結合 IC 卡之校園安全網頁系統的設計與實現

陳軒正，國立高雄師範大學資訊教育研究所研究生

楊中皇，國立高雄師範大學資訊教育研究所副教授

摘要

在危機四伏的網路世界中，一般的網頁系統已難以保護網頁資料及使用者資料的安全，因此本研究主要目的在設計一套**校園安全網頁系統**(Secure Campus-wide Website System)，將學校網頁中較為機密或敏感的資料，透過資料庫及伺服器端網頁語言機制加密儲存起來，並依使用者權限加以分級管理。教職員每人發給一張 IC 卡，做為身份識別及加解密之用，如此充分保障電子資料傳遞與儲存之安全，也不怕因伺服器遭受入侵而洩漏機密資料。

校園安全網頁系統主要採三層式架構設計，分別為伺服器層、資料庫層與使用者層，其中前兩者以自由軟體開發，支援多種作業系統，且軟體建置成本接近於零；使用者層採用 Windows 平台，方便一般使用者操作，並支援更多種類的 IC 卡讀卡機。本系統也使用著名的 RSA 及 AES 密碼學，以確保資料之安全。

關鍵字：IC 卡、WWW、AES、RSA、PHP

1.前言

在資訊安全發生的相關事件中，首頁的改寫、資訊的外洩、電子郵件的窺伺及電腦病毒…等常常是媒體報導的熱門話題。而其中又以首頁的改寫佔全體的四成多，主要包含蠕蟲程式或其他手法所導致；其次就是「檔案的改寫」，約佔一成左右的比例。【1】 由此可知，只要有心人士利用系統的漏洞，或是一些駭客工具軟體，很容易可以入侵系統主機，輕易地竄改網頁或檔案。

隨著電子文件的普及，一些機密資料幾乎都以電子檔案存放，加上人員或管理者的疏失，很容易將不應公開的資訊放置在對外的伺服器上，卻又未妥善控制存取，因而造成資訊外洩事件。該類事件大多是人為因素所造成，而且即使有安裝防火牆、入侵偵測系統等網路設備，也難以防範。

目前大多數網站所提供的全球資訊網(World Wide Web, WWW)或相關的 Web-based 服務，雖然使用率高，但是牽涉到較為機密的內容或個人服務，通常

也只使用簡單的帳號及密碼的認證機制來管制，而且放置於主機中的檔案也少有額外保護機制，因此很容易被他人所攔截、破解，甚至複製。

因此本研究針對校園網路，希望能利用自由軟體開發一套校園安全網頁系統，將原本網頁中較為機密或敏感的資料抽離出來，透過資料庫管理系統(DBMS – Database　Managerment System)及伺服器端網頁程式的機制預先以金鑰加密儲存起來，並透過金鑰資料庫統一管理。而使用者則需要透過 IC 卡來做身份認證【2】，並透過伺服器--資料庫管理機制來檢核權限，當具有足夠權限時才可以取得解密金鑰解開加密網頁來閱讀(一個金鑰僅能解開一筆資料)。如此一來除了可將網頁內容分級管理之外，網頁的儲存及通訊均以加密安全機制來處理，即使網站受到入侵也不怕機密資料的曝光。此外，透過 IC 卡來做使用者的認證，可避免帳號密碼被竊取或遺忘的風險，還可利用 IC 卡做使用者的個人化服務，一舉數得。

2.系統規劃

一般學校(尤其是中小學)在校園網路的建置及維護上，多缺乏專業的網路管理人員，經費也較為有限，所以在資訊安全管理方面常會出現許多漏洞。加上多數教職員對資訊安全的概念及習慣尚未建立，更容易讓不肖人士有機可乘。

校園安全網頁系統(以下簡稱本系統)設計時，針對校園電腦及網路環境，希望能建構一套容易安裝、維護及使用的系統，並優先考量以下幾點：

(1)如何加密及儲存資料使其安全無虞，並具備內容分級功能。
(2)以何種個人認證方式可以確定本人身份，認證機制要方便使用且花費不高。
(3)如何以最低成本建置系統。
(4)用何種軟體、套件可以搭配原有主機架構(具備跨平台特性)。

