以行動代理人為基礎之多 host-based IDSs 之入侵偵測平台

林宣燁　　李忠憲
國立成功大學電機系
q3693164@ccmail.ncku.edu.tw jsli@mail.ncku.edu.tw

摘要

惡意攻擊者入侵的第一個步驟為掃描端口(port scan)，因此只要能即早且正確地偵測 port scan 行為就能即早防範入侵行為。然而為了躲避防火牆、入侵偵測系統，因此產生了許多種能閃避偵測的 port scan 方式，例如：slow scan、distributed scan……等。Slow scan 顧名思義就是放慢掃描的速度以躲避偵測，而 distributed scan 則是將掃描行為執行於多台電腦之上。我們提出一套使用行動代理人偵測 slow scan、distributed scan 的入侵偵測系統且實作之，並證明行動代理人移動的速度與偵測效能的關係。

關鍵詞：入侵偵測系統、行動代理人、安全、slow scan、distributed scan。

1. 前言

網際網路時代的來臨使得資訊廣泛且快速地散佈、e 化政府、e 化交易……等新的服務模式也隨之出現，理所當然地也吸引不少惡意攻擊者。惡意攻擊者若欲入侵主機，首先會做 port scan 的動作，以得知哪部主機的哪一個 port 是開啟的，接著惡意攻擊者即可得知該主機提供何種服務、何種作業系統，最後惡意攻擊者會根據此資訊去執行探測程式，進而獲得管理者的權限。簡言之，惡意攻擊者入侵可分為下列幾個步驟：port scan、入侵並常駐於系統，最後植入特洛伊程式並清除相關記錄，此時該主機就被惡意攻擊者操控。現今愈來愈多新的 port scan 方式用來閃避偵測行為，例如：stealth scan、sweep、slow scan、distributed scan…等，使得偵測的過程愈來愈困難且複雜，換言之即入侵行為愈來愈難偵測。

從演算法的觀點來看，一般的入侵偵測系統(Intrusion detection system，簡稱 IDS)可分為二類，一是 misuse detection，另一是 anomaly detection。前者以攻擊特徵為基底並利用此與審核資料(audit data)比對，若審核資料與攻擊特徵相符即為一攻擊行為，因此其誤判率較後者低；後者則是建立正常行為的檔案，而這些正常行為的資料通常是由統計的方式獲得，若一行為並未出現與此檔案中，則視此行為為攻擊，故其漏報率較前者低，此外，由於電腦與網路的使用日益普及，須儲存的審核資料量亦隨之大增，進而造成儲存空間不足、系統資源不足…等困擾，因此減低資料量為一刻不容緩的課題。

由架構的觀點，IDS 則可分為 Network-based IDS(NIDS)、Host-based IDS(HIDS)、hybrid IDS，NIDS 分析封包進而得知網路行為是否異常，HIDS 則是檢視稽核檔以得知本地是否有異常狀況發生，hybrid IDS 則是混和上述二種 IDS，目前企業或機關則是將這些不同種類的入侵偵測系統佈署於企業網路中的不同的地方。並且利用階層、集中式的架構進行分析。所謂階層是指最底層的 IDS 將搜集到的資料進行分析，並將初步分析結果以及無法以單一台 IDS 判斷的資料交于第二層的 IDS，第二層再交于第三層…一直到最上層。而所謂的集中是指整個入侵偵測系統最上層的那部主機，我們姑且稱之為中控系統，其負責最終的搜集、分析、產生警告以得知整個企業網路安全的狀況。此網路架構至少有二項缺點：一是監視範圍無法括大[9]，另一是若中控系統被攻毀，則整個 IDS 將完全垮台。

行動代理人(mobile agent，簡稱 MA)為一種可以在網路上任意移動，並可以在到達執行地點後決定自己的行為的程式。MA 除了本身程式碼的移動，亦可將程式執行的狀態，甚至整個物件移動，故其具有機動性。因此愈來愈多研究著重於如何將 MA 應用於網路管理的領域上。由於 MA 需於網路中穿梭，因此每部主機必須要有一共同平台使行動代理人能夠執行與移動，例如 JAVA 平台就是一理想的行動代理人平台。

