Wrapper-based 數位版權管理機制

廖鴻圖 世新大學資訊管理學系 htliaw@cc.shu.edu.tw

郭明煌 世新大學資訊管理學系 mhguo@cc.shu.edu.tw

林金龍 中央研究院資訊所 世新大學資訊管理學系 eddy@iis.sinica.edu.tw

陳貴青 中央研究院資訊所 ching64@iis.sinica.edu.tw

摘要

隨著網際網路的快速發展及電子數位產品不斷的出現，數位化的資訊得以更簡便的複製與傳遞。這些數位化的內容資料不僅可以輕易地在電腦上播放，更可以毫不受限的重製、傳播。但是數位內容在形式上有別於傳統有形的著作，它必須面臨許多不可避免的管理問題與挑戰。目前許多資訊擁有者最關心的議題，即是如何對數位內容的版權做保護管理。

現有的數位權利管理 (Digital Rights Management, DRM) 保護機制往往需要使用額外的數位內容播放器 (Digital Content Player) 來瀏覽受保的數位內容，為了簡化瀏覽數位內容時的處理流程及避免改變使用者習慣，在本研究提出的 Wrapper-based 的數位版權管理機制下，使用者將不須使用額外的數位內容播放器，即可瀏覽受保護的數位內容，且藉由 API HOOK 技術的支援以達到數位內容保護與管理的目的。

關鍵詞：數位權利管理技術、API HOOK、Wrapper、權利描述語言、智慧卡。

1. 前言

數位權利管理技術是一個結合硬體與軟體的存取機制[13]。它將數位內容設定存取權限，並與儲存媒體結合，使得數位內容在其生命週期內，從產生到消除都會受到保護。同時不管在其使用過程中是否有複製行為發生，仍然可以持續追蹤與管理數位內容之使用狀況。現存的數位版權管理系統，例如：微軟公司的 Media Player、蘋果電腦的 iTune、及 Adobe Systems 等，在各自的獨立發展下，各自擁有不同的管理機制與不同格式的數位內容。系統之間不能相容，且均須使用特定的播放工具，同時對使用者的限制繁多。

目前大部分的數位內容保護機制，在解決方法上都僅止於起始階段的身份認證，而且這些認證系統均須在網路上即時驗證。如此一來，數位內容的即時性與方便性的提供將受影響。因此，現今的數位版權管理機制，除了數位內容保護的需求外，還必須包含以下功能[2]：

(1) 使用者認證：

確認合法使用者的身份。未經合法授權的使者，就算獲得該數位檔案，也無法對此數位內容進行瀏覽或播放。

(2) 存取限制：

相同的數位內容檔案，可依授予的存取權限不同，制訂出個別的存取限制。例如部分數位內容不提供給一般使用者瀏覽，必須成會員或為特定群組，才可以瀏覽整個內容。

(3) 用途限制：

依授權內容不同，授予不同的用途限制。例如：限制複製、列印等操作功能。授權中未明白指名的其他操作動作就無法使用，除非另外取得合法授權。

(4) 時間限制：

依授權內容不同，授予不同的使用期間。例如：一個月、一週或指定期限等。使用時間一到，就無法再利用原先合法取得的內容，除非再次取得合法授權。

(5) 次數限制：

依授權內容不同，授予不同的使用次數。例如：3 次、10 次或指定次數等。使用次數用完，就無法再利用原先合法取得的內容，除非再次取得合法授權。

另外，還可根據以上功能需求組合出各種不同的授權內容，例如：授權給 Eddie 在一個月內只能瀏覽該數位內容十次，且無法列印或複製該數位內容。

本研究將整合數位內容及使用權限，並利用 API HOOK 技術達到數位內容保護之目的。同時為了不改變使用者習慣，本研究的方法將不須使用額外的數位內容播放器，即可讓使用者瀏覽受保護的數位物件。

本研究並整合了密碼學(Cryptography)、身份認證授權(Authentication)、智慧卡(Smart Card)及權利描述語言 (Rights Expression Language, REL)。密碼學技術將被應用在限制數位內容遭非法取得之保護，以避免不必要的數位資產損失；身份認證授權、整合智慧卡及權利描述語言可以讓使用者有不同的使用權限，並方便內容擁有者管理個別的數位內容權限。本研究的方法將提供數位內容權限管理達到完整性、持續性及安全性之訴求。

2. 相關理論與技術

本章節將針對本研究所使用之相關理論與技術做一個簡單的介紹。

2.1 API HOOK

API HOOK 的使用是讓 Windows 32 API 函數在被執行實際功能之前，可以讓程式設計者先做一些"預先的處理"，然後再回到函數的實際功能執行流程。如此流程的改變可以讓程式設計師監視或訂製某個 Windows 32 API 的使用，以實現一些特殊的功能。簡單的說 API HOOK 就是 Windows 系統控制權的攔截程式，利用此一方法即可攔截或監控 Windows 系統上所有的指令。