針對上述幾項考量，本系統做出以下四項規劃，可以改善現行網頁系統的缺點，讓網頁系統更加安全：

2.1 可靠的加密機制：

傳統的超文件傳輸協定(HyperText Transfer Protocol, HTTP)是完全沒有加密的，因此有心人士只要利用網路監聽軟體，就可以輕易擷取出封包中的帳號及密碼。而一般在電子商務中常使用的 SSL(Secure Sockets Layer)加密技術雖然安全，但是仍無法確定利用該輸入帳號登入的就是本人。因此本系統規劃使用密碼

學技術，預先將網頁的內容加密儲存，使用者讀取網頁時所使用的雖然是 HTTP 協定，但是內容全為密文，即使封包被監聽、甚至網站被入侵，都無法得知網頁的內容，因此確保資料的絕對安全。

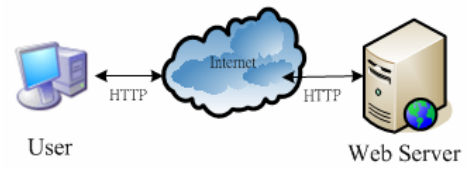
2.2 IC 卡的使用：

傳統網頁僅以帳號及密碼來做身份的確認，無法真正確認是否為使用者本人，比較可信的是利用生物科技認證，或利用僅個人所有的物品來認證，如 ID 卡、IC 卡。然而目前生物科技認證（如虹膜、指紋、聲紋）在辨識準確率方面仍待加強，且使用者資料庫的建立相當困難，辨識設備也較昂貴，因此目前較適合用來做電子認證的，就非 IC 卡莫屬了。IC 卡及讀卡機不但成本低，而且使用方便(體積小、有各式傳輸介面)，功能多樣(有 IC 記憶卡、微處理晶片 IC 卡)很適合用於網路身份的認證，因此為本系統採用。【3】

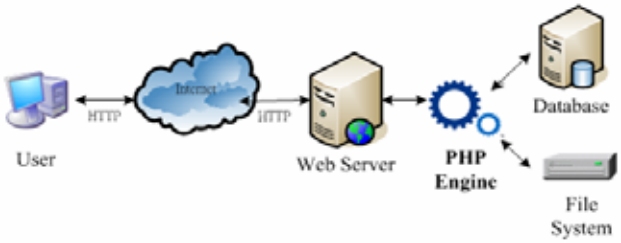
2.3 伺服器端網頁程式語言及資料庫的使用：

將原本一些不安全的靜態網頁(如圖一)或動態網頁，改以具加密功能的伺服器端網頁程式語言(Server-side Webpage Language)，並將資料加密後儲存於資料庫中(如圖二)。加密動作統一由伺服器執行，並將內容及金鑰計算雜湊值儲存，以確保資料之正確性與安全性。

目前常見的伺服器端網頁程式語言有：Perl, PHP, ASP, JSP …等；而常使用的 DBMS 則有 Oracle, Sybase, MS-SQL, MySQL, PostgreSQL … 等。



圖一、現行網頁系統的架構



圖二、改良後加密網頁架構

2.4 採用自由軟體：

目前大多數的校園網頁伺服器運行於 Unix-like 與 Windows 作業系統上，因此本系統規劃採跨平台的方式，希望能搭配原有的系統架構，直接安裝於原來的伺服器上，一來可以節省硬體的購置成本，二來網管人員也方便管理。

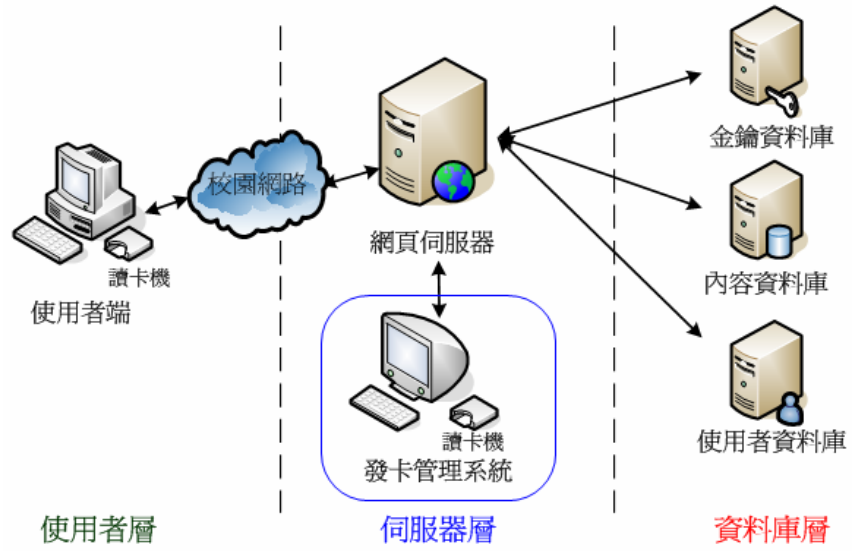
其次考量到節省軟體成本，本系統因此決定採用能跨平台的自由軟體來設計，所選用的是:Apache、PHP、MySQL 及 OpenSSL。

