# 國立交通大學

## 資訊學院 資訊學程

## 碩 士 論 文

使用特徵碼資料庫技術來加強以 API Hook 為基礎之防側錄系統

Using Characteristic Code Database Approach to Prevent Screen Recording System by Using API Hook

研 究 生：林頌閔

指導教授：陳登吉　教授

中 華 民 國 九 十 八 年 一 月

# 使用特徵碼資料庫技術來加強以 API Hook 為基礎之防側錄系統

Using Characteristic Code Database Approach to Prevent Screen Recording System by Using API Hook


研 究 生：林頌閔　　Student：Sung-Min Lin

指導教授：陳登吉　　Advisor：Dr. Deng-Jyi Chen


國 立 交 通 大 學

資訊學院 資訊學程

碩 士 論 文

A Thesis

Submitted to College of Computer Science

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Computer Science

January 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年一月

# 使用特徵碼資料庫技術來加強以 API Hook 為基礎之防側錄系統

學生：林頌閔　　指導教授：陳登吉 博士

國立交通大學 資訊學院 資訊學程碩士班

## 摘要

為因應資訊時代的來臨，各行各業均不斷加速其數位化的腳步，許多資訊的獲得只在彈指之間，尤其在網際網路的環境下，若未經保護的資訊內容，如圖片、聲音、視訊檔案，很容易被下載與複製。因此，資訊取得的便利卻破壞了資訊的智慧財產權(IPR)，一旦盜版行為嚴重，即會抑制原創者的創作意願，使原創的發展退步，如何預防惡意的侵權行為又不影響多媒體產業的正常發展，已成為目前亟須正視的重要課題。

有鑑於以上問題的產生，原本智慧財產權的演進已經無法跟上數位時代的變遷，進而開始衍生出許多的數位內容保護技術，如數位版權管理系統(DRM Systems)、選擇性編碼(Selective Encryption)技術與可變轉碼(Scalable Encoding)、串流加密技術(Stream Cipher Systems)、防側錄技術等研究，目的都是在確保多媒體產業能朝向正面、網路生態以及兼顧權利之維護。

其中，防側錄技術也是目前數位內容保護的一項重要技術，目的是為了防止使用者在開啟加密課程之後，使用桌面或視窗錄影程式錄製課程內容並散佈的一項衍生技術。目前的作法大多是以 Windows API Hook 的方式，來監視系統或進程中有關擷取螢幕畫面的各種事件消息，進而攔截或阻擋此行為，這項技術在業界已經相當成熟且可以有效防止大部分使用軟體側錄行為的發生。

然而若單純只使用 Windows API Hook 方式，也會有其不足之處，例如無法有效防止遠端遙控軟體的錄製，亦或是被防毒軟體視為可疑行為等等。在本研究中，我將針對幾項防側錄模組之不足的部份，提出新的防側錄方式，以補強原本技術不足的部份。

# Using Characteristic Code Database Approach to Prevent Screen Recording System by Using API Hook

student：Sung-Min Lin Advisors：Dr. Deng-Jyi Chen

**Degree Program of Computer Science**

**National Chiao Tung University**

## ABSTRACT

In response to the advent of the information age, all walks of life continue to accelerate its digital footsteps and become easily to get it, especially in the internet environment. If without the protection of information, such as pictures, audio, video files, can be easily downloaded and copied. As a result, information was obtained to facilitate the destruction of intellectual property information (IPR). Once the serious acts of piracy, that would inhibit the creative will of the original, so that the original into the development of the backward. How to prevent malicious violations and don't affect the normal development of the multimedia industry, has become an important issue.

In view of the above problems, the evolution of intellectual property rights had been unable to keep up with changes in the digital era, which began to rise to the development of a number of digital content protection technology, such as DRM Systems, Selective Encryption, Scalable Encoding, Stream Cipher Systems and Screen Recording Preventing Technology. The purpose is to ensure that the multi-media industries in a positive direction, as well as the ecological balance between the rights of a network of maintenance

The Screen Recording Preventing Technology is an important technology currently to protect digital content. The purpose is to prevent users to use windows or desktop capture program to record the content of the course and spread it. Most of the common technology is to use 「Windows API Hook」 to surveillance systems or processes events related to the capture screen images and block or stop this behavior. This technology is already quite mature and can effectively prevent the use of most of the software of the recorded behavior.

However, if only a simple way to use 「Windows API Hook」, also has its shortcomings. For example, use recording software in the remote control can't be prevent effectively, or might be regard as suspicious behavior by anti-virus software, and so on. In this study, I will bring up a new method of Screen Recording Preventing for some shortcomings and tend to reinforce it.

# 致謝

首先感謝我的指導教授陳登吉老師，老師無論在課業、或研究上，每當我遇到問題與挫折時，都會耐心的指導及教誨，並給予我實質的建議與方向。即便在研究的過程中受到飽受打擊，甚至得重新開始，老師依舊對我有信心，也給了我更多的鼓勵，讓我能順利的走下去，我由衷地感謝老師，也祝福老師的身體安泰。

再來要感謝的是我的同窗好友們，能夠在課業上相互扶持，有問題大家一起討論解決；當研究上遇到瓶頸時，這些好同學能給我很多建議，讓我能夠完成這項研究。

最後我要感謝在背後默默支持我的家人，不斷地給予我正面的鼓勵，也相信我一定可以做得很好；朋友們也時常噓寒問暖，了解我的近況，分享我的心事，做個專職的傾聽者。有了支持我的家人與朋友，讓我更堅決自己所要走的路，並能義無反顧的做下去。

# 表 目 錄



# 圖 目 錄

# 目 錄

# 一、 緒論

## 1.1 研究動機

隨著網際網路的普及，原本傳統紙本的課程內容，都漸漸轉換成以數位內容的形式來呈現，例如多媒體互動學習課程、電子文件或網頁圖文等數位形式，並透過網際網路發佈，進而提供讀者可以直接線上閱讀觀看。這樣的學習方式，可以應用在許多的環境中，例如：會議室、辦公室、講堂、甚至家中個人電腦上都可以透過網路下載數位內容來做學習或是講授，改變了原本需要到現場學習的限制，讓學習方式更加多元化，因此課程內容數位化的程度不可同日而語。

隨著上述學習方式的變化，對於未經任何保護的數位資訊內容，如圖片、聲音、或視訊檔案，在網路的開放式環境下，就變得很容易被下載與複製，這是網際網路所帶來的便利性，但卻也同時觸犯了數位內容的智慧財產權(IPR)，一旦這種未經創作者許可的行為越來越氾濫，就會漸漸抑制了原創者的創作意願，進而使原創的發展退步，因此如何預防惡意的侵權行為且又不影響多媒體產業的正常發展，已經成為目前亟須正視的重要課題。

針對以上的議題，除了智慧財產權法的保護之外，這些年來在數位內容保護的技術開發部分(圖 1 所示)，如數位版權管理系統( DRM Systems )、選擇性編碼( Selective Encryption )技術與可變轉碼( Scalable Encoding )、串流加密技術( Stream Cipher Systems )、防側錄技術等研究，在維護多媒體智財權上都佔有重要地位，而這些技術的目的都是在確保多媒體產業能朝向正面的方展、以及兼顧創作者權利之維護。

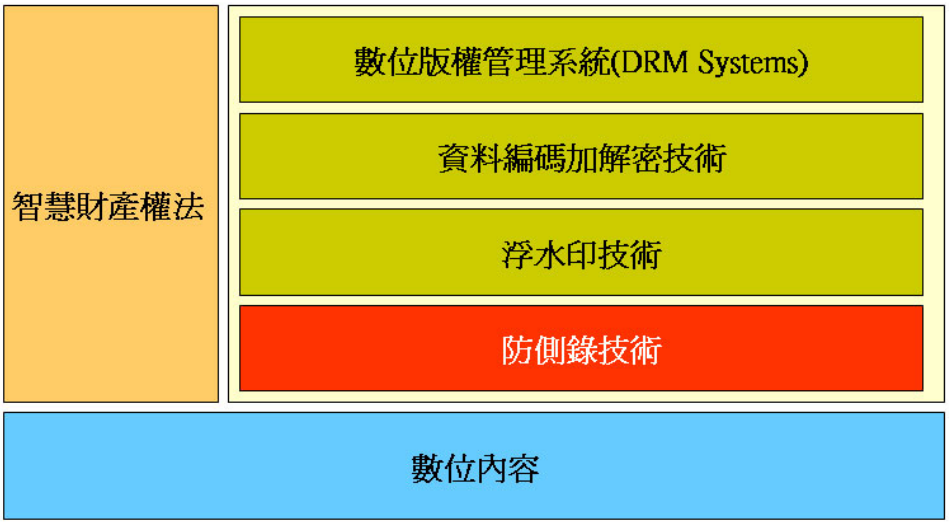



圖 1 數位內容保護技術架構

在數位內容保護技術中，大部份的技術都是針對檔案本身作進一步的處理，例如使用加解密演算法將檔案編碼，並在 DRM 數位版權管理系統中管理加解密金鑰，或是在檔案內容加入可視或不可視的浮水印驗證等等。另外也有針對當課程已通過正常認證程序開始播放之後的保護機制，如防測錄技術，這項技術主要是防止閱讀端使用螢幕或視窗錄影軟體直接錄製播放內容而衍生的技術，如圖 2 所示。


圖 2 側錄播放內容示意圖

防側錄技術的處理方式並非針對檔案本身，而是阻擋閱讀端開啟此課程後，在作業系統中執行與錄製相關行為的軟體，避免將錄製好的課程內容，再次轉為未保護的檔案而散佈。目前，防側錄的作法都是使用 Windows API Hook[6] 的技術，主要可用來監視系統中處理程序的 API 內容，是否有擷取螢幕畫面的行為事件，若發現有不允許的行為，則會進一步攔截或阻擋此事件發生。這項技術在業界已經相當成熟，並且效果不錯，可用來阻擋大部分使用錄影程式側錄的行為發生。

但是，若只是依靠上述的技術，是否就已經可以滿足所有的防側錄需求呢? 在實際應用中發現，若只使用 Windows API Hook 來偵測並阻擋側錄行為，還是會有其不足的地方，例如無法有效阻擋遠端遙控的軟體錄製、或是可能被部分防毒軟體視為可疑行為而造成課程無法正常觀看等問題。另外，若課程需在 Linux 系統上觀看，則原本以 Windows API Hook 的技術並無法相容於 Linux 中使用。

因此，在本研究中，我將針對上述的問題逐一做詳細的說明，並提出一個新的防側錄方法，除了可改善並且補強原本以 API Hook 技術不足的部份，同時也考慮到系統架構的可移植性，使其可以相容於 Linux 環境。

## 1.2 研究目標

本研究共有兩大目標:「加強防側錄系統」與「Linux 系統可移植性的評估」。

在「加強防側錄系統」的部分，除了滿足在 Windows 中阻檔錄影程式的功能外，同時也必須可以阻擋使用遠端遙控軟體的錄影行為，另外，為避免被防毒軟體視為侵害系統的可疑行為，本系統將會以資料庫比對的方式來做深入的研究與探討，找出一適合的比對演算法，與系統中正在執行的處理程序作比對，若發現疑似側錄程式，則停止播放並隱藏課程內容，以達到防側錄效果。在實作的部份，則選擇網頁格式的數位課程作為主要的保護對象，因此會將系統功能實作為網頁的外掛模組，如 Internet Explorer 的 ActiveX，以及 FireFox 的 Plugin。

而在「Linux 系統可移植性的評估」部分，則會從本研究的系統架構內容以及系統中實做的方式，在 Windows 以及 Linux 中互相比較，最後以評估的方式，驗證本模組的系統架構可以在 Linux 中開發運行，以證明本研究提出的方法具有可移植性，改善傳統使用 API Hook 技術無法相容於 Linux 上的問題。

## 1.3 研究方法與步驟

由於現有的加密課程的類型不計其數，因此本研究將使用智勝國際科技公司所研發的 BWDRM Flash 加密課程網頁作為主要的實做以及實驗對象，並說明防側錄行為的機制是如何運作。研究的方法與步驟大略分為下列幾點：

(a). 了解與評估現存系統

針對課程防側錄保護的運作流程，著手尋找現有已完成的 DRM 文件保護系統，了解其防側錄的使用方法與運作過程，經整理分析之後，針對缺點部分，衍生出本研究的設計方式與運作機制。

(b). 了解保護格式

針對本系統的實驗對象 BWDRM Flash 加密課程網頁，研究其運作方式以及檔案格式，並研究與規劃適合的外掛模組。

(c). 分析系統需求

依據本研究所規劃的課程防側錄保護運作機制，分析所需要的功能模組，及模組間的溝通方式；接著分析智勝國際科技公司的BWDRM Flash 轉檔工具對資料的處理方式，設計可與 BWDRM Flash 溝通的功能模組。

(d). 架構與流程的說明

接著介紹本研究的系統架構，以及課程防側錄保護的系統運作流程。

(e). 系統實做

依照之前分析的系統需求與實驗的對象，建置開發環境與工具，並著手實作出各個模組，進而完成整個系統。另外，為了證明本系統可有效阻擋側錄軟體錄製課程內容，因此將建立一資料庫，蒐集目前在市面上提供之側錄軟體，以提供測試樣本，其中也包含有遠端操控功能之軟體，用來驗證本研究模組是否可有效阻擋該軟體的使用。

最後，在Linux的驗證部分，將以評估的方式，詳細說明為什麼本系統之架構可以相容於 Linux 系統。

(f). 應用實例與結論

最後會逐步說明如何在產出 BWDRM Flash 課程的同時，嵌入本研究所擬訂的防側錄保護機制，使用測錄軟體實際測試並驗證其阻擋的效果。

在最後一個章節中，則會總結本研究在課程保護規劃上的彈性，對課程原始作者所提供的安全性、便利性，並說明本系統與一般防測錄模組的不同，以及改善與補強原本不足的部份。

## 1.4 相關名詞解釋與研究範圍

本研究根據1.2節所提出的研究目標，提供相關名詞及參考範圍，茲描述如下：

(a). API Hook：[6]

在windows系統下編程，應該都會接觸到API函數的使用，常用的API函數大概有2000個左右。透過 Hook 技術可動態連接到需要修改的API函數入口點，改變它的位址指向新的自定義的函數。

(b). BWDRM Flash 檔案格式：[8]

為智勝國際科技公司所開發，針對 Flash 格式的一種加密技術，並可搭配 DRM 的數位管理機制。

## 1.5 章節說明

本論文共分為七個章節，以下簡單說明各章節內容：

第一章，首先介紹數位內容保護技術，並說明目前防側錄模組的發展與其重要性，進一步從目前技術的問題中導引出本研究的動機與目標，簡略介紹本研究的方法與步驟。

第二章，介紹目前的防側錄模組技術，如 TrustView、Osafe Mirage、W&Jsoft 等公司所研發的防側錄模組，並以 TrustView 說明目前的主要作法，並進一步引出目前在實際應用上所遇到的問題，以及解決的方式。

第三章，根據第二章提出的問題與解決方法，衍生本研究所提出防側錄模組的運作機制，並規劃出所需的功能模組，最後開始著手分析各模組的功能、以及各模組間的溝通方式。

第四章，介紹本研究的系統架構，以及說明在此架構之下，系統的運作流程是如何進行。

第五章，詳細介紹各個模組的設計概念、與實作方式，並且說明如何與現有的 DRM server 溝通，達到對文件的控管；另外，詳述如何與智勝國際科技公司的 BWDRM Flash 作配合，完成對課程內容的防側錄保護。

第六章，將實作出來的系統，透過畫面擷取、與文字解說，逐步說明系統如何從一份未受保護的 Flash 網頁，轉換成一份受 DRM 保護的加密內容、並同時嵌入本研究之防側錄模組，最後實際在閱讀端模擬使用錄影軟體以及遠端遙控軟體，驗證課程內容是否可以有效被防止錄製。

第七章，本章為總結，說明本研究的研究結果，所達成的研究目的，以及未來展望。

# 二、 相關研究與探討

在 Windows 作業系統中，大多數可用來錄製桌面或視窗畫面的軟體，都是透過 Windows API 技術完成，進而再儲存成視訊或圖片格式等檔案，因此為了能夠即時偵測出 Windows API 的行為，所以都會使用 Hook 技術來監控執行程式裡所有運作中的 API，若發現使用與擷取畫面相關的 API 時，則會立即阻擋該行為的結果，例如將擷取的圖像內容清空，並修改為自訂的圖像內容。目前有許多的數位內容保護控管公司，例如：Osafe Mirage[10]、TrustView[11]、W&Jsoft[12] 等，都是採用此項技術來發展防側錄功能，以保護這些經過權限控管的文件或課程內容。

在接下來的小節中，我將使用 TrustView 所提供的受數位版權保護(DRM)文件，針對其防側錄的流程做說明，並歸納出使用上的問題，最後，再對本研究所使用的相關技術作詳細的介紹與探討。

## 2.1 現有防側錄模組的運作方式

在此一章節中，我將展示由 TrustView 所提供的受數位版權保護文件，此文件為一份經過編碼的 Word 檔案，因此無法直接使用 Word 開啟，需要再安裝一外掛模組才可解密文件。接下來我將針對防側錄的流程逐步做說明，在說明中我將會安裝 TrustView 的外掛模組 TrustView For Office and PDF 3.5.0，並使用 Microsoft Office Word 2003 開啟受保護的文件，最後再歸納使用時所發現的問題。

Step1、首先開啟 TrustView 所提供的受數位版權保護 word 文件，在尚未安裝任何 TrustView 所提供之外掛模組前，該份文件是無法正常觀看且出現圖 3 的提示訊息，訊息指示須先安裝 TrustView 的外掛模組才可正常觀看此文件。

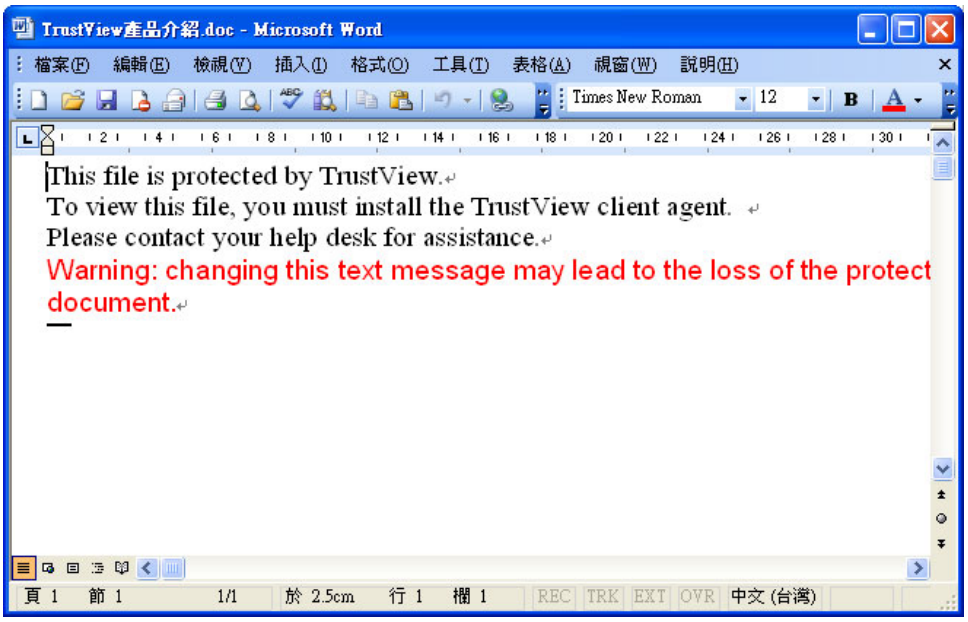



圖 3 開啟受 TrustView 保護的 Word 文件

Step2、開始安裝 TrustView 外掛模組，圖 4 為安裝畫面。


圖 4 TrustView 外掛模組安裝畫面

Step3、安裝完畢之後，再次開啟該份文件，此時可檢視 Microsoft Word 視窗的工具列中是否出現 TrustView 的外掛模組項目，若有即代表已經成功安裝外掛模組，如圖 5 所示。此時可測試鍵盤的 Print Screen 鍵，此鍵的功能是用來擷取整個電腦螢幕的畫面，再開啟附屬應用程式中的小畫家軟體，並使用貼上的功能，將擷取的畫面貼上，但發現此動作已經失效且無法成功截取畫面。