2.1.1 API HOOK 的原理

API HOOK 是在原函式中插入 Jmp 指令來將控制權轉移至自訂函式。Jmp 有好幾種二進位元指令，這裏採用 0xE9 為例，彙編格式為下：

Jmp XXXX　　// 其中 XXXX 為距離當前指令的偏移。

一般而言，系統函數都是以 DLL 封裝，當應用程式呼叫系統函數時，首先包含被呼叫函數的 DLL 會被載入到當前的程序空間中，被呼叫的系統函數的入口位址可以 GetProcAddress 函數進行獲取。當系統函數進行調用的時，首先把必要的資訊保存下來（包括參數和返回位址等資訊），然後轉跳到函數的入口位址繼續執行，而函數位址其實就是系統函數"可執行代碼"的開始位址。

2.1.2 Detours

Detours Tool[8] 是 Microsoft 為實作 API HOOK 介面所開發出來的程式，是一個可以在 x86 系統上攔截任何 Windows 32 API 的工具。Detours 能覆寫目的 Function，並能指定任何的 DLL 和資料片斷到 Windows 32 的程式中。

Detours 能無條件的轉移目標功能，圖. 2-1 顯示原有的系統邏輯流程和加入攔截程式後的系統邏輯流程，系統在未加入 Detours 時在系統在執行完目的 Function 後即返回啟始 Function，而在加入 Detours 後，系統在呼叫目的 Function 時會被 Jump 至 Detour 執行改寫的程式，當改寫的目的功能執行完成時，再呼叫 Trampoline 內存放的原始目的程式及 Jump 至 Target 中將流程完成。Detours 實際重寫了二個函式：Target Function 和 Matching Trampoline Function。

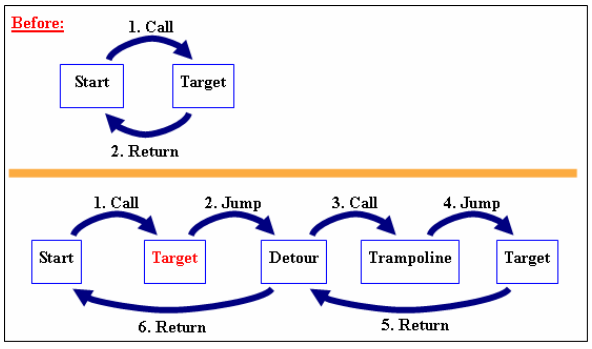

圖. 2-1 加入攔截程式後的系統邏輯流程[8]

2.2 Wrapper

以程式語言的領域而言，Wrapper 通常是指一個程式用來控制另一個程式的存取權。譬如第一個程式將第二個程式包裝起來，藉此可以加強程式的安全等級；此舉就像多加一道防線來避免有心人士的破壞。

Software Wrapper 是一種軟體包裹的技巧，其主要概念為在應用系統的介面層間插入一層中介軟體，用來實作必要的功能擴充，使應用程式本身的原始碼不需修改。此一技巧被使用在許多的應用系統上，如 TCP Wrapper 等。

近來有不少的研究嘗試在系統核心介面利用 Wrapper 的技術藉以擴充作業系統的功能[3]，在此我們稱之為系統軟體包裹技術(System Software Wrapper)，其架構如圖. 2-2 所示。作業系統核心在系統呼叫介面層的部份需要局部的修改，以允許插入適當的 Wrappers。當應用程式透過 System Call 向系統索求服務時，這些 Wrappers 即可執行必要的檢查或提供新增的功能和服務。因為系統呼叫介面(API)並未更改，應用系統本身並不需要任何的修改。

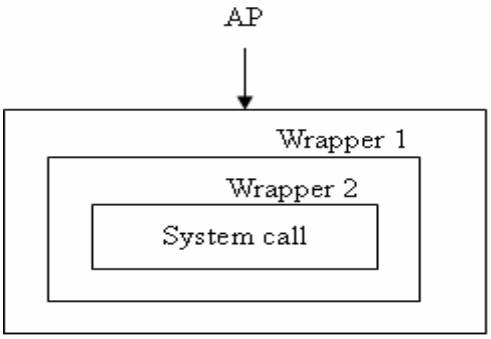

圖. 2-2 系統軟體包裹技術架構圖[3]

2.3 密碼學技術

加密 (Encrypt) 是將明文資料 (Plain Text) 轉變為密文資料 (Cipher Tex) 之處理過程；解密 (Decrypt) 是將密文資料反轉為明文資料之處理過程。在密碼學系統中，有許多的加、解密方法可以運用以達到秘密通訊的目的。在數位權利管理技術中通常也是以密碼學技術來限制數位內容的存取，並進一步達到複製保護(Copy Protection) 的目的。目前較常用的加密方式依照加解密金鑰設計的不同可分為兩種：對稱式加密 (Symmetric Encryption) 或是傳統加密 (Conventional Encryption)，以及非對稱式加密 (Asymmetric Encryption) 或是公開金鑰加密 (Public-key Encryption) [4][17]。

對稱式加解密系統為加密端與解密端均使用同一把金鑰 (即 Secret Key)，此種系統由於執行速度較非對稱式加解密系統快，因此常被用來保護數位物件本身。常見的對稱式加解密系統包含 DES[14]、AES[5]及 RC4[13] 等演算法。

