}, K_{Ra}\}$），其運作程序如下：

傳送訊息：M

會議鑰匙：K（發信者選取亂數）

加密系統：E（如 3DES 演算法）

訊息加密：E_K [M]

會議鑰匙加密：E_{KUa} [K]

則信件的內容為：

信件標頭
內文型態：encrypted; session key
Key-info：E_{KUa} [K]
E_K [M]

對方收到信件後，首先利用自己的私有鑰匙解開會議鑰匙的加密，再利用會議鑰匙解開訊息的加密，如此一來，除了可以達成隱密性功能，又可降低公開鑰匙系統低效率的加/解密處理。

【（C）主密鑰加密】

一般辦公室自動化系統裡，大多利用 Kerberos 系統來發給傳送者通往對方的『通行票』，發信者便可以利用通行票內的主密鑰向訊息加密，收信者亦同樣由通行票中取出主密鑰來解密，這是一般公文交換的基本運作模式。另外，其他安全性公文交換也大多採用此模式，以下的介紹不再另述。

9-2-3 完整性功能

完整性功能是要檢視資料是否遭受竄改，看起來似乎沒有其他安全功能重要，但許多地方仍會使用到它。譬如，利用網路公佈訊息或公告公文，基本上這些訊息無需隱密，並且希望所有人都能觀看其訊息內容。如不幸遭攻擊者竄改公佈的訊息，其傷害不容忽視，因此，公佈訊息者必須將該訊息作完整性的處理，任何人都可以在不受任何限制之下，隨意檢視該訊息的完整性。最簡單的做法只要將訊息經過雜湊函數計算出一個雜湊值，再將訊息與雜湊值一併放置在信件內傳送，通常郵件系統都會指定使用一種雜湊函數（如 SHA-1 演算法）。譬如，Bob 發送一封具有完整性功能的信件給 Alice，其運作程序如下：

傳送訊息：M

雜湊函數：H（如 SHA-1 演算法）

雜湊值：H [M]

則信件的內容為：

信件標頭
內文型態：text
M（明文訊息）
內文型態：MIC，sha-1
H [M]

其中 MIC（Message Integrity Code）表示該內文所儲存的訊息完整碼。

9-2-4 確認性功能

確認性是確定信件並非他人偽造與訊息內容未遭受竄改；簡單的說，即是發信者向該郵件簽署保證該信件是自己所發送的，並保證信件內容是自己所寫的。確認性功能大多採用數位簽章技術來達成，它的做法是首先發信者將訊息經過雜湊演算法，計算出一個雜湊值，再利用發信者的私有鑰匙向該雜湊值簽署得到一個簽署碼，之後簽署碼連同訊息一併傳送給收信者，收信者收到信件之後，利用對方的公開鑰匙向簽署碼解密，得到發信者的雜湊值，再使用同樣湊演算法計算所收到的訊息，得到另一個雜湊值；如果兩個雜湊值相同的話，則能確定信件內容與發信者身份無誤；否則不是訊息有誤，就是該信件是他人偽造的。一般確認性含兩個層次：僅確認性功能與確認性功能附加隱密性功能，以下說明之。

【(A) 僅確認性功能】

僅確認性功能是以明文方式傳送訊息，但有附加簽署碼。假設 Bob（$\{K_{Ub}, K_{Rb}\}$）傳送一封自己所簽署的信件給 Alice，其中 K_{Ub} 為 Bob 的公開鑰匙、K_{Rb} 為私有鑰匙。運作程序如下：

傳送訊息：M

雜湊函數：H

雜湊值：H[M]

簽章函數：SIG（如 DSS 演算法）

簽章碼：SIG_{KRb} [H[M]]；利用 Bob 私有鑰匙簽署雜湊值。

則信件的內容為：

信件標頭
內文型態：text
M（明文訊息）
內文型態：signed
SIG_{KRb} [H[M]]

【(B) 確認性附加隱密性功能】

兼具確認性與隱密性功能的郵件有兩種做法，一者為訊息加密後從事簽署處理；另一者為訊息簽署後，簽署碼再連同訊息一併加密的處理。目前一般安全性郵件較傾向於後者的做法。假設 Bob($\{K_{Ub}, K_{Rb}\}$)傳送一封自己所簽署並加密的信件給 Alice($\{K_{Ua}, K_{Ra}\}$)，運作程序如下：

