東海科學第三卷：45–55
90 年 7 月

植基於智慧卡的軟體保護機制

林祝興* 李鎮宇* 葉義雄**

摘 要

在我們所提出的機制中，使用者透過智慧卡與一個可以信任的金鑰資訊中心的應用，從網路上購買所需要的軟體後下載到使用者的電腦中只能安裝一次，並且若要再次安裝則一定要再次地購買。運用本機制可以加強對軟體的保護以避免遭受到重製與散播等攻擊。

關鍵詞：一次安裝方案、軟體保護、智慧卡、單向雜湊函數、對稱式密碼系統。

一、背景說明

由於網路技術的突飛猛進，讓越來越多的電腦可以藉此交換資訊以及分享系統資源。電腦的普及擴大了各種軟體的需求，從過去使用者必須到軟體銷售商店購買軟體，進步到透過網路可以直接向軟體公司訂購。預計未來不只是透過網路訂購，更能夠直接地經由寬頻網路下載所購買的軟體到使用者的電腦中。然而，當科技帶給人們無限便利的時候，智慧財產權(Intellectual Property)的觀念也逐漸地浮現，而所相關的保護措施也正就是我們在此所要討論的議題---軟體保護(Software Protection)。

許多的軟體開發者研發出一套商業軟體時，常常會在網路上提供一份試用版的共享軟體供使用者試用。假若使用者願意購買正式版軟體以長期使用，就必須對軟體開發者付費以取得相關的註冊資訊，通常所要輸入的註冊資訊多半是註冊序號以及其相對的密碼，或者是一個註冊程式。但是若有人將它們惡意散播出去，便可以讓許多的使用者不必經由付費程序而非法地使用正式版軟體。

另外最常見的軟體保護方法是將所有的安裝資訊燒錄在唯讀光碟片中(可能是 CD 或是 DVD)，然後與安裝序號(Serial Number)等等資訊包裝在一起銷售給顧客。這種的保護法只

* 東海大學資訊工程與科學系 E-mail: chlin@mail.thu.edu.tw
** 交通大學資訊工程所

需要複製那唯讀光碟片以及紀錄下所需的安裝序號等資訊依然可以非法地使用該軟體。而許多的研究人員則不斷地研究如何防止複製唯讀光碟片的技術。

在本論文中，基於以上所提到的軟體保護機制中，前者是幾乎難以抵抗非法惡意散播的攻擊，而後者是基於唯讀光碟片的保護技術。但是隨著燒錄技術的進步，透過 Burn Proof [1]或是 Just Link [2]的技術加上 Clone CD [4]、Disk Juggler [6] 或是 Blind Read/Write [7]等軟體的支援便可以突破幾乎所有唯讀光碟片的保護技術。一但軟體被破解並且廣為散播時，這對於軟體開發者而言無疑是一種非常大的打擊。

在 C. H. Lin 與 C. Y. Lee 的「A Software Protection Scheme on Internet: One-Time Install Solution」[10]所提出的方案中，需要以視覺秘密分享(Virtual Secret Sharing, VSS)[8]的技術輔助以確認使用者確實是該軟體的合法所有人，並且於購買後仍必須要在網路連線(On-Line)的狀態才能完成所需的步驟。但是使用 VSS 的技術並不是十分的合適，因為 VSS 的利用與保護的標的是影像資料(Image)，所以每一次使用者要購買軟體便需要找尋並利用一影像資料，這樣對於使用者而言是一種麻煩與負擔，本論文將針對以上的問題加以改良。本論文參酌了 W.H. Yang 與 S.P. Shieh 的「Password Authentication Schemes with Smart Cards」[18]中的智慧卡驗證技術，利用智慧卡的唯一性(Unique)取代了使用 VSS 的技術，並且在購買後下載了所購買的軟體後，所有的安裝過程都可於網路離線(Off-Line)狀態下完成。如此不僅讓使用者更方便購買軟體，而且可以減少軟體安裝步驟的困難度。