非對稱式加解密系統每位使用者則會有一組金鑰對 (Key Pair)，即公開金鑰(Public Key) 與私密金鑰 (Private Key)。成對的金鑰對才可以相互加解密，即為加密端與解密端使用不同的金鑰。由於該系統所需的計算量較大，因此較適合用來保護重要的小量資料，例如用來保護對稱式加解密系統中的金鑰等。較著名的非對稱式加解密系統為RSA[10][12] 演算法及最近較為熱門的橢圓曲線密碼學 (ECC) [9][15]等。

2.4 權利描述語言

數位內容其包含實體面與權利面，實體面指的是數位內容的部分，包括取得、加值等方面；而權利面即指著作權管理的部分，也是大眾最為關心的，因為它牽涉到數位內容是否有效傳播的關鍵性問題。數位權利管理技術能在數位內容生命週期內做到事前防範的安全管理，其技術不但可以將數位內容加密，並可以設定使用者存取權限(如複製、列印權限、下載次數、到期日設定等)，以及設定追蹤控管使用行為等，讓數位內容在其生命週期內透過數位權利管理機制，並提供較完善的文件存取及使用政策，使得硬體與軟體在最佳狀態下相互結合，進而做到數位內容保密，解決數位內容隨意被洩漏、傳遞、複製、修改，以確保數位內容受到完整的保護與維護創作人的權利[1]。

因此，除了技術面的數位物件保護技術外，權利模型 (Rights Model) 的設計可以說是另一項重點，所謂的權利模型是用來表達有關使用者(User)和數位內容(Digital Content)的允許權利(Allowable Permissions)、限制(Constraints)、義務(Obligations)、和其他權利相關的資訊，權利模型包含了權利的種類 (使用者想要對數位內容進行什麼動作)及權利的屬性(允許的次數、允許的時間、授權使用的對象…等)。學者 Marj Stefik[16]曾對權利模型進行深入的研究，並定義了權利的三種基本類型：執行的權利 (Render Rights)、傳遞的權利 (Transport Rights)、衍生操作的權利(Derivative Work Rights)。

有了權利模型之後，數位物件的存取就可以依照該模型所訂定的規則運作。而權利模型的實作則通常借助權利描述語言 (Rights Expression Language, REL)工具，目前較常見的權利描述語言包含 eXtensible rights Markup Language (XrML)[6]與 Open Digital Rights Language(ODRL)[11]。

2.5 智慧卡

所謂的智慧卡(Smart Card)是一張與一般常用的信用卡或提款卡大小相同的塑膠卡片，兩者不同的地方在於智慧卡上多嵌入了一片 IC 晶片。這片 IC 晶片除了有記憶的功能之外，還具有運算、統計及處理資料的功能

除了上述提到的 IC 智慧卡與傳統磁卡之間的不同，我們可以將 IC 智慧卡的功能與特性歸納為安全性、方便性、應用多元化、離線作業功能、個人資訊管理等的特色。

2.5.1 智慧卡通信介面

PC/SC(Personal Computer/Smart Card) 支援 ISO7816-4[22]的基本指令集，界定了 IC 卡、讀卡機及作業系統的責任與分工，各家讀卡機廠商只要遵循 PC/SC 所定義之介面與方法開發驅動程式，即可結合到 Windows 或其他作業系統，應用程式只要透過單一標準介面與作業系統溝通，就可輕易操控各種讀卡機讀寫 IC 卡，其中應用協定資料單元(Application Protocol Data Units，APDU)[7]即為此智慧卡之通信介面。APDU 既制定了命令格式，也制定了回應格式。在卡的領域中，卡始終處在“主從”關係中的“從”的地位，即智慧卡只能等待讀卡器或終端向它發送 APDU，收到 APDU 後，智慧卡執行 APDU 中的命令，而後返回 APDU 回應。透過 APDU 命令和它的回應，卡就完成了與讀卡器或終端的通信。

3. Wrapper-based 數位版權管理機制

以下將針對本研究所提 Wrapper-based 數位版權管理機制做進一步的介紹。本研究架構主要包四種成員：內容提供者(Content Provider)、數位內容管理平台(Digital Content Management Site)、權限管控中心(Rights Service)及使用者(User)；圖. 3-1 為數位內容管理機制示意圖。

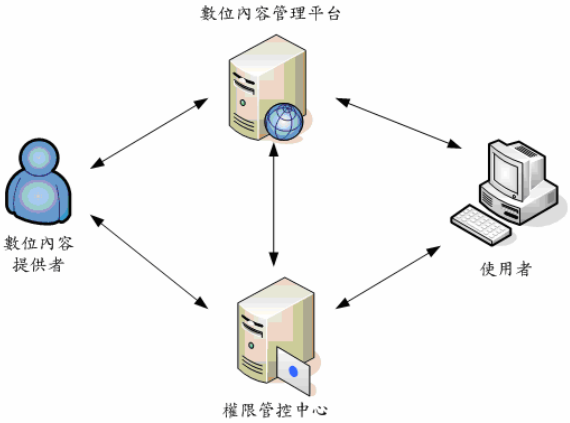

圖. 3-1 數位內容管理機制示意圖

3.1 系統架構

為了讓數位內容權限管理達到完整性、持續性及安全性之訴求，在本研究中設計的數位內容保護機制是以 Wrapper 架構來保護，並透 API Hook 之方式達到數位檔案之監控及播放，並以 XrML 權利描述語言來描述數位內容之使用權限。而為了提供更多元且安全之保護機制，本研究架構之管理介面將提供數位內容提供者設定更多元的保護機制，包含 Smart Card 認證機制、線上認證機制及硬體識別碼限制。