如果我們能即早且正確地偵測出 port scan 的行為，我們即可將入侵行為所造成的傷害減到最低。圖 1 為一最基本的掃描方式之交握圖，稱為 vanilla TCP connect() scanning，其缺點為容易被入侵偵測系統或防火牆…等發現。圖 2 為一種 stealth scan 之交握圖，是利用錯誤的封包來判斷主機的埠號(port)是否開啟。圖 1、圖 2 中的 x、y 代表攻擊者發出封包的 port，其為任意值，N 則是攻擊者欲掃描的 port。

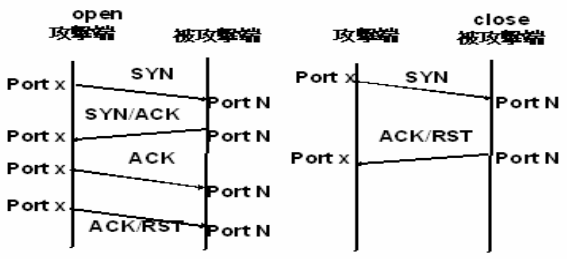

圖 1 vanilla TCP connect() scanning

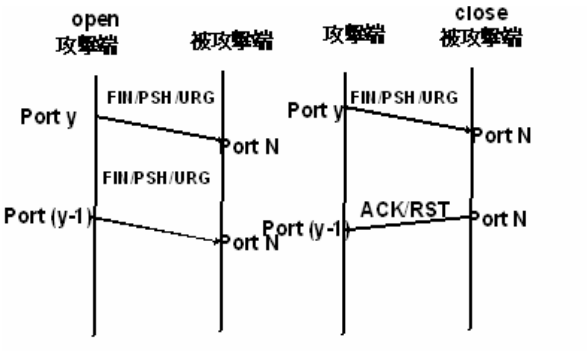

圖 2 TCP Xmas scanning

常見的掃描工具有：nmap、SAINT、nessus。其中最主要的掃描工具是 nmap。在執行掃描時，nmap 會先丟出 ACK 封包以判斷該端口為 unfiltered 或者是 filtered[1]，若該端口對掃描者做出回應，但掃描者確無法判斷出該端口狀態，則此端口會回傳一個 RST；若該端口被防火牆阻擋，則此端口不會回傳任何訊息給攻擊者。因此，nmap 掃描時，這個前置的 ACK 封包是為了確定該端口有沒有被防火牆擋掉，由於 port 80 通常不會被防火牆阻擋，因此 nmap 假設 port 80 沒被擋掉，則該主機必存在。確定好欲掃描之主機存在之後，nmap 便會依照掃描者指令執行不同的掃描行為。

我們可將 slow scan、distributed scan 甚至是綜合上述二種方式的掃描行為以集合 S={p,h,i,f}表示，p 代表掃描者欲掃描的埠號(port number)，h 代理掃描者欲掃描之主機(host)，i 則是每一次掃描之時間間隔(interval)，而 f 為掃描者停止掃描的方式、原因。前三個參數分別可能為隨機或擁有某些特定規則，以參數 h 為例：假設有五台主機，編號分別為 1，2，3，4，5，則掃描者可能的掃描順序為 5533221144 如此重複下去。至於 f 參數則有相當多種可能，例如「無論是否掃到 open 的 port，皆於指定時間(或次數)結束掃描」。若能在這些參數中找到特定規則，則可以加速偵測 port scan 行為，找到愈多參數的特定規則，對整個偵測的效能愈有幫助，但若為隨機的方式則無法利用此方式加快偵測速度，換言之，如果參數為特定規則，我們就可以縮短 port scan 攻擊時間上的定義，故能提早偵得攻擊行為。本篇論文主要探討前三個參數皆為隨機的情況、無法利用某些特定規則預知行為的情況下之 slow distributed scan 行為。

