使用資料探勘技術偵測 P2P 殭屍網路

廖文華
大同大學資訊經營系
e-mail: whliao@ttu.edu.tw

張家慶
大同大學資訊經營所
e-mail: kakeinoo@yahoo.com.tw

摘要

殭屍網路(Botnet)是一群受到殭屍病毒感染的電腦，造成網際網路安全重大威脅。攻擊者先在正常使用者的電腦中植入殭屍病毒，再經由網路下達命令操控所有的受害電腦，執行分散式阻斷服務攻擊(DDoS)、偷竊私密資訊或散佈垃圾郵件等進行各種的惡意行為。對等式殭屍網路模仿對等式軟體使用多主控端架構避免單點故障問題，並搭配加密技術，讓各種 Misuse 偵測技術無法發揮效果。由於對等式殭屍網路不同於一般的網路行為，會建立大量連線且不會消耗大量頻寬的特性，故仍可用 Anomaly detection 技術來偵測它的存在。本論文提出一個使用資料探勘的技術來偵測對等式殭屍網路。並實作於一個網路環境，並驗證其可用來尋找出對等式殭屍網路的宿主。其關鍵作法在於使用對等式殭屍網路與對等式軟體的原生相異點作為資料探勘參數，透過資料探勘技術加以分群、分辨，並達到可接受的正確率。

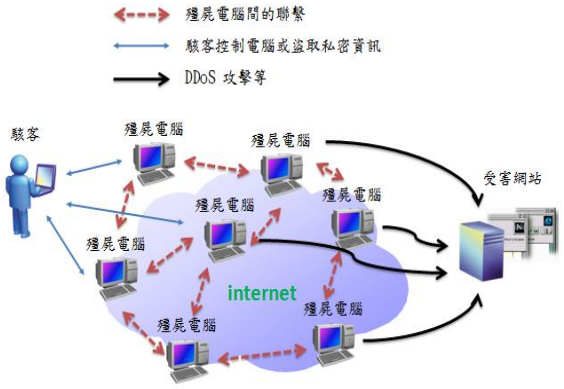
關鍵詞：資料探勘、殭屍網路、P2P

1. 簡介

帶給我們便利的網際網路（internet）近期受到了巨大而頻繁的衝擊，而這些衝擊的原因，源於一連串的惡意軟體與惡意活動。電腦病毒在 1980 年代末期造成了巨大的損失，電腦蠕蟲則在 90 年代開始，一直到 21 世紀初持續困擾著資訊網路。雖然各種電腦防護技術如防毒軟體、入侵防禦系統的進步使著病毒和蠕蟲問題稍受改善，但新興起的 Botnet 卻儼然成為網際網路最大的威脅[1]。

Botnet 是一群受到殭屍病毒感染的電腦，而被感染的電腦則稱為 bot。被感染的機器會自動登錄到一個中心位置，通常是一台網路聊天（IRC 協定）的伺服器，真正的攻擊者就利用這個聯絡的機制，命令和控制（Command & Control），然後可以向他們發出命令來發送垃圾郵件、偷取受害者的帳號密碼或發動 DDoS 攻擊[8]。Botnet 也因技術上的演變，逐漸發展出各種不同的架構，如 IRC、HTTP 還有 P2P[2]等類型。P2P 殭屍網路為新發展出來的 botnet 類型，它模仿 Peer-to-Peer（P2P）軟體的技術與架構，圖一是 P2P 殭屍網路的運作示意圖。P2P 殭屍網路一方面模仿 P2P 軟體使用多主控端架構來避免 single point failure 的問題，另一方面 P2P 殭屍網路 也開始使用加密技術[4]，讓別人無法分析通信的內容，更難以找出混在合法流量中的 botnet 通信，更別說去追蹤駭客的行蹤。網路攻擊的偵測方法可分成兩大類 Misuse detection、Anomaly detection[5]。Misuse detection 即為特徵(signature)比對的偵測方式，此種偵測技術需要對於惡意軟體的通訊內容做深入掃描才能做判斷，用在未搭配加密技術的 IRC bot 通訊的偵測上雖可行，但用在使用加密技術的 P2P 殭屍網路上就行不通了。而 Anomaly detection[7]運用在 botnet 的偵測上也有困難之處，除了它需要一群已知結果的數據來定義正常與異常的行為之外，也無可避免的會有 false positive 和 false negative 的錯誤判斷可能性。