因此本系統能改善現行網頁系統的許多缺點，如表一。

表一、現行網頁系統與本系統之比較

	現行網頁系統	本系統
使用者認證	只藉著帳號及密碼認證， 無法判定是否為本人	藉著 IC 卡與密碼雙重認證， IC 卡可代表本人
帳號安全	容易被竊取或以暴力式破解法等 方式猜得密碼	帳號存於 IC 卡上難以複製， 密碼錯誤三次 IC 卡即停用
使用通訊協定	HTTP / HTTPS	HTTP
伺服器端資料	多以明文方式儲存	全部以密文儲存
資料安全	HTTP 傳輸過程無加密 伺服器若被入侵則資料全數曝光	傳輸過程皆以密文傳遞 伺服器若被入侵資料不受影響

3.系統架構



圖三、系統三層式架構圖

根據上述系統規劃，本系統採用三層式架構設計(3-tier model)來設計，分為資料庫層、伺服器層與使用者層(如圖三)。

資料庫層包含金鑰資料庫(Key Database)、內容資料庫(Content Database)及使用者資料庫(User Database)，金鑰資料庫負責紀錄文件加密所需的加密金鑰(encryption key)，內容資料庫用來儲存加密文件的內容，使用者資料庫則是儲存使用者相關資訊，含識別碼、個人資料、權限及公開金鑰等。

伺服器層主要以 Web Server 及動態網頁程式語言構成，並且包含一個發卡管理系統。Web Server 負責使用者身份的認證，確認後傳遞加密內容及用來解密的金鑰；發卡管理系統主要在控管 IC 卡的發行，並於使用者資料庫中建立使用者的身份、權限及公開金鑰。

使用者層主要是利用 IC 卡的認證，讓使用者能建立加密的文件、檔案於內容資料庫中，並設定讀取者的權限，以達到文件分級的功能。不論是發卡管理系統、使用者端，所有動作皆需透過伺服器端來達成，以確保資料庫之安全性與一致性。

4.密碼學

密碼學技術是目前唯一能有效在不安全的網路上安全傳遞訊息的工具，選用良好的密碼系統可以確保資料的安全性、正確性以及良好的傳輸速度。

本系統所使用的密碼系統有私密金鑰系統(Secret-key cryptosystem)與公開金鑰系統(Public-key cryptosystem)兩種【4】。私密金鑰密碼系統我們使用 Rijndael 演算法，也是美國政府機構的先進加密標準---AES(Advanced Encryption Standard)【5】；至於公開金鑰系統則採用知名的 RSA 演算法【6】。這兩種演算法都經過了嚴格的測試與考驗，並廣為業界所採用，在安全方面是無庸置疑的。

另外，本系統也使用了雜湊演算法—SHA-256【7】來確保資料及金鑰的正確性，安全性也較目前常見的 MD5 與 SHA-1 來得高。

表二 、本系統所採用之密碼演算法及其功能

	演算法	加密長度	功能
私密金鑰演算法	AES(Rjindael)	128 bit	負責資料庫內資料之加密
公開金鑰演算法	RSA	1024 bit	負責解密金鑰之傳遞
雜湊演算法	SHA-256	256 bit	驗證傳送內容之正確性

5.IC 卡

本系統目前所使用的 IC 卡為 Simems SLE 4428，記憶體容量為 1 KB，足以儲存使用者的個人資料及私密金鑰。

6.系統執行流程

本系統所提供之功能，主要有 IC 卡的發行(發卡管理系統)、使用者上傳資料至伺服器及使用者下載資料等三項。這些功能在執行時均經過嚴格的加密過程，其流程分別說明如下：