在本機制中，使用者以智慧卡從網路上購買所需要的軟體後下載到使用者的電腦中只能安裝一次，並且若要再次安裝則一定要再次地購買，以達到保護智慧財產的目標。本論文的內容安排如下：第二節將會介紹何謂「基於智慧卡的一次安裝方案」，第三、四、五節將介紹本機制中各個階段的詳細運作方法，最後，將說明與分析本機制的安全性。

二、系統架構

在本機制中所參與角色的共有三者，一是需要購買軟體的使用者(Buyer)，另一是出售軟體的網路商店(Merchant)，最後則是負責管理、發放智慧卡(Smart Card)的「金鑰資訊中心」。我們預設網路上有一個可以信任的「金鑰資訊中心」(Key Information Center, *KIC*)，負責管理、發放智慧卡給需要的使用者。當使用者要向 *KIC* 申請智慧卡時，必須以該使用

者於真實世界中的身分申請。本機制分為三個階段，註冊階段(Registration Phase)、購買階段(Purchasing Phase)與安裝階段(Installation Phase)。首先在註冊階段中，使用者必須先於 *KIC* 中註冊並取得一合法使用之智慧卡。其次在購買階段中，當使用者要在網路商店(Merchant)購買軟體時，必須透過電腦終端機(Terminal)直接連接之讀卡機(Smart Card Reader)使用智慧卡並輸入智慧卡密碼。當網路商店收到了顧客的購買要求以及透過由讀卡機所傳來的訊息後，便會請求 *KIC* 驗證該顧客是否為合法智慧卡使用者。假若確認無誤，則網路商店開始以一次安裝方案處理顧客所訂購的軟體而產生所需的一次安裝套件(One-Time Install Package)供顧客下載使用。最後在安裝階段中，使用者下載了所購買的套件之後，必須要立即進行安裝。執行該套件時會要求使用者透過讀卡機使用智慧卡以及輸入智慧卡密碼等資訊以供驗證使用者身分是否為該套件的原購買者。若所有的驗證皆無誤，則會正式地開始進行軟體安裝。

本機制基本上是植基於以下的幾點假設：

1. 存在可信賴的金鑰資訊中心(KIC)，負責管理、發放智慧卡給使用者。
2. 所有透過網路的訊息交換皆必須在安全通道(Secure Channel)中進行，例如 SSL/TLS [11, 17] …等。
3. 所有的程式安裝動作皆於使用者電腦中的一個暫時目錄中進行，並於程式安裝結束時自動刪除該目錄。

三、註冊階段

首先，使用者 U_i 必須以其真實世界中的身份於金鑰資訊中心(Key Information Center, *KIC*)申請註冊並取得一智慧卡(Smart Card)。當 U_i 於 *KIC* 註冊時，會得到一組帳號 ID_i 以及其自行所選用的智慧卡密碼 PW_i、一對公開金鑰加密系統(Public Key Cryptosystem)的金鑰 (sk_i, pk_i)和一數位憑證 $cert_i$。如同[8]中所描述的，*KIC* 會進行以下動作(圖示於 Figure 1)：

1. 產生兩個長度至少 512 位元的質數(Prime Number) p、q，使得 $n = p*q$。
2. 選擇出一質數 e 和另一整數 d，使滿足 $e*d \pmod{(p-1) * (q-1)} = 1$，而 e 與 d 分別為 *KIC* 之公開金鑰與私密金鑰。
3. 選擇一整數 $g \in GF(p)$與 $GF(q)$。

4. 計算出使用者的秘密資訊 $S_i \equiv ID_i^d (mod\ n)$。由 RSA [12]中得知 $ID_i \equiv S_i^e \pmod{n}$。
5. 產生使用者 U_i 的智慧卡卡號 CID_i，並且計算 $h_i \equiv g^{PW_i \bullet d} \pmod{n}$。
6. 將 n、e、g、ID_i、CID_i、S_i、h_i 、pk_i、sk_i 和 $cert_i$ 寫到智慧卡的記憶體中，並將智慧卡發給使用者 U_i。

經過了以上的步驟表示使用者 U_i 已經向 *KIC* 註冊成功並取得了卡號為 CID_i 的一個合法智慧卡。之後，使用者 U_i 準備利用該智慧卡向網路商店購買軟體，而詳細的步驟請參考下一章節「Purchasing Phase」。