2. 相關工作

以 MA 做為 IDS 最少有下面五項好處：(1)由於行動代理人可於網路上任意位置移動，因此惡意攻擊者不容易找出 agent 的確切位置，然而靜態的 IDS 卻容易受到惡意攻擊者的攻擊[5][4]，(2)可應用於不同的電腦網路，例如：wireless ad hoc network、sensor network…等網路，(3)更新客戶端軟體時可直接由被允許的主機進行更新，而不用到需更新軟體的主機做更新的動作[7]，(4) 當主機被攻擊或者系統資源減少時，在本地行動代理人可以離開此地，在外地的行動代理人則避免前往[4]，(5)可括大整個入侵偵測系統之監視範圍[9]。

GrIDS[8]為一種以圖形為基底的 IDS，其中節點(node)代表主機(不一定只有一部主機)，箭號則是代表一些在節點之間的網路流量，它可利用上述元素為立活動圖形偵測 port scan 行為，並判斷是否為惡意掃描(如：掃描次數是否大於臨界值)，但此 IDS 無法偵測 stealth scan。

Snort[6]為一種相當普及的 NIDS，其對 port scan 的定義為：在 y 秒內觀察到從同一個 source IP 對任意主機、任意 port 發出 x 個 TCP(或 UDP)封包，其中 x、y 為使用者自訂的參數。但若掃描者延長掃描時間，並分散掃描行為於不同網段的主機，則無法偵測出，亦即 Snort 無法偵測出 distributed scan 以及 slow scan[2]。

基於上述理由，我們提出一套使用 MA 來偵測 slow、distributed scan 的方法，並實作之。

3. 架構與實作

首先我們於各網段的入口架設防火牆(iptables)，並開啟封包記錄的功能，因此所有流經此防火牆之封包將會被記錄下來。同樣地，我們將 MA 的平台架設於各網段的入口，也就是 MA 平台與防火牆架於同一部主機之上。

仿照[3]，我們提出由七個變數所組成的 capability：

Capability(src, dst, src_port, dst_port, protocol, flag, num)

這七個變數分別為來源端 IP、目的端 IP、來源端埠號、目的端埠號、通訊協定、旗標、掃描次數，每個攻擊狀態可由一個或數個 Capability 所組成。由於瞬時封包數量的多寡對 IDS 判斷上會造成影響，因此若以 time-based 的方式判斷各封包會很容易造成 IDS 偵測上的錯誤，如果利用 finite state 的方式來判斷掃描行為即可比前者降低其誤判率(false positive)以及漏報率(false negative)。目前我們能判斷出的掃描方式有七種，我們可將這七種掃描方式分成 S0-S10，共 11 個狀態，其中 S0 表示掃描行為最初的狀態，S10 則是確認此掃描為一攻擊行為，S1 為七種掃描所共有的行為，S2-S8 可判斷出不同的掃描行為，S9 則是用來判斷掃描行為是否為攻擊。

我們創造出 PortScanManager agent、AlertAglet

agent 、PortScanIDS agent 、GoorStop1 agent 、GoorStop2 agent 等五個 agent。前二者為可移動的代理人，後三者為駐守於本地之代理人。PortScan Manager agent 之主要功能為到各主機上搜集 port scan 攻擊偵測之初步分析結果，並判斷、儲存(或暫存)port scan 攻擊之狀態，當 port scan 的次數到達臨界值時，立即產生一個全新的 MA(即 AlertAglet)，AlertAglet 馬上前往網管人員所在之主機，並通知網管人員目前受到哪個 IP 的攻擊。PortScanIDS 則是利用 finite-state 為基底進行 port scan 之偵測行為，可判斷出哪個 IP 進行何種掃描動作，並將初步分析之結果傳給 PortScanManager agent。