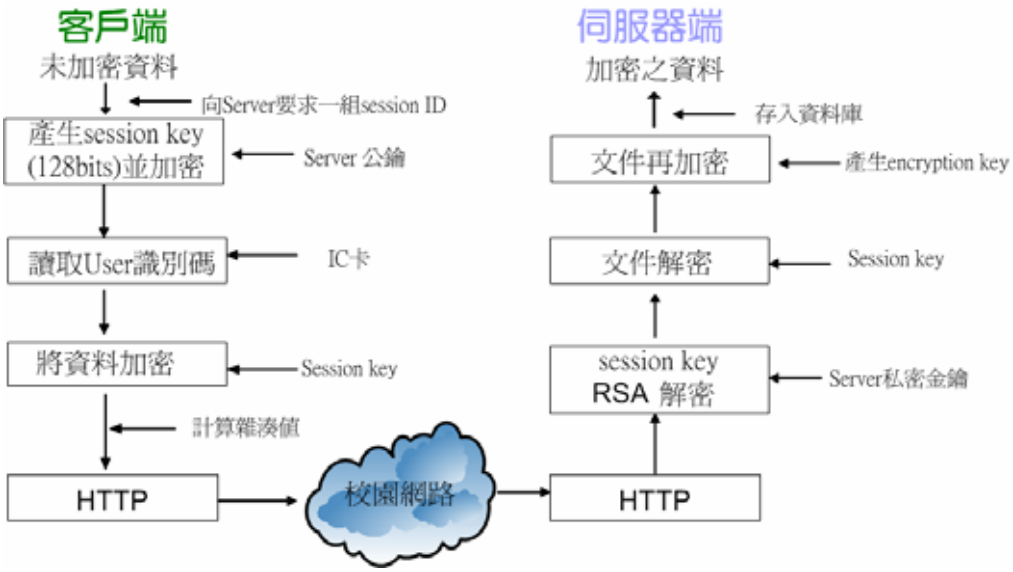
6.1 IC 卡發卡管理系統建立帳號過程

由管理者設定使用者的資料、權限，以亂數產生一組識別碼(ID)(8 bytes，以 Server 端的公鑰加密)，並產生使用者 RSA 金鑰對(key pair)。將上述資料寫入 IC 卡(金鑰則只有寫入私密金鑰)後，使用者資料、識別碼，及使用者金鑰傳遞至伺

服器，由伺服器產生公鑰憑證檔後將各項資料寫入使用者資料庫，並計算上述資料之雜湊值，儲存於資料庫中。

6.2 上傳加密過程

使用者將資料上傳至伺服器的流程如下圖：



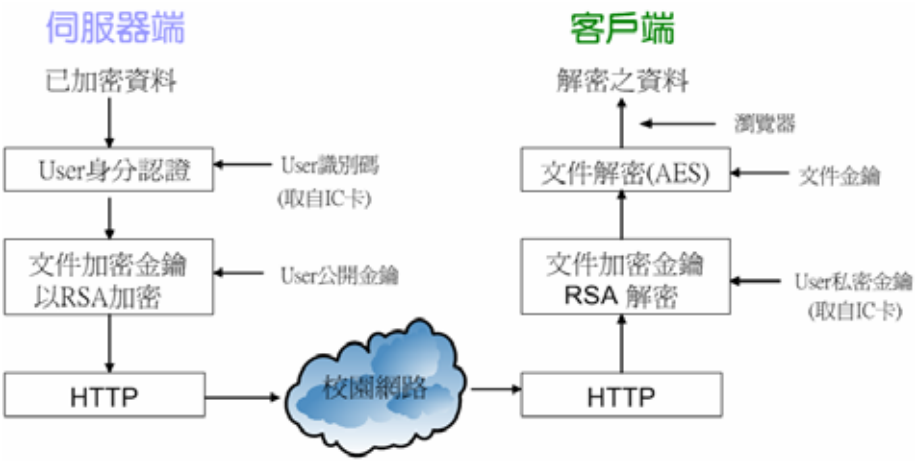
圖四、上傳加密流程

傳輸開始前會以亂數產生一組階段傳輸金鑰(session key)並向 server 要求一組階段識別碼(session ID)，以確保上傳資料之單一性，避免駭客利用偽造封包攻擊。然後系統由讀卡機讀取自己的 IC 卡內的使用者識別碼(user ID)並以 Server 端的公鑰加密。