傳送訊息：M

雜湊函數：H（如 SHA-1 演算法）

雜湊值：H[M]（訊息經過雜湊函數計算所得）

簽章函數：SIG（如 DSS 演算法）

簽章碼：SIG_{KRb} [H[M]]（利用 Bob 私有鑰匙簽署雜湊值）

加密系統：E（如 3DES 演算法）

會議鑰匙：K（Bob 選取亂數而成）

密文：E_K [M || SIG_{KRb} [H[M]]]（利用會議鑰匙向訊息及簽章碼加密）

會議鑰匙加密：E_{KUa} [K]（利用 Alice 公開鑰匙向會議鑰匙加密）

則信件的內容為：

信件標頭
內文型態：encryped; session key
Key-info：E_{KUa} [K]

E_K [M || SIG_{KRb} [H[M]]]（密文）

Alice 收到信件之後，首先利用私有鑰匙（K_{Ra}）解開會議鑰匙的加密，再利用會議鑰匙向密文解密，便可得到訊息的明文與簽章值，最後再利用 Bob 的公開鑰匙（K_{Rb}）解開簽章值，並得到原訊息的雜湊碼。Alice 以同樣的雜湊演算法計算所收到的訊息，亦得到另一個雜湊碼，如果兩個雜湊碼相同的話，則表示訊息內容與發信者（Bob）身份都是正確的；否則可能訊息已遭受竄改，或是該信件是冒名發送的。

9-2-4 不可否認性功能

如果某一信件經由發送者以它的私有鑰匙簽署的話，則此信件除了具有確認性功能之外，還具有不可否認性功能。收信者可以持有原來信件（經過簽署過的）向公正單位（如法院），提出發信者發出此信件的證明。

9-3 MIME 郵件標準

之前我們介紹所謂安全郵件，即是將安全措施嵌入於分段郵件上，而此分段機制就是 MIME（Multipurpose Internet Mail Extension）的主要特性。早期制定 MIME 目的是欲將多媒體功能植入於死板的文字郵件裡（RFC 822），它的做法是分別將聲音、影像、視訊植入於各個區段郵件裡，每一區段由一個區段標頭描述所承載的訊息型態（聲音或影像），再由另一個實體（Body）承載該訊息。欲瞭解 MIME 郵件格式還是必須先由 RFC 822 郵件開始，以下分別介紹之。

9-3-1 RFC 822 封裝格式

在 1982 年制定 RFC 822 標準時，網路上信件內容非常單純，大多僅傳送文字而已。因此，RFC 822 所制定的文件格式非常簡單，只包含『標頭』（Header）與『主體』（Body）兩部份，兩者之間是利用一列空白列來區隔。訊息內容是由 ASCII 文字所構成，第一行空白列之前的文字列都被當成標頭列，當郵件被轉送時，就是利用這些標頭列所標示的內容被轉送，標頭列也是採用 ASCII 文字表示。

一封信件的標頭可能包含多個『標頭列』（Header Column），每一標頭列代表文件的控制訊息。至於一個標頭列則是由一個關鍵字、一個冒號，以及所攜帶參數所構成，

較常用的關鍵字為 From（發信者地址）、To（收信者地址）、Subject（信件主旨）、Cc（副本收信者）以及 Data(發信日期時間）等等。另外，信件主體(Boby)可利用 SACII 表示任何文字，並以 <LF> 與 <CR> 表示主體的結束。以下為簡單 RFC 822 的信件範例：

```
From: 志明 <bob@cc.cma.edu.tw>
To:春嬌 <alice@pchome.com.tw>
Subject: See you tomorrow
Date: Fri. 26 Dec 2003 10:12:37 – 0400

      Please come to meet me at tomorrow.
<LF>&<CR>
```

9-3-2 MIME 延伸功能

直到 1993 年，僅文字傳輸的郵件系統，已不能滿足環境需求，『多目的網際郵件擴充』(MIME)郵件格式因而誕生。吾人所期望的郵件豈是文字模式所能滿足，沒有聲音與影像根本不符時代潮流。基本上，MIME 是 RFC 822 版本的延伸其延伸功能如下：