相對的，Anomaly detection 的好處是它不需要解讀加密過的通訊內容，它只針對該通訊行為上、統計上的特徵來做判斷，對於被偵測體的內容上的變化較不敏感，即使惡意通訊經過變形或是被加密也能適用。故本論文使用 Anomaly detection 作為偵測的方法。Anomaly detection 著重的是該目標物的行為特徵，故利



圖一：P2P 殭屍網路的運作示意圖

用 P2P 殭屍網路的原生特性來整理出統計上的特徵是相當重要的。本研究所提出的偵測方法乃基於三個假設，第一是 P2P 殭屍網路的通訊會模仿 P2P 軟體的架構大量建立連線；第二是 P2P 殭屍網路為了保持這個網路，bot 連線會保持傳輸資料；第三是為了保持隱密不被發現，botnet 的通訊會盡量用最小的資料量。為了提高 Anomaly detection 的準確性，除了在實驗室的網路環境下來提供實驗所需的數據外，也使用資料探勘的方法以提供更精準的判斷。殭屍病毒的選擇上是選擇 Trojan.Peacomm 殭屍病毒。此病毒的行為已在論文[4][6]中被探討過，病毒樣本的取得也相對容易。

本論文的架構如下：第二章探討殭屍網路的相關文獻。第三章說明偵測架構之設計與系統實驗的作法。第四章說明實驗的結果與討論。第五章是結論與未來的研究方向。

2. 文獻探討

目前已經有許多文獻探討如何分辨正常流量與殭屍網路的流量。有些研究使用資料探勘的技術作為判別的方法，如[7]中，他們提出一個使用 machine learning 的方法，可以用來分辨 IRC 與非 IRC 的流量，並期望從 IRC 的流量中篩選出 IRC botnet 與非 IRC botnet 的流量。在[6]中，他們利用廣被使用的 Self-organizing map 演算法，專注於偵測 P2P 殭屍網路的研究，其基本假設為 P2P 殭屍網路和 P2P 軟體一樣在防火牆上會有大量由外而內的失敗連線的特性作為資料探勘的關鍵參數，並用在校園的網路做驗證。在[8]中，他們則提出針對 IRC 聊天的內容與中毒電腦所運作的程式名稱作關聯性分析，同時並以 IRC 連線為單位，針對 IRC 聊天行為做檢驗的作法。若封包內容與電腦程式運作的相關性有不正常，或是 IRC 聊天速度不正常的快，就特別記錄下來把其他的資訊交給資料探勘技術加以訓練並加以分辨。在[1]中，他們使用了類神經網路的演算法 Dendritic Cell Algorithm (DCA) 把 keylogging, SYN or UDP flooding 攻擊、敏感檔案存取或是潛在可能的 bot 通訊加以關連性分析，並以實驗證明此法有相當好的正確性。

有些研究則專注於 botnet 的特性做偵測方法。在[9]中，他們提出了一個利用 botnet 的 IRC 通訊具有群聚並緊密相關的特性，成功的在大量的連線中排除低疑慮的通訊，並在 botnet 通訊中做到分群以提供後續的分析可能性。在[3]中，他們則提出一個類似 honey pot 的作法，利用完全沒有提供 internet 服務的伺服器記錄網際網路上不必要的連線或探測來統計可能是殭屍網路的電腦，描繪出目前的資安問題嚴重性，並提倡 ISP 導入隔離區(walled garden)的作法來整治目前日趨嚴重的殭屍網路的問題。有些論文則針對殭屍病毒做分析以提供後續的研究。在[4]中，他們針對 Trojan.Peacomm 這隻 P2P 殭屍網路 病毒做深入的封包記錄、感染動作與行為分析，並對殭屍病毒的發展歷史做條列的說明。