由於封包數量可能瞬時暴大量，因此一個掃描行為所產生的各封包可能不會同時產生，若 MA 於這些封包所產生之時間差內離開，那麼 MA 便無法立即觀測出攻擊行為，MA 必須等待下一次回來此地時才能判斷出有掃描的行為。有鑑於此，我們提出了 GoorStop1 agent 以及 GoorStop2 agent 幫助 PortScanManager agent 判斷是否該離開本地。但為了預防封包遺失等造成攻擊狀態停留…等因素，因此 PortScanManager agent 的等待必須有時間限制，以免 MA 停在本地而不離開。

PortScanManager agent 可由 LocalWork 、ReadSecure、ReadState1、ReadState2 四個子物件組成。LocalWork 顧名思義，就是在本地端所需執行的工作，它會呼叫另三個物件並得到其回傳之訊息，若為須儲存的訊息則由代理人本身攜帶著。ReadSecure 負責呼叫 PortScanIDS，並可判斷同一 IP 掃描行為是否在特定時間內(如 3 天)超出臨界值(如 10 次[2])並發出 AlertAglet agent。至於 ReadState1 以及 ReadState2 這二個物件則是在當攻擊行為還停留在 finite-state 的流程之中才會啟動的物件，簡單來說就是此行為非正常行為，也非攻擊行為，而是一個未完成的攻擊行為，它們會呼叫 GoorStop1 agent、GoorStop2 agent，其中 ReadState2 可判斷臨界值並發出 AlertAglet agent。

當 PortScanManager agent 到達本地時，首先執行 LocalWork 物件，接著 LocalWork 會呼叫 ReadSecure 以獲得初步分析結果以及判斷是掃描次數是否到達臨界值(狀態 S10)，若達到臨界值即立刻產生 AlertAglet agent 以發出警告。之後，ReadSecure 便向 LocalWork 回報攻擊狀態，若攻擊狀態為 S0，LocalWork 會呼叫 ReadState1，若為 S1 則呼叫 ReadState2，接著 ReadState1、ReadState2 向 LocalWork 回報目前攻擊的狀態，以及停留次數(詳細說明於下一段)，最後 PortScanManager 由子物件 LocalWork 獲得狀態儲存值(若有必要)或暫存狀態。

為了預防狀態停留，因此必須要有停留次數的限制(例如 5 次)，每次停留也要有一定的時間(例如 10 秒)，我們姑且稱停留的次數為「多停留次數」，停留的時間為「多停留時間」。換句話說，PortScanManger 需要「多停留時間」、「多停留次數」的目的是為了加快偵測速度，而限制「多停留時間」以及「多停留次數」也是為了加快偵測速度。整個判別的流程如圖 3 所示，我們可以清楚地發現，只要不是攻擊行為或正常行為都會進入此機制，當 GoorStop1 agent 以及 GoorStrop2 agent 執行後，「多停留次數」會主動加 1，若能於限制的「多停留次數」之內完成偵測，則此狀態暫存值會馬上清除，若無法於限制次數內完成，則此狀態會被記錄起來，待下次 PortScanManager agent 回到本地後再重複圖 3 之流程。

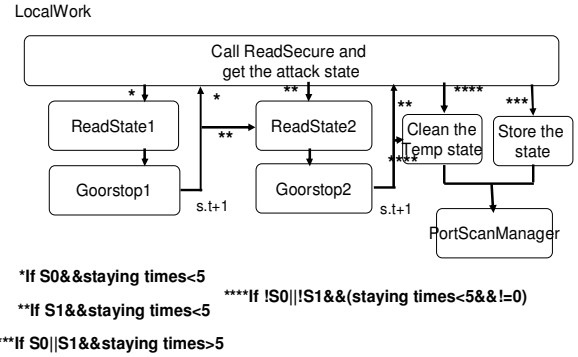

圖 3 PortScanManager 判斷是否離開本地之方式

PortScanIDS agent 為一靜態的代理人，駐守於各網段入口，亦即防火牆所在之主機。它可以由 iptables 所產生的 log 檔加以分析得知是否有掃描行為?若有，為何種掃描行為?為了避免 log 檔過大，因此 iptables 的 log 檔必須常清除，因此為了偵測 slow scan，我們必須將這些封包儲入資料庫。PortScanIDS agent 會由 log 檔開頭讀資料，若此資料已被存入資料庫表示此筆記錄已分析過，若資料未存入資料庫，則 PortScanIDS agent 會將此筆記錄丟入資料庫，並進行分析。