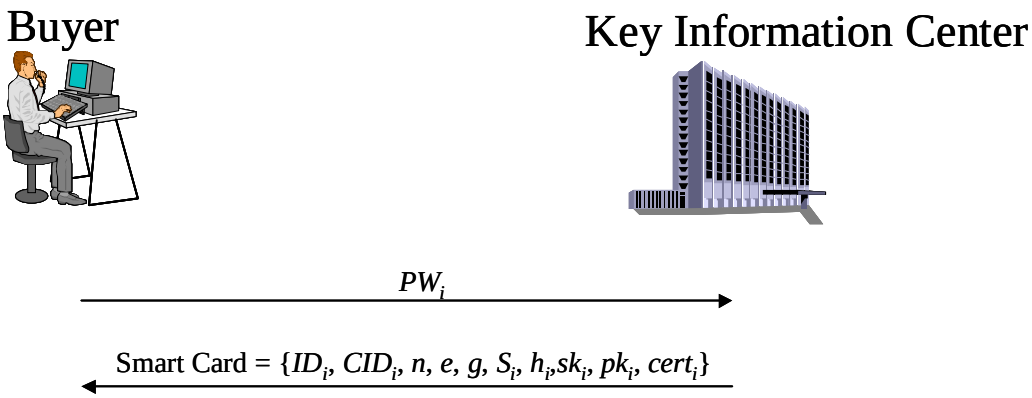

Figure 1.

四、購買階段

當使用者 U_i 欲於網路商店購買軟體時，便需要透過讀卡機使用智慧卡並將購買訂單與其數位簽章和自己的數位憑證等相關資訊一併送到該商店。當商店收到了顧客的要求訊息後，首先商店會驗證使用者的數位憑證與訂單的數位簽章以確認訂單與使用者的真實性，再透過網頁取得顧客端的電腦時戳(Time Stamp) $time_B$，並以單向雜湊函數(One-Way Hash Function)，例如 SHA-1 [3]、MD5 [5]…等，計算出一次安裝金鑰 Key_{OTIS}。商店利用對稱式加密系統(Asymmetric Cryptosystem)，例如 TripleDES [13]、Rijndael [14]…等，將原始安裝程式(Setup Files, *SF*)進行加密成 *ESF*，並且對 *ESF* 產生一數位簽章 $MSig_{ESF}$。將(*ESF*、ID_i、CID_i、$MSig_{ESF}$、$cert_M$)包裝上一個方便的使用者介面即成為一次安裝套件(One-Time Install

Package, *OTIP*)，其中 $cert_M$ 為商店的數位憑證。以下是詳細的步驟描述(圖示於 Figure 2)：

1. 使用者 U_i 於網站上產生一購買資訊 $Info_{order}$。讀卡機讀出智慧卡中的資料(ID_i、CID_i、S_i、h_i、n、e、sk_i、$cert_i$)以及使用者鍵入之 ID_i'。
2. 首先讀卡機會先驗證 $ID_i \stackrel{?}{=} ID_i'$，並計算出 $Y_i \equiv S_i^* h_i \pmod n$、$BSig_{Order} \leftarrow Sig_{sk_i}(Info_{order})$，其中 *Sig* 是公開金鑰簽章技術。最後將($ID_i$、$CID_i$、$Y_i$、$e$、$Info_{order}$、$BSig_{order}$、$cert_i$)透過電腦終端機送往商店。
3. 當商店收到了購買資訊後，便驗證使用者的數位憑證 $cert_i$ 以及 $Info_{order}$ 之數位簽章 $BSig_{order}$，並透過網頁取得 U_i 的電腦時戳 $time_B$。
4. 商店以單向雜湊函數 *f* 產生一次安裝金鑰 $Key_{OTIS} \leftarrow f(Y_i^e \text{ xor } time_B)$，並加密原始安裝程式為 *ESF*，即 $ESF \leftarrow E_{Key_{OTIS}}(SF)$，以及對 *ESF* 產生數位簽章 $MSig_{ESF} \leftarrow Sig_{sk_M}(ESF)$，$sk_M$ 為商店之私密金鑰。
5. 商店將(ESF、ID_i、CID_i、$MSig_{ESF}$、$cert_M$)包裝上一個方便的使用者介面即成為 *OTIP*。
6. 使用者 U_i 將 *OTIP* 下載到自己電腦上。