由於 P2P 殭屍網路為一新興的 botnet 族群，相關的研究相較之下較為不完整，或是不足以應付 P2P 殭屍網路的獨特行為。如在[7][9]中，他們的研究成功的分辨出 IRC 和一般網路行為的流量，但並未成功的分別出 botnet 與 IRC 的流量。在[8]中，他們成功的分辨出 IRC 和 botnet 的流量，但他們的研究方法必須同時監控網路與監控電腦的執行程式來加以比對，無法單純的分析連線行為就達到分辨的目的，而且必須要看封包的 payload，遇到加密的 P2P 殭屍網路流量就無法適用。在[6]中，他們的研究是使用 SOM 演算法來找 botnet，他們的研究只提供其作法而沒有提出實際的數據結論。在[1]中，他們的研究則把把攻擊行為(如 syn flood, port scan)納入判斷的參數，無法完全依賴感染初期的行為特徵做判斷。

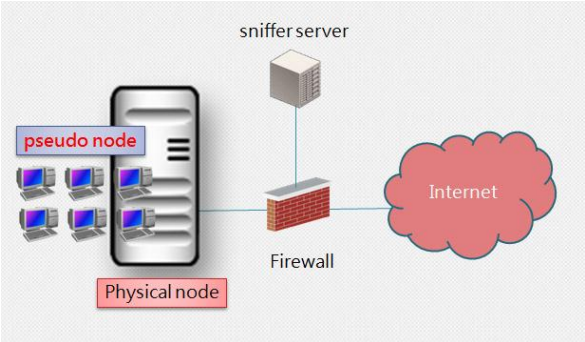
3. 實驗設計與病毒觀察

3.1 實驗的步驟

首先，建立一個實驗環境來觀察殭屍網路，檢視其是否有異於一般正常網路行為的地方，並側錄實際的網路流量來準備 ground truth 作為資料探勘之用。第二個步驟是把側錄的封包做數據上的解析，從數據上來解釋殭屍電腦的行為，並進一步做資料探勘的工作。第三個步驟是把解析好的數據用資料探勘的工具對蒐集到的資料集做判別，並針對其結果做分析整理。

3.2 實驗環境的建立

我們建立了一個實驗環境如圖二。使用一台 PC 做為載體，是為「Physical node」。作業系統為 Windows XP service pack 3，並於其上安裝 VM ware 模擬大量的 PC，是為「pseudo node」，pseudo node 的作業系統也為 Windows XP，軟體方面則隨機安裝 P2P 軟體、瀏覽器、網路遊戲與 P2P bot 病毒：Trojan.Peacomm。網路架構的部份則是透過一台防火牆與 internet



圖二：實驗環境圖

連線，並選擇在防火牆的 LAN 作為監控點，以避免側錄到不必要的 internet 雜訊，此外並準備一台 sniffer server 專門負責側錄，並負責實驗結果的運算統計。Internet 連線則為 8M/512K 的 ADSL 線路。

此實驗環境為一封閉架構，架構內除了計畫內的節點之外並無其他設備，而流量內容除了 Windows XP 本身的服務流量如網路芳鄰、網路廣播外，僅有準備好的 P2P 軟體、瀏覽器、網路遊戲與 P2P bot 病毒之流量，而不會有非預期的流量出現。P2P bot 病毒 Trojan.Peacomm 的流量也只側錄其發動攻擊之前的通訊流量，而非攻擊流量本身，期待做到事前預警而非事後處理的研究目標。