本研究架構中主要分為三個階段，分別為數位內容上傳階段、數位內容下載階段及數位內容播放階段，這三個階段分別對應系統架構之 DC (Digital

Content) Management、DC Package 及 DC Wrapper 三個功能模組，圖. 3-2 即為本研究架構圖。

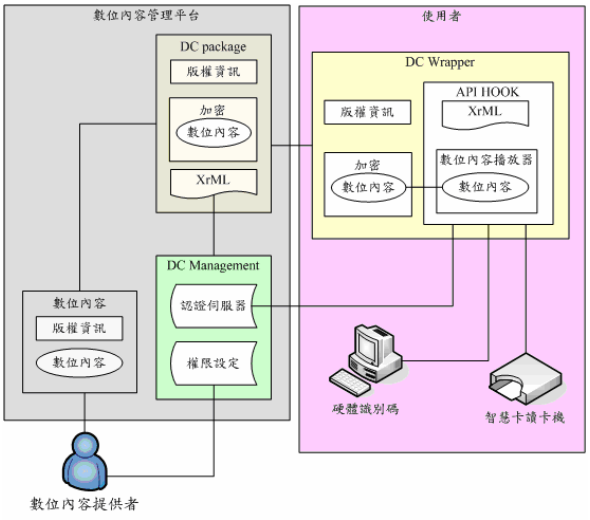

圖. 3-2 系統架構圖

3.1.1 DC Management

本研究將設計一個數位內容管理平台供數位內容提供者來管理所屬的存取權限，其內容包括可否列印、使用次數、使用時間...等。在此模組中將包含 DC 上傳管理及 DC 權限設定兩階段，以下將此二階段詳細說明。

(1) DC 上傳管理階段

數位內容提供者，將透過本機制所提供之數位內容管理平台，來上傳數位內容，且並利用此平台寫入此數位內容所屬之版權資訊，供日後追蹤版權及權利歸屬之用。

(2) DC 權限設定階段

在本階段中，數位內容提供者將透過此數位內容管理平台對所上傳之數位內容做使用權限之管理。因本研究採用 XrML 之權利描述語言，故數位內容提供者可以彈性且多樣的組合各種所須要設定的使用權限。可設定的使用權限如列印禁止、使用次數、使用時間、修改限制、On-Line 認證、Device 限制及使用 Smart Card 等。

3.1.2 DC Package

現行的 DRM 保護技術中，一般都是將數位內容加密，再將數位內容發行，任何人都有可能會得到此一數位檔案，但只有合法擁有解密私鑰的使用者才能將加密過的數位內容解開並播放。若有使用者透過非法管道取得解密私鑰，或者是將解密私鑰放在網路上傳播，則此受保護的數位檔案形同虛設，任何人都可輕易取得並播放瀏覽。為解決這個問題，本研究採用軟體 Package 的方式將數位內容、版權資訊、使用權限(XrML File)及相關的 API Hook 程式，依不同的存取權限重新包裝成一個新的數位檔案，供內容發行及傳遞。如此不管是合法或非法取得此數位檔案，都將只有合法的使用者才能對此數位檔案解開(解密)並瀏覽。

此功能模組中又細分為三個階段：DC 下載請求階段、DC 加密階段、及 DC Package 階段，詳細的系統流程如下所示：

(1) DC 下載請求階段

當使用者對數位內容管理平台提出下載請求時，系統則會依據數位內容所設定之使用權限產生一相對之 API Hook 程式 (本研究採用 Microsoft 所提供之 Windows 32 API 攔截模組：Detours Tool)。API Hook 程式的主要功能是產生內容提供者對此數位內容所設定之使用權限的對應系統監控模組，且此監控模組將在使用者播放數位內容時提供權限監控之目的。

(2) DC 加密階段

在此階段中將對此數位內容加密，以達到內容保護之原則。但此數位內容為一實體檔案，若使用安全性較佳之非對稱式加密法，則須耗費相當多的系統資源，故本研究利用對稱式加密法 RC4 來保護數位內容，並使用隨機取得的 128 位元的私密金鑰，套用對稱式密碼學演算法 RC4 來對此數位檔案加密。

(3) DC Package 階段

此階段主要的目的是要將加密過的數位內容檔案、內容提供者所設定之權限描述語言 XrML File、依權限產生之 API Hook 程式及版權資訊一併包裝成一個新的數位內容檔案，供使用者下載使用。

3.1.3 DC Wrapper

在一般的數位內容版權保護機制中，當要播放內容時，大都需要一個特殊(或指定)的播放工具，如 Apple iTunes、iPod 等。在本研究藉由 API Hook 即時監控的能力，在使用者利用原有播放工具開啟受保護之數位內容檔時，依所設定之使用權限及合法性即時解開此數位內容，讓使用者以原有播放工具觀賞數位內容檔。此功能模中細分為三個階段：API HOOK 階段、DC 解密階段、及 DC 播放階段。