將資料以 session key 加密(AES 演算法)，並將資料與 session key 合併計算其 SHA-256 雜湊值，然後將階段 id、使用者 id、階段金鑰、加密後資料及雜湊值傳遞給伺服器。伺服器以其私鑰解開使用者識別碼，與資料庫比對以確認使用者身份。無誤後以私鑰解開 session key，再解開加密資料，並比對雜湊值。

資料驗證正確後，伺服器亂數產生一組 128 bits AES 金鑰，寫入「金鑰資料庫」；利用該金鑰將使用者上傳內容加密，並與金鑰合併計算雜湊值後寫入「內容資料庫」。(金鑰資料庫與內容資料庫所關連之「金鑰序號」欄位、「金鑰」欄位及「使用者權限等級」欄位另行設計演算法加密之，以確保資料庫之安全性。)

6.3 下載加密過程



圖五、下載加密流程

使用者由伺服器下載資料之流程如圖五，先執行客戶端程式，由資料清單中選擇要下載閱讀的內容，然後系統由讀卡機讀取 IC 卡中的識別碼，將識別碼傳遞給伺服器做身份認證。伺服器連結至使用者資料庫確認並檢核其權限，如果符合則將該筆資料的加密金鑰以使用者的公鑰加密，並連同加密內容傳遞至客戶端。使用者收到內容後，系統讀取 IC 卡內的私密金鑰將加密金鑰解密，再利用加密金鑰將內容解密，並顯示解密後的內容(使用嵌入瀏覽器)。

7.系統軟體開發

本系統可分為四大部分：資料庫系統、網頁系統、管理系統及使用者系統。在資料庫及網頁系統方面使用了自由軟體，且具備跨平台功能，能配合原本的伺服器架構；而管理系統及使用者系統則配合多數人使用的的作業系統 — Windows，在視窗環境下開發，以方便使用。分別說明如下：

7.1 資料庫及網頁伺服器系統：

本系統開發所使用的軟體，在伺服器方面使用了 Apache【8】、PHP【9】、MySQL【10】]及 OpenSSL【11】等四種軟體，並架設於 Linux 作業系統上，其共同優勢是在非商業用途的使用上完全免費，而且跨平台，運作穩定，效能良好，都有非常龐大的使用者。

由於 PHP 並無內建 Rijndael(AES)加密演算法，因此使用 Libmcrypt 套件【12】。另外為了保護伺服器端網頁程式語言原始碼的安全，本系統使用了 Turck MMCache for PHP 套件【13】，不但可以將 PHP 原始碼加密，而且可以提升執行效率。 伺服器端所安裝軟體套件如表三。

表三、伺服器端安裝套件

用途	套件名稱及版本
網頁伺服器	Apache 1.3.29
伺服器端程式語言	PHP 4.3.4
資料庫(DBMS)	MySQL 3.23.49
公開金鑰加密—RSA	OpenSSL 0.9.x
對稱式加密—AES	Libmcrypt 2.5.7 [註]
程式原始碼加密	Turck MMCache

表四、管理及使用者系統開發套件

用途	套件名稱及版本
開發環境及編譯器	Borland C++ Builder 6.0
公開金鑰加密—RSA	OpenSSL 0.9.x
對稱式加密—AES	TSM Rijndael (AES) Component
IC 卡開發	IC 卡讀卡機(EZ100PU)開發 API

7.2 管理系統及使用者系統：

因目前一般使用者多數使用 Windows 作業系統，且 IC 卡讀卡機驅動程式及開發介面也多為 Windows 環境，因此本系統在管理及使用者系統部分採用 Windows 平台開發，所使用之開發環境為 Borland C++ Builder 6.0，並運用相關套件之 API 作為加解密及 IC 卡存取之用。如表四。

7.3 資料庫設計

本系統資料庫採用關連式資料庫設計，可分為三個資料庫：使用者資料庫金鑰資料庫與內容資料庫。主要欄位如下表五。(加底線者為主鍵)