3.3 殭屍電腦的行為假設與觀察結果

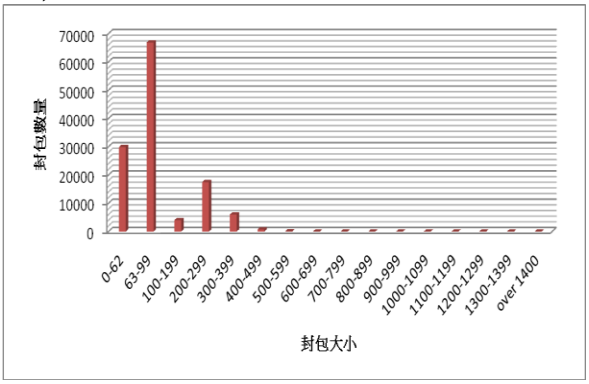
殭屍病毒 Trojan.Peacomm 的寄生和通訊行為已在[4][6]研究過，但其底層的 session 特性和封包特性則未被充分討論過。封包特性與連線特性有太多的參數可以研究，故必須先針對殭屍電腦的行為做參數的收斂才能夠進行資料探勘。針對 P2P 殭屍網路的連線行為特性我們提出三個假設：第一是 P2P 殭屍網路的通訊會模仿 P2P 軟體的架構大量建立連線和其他 bot 進行通訊以達到 multiple controller 和 mesh network 架構的目的，也就是說該殭屍電腦連線數瞬間有效連結數會極多。第二是 P2P 殭屍網路為了保持這個網路，會和其他的 botnet 成員保持連線與交換資料，而非完成傳輸之後就不相通訊，也就是每條連線都會有一定的傳輸量。第三是殭屍電腦為了保持隱密不讓受害者發現，bot 通訊會盡量用最少的資料量做傳輸，避免對受害者的網路使用造成衝擊直到真的要對外發動攻擊（DDoS、發送垃圾郵件）為止。這代表著連線雖多，但每條連線應該都是小流量。為了求證上述的假設，實驗以十分鐘為側錄的時間單位，在防火牆上側錄封包與統計連線來觀察是否真的和其他網路行為有所區別。經統計後把 Trojan.Peacomm 的封包分布圖(圖三)，與一般上網行為(圖四)、網路遊戲（圖五）、P2P 軟體（圖六）一一列出做比較。

從分布圖（圖三）可以發現，P2P 殭屍網路 除了少數無資料的封包之外，封包大小都集中在 63~400 byte 之間。而其他正常網路通訊行為除了部分無資料的封包（如 tcp 三方交握的封包）之外，大部分的封包因作業系統追求效率的關係都會以最大封包大小(MTU)來做傳輸。唯一小封包較多的應用是線上遊戲，但線上遊戲的小封包數量比起 P2P 殭屍網路的小封包數量仍然明顯少的多。接著透過 wireshark [11]軟體來計算每種網路行為的封包與流量的分佈。並把 botnet、P2P 軟體、一般瀏覽行為疊在一張圖如圖七，並加以局部放大做對比。透過比較圖可以發現 P2P 殭屍網路的封包數雖然不少，但依連線來看封包數量和流量卻異常的少，遠不及正常的連線行為的封包和流量。

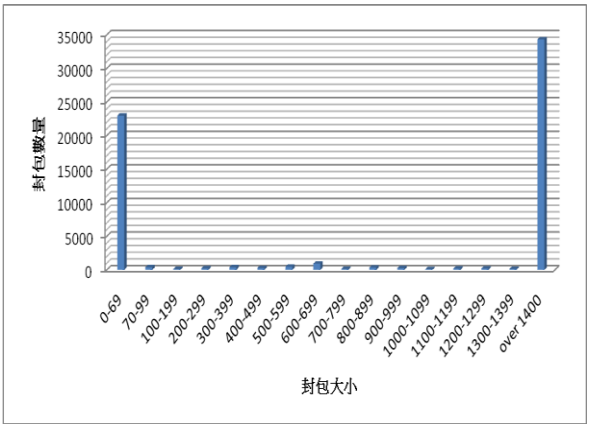
我們將封包、連線行為與最大連線數整理成表一，可發現先前的三項假設皆被驗證，並可以用來作為稍後的資料探勘的重要特徵。為了探勘上的需要，特別把 63-399 byte 大小的封包提為一個參數，並命名為 “small packet”，流量介於平均流量上下一個標準差與封包數介於平均封包數上下一個標準差的連線也作為一個參數，並命名為 “small session”。