- ◈ 規範 5 個新的標頭列：新的標頭列都可以被包含在 RFC 822 標頭上，每一個標頭列描述與主體（Body）有關的訊息。
- ◈ 規範 7 個內文型態：原來 RFC 822 只能攜帶文字模式，MIME 為了增加攜帶其他訊息，定義了 7 種與攜帶訊息有關的型態，如聲音、影像、或其他訊息等。
- ◈ 規範郵件編碼方法：原來 RFC 822 只能以 ASC II（7 bits）編碼文字，MIME 增加其他訊息編碼方式。

接下來，我們來介紹 MIME 所增加的功能。

9-3-3 MIME 標頭列

MIME 所定義的 5 個標頭列，說明如下：

◈ MIME-Version：目前 MIME 版本為 1（MIME-Version:1）。

- Content-Type：此標頭列表示主體內訊息的格式，其包含 7 種主要型態及 15 種次型態（Subtype，容後說明）。
- Content-Transfer-Encoding：指定郵件編碼方式，MIME 定義 7 種訊息編碼方式，其中 Base64 編碼與安全機制較有關係。
- Content-ID：內文的唯一識別碼。
- Content-Description：此標頭列是說明訊息內文的格式時使用。

基本上，後面兩個標頭列（Content-ID 與 Content-Description）甚少使用，只要的重點在 Content-Type 標頭內如何增加其他訊息的傳輸。

9-3-4 MIME 內文型態

MIME 郵件如何達到『多用途』（Multipurpose），這完全視『內文型態』（Content Type）的變化。其實它的道理非常簡單，發信者只要利用 Content-Type 表示目前訊息是何種格式（如 Jepg 圖片）；收信者由 Content-Type 知道所接收訊息是何種格式，再利用該格式來觀閱訊息（如 Jpeg 顯示器）即可。也就是說，Content-Type 是用來通知雙方採用何種格式來存放訊息的意思。表 12-2 為 MIME 所制定的 Content-Type，每一種 Content-Type 又包含若干個次型態（Subtype），說明如下：（RFC 10

表 9-2 MIME 內文型態

型態	次型態	功能說明
Text	Plain	不具格式的文字，如 ASC II。
	Enrich	較有彈性的文字格式。
Multipart	Mixed	區段之間為獨立的，但需依照次序。
	Parallel	區段之間為獨立的，不需依照次序。
	Alternative	各區段皆表示同一訊息，但格式不同。
	Digest	與 Mixed 相同，但採用 rfc822 格式。
Message	rfc822	訊息經由 rfc822 格式包裝。
	Partial	訊息是經過切割包裝而成。

	External-body	訊息內容再另一個指定位置。
Image	Jpeg	訊息為 JPEG 影像格式。
	Gif	訊息為 GIF 影像格式。
Video	Mpeg	訊息為 MPEG 視訊格式。
Audio	Basic	訊息為 Basic 聲音格式。
Application	PostScript	訊息為 PostScript 格式。
	Octet-stream	訊息為 8 位元二進位格式。

◈ Text：如果訊息主體內是文字的話，則利用 Text 來表示文字型態，它有兩個次型態：

- Text/plaint：表示沒有指定特殊文字模式。
- Text/Enriched：表示較具有彈性的文字型態，另由 RFC 1341 所制定。

◈ Multipart：此內文型態表示訊息是由多個獨立部份（或稱區段）所組成，然而這些獨立區段之間的關係，又可區分為：

- Multipart/Mixed：表示各個區段之間是彼此獨立的，但收信者還是必須依照發信者所排列的次序接收。範例如下：(取自 RFC 1046)

```
From: Nathaniel Borenstein
To: Ned Freed
Date: Sun, 21 Mar 1993 23:56:48 -0800 (PST)
Subject: Sample message
MIME-Version: 1.0
Content-type: multipart/mixed; boundary="simple boundary"

        This is the preamble.  It is to be ignored, though it
        is a handy place for composition agents to include an
        explanatory note to non-MIME conformant readers.

--simple boundary

        This is implicitly typed plain US-ASCII text.
        It does NOT end with a linebreak.
```

```
--simple boundary
Content-type: text/plain; charset=us-ascii

        This is explicitly typed plain US-ASCII text.
        It DOES end with a linebreak.

--simple boundary--

        This is the epilogue.    It is also to be ignored.
```