(1) API HOOK 階段

當使用者點選此受保護的數位內容檔案時，系統將自發性的啟動 API HOOK 機制，在此機制中將有幾項主要的工作，如下所示：

1.取得 XrML File

從受保護的數位內容檔案中取出並解析所包含之使用權限描述(XrML)。

2.取得版權資訊

從受保護的數位內容檔案中取出所包含之版權資訊。

3.啟動 Detours Tool

從數位檔案中取出所包含之 API Hook 程式 Detours Tool，並於數位內容播放器中監控 Windows 32 API 的所有行為，再依據權限描述檔(XrML)所定義之使用權限執行監控與

限制。

(2) DC 解密階段

從受保護的數位內容檔案中取出所包含之加密過之數位內容，並經由密碼學對稱式演算法 RC4 及原始 128 位元的私密金鑰解開數位內容。

(3) DC 播放

將解密後之數位內容傳送至為 API Hook 所監控的數位內容播放器中播放。

3.2 系統流程

本研究提供了多元的驗證機制，以因應各種不同的需求。如傳統的線上認證機制，使用者在播放數位內容時，須先向認證伺服器確認身份，通過認證後，方可播放受保護的數位內容。若使用者是在一個沒有網路的環境下，則將面臨無法連線認證，而不能順利的播放數位內容。為避免此種情形，本研究使用硬體上唯一的識別碼 (Machine Code) 來對數位內容驗證，以確定該數位內容使用者是當初合法授權的使用者。每一個與數位內容相對應的使用權限，都會被這唯一的識別碼所指定。

這個方法雖好，但會造成數位內容被綁在使用者的電腦上。當使用者嘗試將合法取得的數位內容轉移至其他電腦時，會因為接收方硬體上的唯一的識別碼不符而無法使用該數位內容。為解決這個問題，本研究採用智慧卡(Smart Card)來提高數位內容之可攜性。以下將說明上述所提的線上認證機制、硬體上唯一的識別碼、及使用智慧卡驗證之系統流程。

3.2.1 認證伺服器驗證流程

在使用認證伺服器驗證之前，須假設使用者已經在認證伺服器上完成註冊程序，且與認證伺服器的通訊系經過可靠的安全通道(如 SSL)或已經過加密之動作。圖. 3-3 為受認證伺服器保護之下載流程，圖. 3-4 為此認證伺服器之驗證流程。

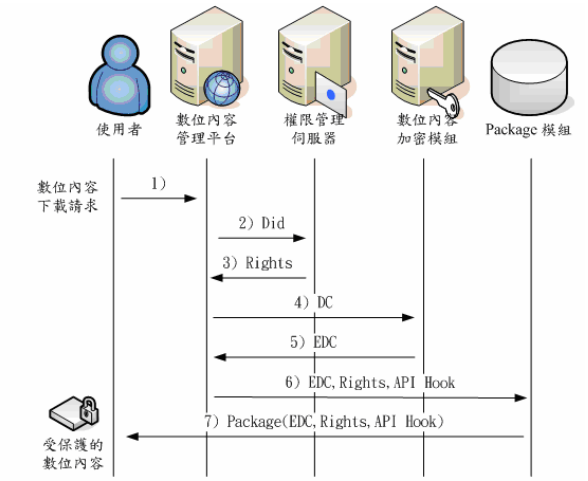

圖. 3-3 由認證伺服器保護之下載流程

認證伺服器保護之數位內容其詳細下載步驟如下：

(1) 使用者對數位內容管理平台請求下載數位內容。

(2) 數位內容管理平台將此數位內容之識別碼傳送到權限管理伺服器，並取得相關之使用權限。

(3) 權限管理伺服器將此內容之使用權限回傳給管理平台。

(4) 管理平台收到數位內容之使用權限後，即將此數位內容交由數位內容加密模組加密。

(5) 數位內容加密模組將加密後的數位內容傳回給管理平台。

(6) 此時，管理平台將加密過的數位內容、使用權限、及依權限產生之 API Hook 模組一併交由 Package 模組重新封裝。

(7) 將重新封裝後的數位內容提供給使用者下載。

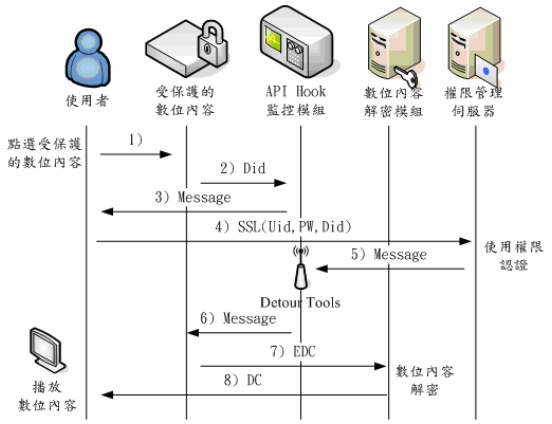

圖. 3-4 認證伺服器驗證流程

認證伺服器驗證流程的詳細步驟如下：

(1) 使用者點選受保護的數位內容檔案。

(2) 將此數位內容之識別碼 Did 傳送至 API HOOK 程序。

(3) 由 API HOOK 判斷此數位內容之驗證方式，並將結果傳回使用者。

(4) API HOOK 判斷此數位內容須由認證伺服器驗證，故將使用者之認證資訊(Uid,PW)及數位內容識別碼經由安全通道(如 SSL)傳送至認證伺服器(在此以權限管理伺服器代替)驗證。