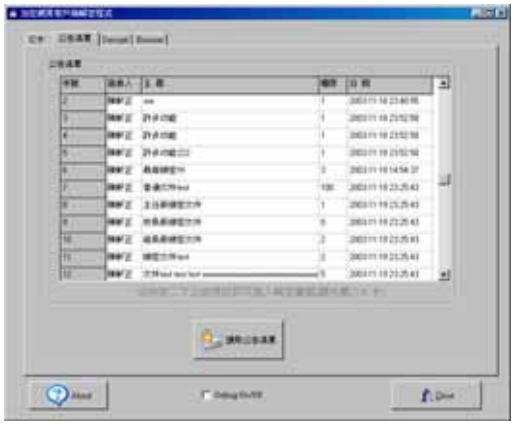
表五、資料庫主要欄位

使用者資料庫	金鑰資料庫	內容資料庫
識別碼(ID) 個人資料 公開金鑰 權限等級(加密) 雜湊值(上述四項)	金鑰序號(加密) 金鑰(密文)	內容序號 標題、日期、發佈者 內容(密文) 閱讀權限 雜湊值(明文+金鑰+權限) 金鑰序號

7.4 程式執行畫面

圖六、客戶端執行畫面 (IC 卡)

圖六為客戶端讀取 IC 卡時之畫面，畫面中顯示讀卡相關訊息。若 IC 卡讀取成功則進入瀏覽文件清單(圖七)，畫面上會顯示讀取該文件所需之權限，若權限足夠時則可將該內容解密，並呼叫瀏覽器(已嵌入程式內)觀看(圖八)。



圖七、客戶端執行畫面 (資料清單)

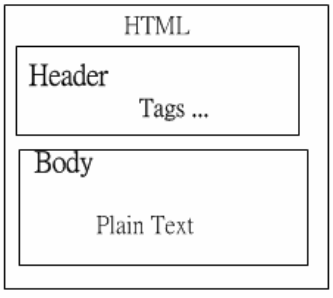
圖八、客戶端執行畫面 (解密後資料)

8.安全性探討

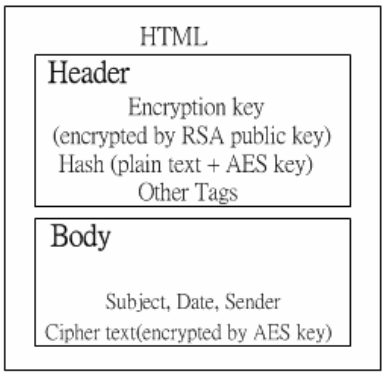
8.1 封包結構的改進

傳統的 HTTP 封包，主要分為檔頭(header)與主體(body)兩個部分，如下圖九，檔頭用來記錄文件相關資訊，主體則是網頁的內容，完全為明文。

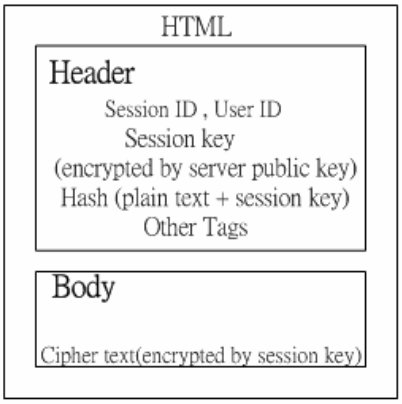
經本系統改良過之 HTTP 封包結構，如下圖十、十一，在檔頭的部分加入的加解密所需的相關資訊，含加密金鑰、雜湊值等，以 HTML 語言的標籤(tag)方式儲存，並以 BASE64 編碼。內容部分則為密文，經解密後恢復 HTML 格式，呼叫瀏覽器觀看。



圖九、傳統 HTTP 封包結構



圖十、改良後下載封包結構



圖十一、改良後上傳封包結構

8.2 可能遭受的駭客攻擊方式：

(1)偽裝成使用者入侵：駭客可能破解客戶端程式，偽裝成一般使用者直接向伺服器要資料，即使偽裝成功，所下載的資料也都是密文，解密之金鑰也經過 RSA 加密，非得使用者插入 IC 卡讀取 RSA 私密金鑰才可以解讀密文，攻擊無效。