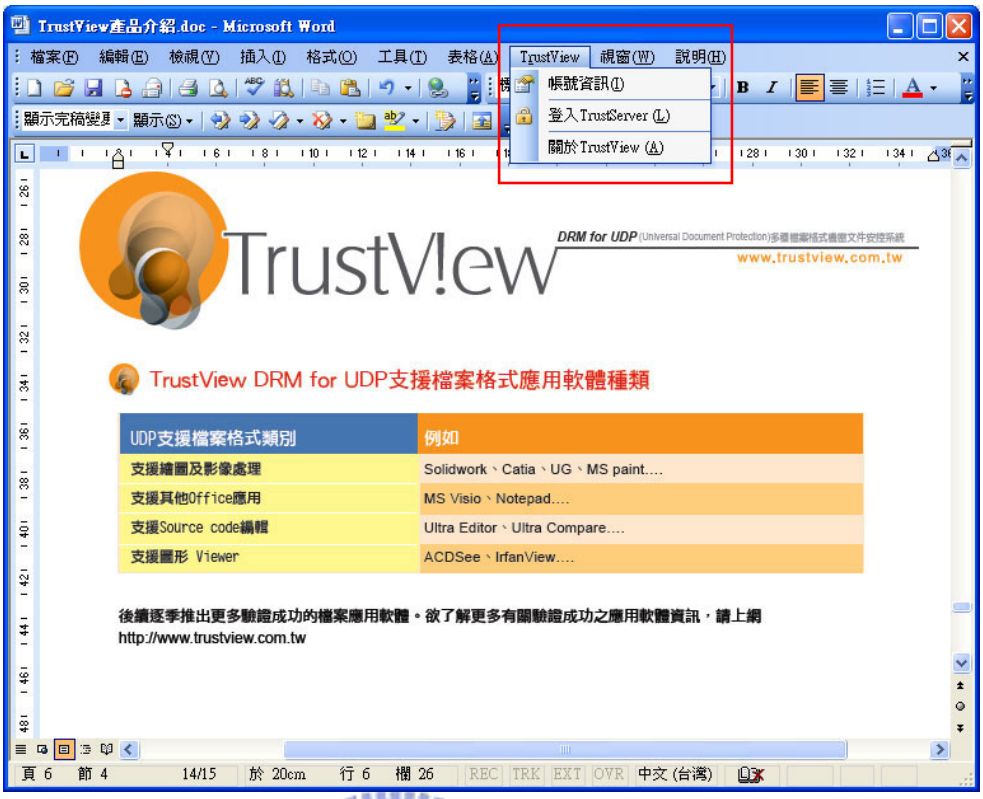



圖 5 開啟受保護的 Word 文件

Step4、開啟具有擷取桌面及視窗畫面功能的錄影軟體，用來測試 TrustView 外掛防側錄模組的效果。此說明中我選擇使用 MWSnap 3.0.0.74 擷取畫面軟體，軟體介面如圖 6 所示。

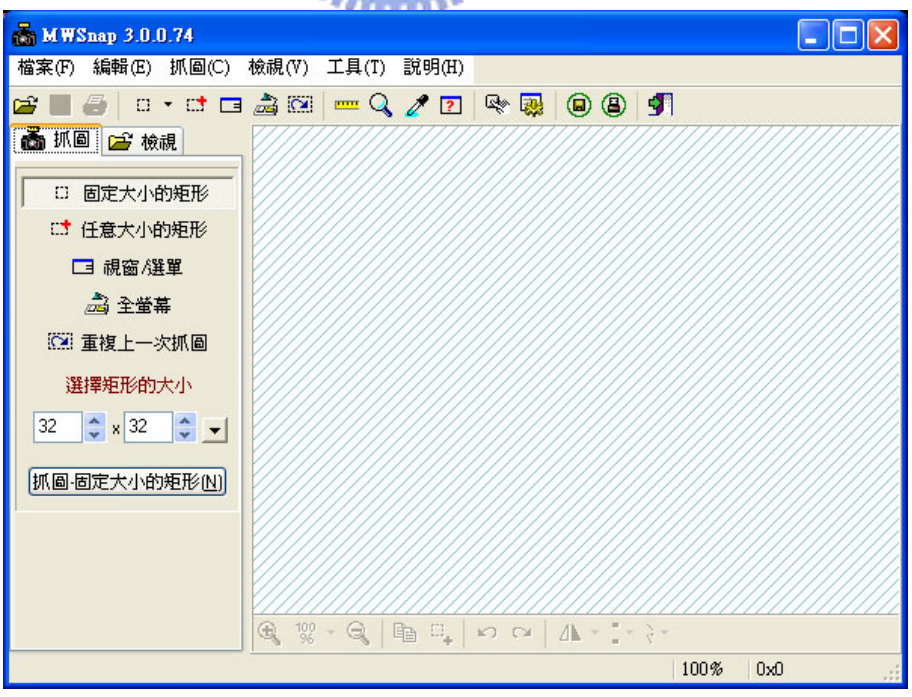



圖 6 開啟 MWSnap 3.0.0.74 軟體

Step5、為了比較受保護文件的防測錄效果，因此我先開啟一份未受任何保護的 Word 文件，接著執行 MWSnap 3.0.0.74 的抓取視窗畫面功能。此時可成功抓下該 Word 文件的內容，如圖 7 所示。

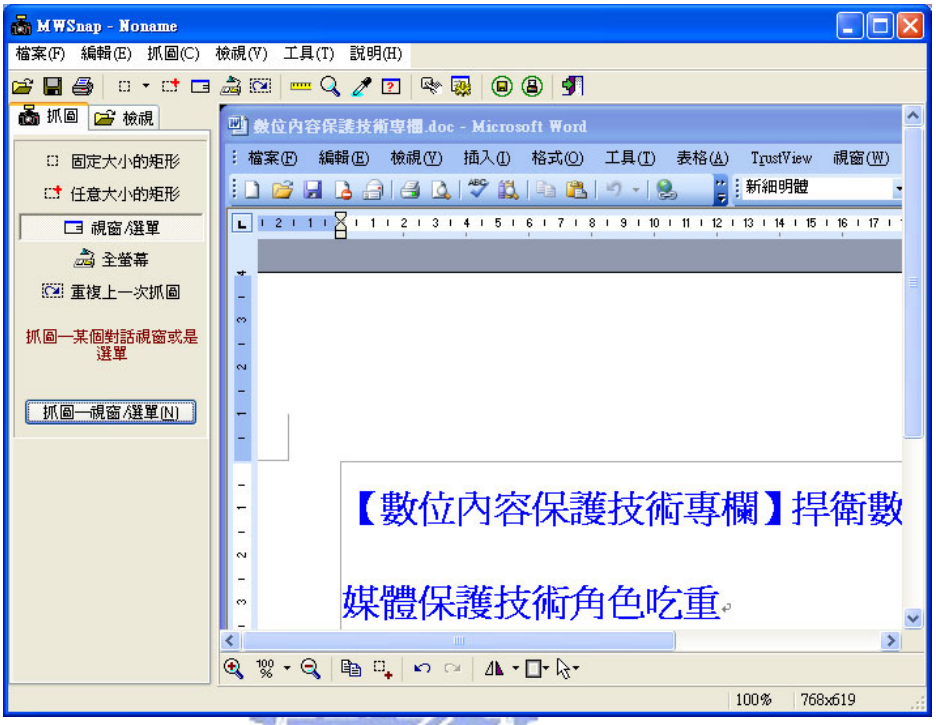



圖 7 未受 TrustView 保護的 Word 文件

Step6、再開啟受 TrustView 保護的 Word 文件，使用相同的測錄軟體以及相同的操作步驟，來抓取文件內容的畫面。結果發現在執行截錄畫面的動作時，已經被 TrustView 視為不允許的行為，並且修改了擷取到的畫面，結果如圖 8 所示，達到了防側錄所需要的效果。

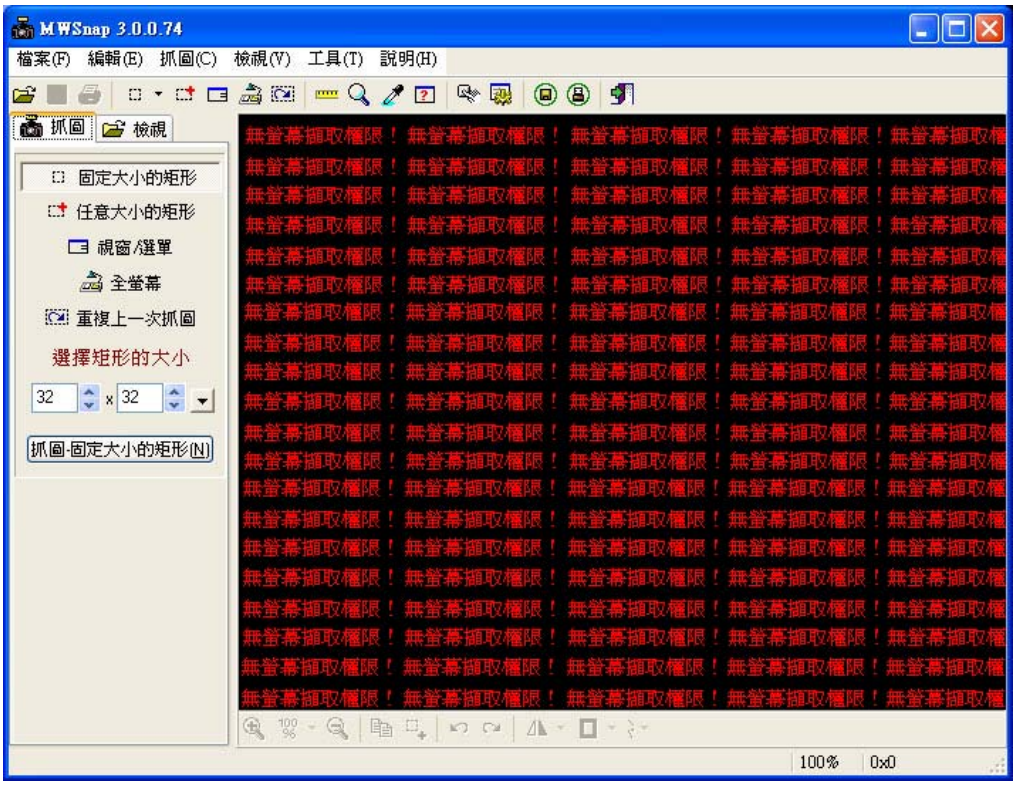



圖 8 受 TrustView 保護的 Word 文件

TrustView 所提供的防測錄模組，僅針對受 DRM 數位版權管控的 Word 文件，因此可以大致猜測出，該防測錄模組的系統架構是以【外掛】的方式依附在該文件的瀏覽器中，例如 Microsoft Word，也就是說當使用 Word 開啟受保護文件的時候，也會同時驅動並執行防測錄模組。

所謂的【外掛】技術，即是非原軟體所提供的功能，而是由第三方開發有特殊目的的功能模組，此模組無法獨立執行，而是在被依附軟體啟動的同時，再將此功能模組載入該軟體的執行程序中。

在 Windows 作業系統中，【外掛】技術的實做方法除原本軟體提供外掛介面之外，其他最常被使用的方式則是 Windows 上 DLL Injection 技術[13]，簡單說，DLL Injection 就是將自行開發的功能模組包裝成一 DLL 檔案，接著找出作業系統中獨立運作的 Process，利用 DLL Injection 的技術將 DLL 載入該 Process，使其成為該 Process 定址空間的一部份，因此可以將 Process 中的指定執行函式的記憶體位置，改變成該 DLL 中的函式位置。

為驗證以上說法，因此我以此份文件為例子，使用 Process Monitor 這套工具來做實驗。 Process Monitor 是一套進階的系統監視工具，可以顯示即時系統所有正在運作的處理程序，並可以顯示該處理程序共 Inject 了多少 DLL 模組。藉由此工具可以清楚的看到當 Word 文件開啟的時候，是否有被

Inject TrustView 相關模組。

圖 9 為開啟 Process Monitor 工具之後，列出目前系統所有正在執行的處理程序。

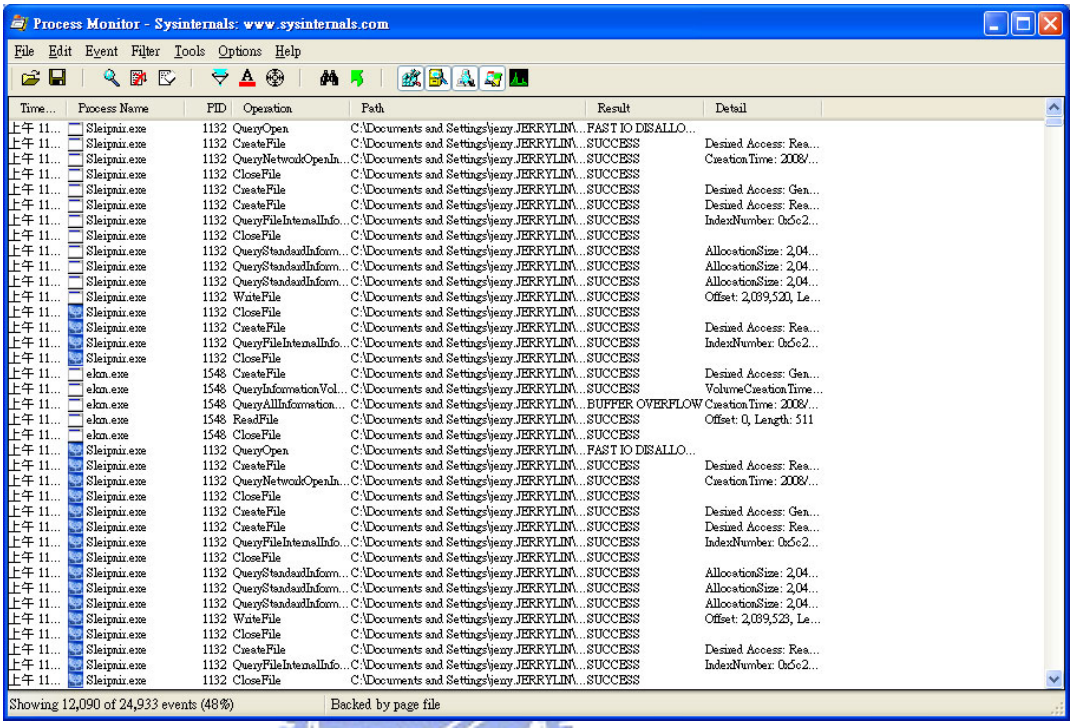



圖 9 開啟 Process Monitor 視窗

開啟該受保護的 Word 處理程序，查看該程序中共 Inject 了多少 DLL 模組，發現裡面有 HOOKTOOL.dll 正在運行，該 DLL 的實際檔案即是安裝在 TrustView 的安裝資料夾位置中，如圖 10 所示。

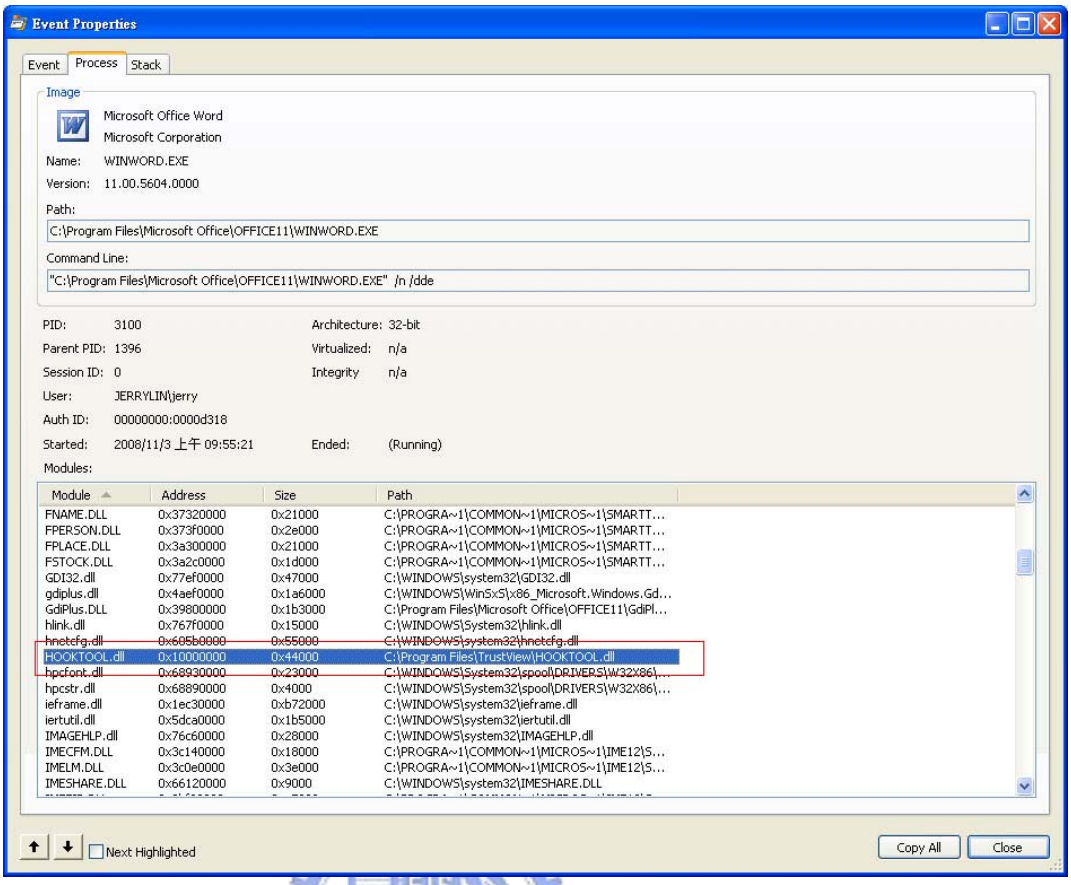



圖 10　WINWORD.exe 中發現有注入 HOOKTOOL.dll

因此可以確認了該防測錄功能，使用了 DLL Injection 的技術，將防測錄功能外掛在處理程序中，除了可以偵測測錄行為之外，也可以對側錄結果的內容做進一步的修改。

## 2.2　使用上的問題

在 2.1 節中，大致確認了 TrustView 的防側錄做法是使用 DLL Injection 的技術，並利用 API Hook 技術來偵測所指定的 API 函式位址，來改變函式的執行內容為自定義的處理函式，因此當測錄模組發現有對桌面或視窗進行錄製行為的時候，該防測錄模組即會將擷取下來的畫面內容清除，畫上自訂的內容，並提示此份文件已經受到該模組的保護。而為了確認以上技術的阻擋效果，我使用更多不同種類的桌面及視窗錄影程式來對這份文件進行測試，結果發現都可以有效阻擋。因此，從實驗中可以確定在 Windows 系統使用 DLL Injection 以及 API Hook 的方法，已經可以擋掉大部分的錄影程式。

針對以上的方法，確實可以有效阻擋測錄行為，但是否能 100% 的阻擋依舊存疑，且此方法僅限於在 Windows 作業系統中實作，若在其他作業系統(例如 Linux)是否也可行，因為在以下小節中，我提出幾項目前防測錄技術存在的問題。

### 2.2.1 無法阻擋使用遠端遙控軟體錄製課程內容

首先，存在第一個問題是否僅使用 API Hook 的作法可以滿足防側錄的需求嗎? 是否存在其他的可能性或方法是使用 API Hook 技術所無法阻擋的? 針對 API Hook 的特性，也就是 API Hook 僅能偵測所在 OS 中所有處理程序的 API 行為，我將該電腦中播放的課程內容畫面傳送到另一台電腦中，是否就能避開偵測了呢?