- Multipart/Parallel：各個區段之間也是彼此獨立的，但收信者可以不按照區段次序收信。

- Multipart/Alternative：所有區段都表達同一種訊息，但各個區段表示方法的版本不同。範例如下：(取自 RFC 1046)

```
From: Nathaniel Borenstein
To: Ned Freed
Date: Mon, 22 Mar 1993 09:41:09 -0800 (PST)
Subject: Formatted text mail
MIME-Version: 1.0
Content-Type: multipart/alternative; boundary=boundary42

--boundary42
Content-Type: text/plain; charset=us-ascii

            ... plain text version of message goes here ...
--boundary42
Content-Type: text/enriched

            ... RFC 1896 text/enriched version of same message
                goes here ...
--boundary42
Content-Type: application/x-whatever

            ... fanciest version of same message goes here ...
--boundary42--
```

- Multipart/Digest：大致上與 Mixed 型態相同，但訊息型態內定為 Message/rfc822 版本。

◈ Message：表示主體內是包含某一種已格式化的訊息，MIME（RFC 2046）制定有下列次型態：

- Message/rfc822：表示主體內訊息已經過 RFC 822 規範包裝而成。
- Message/partial：表示主體內訊息是經由大訊息分割而成，亦即訊息太長而無法由一個信件承載的話，就必須利用此次型態來攜帶分割後的訊息區塊，因此，必須再增加三個參數：id（同一訊息的區段都相同）、sequence number（每一區段的順序號碼）與 total（區段的資料總數）。

```
Content-Type: Message/Partial; number=2; total=3;
                              id="oc=jpbe0M2Yt4s@thumper.bellcore.com"
```

- Message/external-body：表示主體內並沒有承載所欲傳輸的訊息，而訊息是存放在另一位置上。因此，必須有 access-type 參數表示存取型態（如 ftp、mail-server）與 name 參數表示檔案名稱。目前許多廣告信件便用此型態傳送，範例如下：

```
Content-type: message/external-body;
                access-type=local-file;
                name="/u/nsb/Me.jpeg"
Content-type: image/jpeg
Content-ID: <id42@guppylake.bellcore.com>
Content-Transfer-Encoding: binary

      THIS IS NOT REALLY THE BODY!
```

- 其他訊息型態：針對其他無法閱讀的二進位訊息，MIME 另外以 application 型態表示，譬如 Application/Octet-stream。

◈ Image：表示主體內承載影像檔案。基本上，MIME 定義有：

- Image/Jpeg：影像是 JPEG 格式。
- Image/Gif：GIF 格式。

至於其他影像檔由 RFC 2048 規定。

◈ Audio：表示主體承載為聲音檔案。目前 MIME 僅定義 Audio/Basic，其為單聲道、

8 位元 ISDN mu-law 編碼，採樣頻率為 8 KHz。

◈ Video：表示主體內承載為視訊檔案。目前 MIME 僅定義 Video/mpeg（MPEG 格式）型態。

◈ Application：應用型態。簡單的說，上述標準型態無法表示的訊息，都是利用 Application 來表示；目前 MIME 預留作為未來發展使用，為滿足各種不同的應用，MIME 定義下列兩次型態：

- Application/Octet-stream：有關由 8 位元組所構成的二進位資料都可使用此型態。
- Application/PostScript：Adobe Postscript 格式。

上述中，與安全郵件較有關係的是 Multipart 與 Application 型態，S/E-Mail 就是將它的安全機制嵌入於這兩個內文型態之中。

9-3-5 MIME 內容轉換編碼

『內容轉換編碼』（Content Transfer Encoding）是 MIME 另一重要的規範，其功能是指定何種編碼技巧將訊息內容轉換成可傳輸的數碼，MIME 的規範有下列編碼技巧：

◈ 7 bit：將訊息內容轉換成 7 位元的 ASCII 碼。

◈ 8 bit：將訊息內容轉換成 8 位元的 SACII 碼；如果原來為 7 位元的字元，則最高位元為 1。

◈ binary：以二進位碼傳輸，一般使用於執行檔或影像檔傳輸。

◈ quoted-printable：如果訊息內容大部份是 ASC II 文字的話，可將其轉換成可列印的文字編碼格式傳送。

◈ base64：將訊息以每 6 位元為單位，轉換成 8 個位元的可列印 ASCII 字元編碼；此型態為加密信件主要的轉換格式（容後介紹）。

◈ x-token：具有名稱的非標準編碼法。

指定內容傳輸編碼的控制與法如下：

```
Content-Type: text/plain; charset=ISO-8859-1
Content-transfer-encoding: base64
```