(2)透過主機漏洞入侵主機：如果駭客透過漏洞、木馬等方式入侵了網頁主機，只能看到動態網頁的程式，無法竊取到機密資料。如果更高明的駭客入侵到了資料庫主機，甚至破解了資料庫密碼，也只能看到加密後的資料及金鑰，即使是解出了金鑰，也僅能解開其所對應的資料(每筆都有一組 AES 加密金鑰)，想要真正解開某一筆或是所有的密文，機率微乎其微。

(3)封包竊聽、擷取：雖然本系統使用 HTTP 通訊協定，但是其中的機密內容皆已加密，駭客所擷取到的封包只能看到檔頭等不重要的文字，無法解讀密文。

(4)偽造 IC 卡：IC 卡本身具備有身份認證的功能，只要密碼錯誤到一定次數(依照各種 IC 卡的設計而定)就會將卡片鎖住而無法使用，因此即使竊取到別人的 IC 卡，沒有密碼也是徒勞無功，被偽造的機率極低。

9.結論

校園安全網頁系統利用 HTTP 通訊協定，為校園網路提供了一個安全的資訊傳遞和交換平台，讓使用者能放心地在網頁主機上放置較機密的資料，不用擔心資料的外洩或是被不具權限的使用者看到。

本系統針對一般學校(尤其為中小學)的網路及主機環境來設計，依照前述的考量及規劃，使用了安全的加密演算法(RSA、AES)，安全且低成本的 IC 卡，並以自由軟體開發，不但軟體成本接近零，更有跨平台、相容性高等特性，而且易於安裝、管理和操作，因此適合在學校、政府機構甚至是企業使用。

在使用彈性方面，本系統在伺服器端以 PHP 為主體，可以搭配其他的網頁伺服軟體(如 IIS)及資料庫系統(如 Oracle, MS-SQL, Sybase…)，只要程式碼稍做改寫即可；而管理及使用者系統也只要加入其他讀卡機的 API，就可以輕易使用他牌讀卡機及 IC 卡。

如果要再提升系統的安全性，筆者提供以下兩點建議：

(1)網頁主機及資料庫系統安全性，可透過網路架構來加強，如使用內部封閉網路，安裝防火牆，或限制連線主機 IP 位址…等方式。
(2)使用加密性更佳的 PKI(Public Key Infrastructure)智慧卡(smart card)，內建 RSA 金鑰自行產生功能，保障使用者金鑰的安全。

目前 IC 卡的使用日漸普及，已經有許多學校陸續推行了校園 IC 卡。如果將本系統所使用到的個人資料(識別碼、金鑰)與原本的 IC 卡相結合，便可以提供

更安全、更多樣化的服務。

10.參考文獻

1.日本 IPA 安全中心(IPA/ISEC)

http://www.ipa.go.jp/security/crack-report/20020201/01a11.pdf

2.楊中皇，校園 IC 卡的資訊安全系統設計，第十屆國際資訊管理學術研討會論文集，1999 年 6 月，pp. 614-618。

3.賴溪松、葉育斌，資訊安全入門，2000 年，台北市，全華。

4. William Stallings, Cryptography and Network Security: Principles and Practice, 2nd edition, Prentice-Hall, Inc. 1999.。

5. National Institude of Standards and Technology, “Advanced Encryption Standard(AES),” Federal Information Processing Standard, FIPS PUB 197, November 26, 2001. http://csrc.nist.gov/publications//fips/fips197/fips-197.pdf

6. R. L. Rivest, A. Shamir, and L. Adleman, “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”, Communications of the ACM,Feb. 1978, Vol. 21, No. 2, pp. 120-126; “Cryptographic Communications System and Method”, U.S. Patend #4, 405, 829, Sep. 1983.

7. SHA-256， http://www.nist.gov/sha。

8. Apache， http://www.apache.org。

9. PHP， http://www.php.net。

10. MySQL， http://www.mysql.com。

11. OpenSSL， http://www.openssl.org。

12. Libmcrypt， http://mcrypt.sourceforge.net/。

13. Turck MMCache for PHP， http://turck-mmcache.sourceforge.net/。

14.劉良棟(譯)，IPSec-VPN 安全架構與實作，2002 年，美商麥格羅•希爾，台北市。

15.翁木龍、楊中皇、蔡瑞明，Linux 環境下以 AES 及 SHA-256 強化 VPN 的設計與實現，第十三屆國際資訊管理學術研討會，2002 年 5 月，pp. 541-548。