至此，使用者已經成功地從網路商店上購買了軟體。之後，即將開始安裝此軟體至使用者電腦中，有關安裝階段的詳細步驟請參考下一節「Installation Phase」。

五、安裝階段

在本階段中，使用者必須於下載後立即進行安裝，也就是執行 *OTIP*。*OTIP* 首先會以商店的數位憑證驗證 *ESF* 的數位簽章 $MSig_{ESF}$ 以確認 *ESF* 從商店下載到使用者的過程中並未遭到惡意竄改，並要求輸入使用者安裝資訊。因此使用者必須透過讀卡機使用智慧卡並鍵入帳號 ID_i'與密碼 PW_i'供讀卡機驗證該使用者是否為此智慧卡之合法所有人。假若驗證無誤，則讀卡機會開始計算 $X_i \equiv g^{PW_i'} \pmod n$，並將($n$、$X_i$、$ID_i$'、$CID_i$、$PW_i$')送回給 *OTIP*。*OTIP* 以檔案時戳 $time_B$'產生一次安裝金鑰 $Key_{OTIS} \leftarrow f((ID_i' * X_i) \text{ xor } time_B')$將 *ESF* 解密以得到 *SF*。最後用 *SF* 開始安裝軟體。*OTIP* 所有的計算與安裝動作皆於記憶體和暫時目錄中

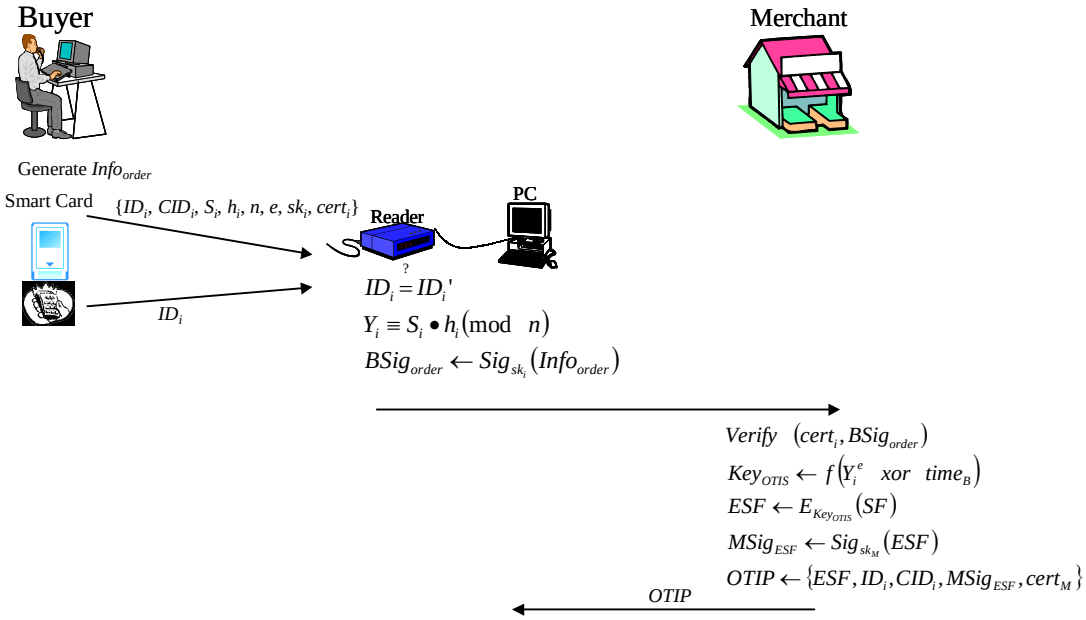

Figure 2.