上述編碼技巧中，與 S/E-mail 較有關係的是 Base64 (即是 radix-64 編碼法)。訊息經過加密或簽署處理之後，會產生一連串的亂碼，這些亂碼並不適合儲存或傳輸，若希望將這些亂碼轉換成可閱讀的字串，需仰賴 Base64 編碼來轉換。Base64 編碼是將一連串的亂碼，以每 6 個位元為單位，連續將它轉換成可閱讀的字元(ASC II 碼)，其轉換編碼如表 12-3 所示。

表 9-3 Base64 編碼轉換表

6 位元輸入	編碼輸出	6 位元輸入	編碼輸出	6 位元輸入	編碼輸出	6 位元輸入	編碼輸出
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	L	26	a	42	q	58	6
11	K	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	-
						(pad)	=

9-4 Secure-MIME 安全郵件

所謂 S/MIME (Secure/Multipurpose Internet Mail Extension) 係指 MIME 郵件再增加安全性的功能，其安全機制是以 RSA 演算法為基礎。由此可見，S/MIME 似乎最能接近目前的郵件系統環境，因此有人大膽預測它極有可能成為未來的標準；另外，S/MIME 認證方式是採用 X.509 v3，亦能結合目前 Internet 上的電子商務環境。也就是說，S/MIME 憑證認證乃是沿用 PKIX 系統。本

只要瞭解 MIME 郵件系統，欲將安全機制嵌入 MIME 其中成為 S/MIME 就不會很困難，但我們還是先來討論 S/MIME 提供那些安全機制。基本上，S/MIME 是由 RSA 公司所發展出來，所以公開鑰匙系統方面大多採用 RSA 演算法，我們依照 S/E-Mail 的安全措施，將 S/MIME 所採用的演算法歸納如下：

- ◈ 訊息摘要：欲產生數位簽章之前，必須將訊息經過某一種雜湊演算法，計算出訊息摘要，再簽署訊息摘要來得到數位簽章。S/MIME 規範中(RFC 2633)有 SHA-1 與 MD5 兩種演算法（第五章介紹），但大多採用產生 160 位元訊息摘要的 SHA-1 演算法。
- ◈ 數位簽章：即是簽署訊息摘要產生數位簽章的演算法。S/MIME 規定傳送端與接收端必須提供 DSS 演算法（第七章介紹），並且傳送端至少必須提供 RSA 加密法，接收端至少應支援 RSA 簽章確認方式。簽署鑰匙的長度可以從 512 位元到 1024 位元。
- ◈ 訊息加密：S/MIME 規範有 RC2/40 與 3DES 加密演算法，但大多採用 3DES 演算法（第三章介紹）。
- ◈ 會議鑰匙加密：即是將會議鑰匙加密後，與密文一併傳送出去的郵件，其中針對會議鑰匙加密的演算法。S/MIME 規定必須採用 Diffie-Hellman 演算法(RFC 2631，即是 ElGamal 演算法)。

瞭解 S/MIME 所採用的安全機制之後，接著來探討如何將這些安全機制嵌入 MIME 郵件之內。

9-4-1 S/MIME 安全郵件型態

S/MIME 係利用 MIME 郵件分段（Multipart）與擴充訊息型態（Application）兩種內文型態（Content Type），將所加密或簽署的訊息嵌入郵件之中，嵌入方法如同 9-3 節所介紹。S/MIME 所定義的內文型態，如表 9-4 所示。

表 9-4 S/MIME 所定義的內文型態

內文型態	次型態	S/MIME 參數	說明

Multipart	signed		訊息主體分為：訊息與簽章兩部份。
Application	pkcs7-mime	signedData	已簽署的 S/MIME 物件。
	pkcs7-mime	envelopedData	已加密的 S/MIME 物件。
	pkcs7-mime	degenerate	只包含數位憑證的物件。
	pkcs7-signature	signedData	已簽署的 multipart 訊息。
	kcs10-mime	signedData	要求登錄憑證訊息。