我以此為出發點，使用了遠端遙控軟體來驗證以上的假設，並建構了以下的實驗架構。

這項實驗使用 VNC ( Virtual Network Computing ) 遠端遙控軟體，VNC 的架構包含 VNC Server 以及 VNC Viewer。 在 Server 端主要是負責傳送電腦畫面以及接收Viewer 端的指令，而在 Viewer 端則是接收 Server 端傳送過來的畫面。環境設置如圖11所示。

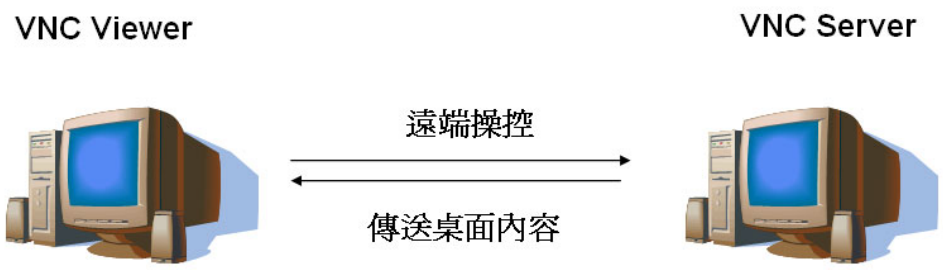



圖 11 遠端遙控軟體環境設置圖

實驗結果發現這樣的方式可以避開 API Hook 的偵測，錄製下受保護文件的內容。圖12為擷取文件內容的畫面。

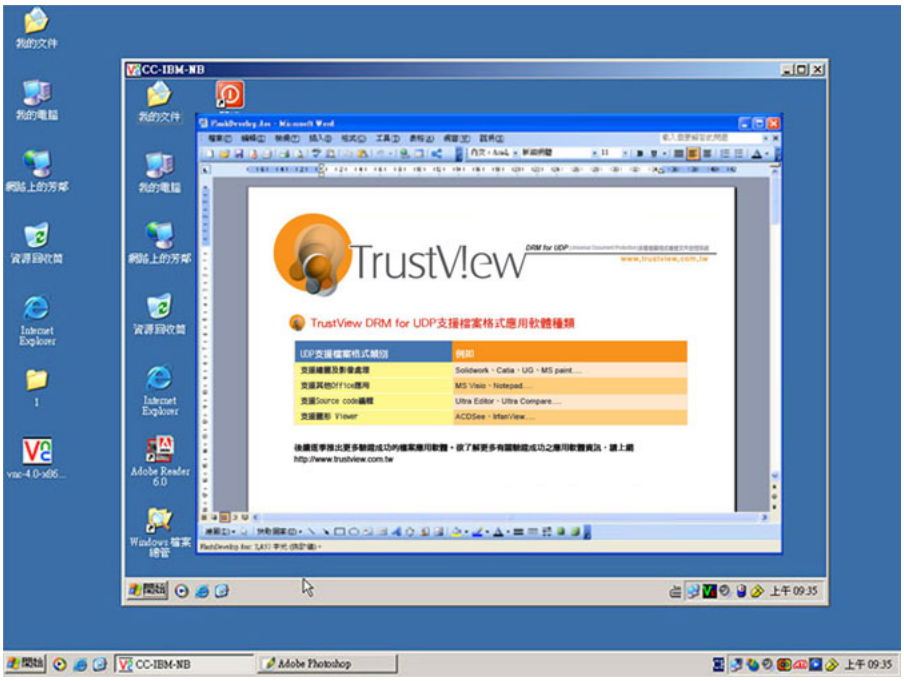



圖 12 使用遠端遙控軟體截取受保護文件的畫面

VNC為了提高擷取與傳送Server端螢幕畫面的效率，因此不使用Hook DLL的技術，而是使用了Mirror Driver[14]的技術，對防測錄模組而言，並無法判斷為一個測錄螢幕的行為，使防側錄模組出現保護的漏洞。

### 2.2.2 API Hook 可能被防毒軟體視為可疑行為

從「2.1 現有防測錄模組的運作方式」中，使用了 Process Monitor 去觀察當此受保護文件開啟時，發現了 Word 的執行序中多了一個 HOOKTOOL.dll 的模組，因此可以判斷該防測錄模組是使用了 DLL Injection 的技術，這項使用技術也是目前 Win32 病毒廣泛使用的其中一種技術，將病毒體本身藏於一個 DLL 中，當 EXE 程序啟動的時候再將這個 DLL 加載至程序中運行。

另外，當 DLL Inject 該程序後，就可以開始使用一些修改指令行為的技術，可以用來攔截、中斷或是監聽程式指令、改變 API 的執行結果，亦即能夠隱藏檔案、目錄、登入檔機碼，乃至系統真正執行的程序。在防測錄模組中，則會修改或刪除擷取下來的螢幕畫面內容。

以上的行為，在某些的防毒軟體中，均會將其視為侵害電腦的可疑行為，造成課程無法正常閱讀。

### 2.2.3 API Hook 技術無法在 Linux 上使用

很多廠商為減少電腦的成本，均以免費的 Linux 作業系統來取代 Microsoft Windows，對數位學習而言，許多課程也必須支援可以在 Linux 中觀看，因此，防側錄功能也必須支援可在 Linux 中執行。針對這項需求，我發現 API Hook 並無法相容於 Linux 環境，主要原因是 API Hook 係用於攔截 Windows 中的 API 事件而提出的技術，所以並不適用於 Linux 環境。

基於以上的三個問題，因此本研究將針對這些問題開始進行研究與分析，進而提出一個新的防側錄方法，除了達到與 API Hook 技術有相同的阻擋效果之外，同時也將補強防側錄系統。另外，也會考量到新架構的可移植性，以滿足相容於 Linux 的需求。

## 2.3 相關技術介紹

此章節將詳細說明幾項在本研究會用到的技術。首先是 PE File 檔案格式的說明，此技術將用於產生本模組所需要的特徵碼。另外，為了驗證阻檔效果，因此也會在本研究中實作一份以 API Hook 技術的防側錄模組，作為本研究之實驗對照組，因此在此章節中，也會一併介紹 API Hook 的相關技術。

### 2.3.1 PE File 檔案格式說明

PE 為 Portable Executable 的縮寫，為 Mircosoft Windows 中可執行的二進位碼，例如 EXE 和 DLL 檔案[2][3]。此格式為 Microsoft 所設計，並在 1993 年由 TIS( Tool Interface Standard ) 標準化。在 B. Luevelsmeyer 之《PE 檔格式 1.9 版》[1] 中對 PE File 的檔案架構有完整的說明。

圖 13 為 PE File 的檔案格式：

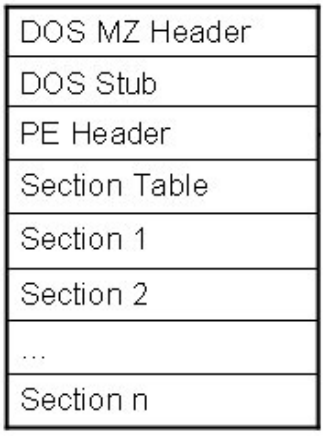



圖 13 PE 檔案格式

從圖 13 的檔案格式說明圖中，DOS MZ Header 是一個 IMAGE_DOS_HEADER 結構，前兩個位元組必須為"MZ"開頭。接下來的 DOS STUB 內容則是代表用於 OS/2 可執行檔，自解壓縮檔和其他程式。對於 PE 檔，它是 DOS2 相容可執行檔，包含 100 位元組內容，若執行時的 OS 不符合，則會輸出一個錯誤資訊："this program needs windows NT"。 PE Header 中則紀錄被載入記憶體時需要用到的參數，程式執行時，PE 裝載器將從 「DOS MZ header」 中找到 PE header 的起始偏移量。因而跳過了 DOS stub 直接定位到真正的檔頭 PE header，而在 PE 標頭之後，為該檔案實際內容。

圖 14 為使用 Ultra Edit 實際開啟一份可執行檔案，並以 16 進位顯示其內容，圖中標示 1 的位址，即為該檔的 PE 標頭位置，此標頭之後即為該執行檔的實際內容。

本研究特徵碼產生的資料來源，是從 PE 標頭位置開始取得，為何要從 PE 標頭之後取出內容? 主要原因是使用同一個 Compiler 編譯而成的 EXE 執行檔，其檔案標頭中所記載的資料內容可能會相同，因此無法確保可以取得完全相異的資料內容，因此決定從 PE 標頭後開始取得資料。

```
00000000h: 4D 5A 50 00 02 00 00 00 04 00 0F 00 FF FF 00 00 ; MZP.........    ..
00000010h: B8 00 00 00 00 00 00 00 40 00 1A 00 00 00 00 00 ; ?.......@.......
00000020h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000030h: 00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00 ; ................
00000040h: BA 10 00 0E 1F B4 09 CD 21 B8 01 4C CD 21 90 90 ; ?...???L??
00000050h: 54 68 69 73 20 70 72 6F 67 72 61 6D 20 6D 75 73 ; This program mus
00000060h: 74 20 62 65 20 72 75 6E 20 75 6E 64 65 72 20 57 ; t be run under W
00000070h: 69 6E 33 32 0D 0A 24 37 00 00 00 00 00 00 00 00 ; in32..$7........
00000080h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000090h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
000000a0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
000000b0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
000000c0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
000000d0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
000000e0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
000000f1: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000100h: 50 45 00 00 4C 01 03 00 19 5E 42 2A 00 00 00 00 ; PE..L....^B*....
00000110h: 00 00 00 00 E0 00 8F 81 0B 01 02 19 00 60 06 00 ; ....??.....`..
00000120h: 00 30 00 00 00 D0 0C 00 A0 2F 13 00 00 E0 0C 00 ; .0...?.?.../.?.
00000130h: 00 40 13 00 00 00 40 00 00 10 00 00 00 02 00 00 ; .@....@.........
00000140h: 01 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00 ; ................
00000150h: 00 70 13 00 00 10 00 00 00 00 00 00 02 00 00 00 ; .p..............
00000160h: 00 00 10 00 00 40 00 00 00 00 10 00 00 10 00 00 ; .....@..........
00000170h: 00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000180h: 34 69 13 00 20 03 00 00 00 40 13 00 34 29 00 00 ; 4i.. ....@..4)..
00000190h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
```

圖 14 以文字編輯器開啟執行檔內容

### 2.3.2 Windows API Hook 技術

Windows 作業系統中，各視窗模組之間的溝通是透過定義的 Message 來傳遞指令內容。而 Hook[6]與 Message 間有非常密切的關係，Hook 的中文名稱是「鉤子」，以字面上的意思，就是要用於監控 Message 在系統中的傳遞，也就是說，透過 Hook 技術可以偵測到該處理程序所發出的任何 Message 內容。

在 Windows 系統下編程，提供了約 2000 個左右的 API 函數給開發人員使用，若在開發的過程中，對 Windows 所提供之 API 函數的功能不甚滿意，想要改變 API 原有的功能，一般的作法就會使用到 API Hook，其作法就是利用 Hook 技術來找到 API 函式的入口點，並修改函式的指標位址為自定義的函式。防側錄模組中即是應用了上述的技術，修改系統中原本用來擷取桌面或視窗畫面的 API 執行函式，攔截此一動作，並修改函式內容將畫面清空。

### 2.3.3 MD5 雜湊演算法

MD5（Message-Digest algorithm 5)雜湊演算法可將任意長度的資料，以 MD5 雜湊演算法運算，得到一組固定長度為 128 位元的結果。

- 雖然不同的資料經由 MD5 雜湊演算法計算所得到的結果有可能相同，但是根據統計顯示，重覆的機率低於百萬分之一。
- MD5 為一個單向雜湊演算法，亦即不易以逆向運算得到原始資料，例如：要計算字串 abc 的 MD5 結果很簡單，但是要將 MD5 計算後的結果逆向運算得到 abc 卻相當困難。

本研究將先採用 MD5 演算法，將取得的資料雜湊計算出一個固定長度的字串，當作代表該側錄程式的唯一特徵碼，未來若有更高安全性的考量，可改用 SHA-1、或 SHA-2 雜湊演算法。

# 三、 系統需求分析

從「2.2 使用上的問題」一節中所發現僅使用 API Hook 的防側錄方式，無法有效阻擋使用遠端遙控軟體的側錄方式，因此在本章 3.1 節中將會提出本研究防側錄模組，分析所需的功能模組與各模組的運作流程，針對先前提出的問題所提出的改善方法，以滿足需求。

## 3.1 防側錄模組的特性

本研究擬發展一特徵碼網路資料庫比對機制，針對解決2.2節所提及的使用上問題，該系統須滿足以下幾點特性：

1. **唯一性**：每一個執行檔需對應一唯一特徵碼。

2. **不可逆**：特徵碼為不可逆，亦即無法由特徵碼猜出執行程式，無法經由修改執行檔名稱或視窗名稱，避過防側錄模組的檢查。

3. **即時偵測**：除特徵碼資料庫可即時更新內容外，特徵碼的比對效率不能影響到課程瀏覽的流暢度，因此計算特徵碼、以及特徵碼比對需最佳化

4. **可移植性**：須考慮整個系統架構將來可以移植到 Linux 上運作。

## 3.2 特徵碼

特徵碼的概念是運用防毒軟體中的其中一項作法 - 「病毒碼掃描法」[7]，以此方法的概念作不同領域的延伸。所謂的病毒碼掃描法，主要是將新發現的病毒經過分析之後，取出其特徵值的部份編成唯一病毒碼，此病毒碼將用來代表此病毒，並加入資料庫中，若在掃毒比對的過程中，發現與病毒碼相符合，則會被判讀為病毒程式。

### 3.2.1 何謂特徵碼

在前文中提到了本研究的特徵碼是運用防毒軟體中的病毒碼概念，根據相同的概念，從執行程式檔案中取出部份可代表該檔案特徵值的內容，透過一演算法轉換成一固定長度的字串碼，用來代表此執行軟體，稱之為「特徵碼」。

因為考量到計算速度的問題，不會以整份檔案的內容做為特徵碼計算的來源，而是根據 PE 檔案格式的架構[1]，從中截取一小段獨一無二且能代表這隻程

式的二進位程式碼(Binary Code) 來做為特徵碼計算的來源。

另外，在比對資料的來源部分，當防側錄模組啟動的同時，會列舉目前系統中所有正在運行中的處理程序，如圖 15 所示，並計算其特徵碼，最後再與資料庫中的特徵碼做比對。


圖 15 Windows 工作管理員中的處理程序列表

## 3.2.2 特徵碼的產生

根據「2.2.1 PE File 檔案格式說明」中，我們得知 Windows 中每一可執行檔案均為 PE 格式，此格式的 PE 標頭之後所記錄的二進位碼則為該程式的執行內容。這部份的內容，由於程式撰寫與程式架構的不同，因此每一程式均大不相同，所以取得此段資料的二進位碼內容，來做為產生特徵碼的來源資料。

圖 16 為根據唯一性的需求，所規畫出的特徵碼產生流程：

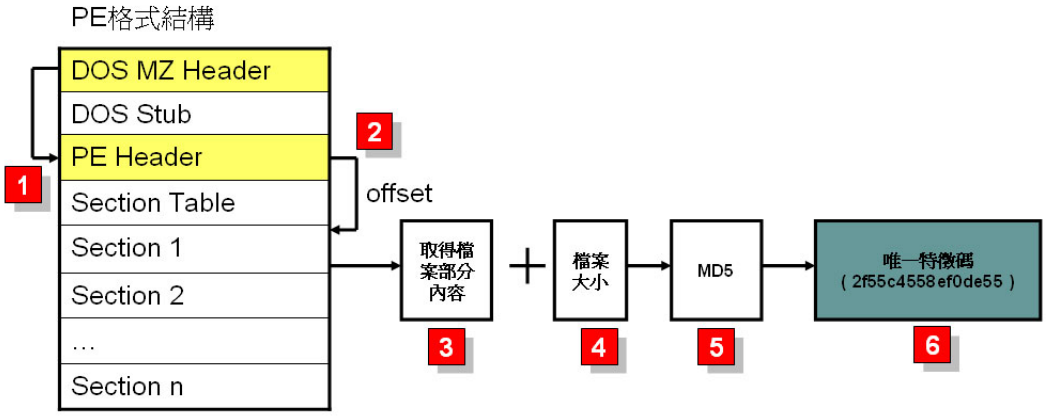



圖 16 特徵碼的產生流程圖

1. 從 DOS MZ header 中找到 PE header 的起始偏移量
2. 定義一個差量值
3. 取得部分檔案內容( 50 個字元 )
4. 取得檔案大小
5. 將步驟 3 與 4 的內容轉為字串，接著做 MD5 的計算
6. 固定取得前 16 字元，即產生該檔案的唯一特徵碼

### 3.2.3 特徵碼資料庫欄位說明

產生特徵碼之後，接下來就要將這些特徵碼記錄在資料庫中，資料庫中的欄位包含檔案名稱、檔案大小、檔案特徵碼、建立者等資訊。以下為各欄位的詳細說明：

| 特徵碼資料庫儲存欄位設計 | |
|---|---|
| **欄位名稱** | **欄位說明** |
| **檔案名稱** | 側錄程式之執行檔名稱或 DLL 檔案名稱 |
| **檔案大小** | 執行檔或 DLL 檔案大小 |
| **檔案特徵碼** | 唯一檔案特徵碼 |
| **建立者** | 資料庫建立者 |
| **備註** | 側錄程式說明 |

表格 1 特徵碼資料庫欄位說明

系統建立一個網路資料庫以及一個 Web 端的使用者介面，此使用者介面是方便系統管理者，用來儲存並建立測錄程式的特徵碼內容。此資料庫將配合防側錄模組即時更新資料庫內容至該模組中，讓每次進行比對工作時，都可以使用到最新的資料，除了滿足即時性需求外，也可提高該模組的有效阻擋率。