GoorStop1 agent、GoorStop2 agent 亦為一靜態代理人，駐守於各網段入口。一開始它們會接受由 PortScanManager agent 所傳來的命令，接著判別目前的攻擊狀態，並將攻擊狀態告知 PortScanManager agent，此時 PortScanManager agent 即可利用此資訊決定其下一步動作。GoorStop1 agent 以及 GoorStop2 agent 的差別在於所能判別的狀態不同，後者主要用來判斷「掃描行為」的狀態，前者則是掃描的前置狀態。將二個功能類似的代理人分開為二個不同代理人是為了更落實分散式系統的優點。

AlertAglet agent 為一警告代理人，當 PortScanManager agent 察覺同一 IP 的掃描次數於限定日期內超過臨界值，PortScanManager agent 會將此臨界值歸零，並發出 AlertAglet agent。AlertAglet

agent 隨即前往網管人員主機，並告知網管人員有 port scan 行為發生，以及攻擊者的 IP 為何。由於將 port scan 行為的時間限制延長，故能偵測出 slow scan，而且因為將偵測系統架於各網段入口，並利用行動代理人做通訊的工作，故能測得 distributed scan。

4. 實驗與結果

我們利用成大計算機與網路中心的測試平台架設環境，此環境由五個網段所組成，每個網段的入口皆架設防火牆以及行動代理人平台。為了方便起見，我們僅以五台主機並架設防火牆代表一網段，流入五台主機的流量作為流入五個網段的流量，我們並把其中一主機作為網管人員所在之主機。此五台主機的 IP 為 192.168.0.1~192.168.0.5。

我們使用 IBM 的 Aglet-2.0.2 作為行動代理人，其提供一名為 Tahiti 的行動代理人平台，可供各行動代理人於其上執行與移動，檔案 algets.props 可提供一些設定功能(例如是否開放代理人讀取本機之檔案)。圖 4 為各主機整體平台架構，作業系統使用 RedHat9 是考量到行動代理人與作業系統的相容性，選擇 Linux 系統則是為了架設防火牆、資料庫以及執行掃描工具…等作業方便。為了更貼近真實環境，我們捉取成功大學網路中的 5 個 IP，並從中過濾出非掃描行為之封包作為我們的 background traffic，並由 tcpreplay 於五台主機做重新播放的動作。掃描行為則由內部網路 192.168.0.6 主機發出，掃描工具為 nmap。

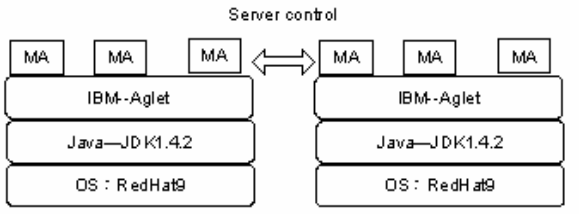

圖 4 平台架構

我們定義同一 IP 於三日內發出 10 次掃描行為為攻擊[2]，接著並執行所有的行動代理人，圖 5 為 PortScanManager agent 離開本地前偵測得未完成攻擊，並於下次回來本地時由記錄之狀態直接繼續判斷攻擊的狀態。而圖 6 則為當確定為掃描攻擊時，PortScanManager agent 產生 AlertAglet agent 的情形，以及 AlertAglet agent 移網網管人員主機的情形。