在 RFC 1847 規範中，定義有 Multipart/Signed 與 Multipart/Encrypted 兩種內文型態，前者為簽署區段、後者為加密區段。但這兩種內文型態所傳輸訊息都必須是為經過轉換的原文，亦即無論加密或簽署後的亂碼，都直接將其放置於郵件區段之內傳送。然而 S/MIME(RFC 2633)只採用 Signed 區段，並另外制定 Application/pkcs7-mime 作為訊息加密與簽署使用，並且可選擇採用何種『內容傳輸編碼』將訊息轉換，一般都採用 Base64 編碼方式，其中 PKCS #7 為 RSA 加密套件標準（容後說明）。

【(A) Multipart/Signed 型態】

如郵件採用 Multipart/Signed 型態則包含兩個區段，第一個區段包含原 MIME 標頭及訊息主體，第二個區段則為數位簽章，並包含下列定義及參數值：

(1) MIME 型態名稱：Multipart

(2) MIME 次型態名稱：Signed

(3) 參數：boundary、protocol、與 micalog

(4) 選項參數：無

(5) 安全考量：傳輸訊息未經過特殊標碼。

其中 micalog 表示採用何種『訊息完整性檢查』(Message Integrity Check, MIC) 的演算法，protocol 表示訊息的表示方式。範例如下：(取自 RFC 1847)

```
Content-Type: multipart/signed; protocol="TYPE/STYPE";
                micalg="MICALG"; boundary="Signed Boundary"

--Signed Boundary
Content-Type: text/plain; charset="us-ascii"
```

```
        This is some text to be signed although it could be
        any type of data, labeled accordingly, of course.

--Signed Boundary
Content-Type: TYPE/STYPE
        CONTROL INFORMATION for protocol "TYPE/STYPE" would be here

--Signed Boundary--
```

【(B) Application/pkcs7-mime 型態】

Application/pcks7-mime 型態是希望將信件包裝成 CMS (Cryptographic Message Syntax，RFC 2630) 的標準格式，其中並使用 PKCS #7 公開鑰匙系統之安全套件。它的郵件操作可區分為 Enveloped-data (密文或明文)、Signed-data (數位簽章)、以及 Certs-only (憑證要求) 等三種，然而此三種操作可以在同一郵件上混合使用。為了增加發信者與收信者之間處理信件的方便，郵件內可以利用參數說明應採用何種安全套件，為了區分不同的郵件操作，可用不同的參數檔案型態來區分，如表 12-5 所示。

表 12-5 Application/pcks-7-mime 參數與檔案型態

MIME 型態	smime-type	參數檔案型態
Application/pcks7-mime	signedData	.p7m
	envelopedData	
Application/pcks7-mime	signedData (Certs-only)	.p7c
Application/pcks7-signature	signedData	.p7s

利用參數檔案有許多優點，接收端可將各種參數檔案儲存磁碟機內。當收到信件時，便可以搜尋磁碟機是否有該型態的參數檔案，如果找不到合適的參數檔案，也可以要求發送者傳送該檔案給接收者，接收者再依此來認證或解密郵件。

【(C) PKCS #7 安全套件】

PKCS #7 是 RSA 公司所制定的安全套件 (RFC 2315)，包含訊息加密與簽署的功能。更重要的是，它將訊息經由安全處理之後，所產生的密文、數位簽章、會議鑰匙加

密、或鑰匙訊息等等訊息，重新包裝成另一個特殊的資料結構；之後再將此資料結構填入郵件中傳送。因此，採用 Application/pkcs7-mime 型態時，郵件主體內只有一個區段，並且無法由主體區段觀察出所傳輸的內容，該資料結構又稱為『數位信封』(Digital Envelope)。

我們以 EnvelopedData 型態來觀察，PKCS #7 的郵件處理過程，該型態僅有加密處理 (或許沒有)，並沒有包含簽署功能。假設發信者發送一封信件給多個接收者，處理步驟如下：

(1) 發信者依照加密演算法 (如 3DES)，選擇一個亂數作為會議鑰匙，並向訊息加密。

(2) 利用收信者公開鑰匙，分別向會議鑰匙加密。

(3) 分別將加密後的會議鑰匙填入收信者的 RecipientInfo 參數內，其中包含收信者其他資料。

(4) 將每一收信者 RecipientInfo 與密文，組合成一個『數位信封』。

(5) 再將此『數位信封』嵌入於 Application/pcks7-mime 郵件之內。

至於 EnvelopedData 數位信封的資料結構如下：(ANS.1 宣告格式)

```
EnvelopedData ::= SEQUENCE {
        version Version,
        recipientInfos RecipientInfos,
        encryptedContentInfo EncryptedContentInfo }

RecipientInfos ::= SET OF RecipientInfo
EncryptedContentInfo ::= SEQUENCE {
        contentType ContentType,
        contentEncryptionAlgorithm
        ContentEncryptionAlgorithmIdentifier,
        encryptedContent
           [0] IMPLICIT EncryptedContent OPTIONAL }

EncryptedContent ::= OCTET STRING
```