## 3.3 API Hook 防側錄模組

此項需求，則是要完成本研究所需要的實驗對照組。

為了實作出一個使用 API Hook 的防測錄模組，因此在實作前我先整理了與畫面擷取以及畫面繪製相關的 Windows API 函式，如表格 2 所示。

| 函式名稱 | 函式說明 |
|---|---|
| GetDC | 取得目前執行視窗的裝置環境代碼( Device Context )。 |
| GetDCEx | 與 GetDC 相同，該函數檢索指定視窗客戶區域或整個螢幕的顯示設備上下文環境的控制碼，在隨後的 GDI 函數中可以使用該控制碼在設備上下文環境中繪圖。 |
| GetWindowDC | 取得整個視窗（包括邊框、捲軸、標題欄、功能表等）的裝置環境代碼。 |
| CreateDCW | 使用指定的名字為一個設備創建設備上下文環境，Unicode。 |
| CreateDCA | 使用指定的名字為一個設備創建設備上下文環境，ASCII。 |
| ReleaseDC | 若是透過 GetDC 函式來取得的話，不用時要用 ReleaseDC 函式來釋放它 |

表格 2 畫面繪製相關的 API 函式

接下來要開始研究如何修改處理程序中 API 函式的入口位址。首先我先從 PE File 中找到 Imported Symbols Table(輸入符號表)，此 Table 是記錄在 PE Header 後的一個 IMAGE_DATA_DIRECTORY 結構中，從它中的 VirtualAddress 位址內容，即可找到 API 函數的入口位址。

最後，使用 DLL Injection 的方式，將防測錄模組 Hook 到指定的處理程序中，

圖 17 為 DLL Injection 程序的說明圖：

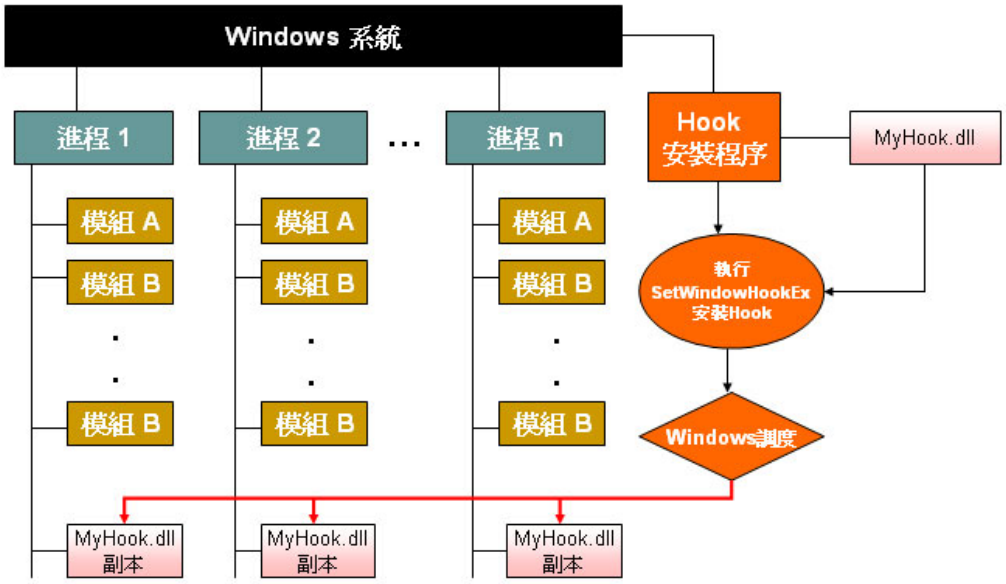



圖 17 API Hook 架構圖

# 四、系統架構與流程

前章中針對本研究所提出的各模組做功能上的需求分析，以及說明各模組的運作方式，接下來則會以這些模組的分析作為基礎來建構整個系統，並對本研究的系統架構、功能模組間的溝通，以及系統流程作詳細的說明。

## 4.1 系統架構

本研究將實做與整合的系統架構如圖 18 所示，包含「製作端」的課程網頁轉換工具以及「伺服端」的特徵碼資料庫與 DRM Server、「閱讀端」的防側錄模組：

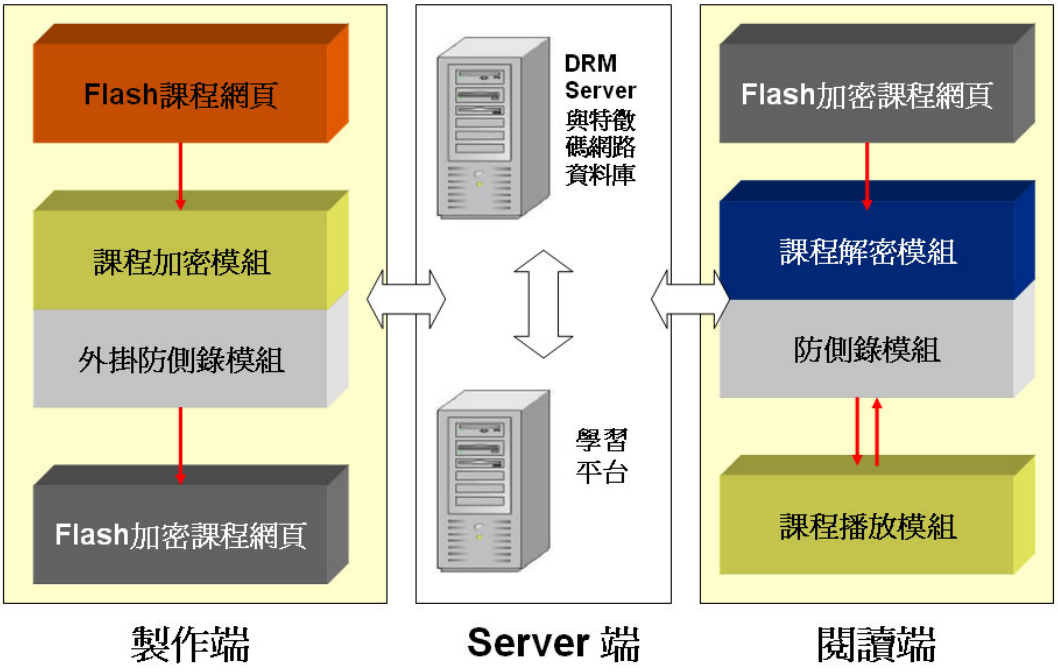



圖 18 系統架構

- 「製作端」- 課程網頁轉換工具

為本研究系統建置的第一個步驟，課程原始作者(Content Creator)會將需要保護的課程，利用課程編碼器來做編碼轉換。課程編碼器開始運作之前，會和 DRM server 取得加密金鑰，使用此金鑰開始加密課程內容。內容加密完成之後，課程網頁轉換工具則會將防側錄模組嵌入此份課程網頁中，嵌入的語法中會依據閱讀端使用何種瀏覽器來觀看課程，而啟用不同的外掛模組，例如 IE 上會使用 ActiveX Control， FireFox 上會使用 Plugin。當課程編碼器處理完畢，即會產生一份受 DRM 保護且含防側錄模組的加密課程。

- 「伺服端」- 特徵碼資料庫

在 Server 端的部分，除了課程權限控管的 DRM Server 外，另外建置一特徵碼資料庫，此資料庫是用來記錄每一隻桌面或視窗錄影程式所產生的唯一特徵碼，並提供輸入與設定的 Web 使用者介面，方便建立資料於特徵碼資料庫中，管理者可以隨時新增、刪除或修改名單中的項目。當閱讀端開啟課程，經過 DRM Server 的身分及權限認證通過後，防側錄模組即會透過網路下載並更新最新的側錄程式名單，暫存在閱讀端的電腦，以提供防側錄模組比對使用。

DRM Server 的部分則是配合智勝國際科技公司所建置的 DRM server，這是一套用來控管課程閱讀權限的系統，例如課程的閱讀時間限制，閱讀電腦 IP Range 限制等等功能，受保護的課程無論在何處，都會受到 DRM server 保護。

- 「閱讀端」- 防側錄模組

閱讀端為一課程播放器，此課程播放器是配合智勝國際科技公司開發之 BWFlash Player[8]，當中包含三個模組：課程解碼器模組、DRM 權限管理模組、與課程播放模組，為整合防側錄功能，因此會再加入本研究之防側錄模組。

當讀者從學習平台上開啟一份受保護的課程時，課程解碼器模組會先透過 DRM 權限管理模組和 DRM server 做連線，而 DRM server 會要求讀者的驗證資訊，一旦身份與使用權限確認通過，DRM server 就會傳送解密金鑰給課程解碼器，此時便可開始執行解碼的動作，而解碼後的記憶體內容再傳送給課程播放模組，待課程正式播放的同時，防側錄模組也會一起啟動，並開始列舉閱讀端電腦的所有處理程序列，執行比對工作，一但發現符合側錄名單中的項目時，會立即要求播放器停止播放課程內容，並隱藏播放區域，接著會出現提示訊息，要求閱讀者先關閉側錄程式課程才能繼續觀看，等待閱讀端將側錄程式關閉之後，會通知播放器恢復播放狀態，如此以達到防側錄效果。

本研究所規劃出的系統架構是依循「製作端←→Server 端←→閱讀端」的模式，如圖 18 所示。製作端有課程編碼器的本地端製作工具；閱讀端有 BWFlash Player 課程播放器以及本研究之防側錄模組，兩者再透過 Server 端來對這些受保護的課程作管理與追蹤，以及更新最新的防側錄資料庫內容，讓整個系統架構趨於完整。

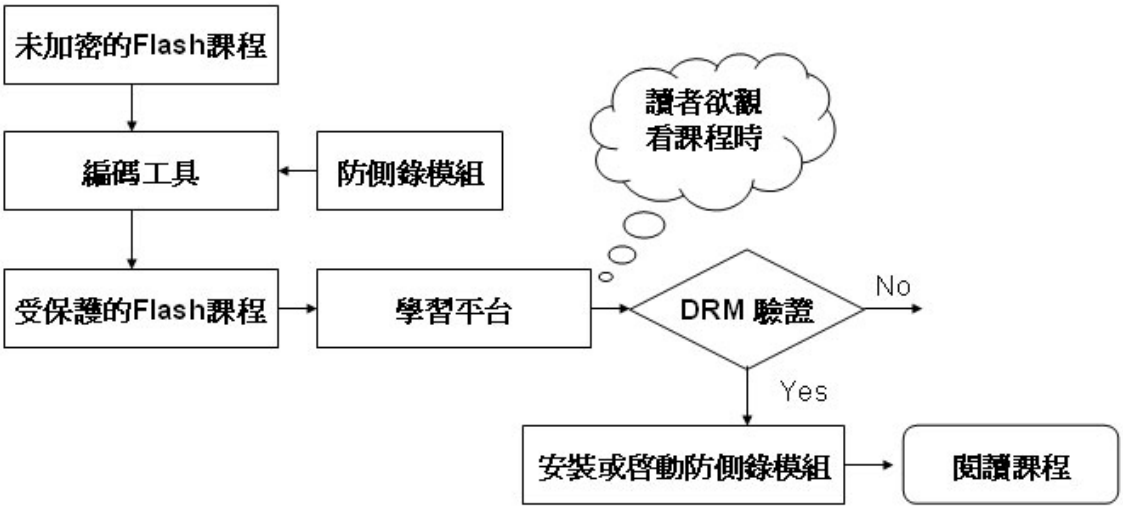



圖 19 系統流程

在系統流程的部份，則會依循圖 19 所示，從課程先經過加密保護機制，接著上傳到學習平台，以及閱讀端開始觀看課程時，經過 DRM Server 的認證，並開始防側錄的偵測比對流程，在下節中我將詳細說明各模組的運作流程。

## 4.2 特徵碼資料庫建立流程

針對「Server 端」的特徵碼資料庫需求，以及根據「3.2.2 特徵碼的產生」以及「3.2.3 特徵碼資料庫欄位說明」的需求分析，特徵碼資料庫是防測錄模組比對機制最基本的資料來源，當資料庫內容蒐集越完整，則閱讀端的阻擋的效果越佳，因此該資料庫需由系統管理員負責管理與建立特徵碼，也就是當蒐集到一筆新的防側錄特徵碼時，必須能立即更新到資料庫中。

針對此項需求，在資料庫的更新流程上需設計得更為簡易，因此捨棄了 Local 端工具的開發，選擇開發 Web 端的使用者介面，並將特徵碼演算法實做成一個 Web 的外掛模組(ActiveX)，管理者只需要透過有網路環境的電腦，開啟特徵碼資料庫管理網頁，輸入帳號密碼取得使用權限之後，就可以立即新增特徵碼至資料庫中。在第五章中，我將會在實作與操作的部份做詳細說明。

## 4.3 課程保護製作流程

在課程保護的製作流程中，我使用的是智勝國際科技公司所研發之 Flash 轉檔工具，主要使用的原因有二：

1. Flash 檔案格式目前已經為內容製造商( Content Creator ) 廣泛使用，而本工具正好滿足 Flash 加密保護的需求，並且本工具所產出的加密課程可搭配 DRM 的權限管控機制，也符合了數位內容保護的需求。

2. 此工具產出格式為 Flash 的網頁格式，適合本研究之外掛模組。

BWDRM Flash Player 是由 Action Script 3.0 [9]版開發完成，因此在與防測錄模組的溝通部份，會有一部份使用 AS3 來開發，目的在當防側錄模組有偵測到測錄行為的時候，會傳送停止播放指令給播放模組，而播放模組接收到此一指令，即會立即停止播放，直到測錄程式被關閉為止。

完整的數位保護製作流程為：將原始未被加密的 Flash 網頁來源檔，匯入轉檔工具，在檔案加密前，會先與 DRM Server 取得課程金鑰，接著透過隨機演算法的決策來決定加密該份 Flash 所使用的演算法，加密工作完成之後，在產出網頁的同時，會同時嵌入一段防側錄模組的外掛 Java Script 語法，使課程在閱讀端開啟的時候，可以根據語法中紀錄的防側錄模組下載位置，自動下載且安裝防側錄模組至閱讀端的系統中。另外，此外掛模組也會判讀版本是否為最新，亦或是從未開裝過該模組，並自動下載安裝，所以當第一次開啟課程的時候，會先要求下載安裝該模組，才可開始觀看課程內容。完成以上的流程之後，即完成一份受保護的課程。

## 4.4 防側錄保護流程

最後，在此章節中我將繼續說明「閱讀端」的防側錄保護流程。為了可以即時更新比對資料庫，因此，在防側錄模組開啟之前，會先透過網路檢查特徵碼資料庫是否有更新，若有更新，則會自動下載並儲存在 Local 端。另外，也考慮到客戶端可能有離線瀏覽的狀態，因此，在防側錄模組第一次安裝的時候，亦同時會安裝一份 Server 端的特徵碼資料檔至 Local 端的電腦中，以提供客戶端在沒有網路連線的狀態下使用。

特徵碼資料庫的更新時機，有以下兩種情形：

1. **Local 端完全無任何資料庫暫存檔時**： 在無任何資料庫暫存檔的情形下(可能被客戶端刪除的情形)，課程開啟時，防側錄模組將會要求課程播放器馬上停止並隱藏播放課程，直到有下載並安裝資料庫檔案為止。

2. **當 Server 端資料庫內容有更新時**： 客戶端下載的資料檔中有紀錄此份資料檔的修改與建立時間，因此當此紀錄時間與 Server 端的修改建立時間不相同時，即代表資料庫有做過改變，此時會再次下載並更新 Server 端的最新資料檔。

另外，為避免下載的資料檔被隨意開啟與修改，因此資料檔將會使用 3DES 的加密演算法保護。

接下來，在圖 20 中則說明了當閱讀端啟動防側錄模組之後，此模組會列舉

Windows 中目前所有正在運行的處理程序(Process)，透過模組中的特徵碼產生演算法，取得每一個處理程序的唯一特徵碼，並紀錄在自定義的資料結構中，接下來就會用此資料，與特徵碼資料檔中的項目一一做比對，若發現有符合的項目，則表示為一個側錄程式，立即發出訊息通知課程播放器停止播放並隱藏課程內容，直到下次的比對程序中，沒有發現任何的符合項目時，才會再重新開啟並顯示課程的播放內容。

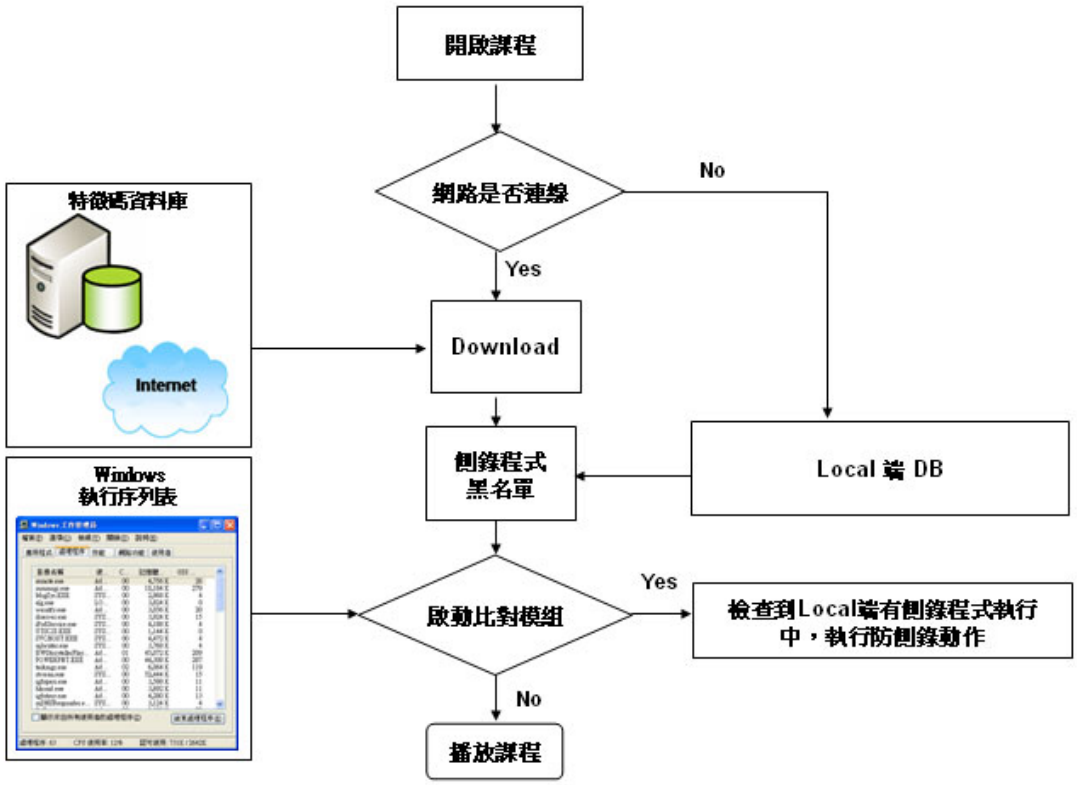



圖 20 防測錄流程

# 五、 系統設計與實作

確定了系統的架構與流程之後，本章將開始介紹實作的部份，在 5.1 節中將說明特徵碼實作、製作端的課程網頁轉換工具的實作；以及分別會在 5.2、5.3、5.4 節中介紹閱讀端播放器，其中播放器是由解碼器、解析模組、與防測錄模組所組成。

## 5.1 特徵碼的實作

引用在第三章的系統需求分析中所提及的特徵碼製作概念，使用開檔的 API 函式開啟該執行檔案，接著再根據取得檔案的控制權(Handle)，使用搜尋的 API 函式找到 PE 標頭的位址，從此標頭位置開始取得所需的二進位碼資料(Binary Data)。另外，在製作端以及閱讀端，必須使用相同的特徵碼產生演算法，否則在比對的時候，會產生誤判的問題。