3.4 資料探勘方法、工具與步驟

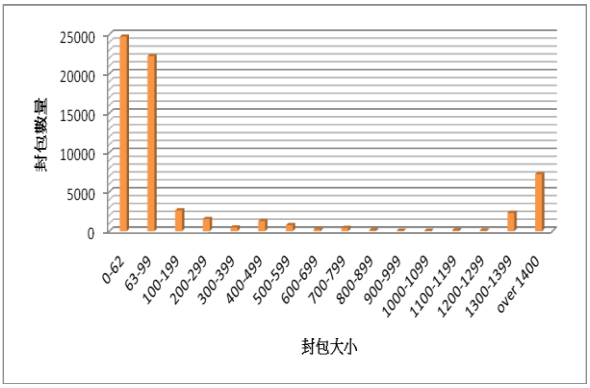
探勘工具軟體所選用的是免費軟體 WEKA[10]，此軟體普遍被運用在學術與商業用途上。資料探勘演算法則選擇了廣被運用的 J48、Naive Bayes 和 Bayesian networks 演算法來交叉測試並作討論。其中，J48 演算法是 WEKA 中以 C4.5 演算法所實作出來的演算法。在這個演算法中，資料會以決策樹(decision tree)的方式來做分類，每一個 internal 節點代表



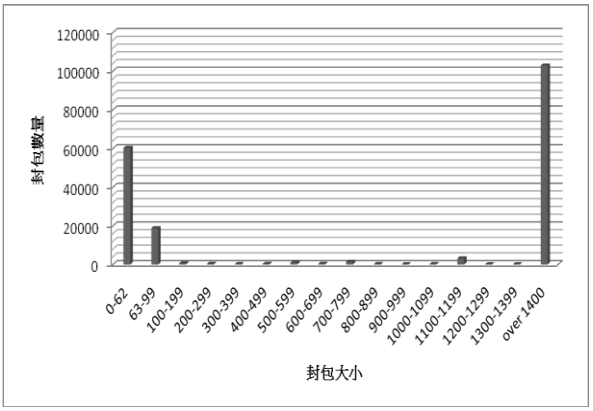
圖三：P2P 殭屍網路的封包分布圖



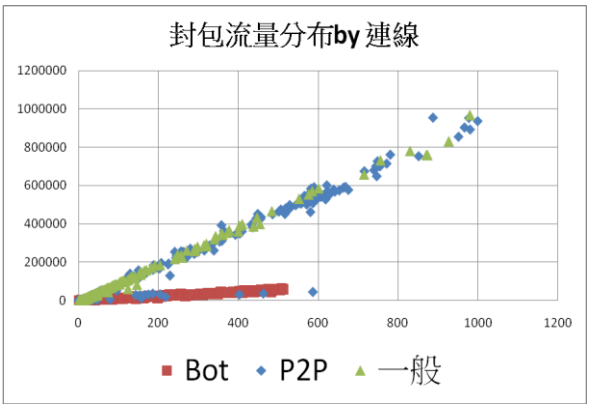
圖四：一般上網行為的封包分布圖



圖五：網路遊戲的封包分布圖



圖六：P2P 軟體的封包分布圖



圖七：三種應用的連線分布圖(局部放大)

表一：各種網路行為的封包與連線比較表(流量單位: byte)

	P2P Bot	P2P 軟體	網路遊戲	一般上網
同時連線數	800 up	3000 up	400 up	少於 100
平均流量	21, 151. 04	63, 288. 46	6, 544. 99	107, 801. 97
平均封包數	194. 22	71. 88	20. 70	118. 37
流量標準差	19, 954. 73	359, 487. 48	28, 742. 35	349, 581. 59
封包數標準差	175. 79	375. 87	224. 21	357. 2
小封包連線比例	**64%**	2%	8%	13%
小封包連線數	**400**	41	240	73
平均封包大小	108. 88	872. 23	311. 48	910. 18
封包大小標準差	90. 91	701. 81	500. 57	697. 04
平均每秒封包	208. 22	315. 38	106. 94	102. 45
小封包比例	75%	10%	42%	2%
小封包數量	**94323**	19737	26859	1127
無資料封包數	29853	60237	24706	22958