(5) 認證伺服器驗證無誤後將結果傳回給 API HOOK。

(6) API HOOK 在收到驗證完成之結果後，便啟動 Detours Tool 且開始監控播放工具。

(7) API HOOK 從受保護的數位檔案取出加密過的數位內容，並傳送至解密伺服器中準備解開加密過的數位內容。

(8) 解密伺服器將解開加密的數位內容傳送至數位內容播放器中，供使用者觀賞。

3.2.2 硬體識別碼驗證流程

硬體識別碼驗證機制為使用者在下載數位內容時，系統將取得的使用者的硬體識別碼，且一併 Package 至數位檔案內。而在後續驗證時，系統將檢查檔案內之硬體識別碼與使用者系統之硬體識別碼的一致性。圖. 3-5 為受硬體識別碼保護之下載流程，圖. 3-6 顯示了硬體識別碼驗證之驗證流程。

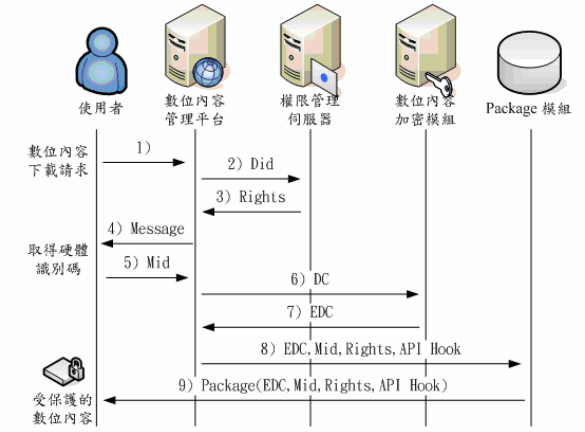

圖. 3-5 由硬體識別碼保護之下載流程

硬體識別碼保護之詳細下載步驟如下：

(1) 使用者對數位內容管理平台請求下載數位內容。

(2) 數位內容管理平台將此數位內容之識別碼傳送到權限管理伺服器，並取得相關之使用權限。

(3) 權限管理伺服器將此內容之使用權限回傳給管理平台。

(4) 管理平台收到數位內容之使用權限後，得知此數位內容須受使用者硬體識別碼保護，故傳送訊息至使用者端以取得使用者之硬體識別碼。

(5) 使用者將本身之硬體識別碼傳送給管理平台。

(6) 管理平台收到使用者之硬體識別碼，即將此數位內容交由數位內容加密模組加密。

(7) 數位內容加密模組將加密後的數位內容傳回給管理平台。

(8) 此時，管理平台將加密過的數位內容、使用權限、使用者之硬體識別碼、及依權限產生之 API Hook 模組一併交由 Package 模組重新封裝。

(9) 將重新封裝後的數位內容提供給使用者下載。

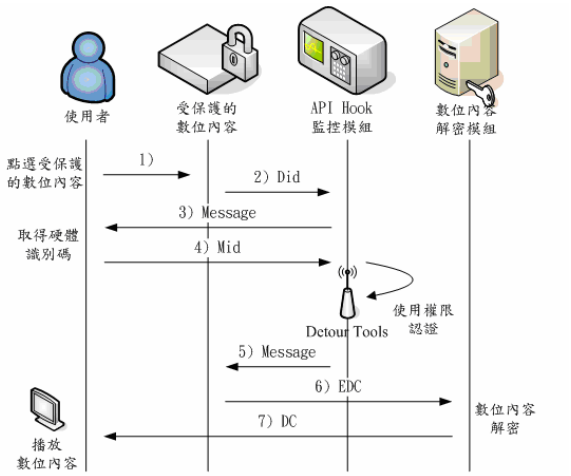

圖. 3-6 硬體識別碼驗證流程

硬體識別碼驗證流程的詳細步驟如下：

(1) 使用者點選受保護的數位內容檔案。

(2) 並將此數位內容之識別碼 Did 傳送至 API HOOK 程序。

(3) 由 API HOOK 判斷此數位內容之驗證方式，並將結果傳回使用者。

(4) 故將取得使用者所使用之硬體識別碼資訊 (Mid)。

(5) API HOOK 驗證數位檔案內之硬體識別碼與方取得之硬體識別碼，驗證無誤後，便啟動 Detours Tool 且開始監控播放工具。

(6) API HOOK 向受保護的數位檔案取出加密過的數位內容，並傳送至解密伺服器中。準備解開加密過的數位內容。

(7) 解密伺服器將加密過的數位內容解開並傳送至播放器中，供使用者觀賞。

3.2.3 智慧卡驗證流程

智慧卡可提高數位內容的可攜性，同時亦可利用智慧卡之唯一識別碼來做系統驗證之依據。智慧卡驗證機制初步與硬體識別碼驗證機制相同。當使用者在下載數位內容時，系統取得使用者之智慧卡的識別碼，並一併 Package 至數位檔案內，在驗證時，系統將檢查檔案內之智慧卡識別碼與使用者所使用之智慧卡識別碼之一致性。圖. 3-7 為受智慧卡保護之下載流程，圖. 3-8 顯示受智慧卡唯一識別碼保護之驗證流程。