本節中將先實作一特徵碼演算法，然後再套用至「Server 端」特徵碼網路資料庫，以及「閱讀端」的防側錄外掛模組中。接下來，我將先詳細說明如何產生特徵碼演算法的實作內容。

首先，使用 fopen 函式開啟檔案，指定欲計算特徵碼的實體檔案位置。

```
FILE* pfile = NULL;
pfile = _wfopen( i_exepath.c_str(), L"r" );

if( pfile == NULL )
{
  ::MessageBoxW( NULL, L"無法開啟檔案!", L"Error!", MB_OK );
  return false;
}
```

取得檔案的 Handle(即下方程式碼中的 pfile 指標變數)之後，再使用 ptell 函式帶入檔案指標，取得該檔案的大小(filesize)，暫時將此檔案大小的資訊存放在變數中，將作為構成特徵碼的來源資料之一。

```
//- 取得檔案大小
fseek( pfile, 0, SEEK_END );
filesize = ftell( pfile );

rewind( pfile );
```

接著開始取得檔案的二元資料(Binary Data)內容，首先找到檔案中的「PE」

標頭位置，從先前的研究得知，若檔案中無任何 PE 標頭，則代表此檔案非 Windows 中可執行的檔案，也就是非合法可在 Windows 中執行的軟體。在下方的程式碼中，主要搜尋檔案中「PE」標頭的位置，若找不到「PE」標頭，則會結束特徵碼的執行程序。

```
bool isPEFile = false;

while( curPos <= filesize )
{
  BYTE pBuffer[2];
  fread( pBuffer, sizeof( BYTE ), 2, pfile );

  BYTE tmp = pBuffer[0];
  BYTE tmp2 = pBuffer[1];
  if(tmp == 'P' && tmp2 == 'E')
  {
    int offset = ( filesize - curPos ) / 10;
    curPos += offset;
    isPEFile = true;
    break;
  }

  curPos++;
  fseek( pfile, curPos, 0 );
}
```

在「PE」標頭的位置之後即為該執行檔實際的二元資料(Binary Data)，每一份執行程式除了 Compiler 自動產生的標頭資訊有可能相同之外，其他部分因為每隻程式的用途與撰寫寫法均不相同，因此「PE」標示之後的資料可用來作為組成特徵碼的要素之一。以下程式碼中，會設定一個 50 字元長度的陣列，用來存放該執行檔中取得的二元資料(Binary Data)。

```
//- 取得檔案部份內容
string filecontent;

while( true )
{
   char pszBuffer[50] = "\0";
   fgets( pszBuffer, 50, pfile );
   filecontent = (string)pszBuffer;
   curPos = ftell( pfile );

   if( filecontent != "" && filecontent.length() > 2 )
   {
        break;
   }

   if( curPos >= filesize )
```

```
    {
        fclose( pfile );
        ::MessageBoxW( NULL, L"超過取得檔案長度!", L"Error!", MB_OK );
        return false;
    }
  }
```

最後，將先前取得的檔案大小以及執行檔中取得的二元資料(Binary Data)轉換為字串相加，再將此字串帶入 MD5 雜湊演算法中運算，產生一固定長度的雜湊值，轉成字串後取前 16 字元長度，即產生本研究所需要的特徵碼值。

```
BWMD5 clBWMD5;
string Md5String = clBWMD5.md5( filecontent ).substr( 0, 16 );
```

以上的過程即完成了特徵碼的實作，接下來我先將此演算法套用到「Server」端的特徵碼網路資料庫上，此資料庫的使用者介面，會開發成一 Web 的使用者介面，透過此介面，系統管理者可直接匯入測錄軟體的執行程式檔案，而在特徵碼的演算法部分，則會以 ActiveX 的方式實作，另外為了符合 Web 化的操作介面，我選擇使用了 Action Script 3.0 與 Flex 工具，開發一個以 Flash 呈現的網頁操作介面，請見圖 21。

在特徵碼演算法的 ActiveX 部份，運算完成後會透過 Java Script 傳送回 Web 端的使用者介面上。ActiveX 嵌入網頁的語法如下：

```
<object id="IBuilder"
    classid="clsid:481248BC-11A0-457B-AE8F-1FABF6992BB5"
    codebase="http://drmx.caidiy.com/bw/dl/IBuilder2/apInfoBuilderActi
    veX.cab#version=-1,-1,-1,-1" allowScriptAccess="always" >
    <embed type="application/x-IBuilder" id="IBuilder"
    allowScriptAccess="always" pluginspage="">
</object>
```

完成了介面與各模組的功能之後，接下來就要開始使用工具來建立特徵碼資料庫內容，以下我將逐步介紹如何在 Web 端的使用者介面上，新增測錄程式的特徵碼：

步驟一、開啟特徵碼資料庫 Web 介面，圖 21。

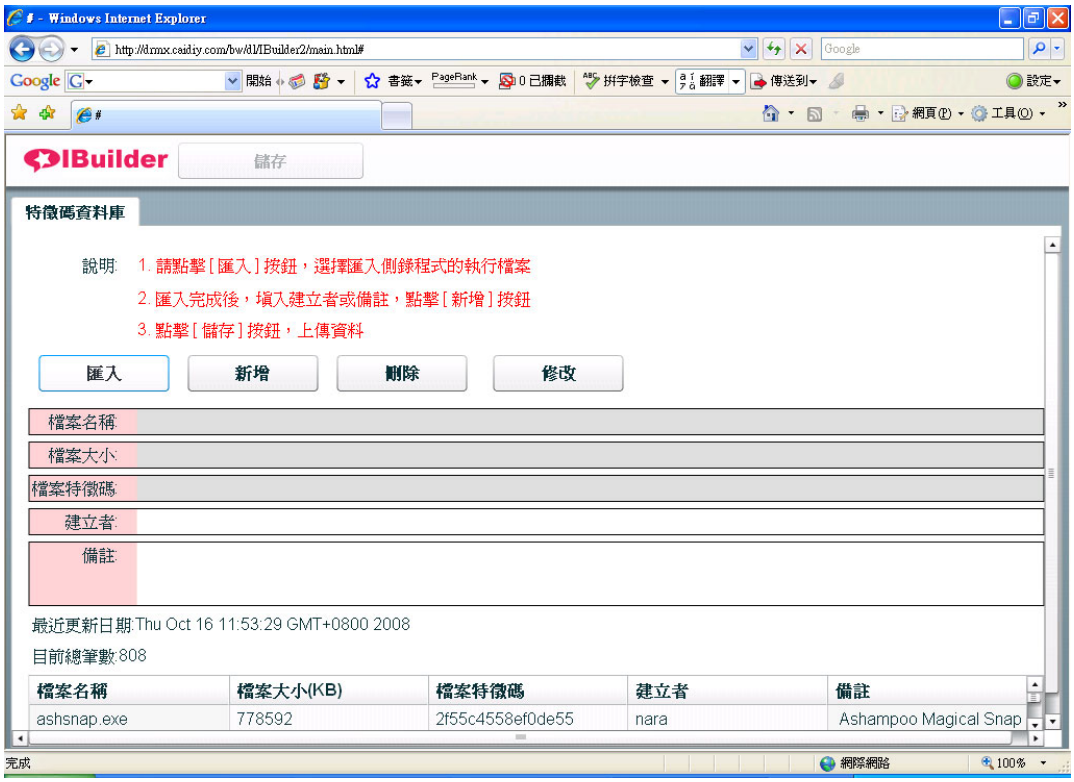



圖 21 特徵碼資料庫系統

步驟二、滑鼠點擊【匯入】按鈕，開啟檔案總管視窗，如圖 22 所示。接著，選擇欲加入資料庫的側錄程式執行檔案位置，點選【開啟】按鈕匯入檔案。

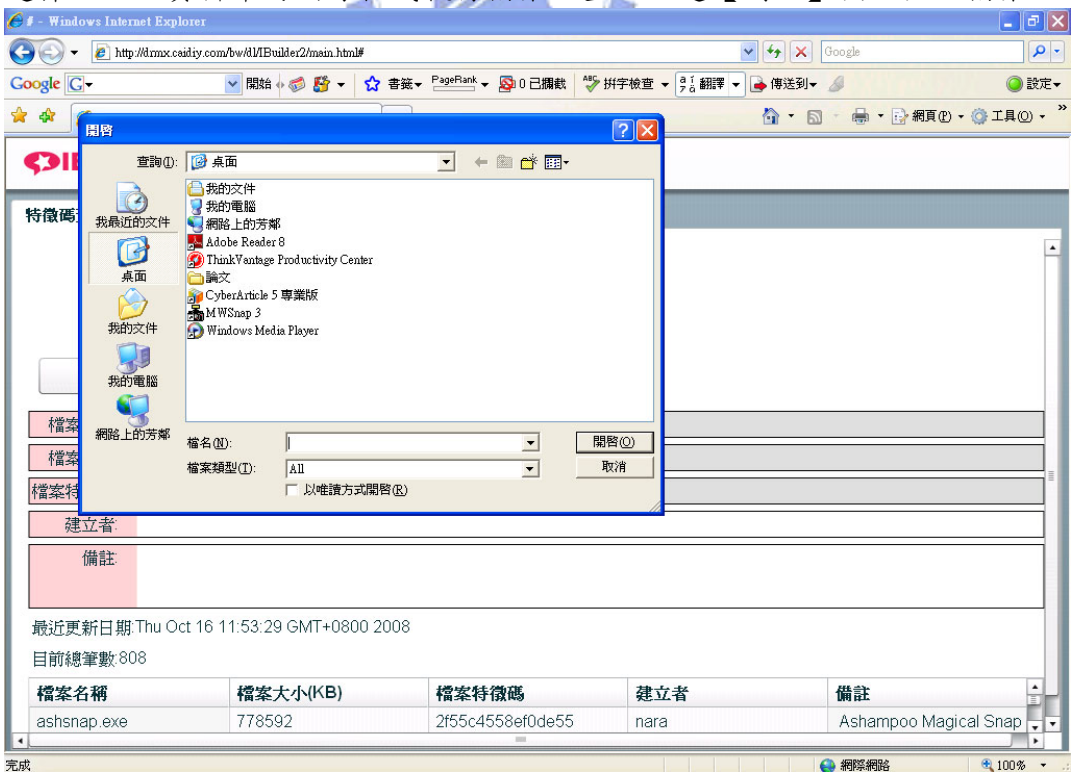



圖 22 選擇測錄程式執行檔案位置

步驟三、匯入檔案之後，透過網頁外掛模組的特徵碼演算法，計算出代表該

程式的 16 字元長度的唯一特徵碼。最後傳回網頁並顯示在欄位中，如圖 23 中所示，除了顯示特徵碼資訊之外，也同時會顯示該檔案的名稱、大小等訊息。

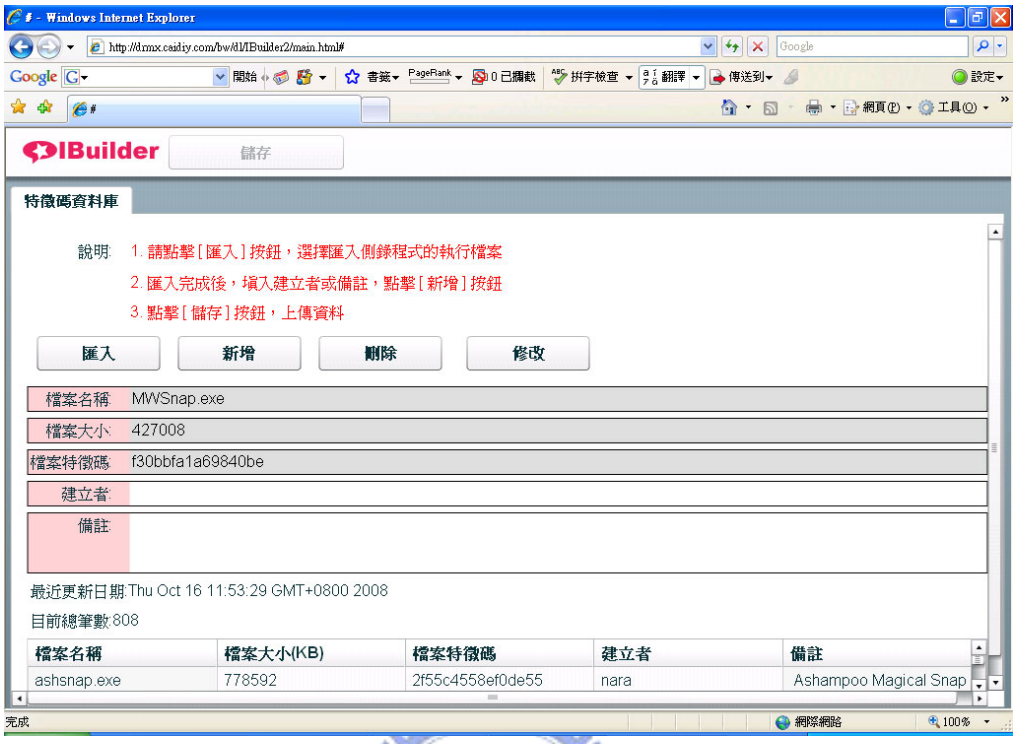



圖 23 計算產生特徵碼

步驟四、介面中也提供了建立者與備註欄位讓管理者填入，使可以記錄更多該側錄軟體的相關資訊，如圖 24 所示。

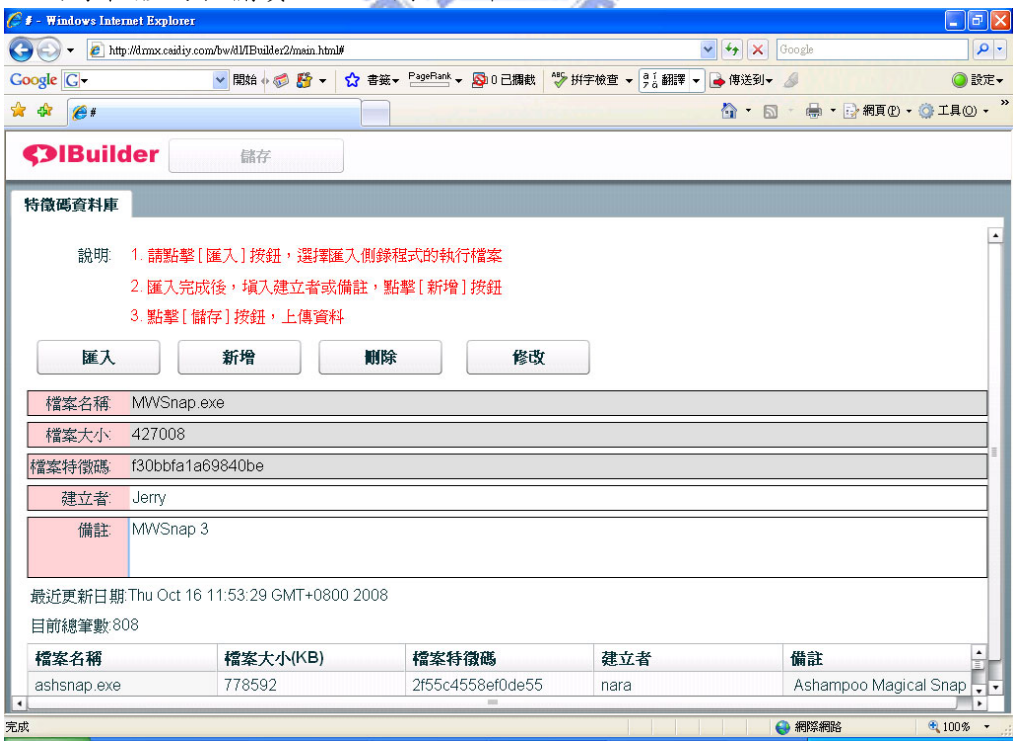



圖 24 加入建立者及備註等資訊

步驟五、最後，再點擊【新增】按鈕儲存建立好的特徵碼。另外，介面上也

提供【刪除】、【修改】按鈕，方便以後維護資料庫內容用，如圖 25 所示。

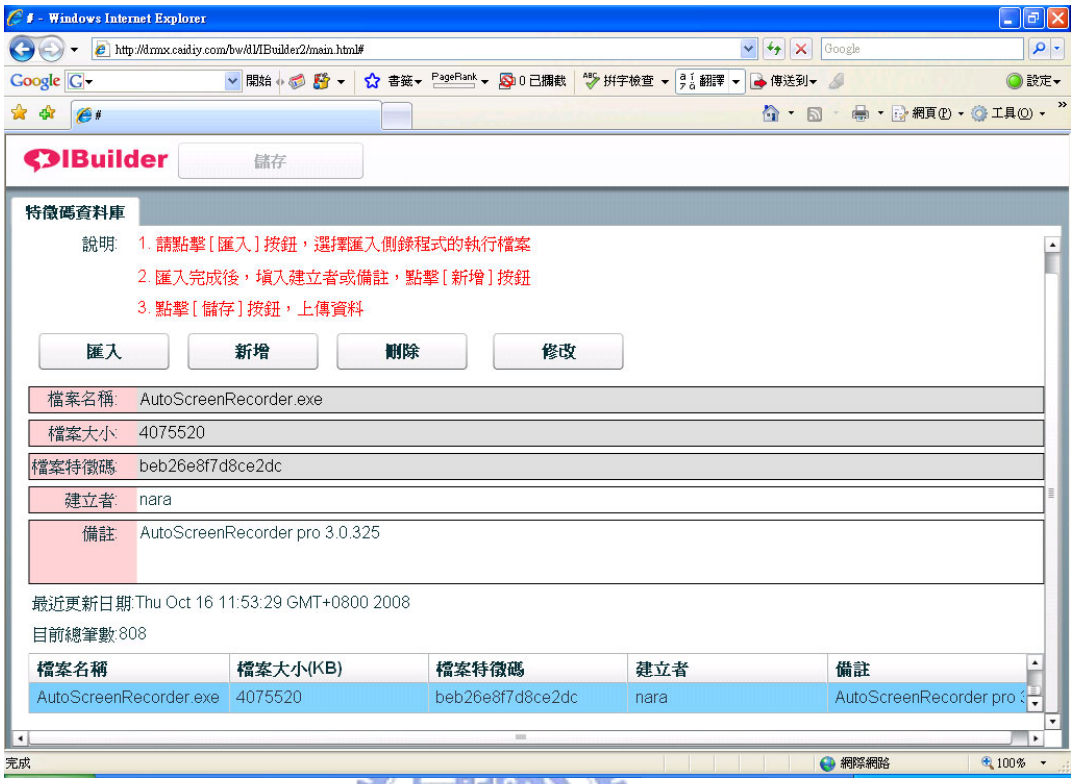



圖 25 儲存特徵碼至資料庫

為了增加本研究實驗時的測試樣本，截至目前為止，我一共收集了網路上擁有錄影桌面或視窗功能的軟體，來作為本研究的測試來源，目前共建立有 828 筆紀錄。

## 5.2 特徵碼比對系統實作

完成了 5.1 節的特徵碼實作，接下來在此一章節中，我將繼續說明如何列舉系統中正在執行的程序，以及特徵碼的比對實作。

圖 26 為本研究特徵碼比對系統實作的流程圖：

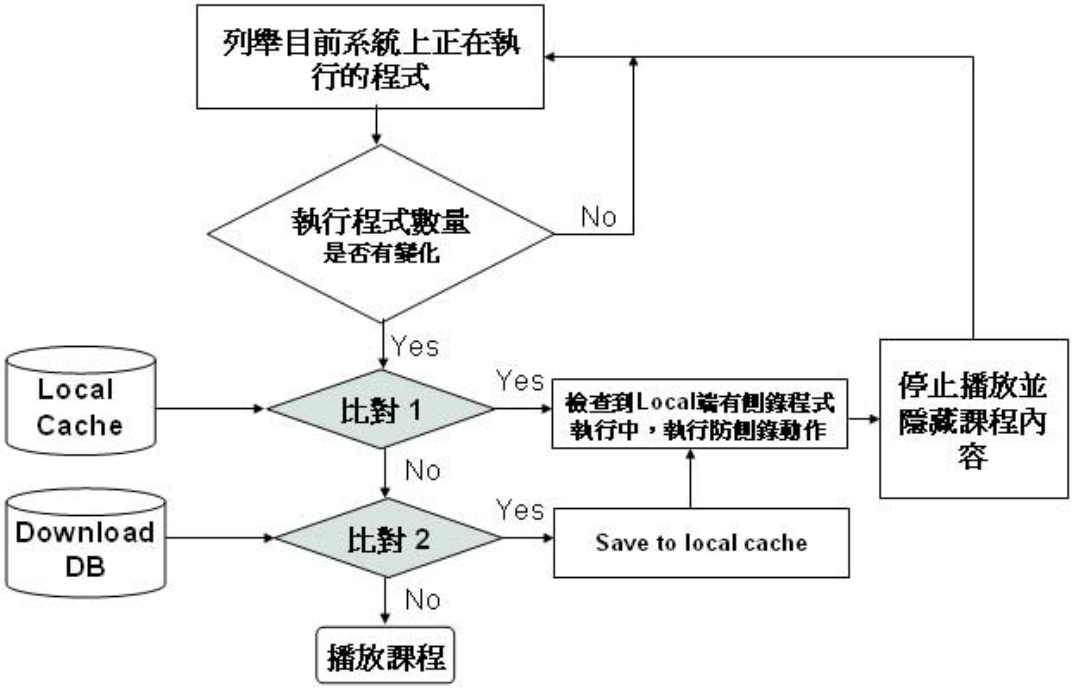



圖 26 特徵碼比對流程圖

首先，在列舉系統處理程序的實作部分，我定義了一個資料結構，目的在儲存該處理程序的相關資訊，如下程式碼內容所示，winHandle 用來儲存該執行緒的控制指標，而 exePath 則是用來儲存啟動該處理程序的實際執行檔案位置。

```
typedef struct _RunningAP{
   HWND winHandle;
   wstring exePath;
}RunningAP;
```