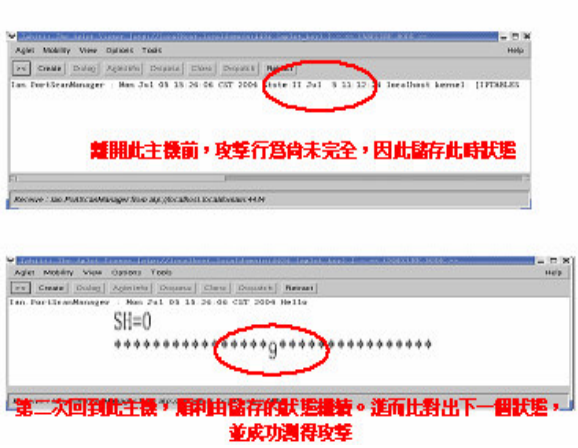

圖 5 PortScanManager agent 儲存狀態並判斷攻擊

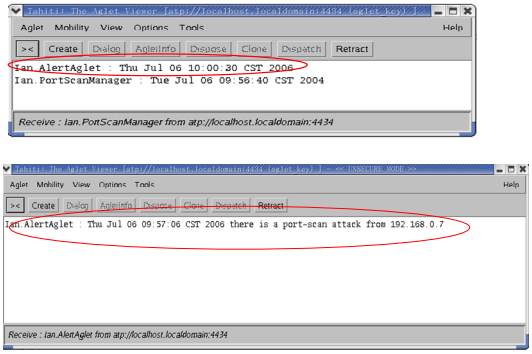

圖 6 AlertAglet agent 的產生以及通知網管人員

最後，我們討論 mobile agent 移動頻率與偵測速度的關係，在 S 中的四個參數與此項有關的為 i，因為 i 與定義攻擊的限制時間會造成偵測之攻擊結果的差異，故另三項參數在此並不討論。若能有一中控系統收集或分析來自其下階層的訊息，在正常情況下固然能做出做即時的判斷，但遭受攻擊的機率也極大，因此我們不採用此種方式。從我們底下的分析可知，當 mobile agent 移動之速率愈快，則愈能及時偵測，我們定義一個能評估偵測速率的數字，稱為時間落後率，當時間落後率等於零表示能於攻擊瞬間立即偵測出，若時間落後率等於無限大則表示無法測出此攻擊。

$$\text{時間落後率}=\frac{\text{攻擊時間}-\text{測得攻擊時間}}{\text{攻擊時間}} \quad (1)$$

我們利用亂數產生器分別產生 uniform 1 秒-1 天以及 mean 為 4 小時、8 小時的亂數各 100 組，每一組皆有 11~13 個數字，每個數字代表同一 IP 每次掃描的間隔，而這 11~13 次的掃描行為皆於 3 天內結束。我們並將 mobile agent 移動的頻率分為 10 小時、5 小時、2 小時、30 分鐘、1 分鐘五種情形討