由上述可以觀察出，無論密文、鑰匙訊息、加密演算法都包含在此資料結構內。接著，再將此資料結構以某一種編碼 (如 Base64)，轉換成可傳輸的數碼。其他數位信封 (如 Signed) 作者不再贅述，有興趣讀者請參考 RFC 2315。接下來將介紹各種 S/MIME 封

裝格式。

9-4-2 僅信封包裝郵件

『僅信封包裝』(Enveloped-only)表示僅將所欲傳輸的訊息包裝成『數位信封』(即是 PCKS #7 格式)，所傳輸的訊息可能經過加密處理或沒有，但必須是沒有經過數位簽章處理。S/MIME 郵件格式如下：(取自 RFC 2633，未包含 MIME 標頭)

```
Content-Type: application/pkcs7-mime; smime-type=enveloped-data;
              name=smime.p7m
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m

    rfvbnj756tbBghyHhHUujhJhjH77n8HHGT9HG4VQpfyF467GhIGfHfYT6
    7n8HHGghyHhHUujhJh4VQpfyF467GhIGfHfYGTrfvbnjT6jH7756tbB9H
    f8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
    0GhIGfHfQbnj756YT64V
```

上述郵件主體是 PKCS #7 資料結構，已經過 Base64 編碼，因此，無法觀察出它的內容。

9-4-3 僅簽署郵件

簽署信件的運作程序如 12-2-3 節介紹。但在 S/MIME 郵件中可以選擇 Multipart 與 Application 兩種信件格式，如下介紹。

【(A) 採用 Application 型態】

採用 Application/pcks7-mime 型態是將訊息包裝成『數位信封』，唯一不同的地方是 smime-type 參數指明是簽署資料 (signed-data)，範例如下：

```
Content-Type: application/pkcs7-mime; smime-type=signed-data;
              name=smime.p7m
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m
    567GhIGfHfYT6ghyHhHUujpfyF4f8HHGTrfvhJhjH776tbB9HG4VQbnj7
    77n8HHGT9HG4VQpfyF467GhIGfHfYT6rfvbnj756tbBghyHhHUujhJhjH
    HUujhJh4VQpfyF467GhIGfHfYGTrfvbnjT6jH7756tbB9H7n8HHGghyHh
    6YT64V0GhIGfHfQbnj75
```

【(B) 採用 Multipart 型態】

採用 Multipart/signed 型態並沒有重新包裝訊息，因此，信封內包含兩個主體區段，一區段為原訊息資料，另一區段為簽署的數位簽章。另外，簽署區段內的數位簽章是採用 PKCS #7 封裝格式 (Application/pcks7-signature)。這種做法是希望訊息部份可以不經處理便可閱讀得到，除非需要認證訊息來源時才處理數位簽章，一般公告公文或廣播訊息都採用這種方式。範例如下：

```
Content-Type: multipart/signed;
              protocol="application/pkcs7-signature"; micalg=sha1;
boundary=boundary42

--boundary42
Content-Type: text/plain

          This is a clear-signed message.

--boundary42
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s

          ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfHfYT6
          4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbB9HGTrfvbnj
          n8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
          7GhIGfHfYT64VQbnj756
--boundary42--
```

此範例是採用 SHA-1 雜湊演算法與 DSS 數位簽章演算法，其中前區段為訊息明文、後區段為數位簽章。

9-4-4 簽署並加密郵件

兼具簽署與加密郵件的運作程序如 9-2-3 節介紹，郵件包裝是採用巢狀方式，即利用 signed-only 與 encrypted-only 兩種型態交替處理。一般是先計算出雜湊值，並經過簽署後得到一個數位簽章，再包裝成『數位信封』的資料結構 (signed-only 處理)。接下來，將此數位信封當作傳輸訊息，再經過加密處理，並包裝成另一個『數位信封』(encrypted-only 處理)，完成之後，再嵌入 S/MIME 郵件上傳送；收信者再以反方向

拆解信件。