接著定義一個 vector，用來儲存所列舉的處理程序內容。

```
vector<RunningAP> m_WndList;
```

在列舉的實作部分，則使用了 Window 所提供的 API -「EnumWindows」來實做，如下程式碼內容所示。

```
BOOL EnumWindows(
      WNDENUMPROC  lpEnumFunc,
      LPARAM  lParam
);
```

在 EnumWindows 函式中第一個參數 lpEnumFunc 傳入的是一個 CallBack 函式指標。在下方的程式碼中，EnumWindowsProc 則為自定義的 CallBack 函式，需要帶入兩個參數 hwnd 和 lparam，第一個引數是視窗的控制代碼；第二個引數則是會帶入之前定義的 vector - m_WndList，用來儲存列舉的所有處理程序。

```
bool CALLBACK EnumWindowsProc( HWND hwnd, LPARAM lParam )
```

以下程式碼則為 EnumWindowsProc 函式中實做的內容：

```
if( GetParent( hwnd ) == 0 )
{
   DWORD cnt;
   HMODULE hMod[255];
   wchar_t szModName[255];
   DWORD dwProcess;

   GetWindowThreadProcessId( hwnd, &dwProcess );

   //- 取得程式執行檔名稱
   HANDLE hProcess = OpenProcess(PROCESS_QUERY_INFORMATION |
                                 PROCESS_VM_READ, FALSE, dwProcess);
   EnumProcessModules( hProcess, hMod, 255, &cnt );

   memset( szModName, 0, sizeof(szModName) );
   GetModuleFileNameExW( hProcess, hMod[0], szModName,
                         sizeof(szModName) );

   CloseHandle(hProcess);

  if( vector<RunningAP>* WndList =
      reinterpret_cast<vector<RunningAP>*>(lParam) )
   {
        if( wcslen(szModName) > 0 )
        {
            bool bExist = false;

             for( unsigned int j = 0; j < l_ExcludeCodeList.size(); j++ )
             {
                 if( IsFindString( l_ExcludeCodeList[j].filename,
                                   (wstring)szModName ) )
                 {
                      bExist = true;
                      break;
                 }
             }

             RunningAP runningAP;
             runningAP.winHandle = hwnd;
```

```
                if( bExist )
                {
                    //- 標示為非側錄程式
                    runningAP.exePath.append(L"*");
                    runningAP.exePath.append( (wstring)szModName );
                }
                else
                {
                    runningAP.exePath = (wstring)szModName;
                }

                WndList->push_back(runningAP);
            }
        }
    }
```

列舉目前系統中正在執行的程序之後，接下來就要開始與資料庫中的特徵碼一一做比對。在實做部分，我設計一個計時器固定時間執行比對動作，而為了加快比對速度，因此設計了 Two Pass 的比對機制，以下是此比對機制的說明。

**First Pass**：首先會比對側錄程式的 Local 端資料庫(Local Cache)，此資料庫主要是記錄先前被偵測為側錄程式的特徵碼紀錄，在下次觀看課程的時候，防測錄模組會先檢查比對此份資料庫內容，由於此資料庫會比網路端的資料庫小很多，所以多了這一個步驟的比對，可以加速產生比對結果。

**Second Pass**：如果 First Pass 中沒有偵測到任何的側錄程式存在，則會開始進行此一階段的特徵碼比對工作，此階段會與網路端下載的資料庫內容做比對。若發現有符合特徵碼項目時，則會立即通知課程播放器停止播放並隱藏課程內容。

## 5.3 安裝特徵碼模組實作

完成了 5.2 節的特徵碼比對實作，接下來就要將實作好的模組，與加密課程網頁做整合。針對目前市面上最常被使用的瀏覽器：IE 與 Mozilla FirFox，我分別研究這兩個瀏覽器的外掛技術：ActiveX 與 Plugin。

首先說明一下，為何要使用 IE 的 ActiveX 外掛技術，主要原因是 ActiveX 是由 Client 端平台、Server 端平台、開發工具、編輯工具、和網路技術所組成，而 ActiveX 將提供給使用者、網頁管理者和開發者一個豐富化的平台，讓他們可以在 Internet 上執行各種的應用軟體。因此，使用 ActiveX 就可以使本研究之防側錄模組與課程播放模組做訊息的溝通，對於課程內容則有較佳的操控性。而在 FireFox 的 Plugin 部分，則與 IE 的 ActiveX 技術的目的相同。

確定本研究中所要使用的外掛技術之後，接下來則在課程網頁上加入以下的 JavaScript 程始碼，此段程式碼代表會自動偵測客戶端開啟課程所使用的瀏覽器，若為 IE 則使用 Object 的方式嵌入網頁，若為 FireFox 則使用 Embed 的方式嵌入網頁，也就是說會依據閱讀端所使用的瀏覽器，使用 ActiveX 或是 Plugin。

```
<object id="BSTDecoder"
        classid="clsid:12943670-5E2A-4EB3-9620-0B2C250DE3EF"'
        codebase="http://drmx.caidiy.com/bw/dl/bwplayerupdate2//bwplayer.cab
                  #version=-1,-1,-1,-1" allowScriptAccess="always"
        <embed type="application/x-BSTDecoder" id="BSTDecoder" width="0"
               height="0" allowScriptAccess="always"
               pluginspage="http://drmx.caidiy.com/bw/dl/bwplayerupdate2/">
</object>
```

圖 27 為防側錄模組在閱讀端安裝的流程圖：

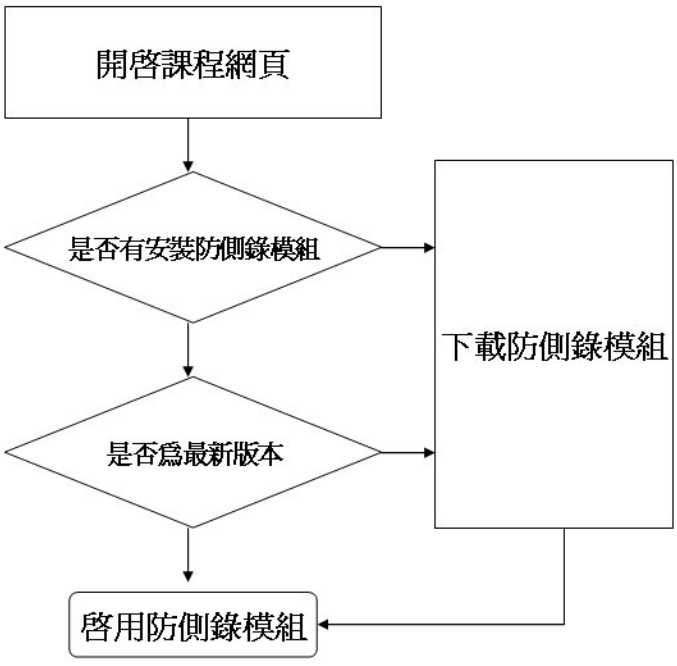



圖 27 閱讀端安裝防測錄流程圖

閱讀端開啟課程網頁之後，會先檢查客戶端電腦是否有安裝過防側錄模組，若無則會透過網路下載並安裝防側錄模組。若已經有安裝過防側錄模組，則會與 Server 端做版本的比對，若有更新版本，則會下載並安裝最新的防側錄模組，等待安裝程序都完成之後，才會啟動模組功能，並開始播放課程內容。

## 5.4 防側錄處理流程

為避免被防毒軟體視為侵害系統運作的可疑行為，因此在防側錄處理的部份，不會使用如同 API Hook 的方法，將擷取下來的畫面清除並置換成自定義的畫面內容。

當本研究之防測錄模組偵測到處理程序中有測錄軟體時，會立即通知目前正在播放課程的 Player，要求停止播放以及隱藏播放內容。

在實做的部分，則是使用網頁的 Java Script 來負責傳遞 ActiveX 以及 Flash 彼此要溝通的訊息。傳遞流程如圖 28 所示：

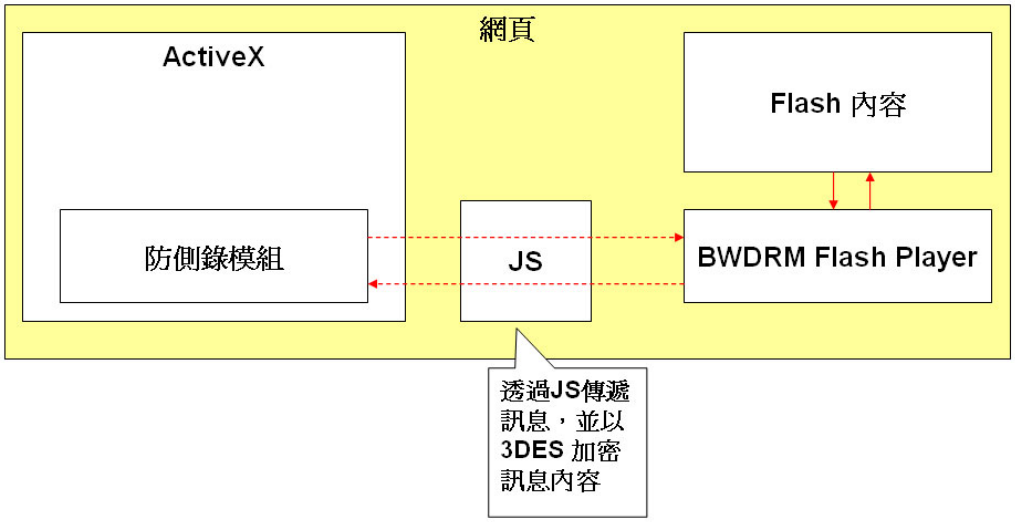



圖 28 網頁 JavaScript 訊息溝通圖

由於智勝國際科技公司所開發的 BWDRM Flash Player，是採用 Action Script 3.0 與 Flex 平台開發，其播放器為一個嵌入在網頁上的 Flash Player，一樣也是以 ActiveX 的外掛模組技術來嵌入網頁。Flash Player 目前已經在大多數的網站被使用來呈現多媒體互動效果，其中也包含很多的數位教學內容，例如巨匠、鼎新等知名教育訓練企業均建立了線上教育學習系統，其中許多課程內容都以採用 Flash 來製作，為現在網路使用者所熟知與接受的格式。

上述提到 Flash Player 也同樣是屬於網頁上的外掛模組，因此與防測錄模組相同，均可透過 Java Script 傳遞訊息，針對這項特點，因此可用來傳遞模組間的指令，達到防側錄處理的效果。於是我先在 BWDRM Flash Player 中新增一個可以停止播放並且隱藏課程內容的函式，目的在當接收到防側錄模組傳來停止播放的訊息時，可以同步進行停止播放的動作。

另外，為了 JavaScript 中傳遞訊息的安全考量，保護防側錄模組不被破解而失效，因此在傳遞訊息的內容部分使用了 3DES 的加密演算法，以加密保護溝通的訊息內容。

該加密演算法 3DES (Triple DES)是美國 NIST ( National Institute of Standards and Technology )使用三階段 DES 所改良的加密方式，是一種在電腦科學業界被廣泛應用、同時具備安全有效特性的資料加密技術，也是國內金融 IC 提款卡採用的安全機制，安全性已被最講求交易安全的金融單位驗證可行。因此，可提高訊息傳遞時的安全性。

圖 29 中簡略說明 3DES 加密的演算法流程，另外，圖 30 則為 3DES 的解密演算法流程。

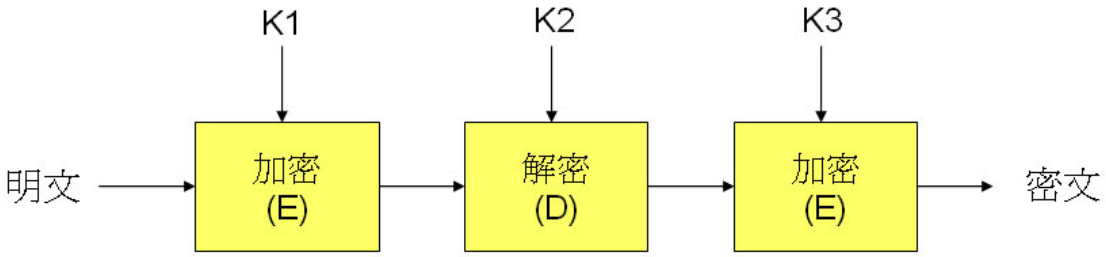



圖 29 3DES 加密

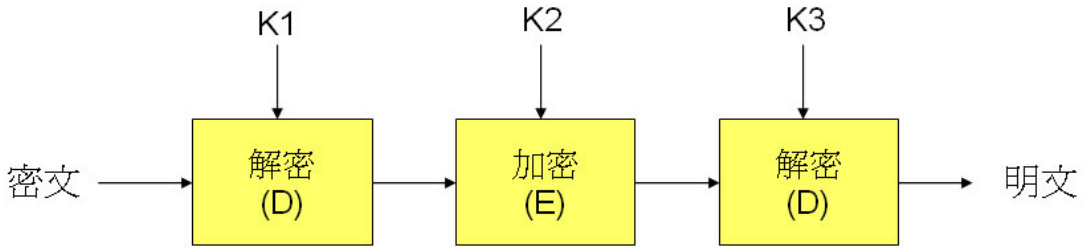



圖 30 3DES 解密

## 5.5 Linux 系統移植評估

本研究最後一個項目，則是評估本系統架構是否可移植到 Linux 中使用，首先，我先針對 Windows 與 Linux 作業系統的開發環境做比較，如表格 3 所示：

| | Windows | Linux |
|---|---|---|
| **OS** | MS Windows XP SP2 | Fedora Linux 8 |
| **可執行檔案格式** | PE 格式 | ELF 格式 |
| **Programming Language** | C/C++ | C/C++ |
| **Development Tool** | Visual Studio 2008 | Code::Blocks IDE |
| **Compiler** | C++ Compiler | GCC Compiler |
| **Browser** | IE / FireFox | FireFox |
| **執行程序列舉方式** | EnumWindows | popen |

表格 3　Windows 與 Linux 作業系統的開發環境比較表

以下則區分幾項關鍵技術來做說明，並同時評估其移植的可能性：

1. **特徵碼：**

相較於 Windows 系統中的 EXE 及 DLL 檔案，在 Linux 系統中則為 ELF（Executable and Linking Format）格式的 object file，此 object file 主要可區分為三類，「relocatable file」 是副檔名為 .o 的檔案、「executable file」 為一般的執行檔、「shared object file」 則是 *.so（shared libraries）的檔案。因此可以針對 Linux 中可執行檔的檔案格式，取出部分內容，並使用特徵碼演算法產生唯一特徵碼。

2. **列舉處理程序：**

Linux 是一個多工的作業系統，所以允許多個使用者同時對系統進行存取，而每個使用者也可以同時執行多個程式。在列舉處理程序的方法，Windows 中是使用了 Windows API「EnumWindows」來列舉，在 Linux 中並不支援這項功能，但可以使用 popen 的指令來列舉，執行「/bin/ps ax」指令，並建立讀取管道，將執行結果存入 buf 變數中，例如以下方程式碼所示：

```
fp = popen( "/bin/ps ax |grep -v 'grep'|grep -v 'ax'|grep -v 'print'|awk '{print
$1, $6}' ", "r" );

ch = fread( buf, sizeof(char), buffer_size, fp );
```

### 3. 特徵碼資料庫更新與讀取機制

資料庫更新與讀取機制的部分，本研究特徵碼資料庫檔案為 XML 格式，所以在 Windows 系統中使用 XML Parser 的工具取得資料庫內容，而在 Linux 中，同樣也可以使用 XML Parser 的工具。

### 4. 特徵碼比對

比對的機制部分單純只用到字串比對的函式，主要是比對資料庫中的特徵碼字串與系統中是否符合。

由以上針對模組中關鍵功能的分析與比較，由於本研究所使用到的方法，與系統核心運作機制無關，因此，在移植的工作部分，可以簡化複雜度以及增加可移植性。

# 六、 應用範例

在 6.1 節會針對「課程保護」與「防側錄模組」的操作方式作說明，其中包括「製作端」與「閱讀端」。在本章節的應用範例，將會使用智勝國際科技公司開發之 BWDRM Flash 來做展示與說明。

## 6.1 課程保護的操作方式

首先展示「製作端」的轉檔工具部分，此工具的主要用途是將未加密的 Flash 網頁，透過 DRM 保護的加密機制，轉換成一份加密的 BWDRM Flash 網頁，在本研究的實做中，我在轉出的網頁中，加入防測錄外掛模組的嵌入語法，使此網頁不僅受 DRM 保護之外，同時也擁有防測錄的能力。接下來，我以一步一步的方式，來說明如何使用轉檔工具轉出一份含防測錄模組的 Flash 加密課程網頁。