進行，並於安裝結束後自行完整地刪除。而以下是詳細描述步驟說明(圖示於 Figure 3)：

1. 使用者 U_i 於上一階段中下載了 *OTIP* 後立即執行它。
2. *OTIP* 以 $cert_M$ 驗證 $MSig_{ESF}$，並要求使用者輸入安裝資訊。
3. 使用者透過讀卡機使用智慧卡，並且鍵入使用者帳號 ID_i'與密碼 PW_i'。
4. 讀卡機從智慧卡中獲得(ID_i、CID_i、n、e、g)，並驗證 $ID_i \stackrel{?}{=} ID_i'$。假如驗證失敗，則立即終止程式。否則計算 $X_i \equiv g^{PW_i'}(\text{mod } n)$，並將($n$、$X_i$、$ID_i$'、$CID_i$、$PW_i$')傳送給 *OTIP*。
5. *OTIP* 以本身的檔案時戳 $time_B$'計算一次安裝金鑰 $Key_{OTIS}' \leftarrow f((ID_i' * X_i)$ xor $time_B')$對 *ESF* 進行解密以取得 $SF \leftarrow D_{Key_{OTIS}'}(ESF)$。
6. *OTIP* 執行 *SF* 開始正式地安裝軟體。
7. *OTIP* 於安裝完成更新自己的檔案時戳並且刪除所有的暫時目錄。

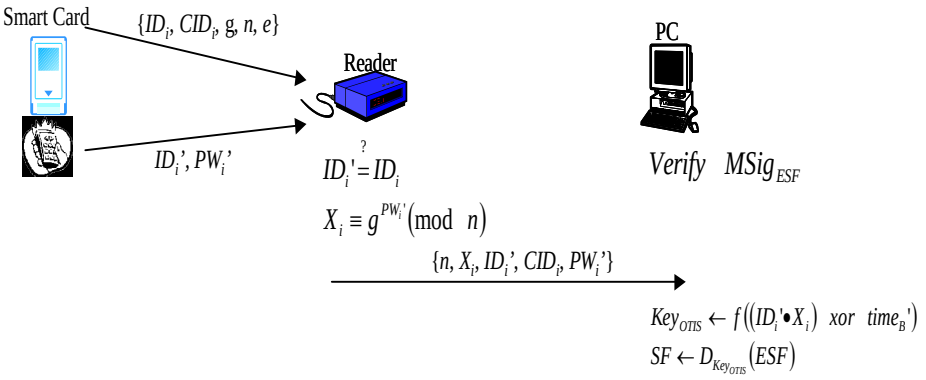

Figure 3.

六、安全分析與討論

在本論文我們所提出的機制中，所有的安全性大致上可以分為三部分討論：第一是有關智慧卡的安全性；第二是有關一次安裝套件的安全性；第三則是說明有關抵抗重送攻擊(Replay Attack)的能力。我們先假設一攻擊者 U_h $(U_h \neq U_i)$ 企圖攻擊本機制。關於智慧卡的安全性而言，假使 U_h 取得了 U_i 的智慧卡並獲得了 U_i 的 *OTIP* 時，由於智慧卡是一個 tamper-proof 的裝置，所以並不可能單純地從智慧卡中攫取到儲存於智慧卡內的所有資料。U_h 將 U_i 的智慧卡透過讀卡機讀取時，由於並不知道 ID_i 與 PW_i，因此並不能讀取到智慧卡內的資料。假使 U_h 攫取到智慧卡中的資訊時便可獲得 ID_i，而與若想要從 $h_i \equiv g^{PW_i \bullet d} \pmod{n}$ 中計算出 PW_i [8] 是不可得的，因為這是一個離散對數的問題(Discrete Logarithm Problem) [16]。當然 U_h 可以透過不斷地嘗試以獲得 PW_i，有關密碼的設定與安全的使用方法在此並不討論，請參考[15]。由於並不知道 PW_i，故不能計算出 $X_i \equiv g^{PW_i} \pmod{n}$，換言之，也不能產生所需要的一次安裝金鑰 Key_{OTIS}。另外，由於讀卡機是直接與個人電腦連接，在傳輸的過程中 U_h 並不可能攫取到 U_i 的資料。