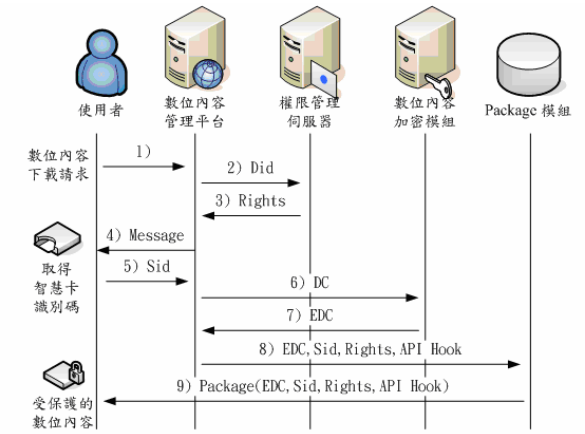

圖. 3-7 受智慧卡保護之下載流程

智慧卡保護之詳細下載步驟如下：

(1) 使用者對數位內容管理平台請求下載數位內容。

(2) 數位內容管理平台將此數位內容之識別碼傳送到權限管理伺服器，並取得相關之使用權限。

(3) 權限管理伺服器將此內容之使用權限回傳給管理平台。

(4) 管理平台收到數位內容之使用權限後，得知此數位內容須受 Smart Card 保護，故傳送訊息至使用者端以取得 Smart Card 之識別碼。

(5) 使用者將使用本身之讀卡機以取得 Smart Card 之識別碼，並傳送給管理平台。

(6) 管理平台收到使用者之 Smart Card 之識別碼，即將此數位內容交由數位內容加密模組加密。

(7) 數位內容加密模組將加密後的數位內容傳回給管理平台。

(8) 此時，管理平台將加密過的數位內容、使用權限、使用者之 Smart Card 識別碼、及依權限產生之 API Hook 模組一併交由 Package 模組重新封裝。

(9) 將重新封裝後的數位內容提供給使用者下載。

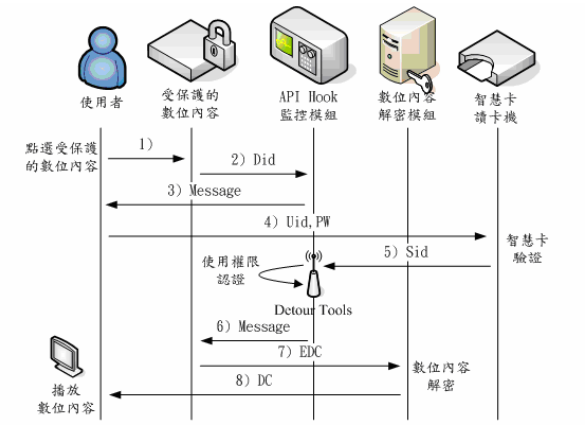

圖. 3-8 智慧卡 Smart Card 驗證流程

智慧卡驗證機制的詳細步驟如下：

(1) 使用者點選受保護的數位內容檔案。

(2) 將此數位內容之識別碼 Did 傳送至 API HOOK 程序。

(3) 由 API HOOK 判斷此數位內容之驗證方式，並將結果傳回使用者。

(4) 將使用者之智慧卡認證資訊(Uid,PW)傳送至讀卡機驗證。

(5) 讀卡機驗證其使用者為合法之使用者後，即將智慧卡之識別碼 Sid 回傳給 API HOOK。

(6) API HOOK 驗證數位檔案內之智慧卡識別碼與方取得之智慧卡識別碼，驗證無誤後，便啟動 Detours Tool 且開始監控播放工具。

(7) API HOOK 向受保護的數位檔案，取出加密過的數位內容，並傳送至解密伺服器中。準備解開加密過的數位內容。

(8) 解密伺服器將解開加密的數位內容傳送置至數位內容播放器，供使用者觀賞。

4. 結論與未來研究方向

數位內容的版權管理保護機制經常都是在各家獨立發展下，各自擁有不同的管理機制與不同格式的數位內容，各個系統之間均不能相容，而且均須使用特定的播放工具，對使用者的限制繁多，造成困擾。因此在本研究機制中制定了一個嚴謹且具彈性的數位內容保護機制，以及在不改變使用者原有的播放工具情況下做到數位內容保護之目的。

此外，本研究以 Wrapper 機制之技術，將數位內容及使用權限合併在一起，讓數位內容與使用權限保持一致，讓受保護的數位內容無法使用其它數位內容的使用權限控制，以達到數位內容與使用權限的一致性，進而達到數位內容保護機制之持續性。