首先先開啟 BWDRM Flash 轉換工具，如圖 31 所示：

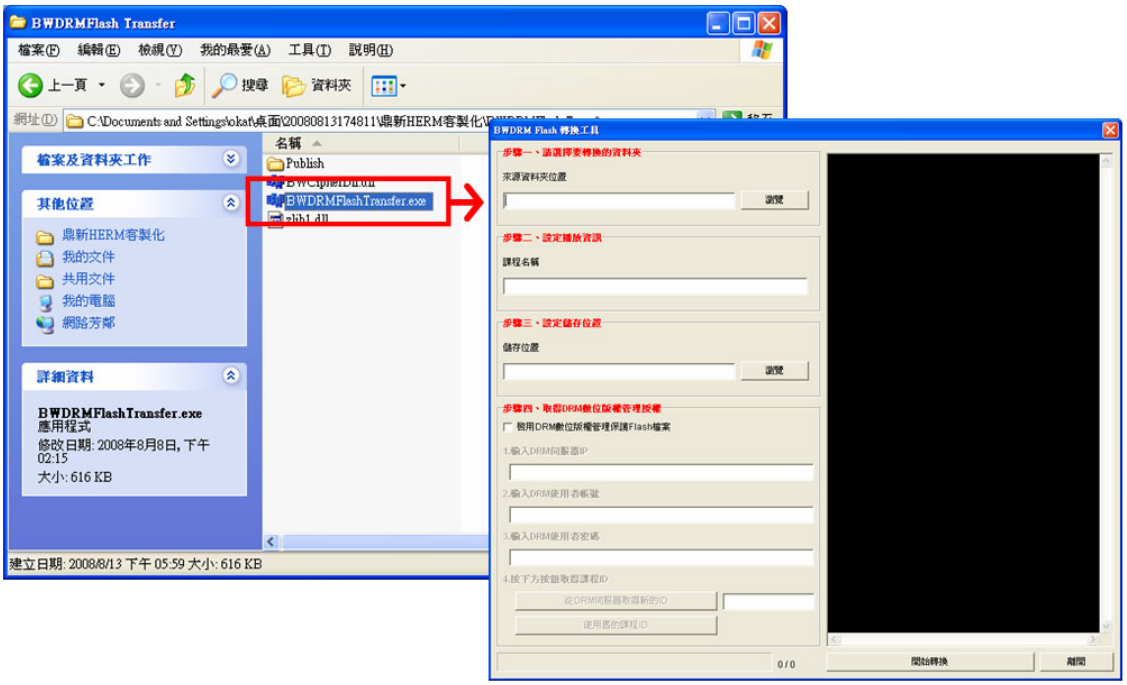



圖 31 開啟 BWDRM Flash 轉檔工具

接著依照圖 32 中的步驟編號操作。

步驟一、選擇來源的資料夾位置。
步驟二、設定課程名稱。
步驟三、選擇輸出的資料夾位置。
步驟四、啟用 DRM 相關設定，若不需要 DRM 帳密認證，此選項則不需要設定。
步驟五、開始轉換。


圖 32 轉檔步驟圖

轉檔完成之後，可以先看原始資料夾內容，如圖 33 所示。

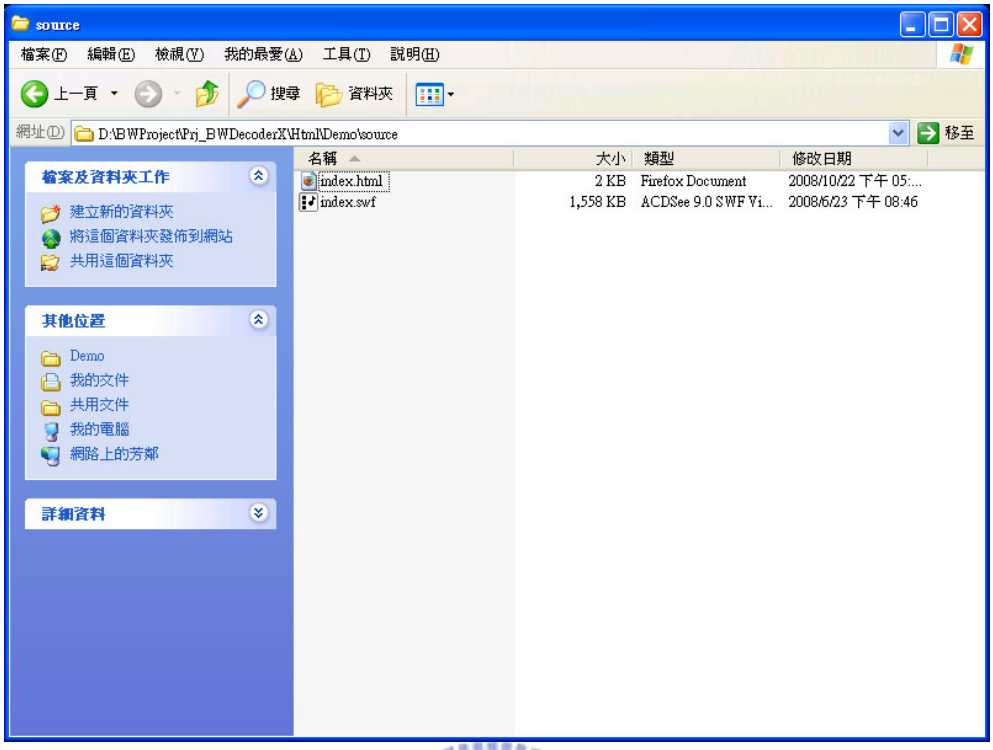



圖 33 未加密保護前的來源資料夾

而在轉檔輸出的資料夾中，可以看到相關模組的檔案，包含解密模組與 JS 檔。其中 pagecom1. js 即為用來寫入嵌入語法的 JS 檔案，如圖 34 所示。

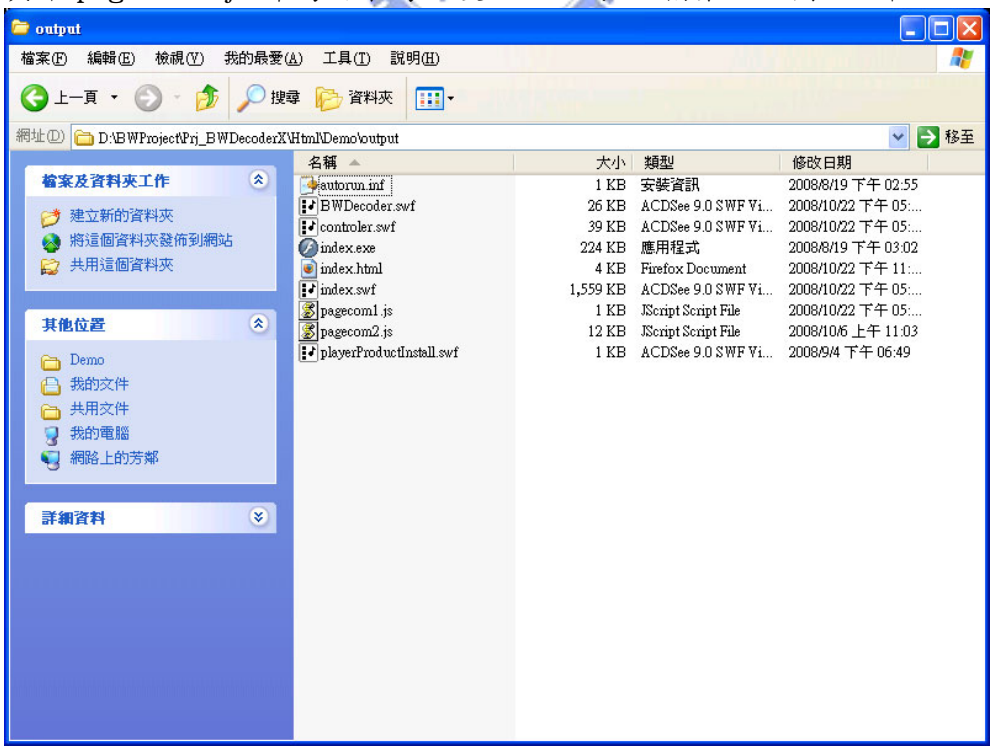



圖 34 加密保護後的輸出資料夾

## 6.2 安裝防側錄模組

完成了 6.1 節中的操作之後，會產出一份受保護的網頁課程。接下來就要實際開啟課程，並使用側錄軟體來測試模組的功能，同時也會使用 VNC 遠端遙控軟體，來測試是否可以有效阻擋。

於是，我在一台乾淨的電腦環境中以 IE 瀏覽器開啟該課程網頁，由於網頁中含有 ActiveX 等控制項，因此會出現圖 35 中的黃色提示列。

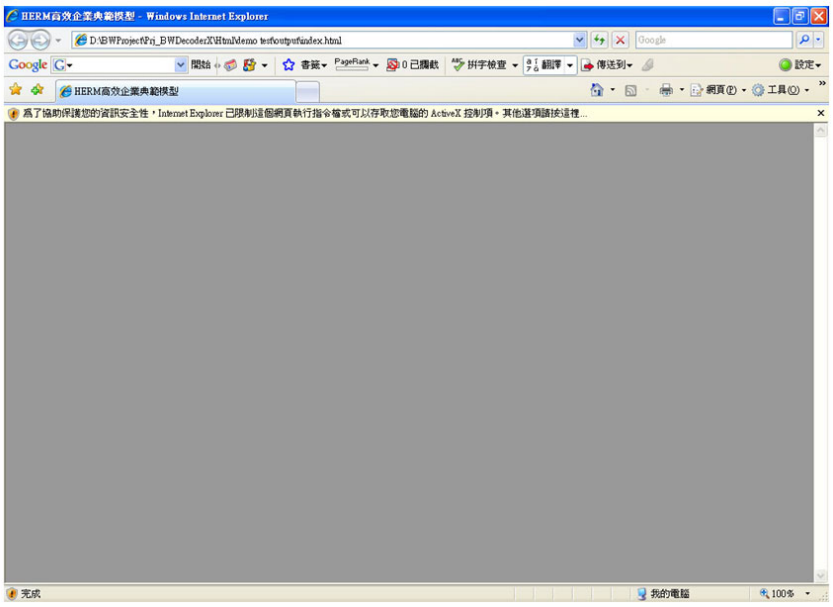



圖 35 出現網頁提示列

接著點擊黃色提示列，選擇允許被封鎖的內容，如圖 36 所示。

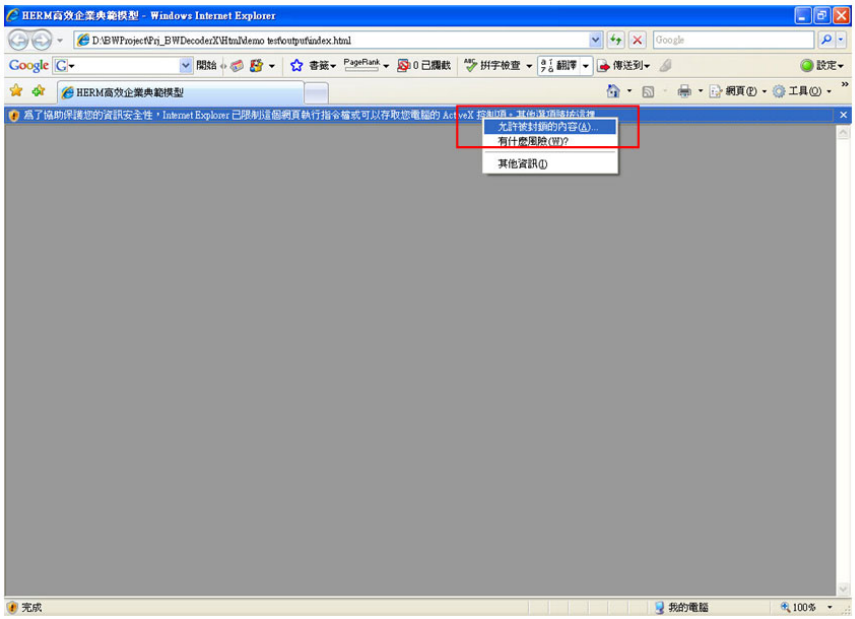



圖 36 允許被封鎖的內容

由於是第一次觀看此課程，因此會要求下載防側錄模組，如圖 37 出現確認安裝的視窗，點選安裝按鈕後，開始安裝防測錄模組控制項。等待下載以及安裝完成之後，即會開始播放課程內容，並同時啟動防側錄模組。

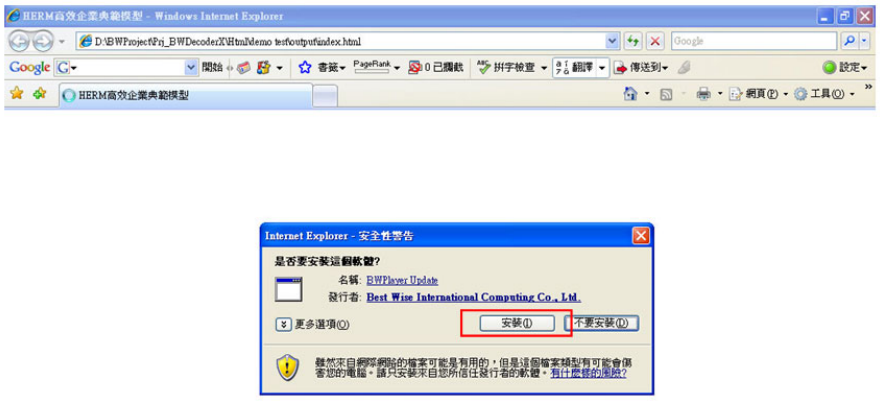



圖 37 安裝防測錄模組

接下來我使用另一種瀏覽器「FireFox」來確認防測錄模組的安裝。如圖 38 所示，因為尚未在 FireFox 中安裝防測錄模組，因此會先要求使用者點擊連結並下載安裝檔。

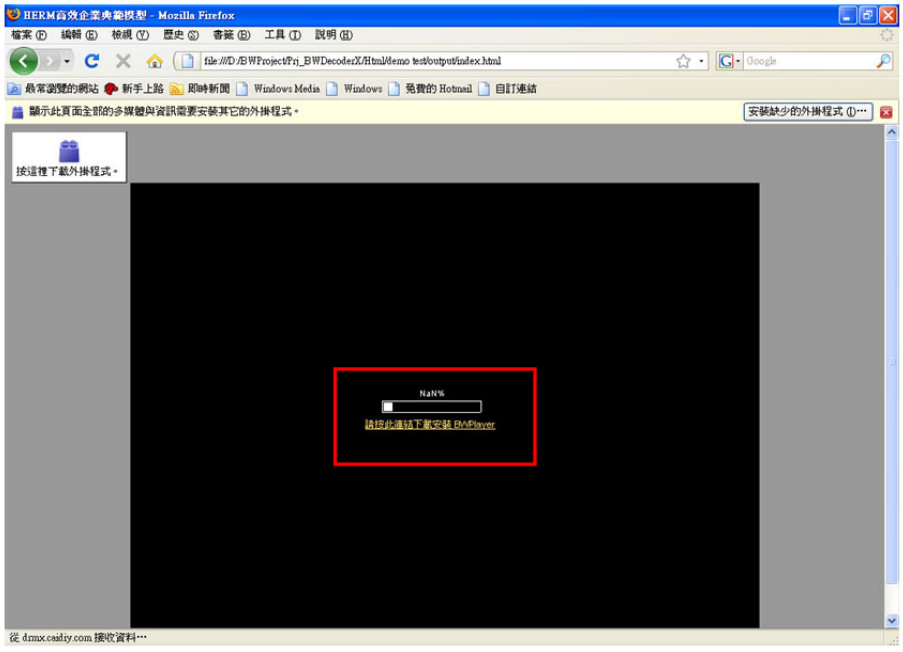



圖 38 在 FireFox 中開啟課程網頁

下載完畢之後，點選開始安裝。

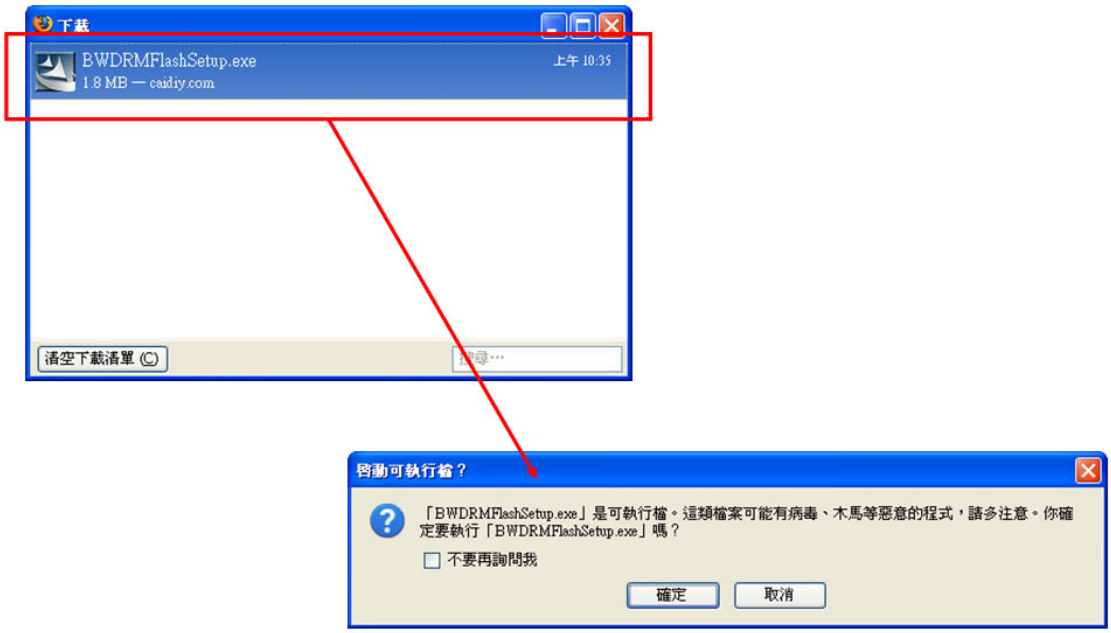



圖 39 安裝檔下載完畢

開始安裝防測錄模組，如圖 40 所示，待安裝完成之後，即可開始在 FireFox 上觀看課程內容。

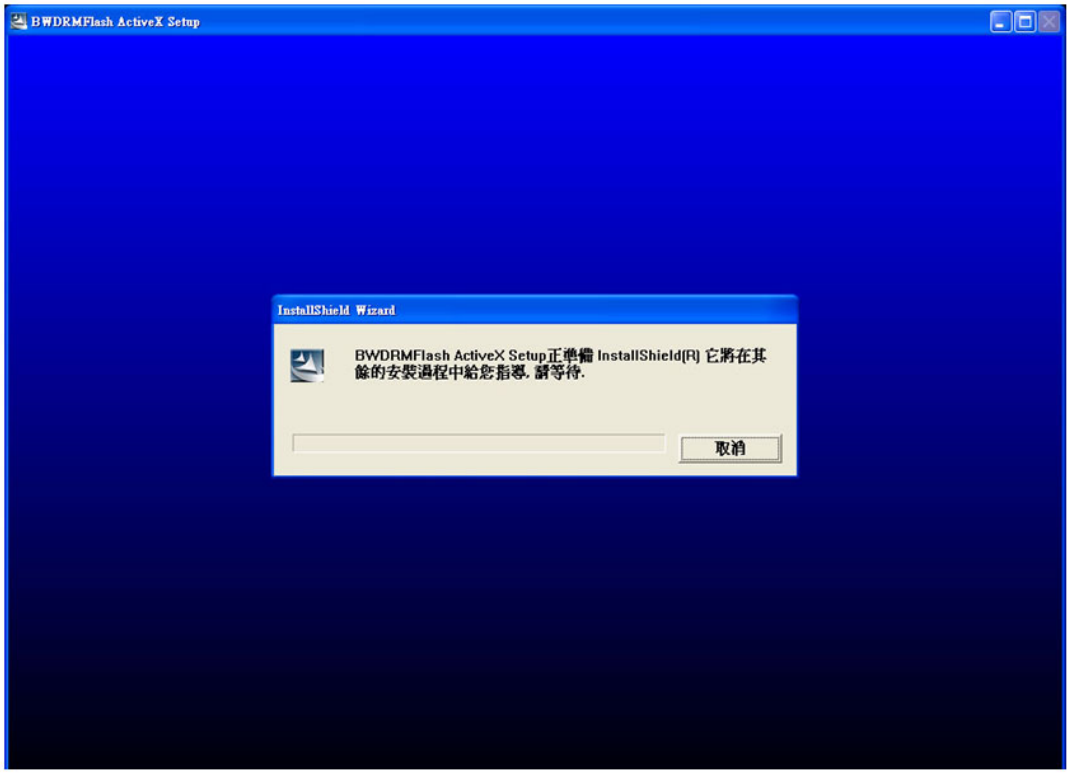



圖 40 防測錄模組安裝過程

## 6.3 防側錄模組的驗證

在本節中，我使用 IE 瀏覽器開啟課程網頁，並實際用一套桌面及視窗錄影程式，以及 VNC 遠端遙控軟體，來驗證本研究防測錄模組的功能。

首先使用桌面及視窗錄影程式來驗證防測錄模組是否可以有效阻擋。這次的實驗中我使用的是 MWSnap 3.0.0.74 這套軟體，這套軟體的主要功能可以指定桌面上某一特定區域或是指定視窗，將指定範圍內的畫面擷取至該軟體中，該軟體介面如圖5所示。