有關一次安裝套件的安全性而言，先假設 U_h 透過讀卡機使用自己的智慧卡突破了讀卡機的驗證後，計算出 $X_h \equiv g^{PW_h} \pmod{n}$ 並將(n、X_h、ID_h、CID_h、PW_h)傳送給 *OTIP*，企圖計算出 $Key_{OTIS}'' \leftarrow f$ ((ID_h * X_h) xor $time_B''$)。因為 $ID_h \neq ID_i$ 以及 $X_h \neq X_i$，故 $Key_{OTIS}'' \neq Key_{OTIS}$ 便不能將 *ESP* 解密。另外，由於 *OTIP* 必須於下載之後立即安裝，因此在

一個適當的時間區段中 $time_B' = time_B$。若沒有立即安裝或是於安裝後再次地安裝，則 *OTIP* 的檔案時戳 $time_B'' \neq time_B$ 將不能產生正確的 Key_{OTIS}''。將 *OTIP* 複製到其他的電腦上，檔案時戳將會重新設定，那麼也如同上述狀況般不能使用。

我們假設一個最壞的狀況，若 U_h 經由監視、偷聽或是竊取而獲得了 U_i 的智慧卡並以特殊方法從中解析出 ID_i 與 PW_i，而且也將檔案時戳更改成跟 $time_B$ 相同，並計算出了 X_i 企圖產生 Key_{OTIS}'以對 *ESP* 作解密。在這樣的情況下的確是可以正確地獲得 *SF* 並進行安裝與使用。不過若由於 *OTIP* 指定要從讀卡機讀取資料並加以驗證，則單純只是解析出正確的資料亦不能使用。而突破的方法只可能是必須完全地複製一張智慧卡。我們可以想見的是只有擁有複製到完全相同的智慧卡使用者才可能進行軟體安裝。因此在這種情況下雖然並不能夠完全地防止非法使用者的使用，但是至少也限定在擁有「完全地複製的智慧卡」才可能使用的範圍。另外，*OTIP* 的程式安裝動作皆於使用者電腦中的一個暫時目錄中進行，並於程式結束時刪除以防止 Dumpster Diving Attack [9]。

重送攻擊法在本機制中，即是當商店 M_1 收到了從使用者 U_i 傳送來的購買資訊後再偽裝成使用者向另一商店 M_2 進行購買軟體的動作，這將可能會危害到使用者的信用、財產的安全。在此我們預設以下狀況成立，一個普通的數位訂單中一般都會有許多的欄位，其中有一個欄位是商店名稱(Merchant ID, *MID*)。該欄位係用以標明使用者所要購買標的物之銷售商店名稱。在本機制中 U_i 的購買資訊裡不僅有 $Info_{order}$，也同時有 U_i 的數位憑證 $cert_i$ 與使用者對 $Info_{order}$ 的數位簽章 $BSig_{order}$。因此若 M_1 要偽裝成 U_i 向 M_2 購買軟體時，若 M_1 將 U_i 所有購買資訊傳給 M_2 將不能成功。因為在 $Info_{order}$ 中的 *MID* 欄位上已經記載為 $M_1(\neq M_2)$ 了，所以不能成功。假若 M_1 更改了 $Info_{order}$ 上的 *MID* 欄位為 M_2，即是 $Info_{order}'$，也由於 M_1 並沒有 sk_i，所以不能再次對 $Info_{order}'$產生出 $BSig_{order}'$讓 M_2 驗證無誤，因此重送攻擊法並不可行。

七、結論

在本機制中最為可能突破的弱點其實是用來產生一次安裝套件的原始安裝程式(*SF*)。假如這個安裝程式所安裝完成的軟體並不能夠抵抗重製，則當合法使用者安裝完成所購買的軟體之後，便可以將它任意地重製到任何地方給任何人使用，那麼本機制也沒有辦法保