論之。圖 7-圖 9 分別為 uniform、mean=4hr、mean=8hr 之攻擊發現率以及時間落後率的比較圖。攻擊發現率=1-漏報率

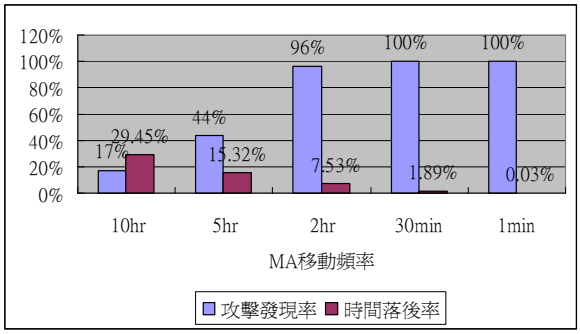

圖 7 掃描間隔為 uniform 之下 MA 頻率與偵測效能之關係

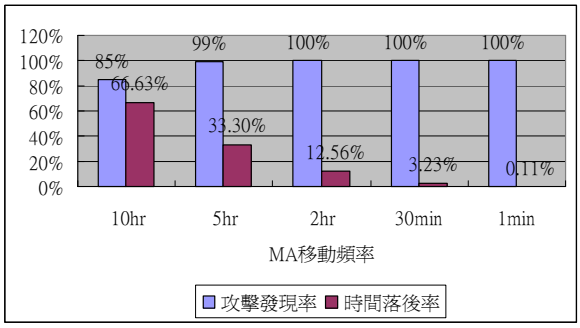

圖 8 掃描間隔為 mean=4hr 之下 MA 頻率與偵測效能之關係

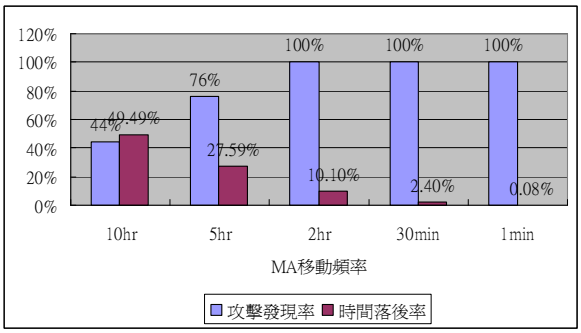

圖 9 掃描間隔為 mean=4hr 之下 MA 頻率與偵測效能之關係

由上面三張圖可清楚地發現，無論掃描時間間隔之隨機的方式為何，只要行動代理人移動的頻率愈高，則攻擊發現率愈高(亦即漏報率愈低)且時間落後率愈低(亦即愈能及時偵測)。故加快 mobile agent 的移動頻率便能加快偵測速度，若能再加上本論文所提的 GoorStop 機制更能相得益彰，快上加快。

5. 結論與未來工作

port scan 僅為入侵過程的其中一步驟，當惡意攻擊者掃描到弱點時便會進行下一步入侵行為，未來我們將把這些入侵行為做更進一步分析以及分類，並使我們的行動代理人入侵偵測系統偵測出更多種類之攻擊，如：蠕蟲攻擊、DDoS 攻擊…等，並利用蜜罐系統(honeypot)建構一網路讓外部攻擊者攻擊我們，以在更接近於真實網路的環境做測試，使我們整個系統更加完善。

誌謝

非常感謝台灣網路安全測試平台(TWANST，國科會計劃編號：NSC 94-2219-E-006-007)、成功大學資通安全研究與教學中心(TWISC@NCKU，國科會計劃編號:NSC 94-3114-P-006-001)所提供之設備與服務，以及「使用直接序列（DS）展頻技術之及寬頻無線通訊網路之研究(國科會計劃編號：NSC94 -2219-E-006-005」的支援，使本實驗能順利進行。

參考文獻

[1] http://www.insecure.org/nmap/

[2] Chunmei YIN, Mingchu LI, Jianbo MA, Jizhou SUN “Honeypot and scan detection in intrusion detection system,” CCECE 2004- CCGW 2004, Niagara Falls, Mayhai 2004.

[3] LIANG-MIN WANG, JIAN-MING ZHANG, JIAN-FENG MA “MODELING THE INTRUSION BY USING CAPABILITY OF ATTACKERS,” Proceedings of the 4th International Conference on Machine Learning and Cybernetics, Guangzhou, 18-21 Aug. 2005.

[4] Noria Foukia, “IDReAM：Intrusion Detection and Response excuted with Agent Mobility Architecture and Implementation,” AAMAS’05, July25-29, 2005, Utrecht, Netherlands.

[5] Peter Mell, Mark McLarmon, “Mobile Agent Attack Resistant Distributed Hierarchical Intrusion Detection Systems,” National Institute of Standards and Technology

[6] Roesch, M. http://www.snort.org

[7] Simon Y. Foo and Michael Arradonodo, “Mobile Agents for Computer Intrusion Detection,” IEEE CNF, 2004.

[8] Staniford-Chen S., S. Cheung, R Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip, and D. Zerkle, “ GPICG- A Graph-Based Intrusion Detection System for Large Networks,”The 19th National Information Systems Security Conference

[9] Yu-Fang Zhang, Zhong-Yang Xiong, Xiu-Qiong Wang, "Distributed Intrusion Detection based on CLUSTERING," Proceedings of 4th International Conference on Machine Learning and Cybernetics,Guangzhou,18-21 August 2005