在實驗開始之前，先依照「5.1 特徵碼實作」的操作說明，新增該軟體的特徵碼到特徵碼網路資料庫中，新增後如圖41所示

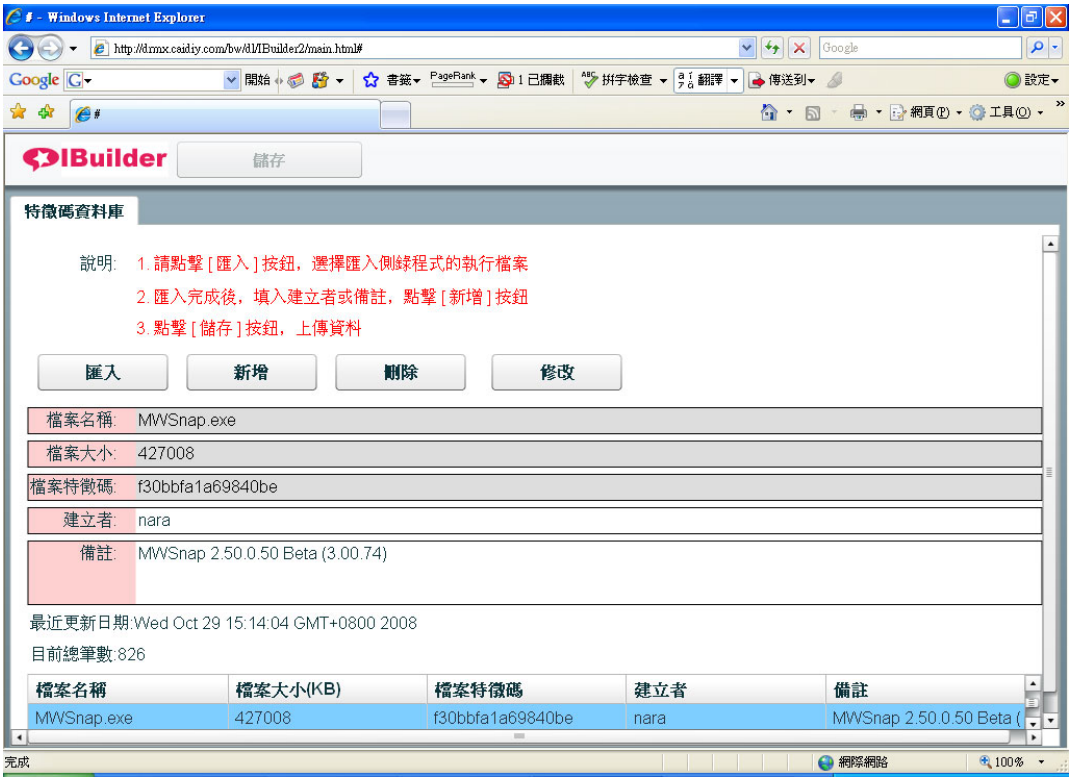
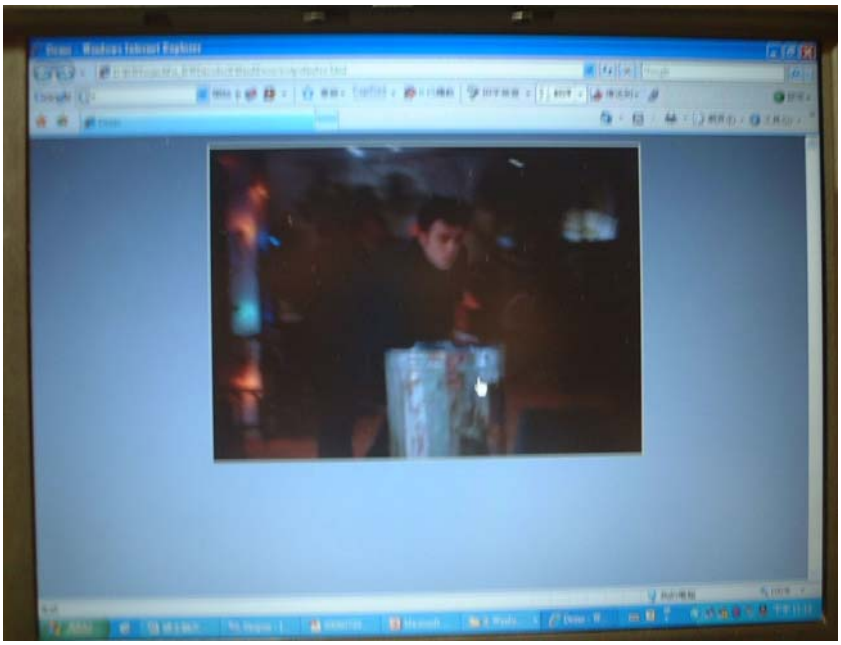



圖 41 新增 MWSnap 3.0.0.74 至特徵碼資料庫

接下來開啟事先使用轉檔工具轉出的課程網頁，此網頁也包含了本研究之防測錄模組。接著先開啟網頁，因尚未有任何的測錄軟體正在執行中，所以可正常觀看課程內容，如圖 42 所示。



開啟課程網頁

接著開啟 MWSnap 3.0.0.74 軟體，此時防測錄模組發現有測錄程式正在執行中，則會立即關閉課程內容，如圖 42 所示，提示使用者先將側錄軟體關閉之後，才可繼續觀看課程內容。由此可以驗證本實驗之防測錄模組可有效阻擋桌面或視窗錄影程式

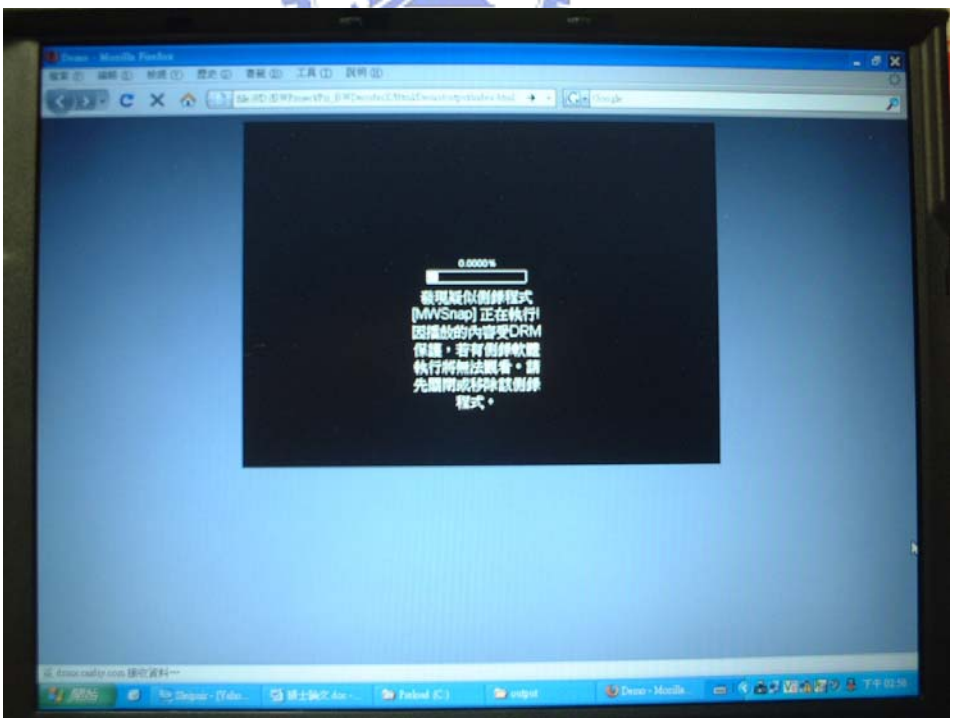



圖 42 發現 MWSnap 3.0.0.74 並關閉課程

接下來，繼續使用 VNC 這套遠端遙控軟體來驗證防測錄功能，而在驗證之前，同樣也是將該軟體特徵碼新增至網路資料庫中，並開啟同一份課程網頁。在還沒開啟 VNC 軟體之前，課程均可正常觀看，如圖 43 所示，而開啟 VNC 之後，則防測錄模組會立即阻擋並關閉課程內容，執行的結果如圖 44 所示。

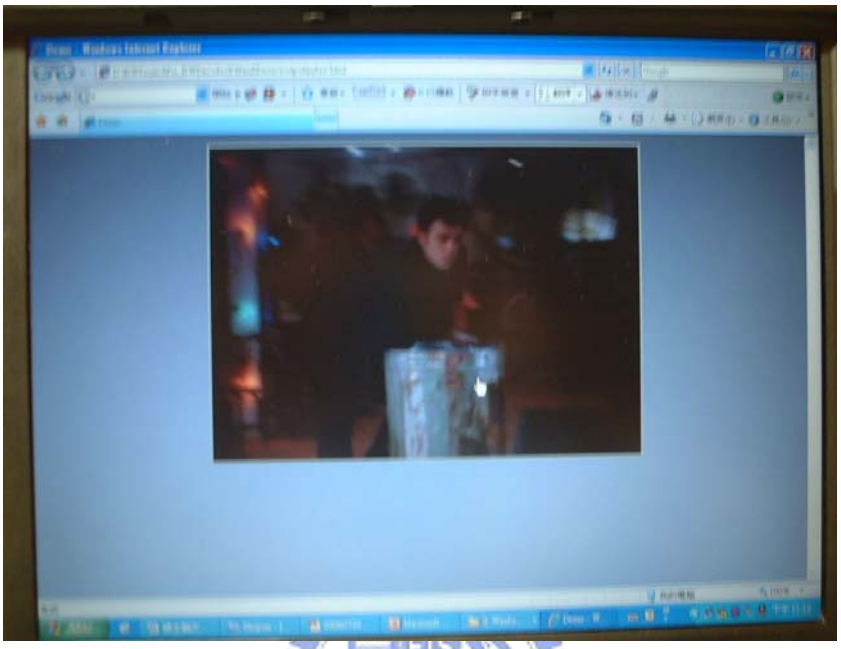


圖 43 未開啟 VNC 時，可正常觀看課程內容。

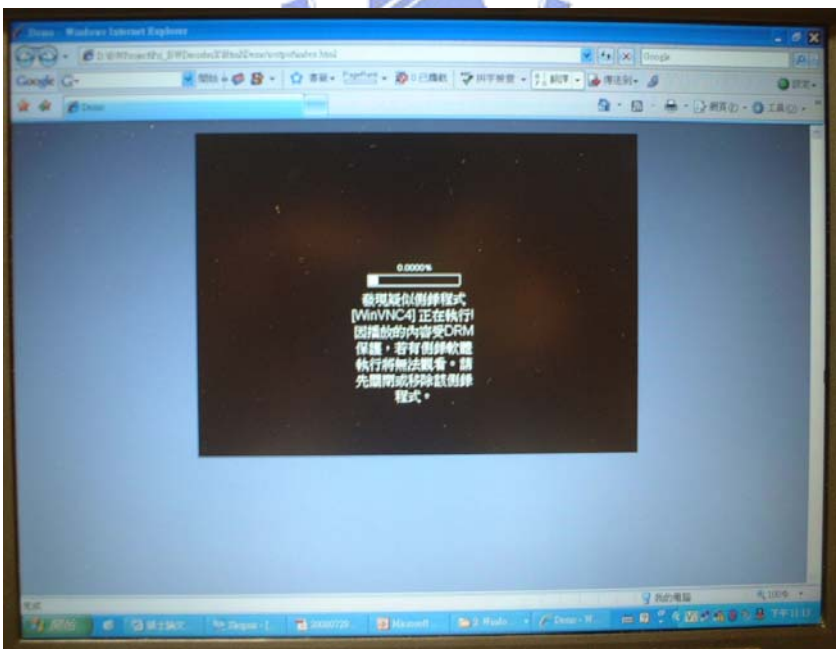



圖 44 開啟 VNC 後，防側錄模組通知播放器關閉播放內容

由以上的實驗得知，本研究的方法，除了可以阻擋原本 API Hook 可阻擋的測錄軟體之外，同時也可有效阻擋使用遠端遙控軟體的方法來測錄課程內容。

# 七、 結論與未來展望

## 7.1 總結

本研究所提出的防測錄機制，目的在

- 以特徵碼資料庫的比對方式，補強單純使用 API Hook 技術的防側錄模組，可阻擋遠端遙控軟體側錄課程播放內容。
- 解決原本以 API Hook 技術之防測錄系統，被防毒軟體視為可移行為，造成課程無法正常閱讀的問題。
- 評估本研究的系統架構可移植到 Linux 中執行。

在這次的實驗中，除了使用桌面及視窗錄影軟體之外，也使用了遠端遙控軟體-「VNC」，來驗證防測錄模組的阻擋效果，卻發現 API Hook 技術並非可以完全阻擋所有的側錄方式。主要原因是遠端遙控軟體的螢幕擷取方式並非使用 Windows API，而是使用了Mirror Driver的技術將畫面傳送到另一台電腦，當接收端收到傳送端畫面之後即可執行測錄軟體，並且避開防測錄模組的偵測，取得課程的畫面內容，實驗結果也確認了 API Hook 技術缺乏阻擋這類型側錄方式的能力。

為了解決以上的問題，因此在本研究中提出了一個結合特徵碼概念與資料庫比對的方式來改善問題。

由於 Windows 是一個多工處理的系統架構，可允許多個處理程序同時進行，這些執行程序的資訊，均可以用「Windows工作管理員」中的「處理程序」來檢視，因此每一個正在執行中的程式，都可以取得他的處理程序以及執行檔案的位置，包含本實驗中測試用的桌面及視窗錄影軟體，亦或是遠端遙控軟體等等。根據這項系統的特點，所以當本論文之防測錄模組啟動時，便能立刻列舉閱讀端所有正在執行中的處理程序，並計算出該執行檔案的特徵碼，最後再與特徵碼資料庫的內容做比對。

以上的作法可有效解決課程觀看的同時，偵測是否有不允許的軟體正在執行測錄行為。另外，因本研究之方法未牽涉到存取及修改重要系統資訊的動作，因此不會被防毒軟體視為可疑行為，而造成課程無法正常閱讀的問題。

而在評估系統架構是否可移植到 Linux 平台中的需求部分，在「5.5 Linux 系統移植評估」中則是針對系統模組中每一項關鍵的功能模組來做各別的評估。

最後，再以本研究所使用的特徵碼資料庫比對方式，與 API Hook 的技術做比較。比較的結果如下表所示：

| | **以特徵碼資料庫方式** | **API Hook方式** |
|---|---|---|
| **Window側錄軟體阻擋效果** | 依據資料庫中所建立的特徵碼項目。 | 可有效阻擋使用Windows API 側錄的行為。 |
| **遠端遙控軟體阻擋效果** | 可阻擋。 | 無法阻擋。 |
| **是否可移植 Linux** | 可移植系統架構。 | 無法移植。 |

表格 4 特徵碼資料庫比對方式與PI Hook 的技術做比較

## 7.2 未來展望

在這一節會針對本研究所提出的防測錄機制，提出一些功能的補強、與未來發展的方向：

- 研究的結果顯示，這兩種防測錄的做法，皆有其優缺點存在，例如特徵碼資料庫適合於非使用 Windows API 擷取畫面的側錄方式，API Hook 則相反，未來可深入研究兩套方式在防測錄中交互運作，以期能加強防測錄的效果。

- 本研究中實驗的對象為網頁的格式，並以外掛功能將模組套用在課程中，未來可發展各種不同格式的外掛模組，例如 Word、PowerPoint等等需要保護的文件。

# 參考文獻與資料


[1] LUEVELSMEYER, PE Format, [On-line].Available: http://webster.cs.ucr.edu/Page_TechDocs/pe.txt

[2] MSDN, Peering Inside the PE: A Tour of the Win32 Portable Executable File Format (M. Pietrek), in: Microsoft Systems Journal 3/1994, [On-line].Available: http://msdn.microsoft.com/en-us/library/ms809762.aspx

[3] Randy Kath , The Portable Executable File Format from Top to Bottom, [On-line].Available: http://www.csn.ul.ie/~caolan/publink/winresdump/winresdump/doc/pefile2.html

[4] Windows NT System-Call Hooking, Mark Russinovich、Bryce Cogswell,Dr. Dobb' s Journal January, 1997

[5] John Robbins, Debugging Applications: Microsoft (Dv-Mps Programming) (Paperback)

[6] 史萊姆論壇, 詳談 HOOK API, [On-line].Available: http://forum.slime.com.tw/thread167916.html

[7] 趨勢科技, 防毒入門-基本概念-認識病毒碼與掃瞄引擎, [On-line].Available: http://www.trend.com.tw/corporate/security/virusprimer_2.htm

[8] 智勝國際科技, BWDRM Flash, [On-line].Available: http://www.caidiy.com/

[9] Joey Lott, Darron Scball & Keith Peters, ActionScript 3.0 Cookbook, O' REILLY

[10] 網核股份有限公司, OSafe 企業內容安全方案, [On-line].Available: http://www.iisc.com.tw/Product/Security/OsafeMirage/tabid/50/language/zh-TW/Default.aspx

[11] TrustView inc., TrustView DRM for Office, [On-line].Available:
http://www.trustview.com.tw/default.aspx?tab=product_office

[12] 以柔資訊, [On-line].Available:
http://www.wnjsoft.com/index.php

[13] DLL Injection, [On-line].Available:
http://en.wikipedia.org/wiki/DLL_injection

[14] VNC Mirror Driver, [On-line].Available:
http://www.tightvnc.org/driver.html