護到軟體的安全。所以一個程式設計師必須於程式開發階段時就要注意到該設計是否能夠抵抗重製的攻擊。

有一種軟體攻擊法---中斷攻擊法(Interrupt Attack)，係當一個安裝程式(*SF*)正在執行時(Run-Time)，從中找尋適當的中斷點(Interrupt Point)並插入中斷向量(Interrupt)以中斷安裝程式的執行，而且更改參數驗證與相關保護機制所需的程式碼，使得原本要進行密碼或相關硬體的驗證機制遭到破壞。此類之攻擊法以往多半係以組合語言(Assembly Language)來進行破解，然而現在的軟體絕大多數係以物件導向的視窗化程式語言(Object-Oriented Window-Based Language)所撰寫，導致一個安裝程式的執行複雜度大幅提昇，相對地也加強了中斷攻擊法的難度。這樣的一個軟體攻擊法就我們目前所知的範圍內，包括世界上幾個著名的軟體研發公司也都沒有任何有效的防範措施可以阻擋。而這也就是我們未來所需要再努力研究的目標。

隨著以硬體設計的方法來保護軟體的技術，如光碟防拷技術…等逐漸地進步時，期望未來若能夠也結合軟體設計的保護方法將會更能有效地提供軟體保護，而這也將會是我們所要努力的方向。

參考文獻

[1] SANYO Electric Co., Ltd., http://www.sannet.ne.jp/BURN-Proof/

[2] Ricoh Company, Ltd., http://www.ricoh.com/

[3] National Institute of Standards and Technology, NIST FIPS PUB 180-1, “Secure Hash Standard,” Apr 1995.

[4] Elaborate Bytes, http://www.elby.de/

[5] Rivest, R.L. (1992) “The MD5 Message Digest Algorithm,” RFC 1321, Apr.

[6] Padus, Inc., http://www.padus.com/

[7] BlindRead Co., http://www.blindread.com/

[8] Naor, M., and Shamir, A. (1995) “Visual Cryptography,” *Advances in Cryptology -EUROCRYPT’94, LNCS*, pp. 1-12, Springer-Verlag.

[9] Icove, D., Seger, K., and VonStorch, W. (1998) *Computer Crime*, O’Reilly & Associates, Inc..

[10] Lin, C.H., and Lee, C.Y. (2000) “A software Protection Scheme on Internet: One-Time

Install Solution", *Proceeding of 2000 Workshop on Internet & Distributed System*, Taiwan, pp.438-442.

[11] Frier, A., Karlton, P., and Kocher, P. (1996) "The SSL 3.0 Protocol," Netscape Communications Corp., Nov 18.

[12] Rivest, R.L., Shamir, A., and Adleman, L. (1978) "A method for obtaining digital signature and public-key cryptosystem," *Communications of the ACM* **21**(2), pp. 120-126.

[13] ANSI X9.17 (Revised), "American National Standard for Financial Institution Message Authentication (Wholesale)," American Bankers Association, 1985.

[14] Daeman, J., and Rijmen, V. (1998) "The Rijndael Block Cipher," *The First AES Candidate Conference*, August.

[15] "Information Security Program of the Year: Sample Policies and Procedures", *Computer Security Journal* **XV**(1) 1999, pp. 47-52.

[16] Adleman, L. (1979) "A subexponential algorithm for the discrete logarithm problem with applications to cryptography," *Proc. 20th IEEE Symp, Foundations of Computer Science*, pp. 55-60.

[17] Dierks, T., Certicom, Allen, C., and Certicom (1999) "The TLS Protocol Version 1.0", RFC 2246, January.

[18] Yang, W., and Shieh, S. (1999) "Password Authentication Schemes with Smart Cards", *Computer & Security* **18**(8), pp. 727-733.

A Software Protection Scheme Based On Smart Cards

Chu-Hsing Lin* Chen-Yu Lee* Yi-Shiung Yeh**

Abstract

We propose an innovative software protection scheme using smart card. In this scheme, a user can purchase software through his (or her) smart card and a trusted key information center on the Internet. After downloading, he (or she) just enables to install once. This scheme strengthens software protection and preventing from illegal copying and redistribution.

Keywords: One-Time Install Solution, Software Protection, Smart Card, One-Way Hash Function, Symmetric Cryptosystem.

* Laboratory of Information Security, Department of Computer Sciences and Information Engineering, Tunghai University, Taichung 407, TAIWAN. Email: chlin@mail.thu.edu.tw

** Dept. of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C