其次，在一個完整的數位內容保護機制中應包含完整性、持續性及安全性之訴求；而且，目前智慧卡之普及率已趨向全民化，每一個人身上皆有一張以上之智慧卡，如全民健保卡、銀行/郵局提款卡、信用卡、會員卡…等。因此，本研究提出以智慧卡之機制達到更完整之授權認證機制，讓授權認證機制更具可攜性。

而且本研究並以硬體識別碼(Machine Code)方式來限制使用者設備之保護方式(如個人電腦)，讓重要的數位內容檔案僅可在下載時經過認證授權過的電腦上播放、瀏覽，即使將此受保護的數位內容複製，也無法在其它沒有經過認證授權之電腦設備上播放，以防止數位內容檔案遭非法的複製、傳播，使有心將數位內容非法傳遞之使用者，無法複製及傳遞，讓數位版權保護機制更具完整性與安全性。

在本研究架構中，其認證機制保留傳統的認證伺服器認證機制，以增加數位內容之便利性，內容提供者可依數位內容之重要性給予不同等級之認

證授權方式，可將較為不重要之數位內容設定為此認證伺服器授權，如此一來，使用者僅須連上網路，即可對此數位內容進行操作、瀏覽。

在數位內容皆可受到完整且安全的保護後，數位內容提供者才能在無安全的顧慮下，將內容開放出來給有需要的人，讓數位內容創作者可以創作出更多的數位內容，或使數位內容提供者更有意願將所收藏的典藏物品提出來數位化，並開放授權給一般大眾使用，這將使得數位內容市場更加的蓬勃發展，也可使得國家文化之傳承更具意義。

4.1.1 未來研究方向

在本研究機制中所提的權利描述語言 XrML，為一國際標準的權限交換格式，但在本研究構架中，所強調的為 Wrapper-based 之數位版權保護管理機制，所以並未提出其權利之交換程序、流程，故在接下來的研究中，可針對不同數位內容管理平台中之權利交換模型加以研究，以增加數位內容版權保護之通用性及一致性。

另外，在本研究中所使用之權限管控技術為微軟所提供之 Detours Tool，且並以此技術攔截 Windows 32 API 來保護數位內容，讓數位內容之使用權限受到限制，但此架構僅限於 Windows 系列的作業系；要如何將本研究機制延伸至其它作業系統中，如 Linux、MAC 等作業系統，甚至將此機制擴大至行動裝置、設備，如 PDA、Smart Phone 等，這將是未來研究的方向。

致謝

本研究承蒙國科會提供補助與相關研究所需之協助(計畫編號：NSC 94-2422-H-128-001)。

參考文獻

[1] 中研院數位典藏技術研發展組，”數位權利管理技術簡介”，數位典藏技術規範會議-數位內容保護技術研討會，頁 89-119，2005 年 3 月。

[2] 林昱仁、徐和謙、葉文熙，”數位內容版權發行管理機制”，第三屆數位典藏技術研討會，頁 297-304，2004 年。

[3] 馮立琪、黃盈源，”軟體包裹技術在系統安全上應用之探討”，資訊安全通訊 第七卷第二期，頁 12-21，2001 年 3 月。

[4] 賴溪松、韓亮、張真誠，”近代密碼學及其應用”，旗標出版股份有限公司，2004 年 1 月。

[5] AES, ”Announcing Request For Candidate Algorithm Nominations For The Advanced Encryption Standard”, National Institute of Standards and Technology, Vol. 62, No. 177, pp. 48051-48058, September 1997.

[6] ContentGuard, ”eXtensible rights Markup Language (XrML) 2.0 Specification”, http://www.xrml.org/get_XrML.asp

[7] Guthery Scott and Jurgensen Tim, ”Smart Card Developer's Kit”, Macmillan Computer Publishing, February 1998.

[8] Hunt Galen and Brubacher Doug, ”Detours: Binary Interception of Win32 Functions”, Published in Proceedings of the 3rd USENIX Windows NT Symposium. Seattle, WA, pp. 135-144, July 1999.

[9] Menezes Alfred, ”Elliptic Curve Public Key Cryptosystems”, Kluwer Academic Publishers, 1993.

[10] NIST, http://csrc.nist.gov/aes

[11] ODRL Initiative, ”Open Digital Rights Language Version 1.1 (Released 8 August 2002)”, http://odrl.net/

[12] Rivest Ronald, Shamir Adi, and Adleman Leonard, ”A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”, Communications of the ACM, Vol. 21, No. 2, pp. 120-126, February 1978.

[13] RSA Security, http://www.rsasecurity.com/rsalabs/node.asp?id=2009

[14] Schaefer Edward, ”A Simplified Data Encryption Standard Algorithm”, Cryptologia, pp. 77-84, 1996.

[15] Silverman Joseph and Tate John, ”Rational Points on Elliptic Curves”, Undergraduate Texts in Mathematics, Springer-Verlag, 1992.

[16] Stefik Mark, ”Letting Loose the Light: Igniting Commerce in Electronic Publication”, Internet dreams: Archetypes, Myths, and Metaphors, Cambridge, MA: MIT Press., pp. 219-253, 1996.

[17] William Stallings, ”Cryptography and Network Security Principles and Practices, 3/e”, Prentice Hall, 2002.