著一個（或更多）用來分類的屬性，而每個葉節點代表著分類的結果。Naive Bayes 則是把每個屬性認定為獨立且不相干的屬性來做統計機率上的判別，其缺點為每一個用來判定的屬性都擁有同樣的重要性，且不必要的屬性也會對正確性有負面影響。至於 Bayesian networks 演算法是一個圖像式的演算法，會把所有的參數視做是彼此相關的參數來做判斷。

為了取得實驗需要的網路封包，隨機在各 pseudo node 上同時執行 P2P bot、上網頁瀏覽、玩網路遊戲與 P2P 軟體下載檔案等行為，換句話說各行為是混雜在同一個 pseudo node 上的，一方面提高偵測難度，另一方面也比較符合實際上的網路使用行為。側錄時間為五分鐘，其結果交給 sniffer server 做運算統計，並寫入一 CSV 文字檔，再把該 CSV 檔案交給 WEKA 做資料 training。初步的 ground truth 為 45 筆資料。雖然我們會側錄完整的封包內容，但實驗並不會參考封包的內容做判斷，來符合 Anomaly detection 偵測技術的定義。Sniffer server 會把側錄結果做運算之後輸出下列的格式作為資料探勘用，其參數說明如下：

- KeyID：獨一序號，用來作為 data column 的識別，探勘不使用此參數。
- Max_Session：該側錄時間內的同時最大連線數。
- Session_avg：該次側錄的所有連線的平均流量。
- Session_pkt：該次側錄的所有連線的平均封包數。
- Session_std：該次側錄的所有連線的標準差。
- Session_pac_std：該次側錄的所有連線的封包數的標準差。

- SmallSession_percent：該次側錄的連線中，符合小連線特性的連線相對於所有連線的比例。
- SmallSession_count：該次側錄的連線中，符合小連線特性的連線的數量。
- Packet_avg：該次側錄封包平均大小。
- Packet_std：該側錄封包大小標準差。
- Packet_persecond：該次側錄的平均每秒封包。
- SmallPacket_count：該次側錄的小封包個數。
- SmallPacket_percent：該次側錄的小封包比率
- Null_packet_count：該次側錄的無內容的封包的個數。
- Bot：若該次側錄有 P2P bot 殭屍病毒的流量混在其中，則此欄位為 yes，反之則為 no。

把數據交給 WEKA 做初步分析之後就可以知道先前的假設是否可以成立，接著第二個步驟是調整其操作參數檢視其準確率是否有所改善。實驗的結果會以 false negative rate (FNR)與 false positive rate (FPR)來評斷以上的分類演算法的效率。較低的 FNR 意味著只有一小部分的 bot 可以閃過本偵測機制；較低的 FPR 則可以保證正常的流量不會被誤判為 bot 的流量，且代表著這個方法有較高的實作性，因為在實務上，攔阻正常的流量會比漏攔惡意流量導致較為嚴重的後果。

本次的實驗蒐集了45筆的側錄資料，其側錄內容之混雜方法與特性如表二所示。

表二：側錄資料內容說明

流量混雜方式	流量特性	筆數
bot + 環境流量	純bot的流量，小封包小連線多	5
bot + 瀏覽器 + 環境流量	bot小連線、小封包多但瀏覽器的大封包稀釋了bot的特徵。	5
bot + P2P軟體：foxy + 環境流量	bot小連線、小封包多但P2P軟體所帶來的大流量稀釋了bot的特徵。	5
P2P 軟體：foxy + 環境流量	P2P軟體連線極多，符合bot的特性，但流量大這點與bot相異。	5
P2P 軟體：bitcomet + 環境流量	P2P軟體連線極多，符合bot的特性，但流量大這點與bot相異。	5
P2P 軟體：bitcomet慢速下載 + 環境流量	因下載的檔案較難下載，本數據有極多的小封包與小連線，流量也小，具有較多P2P bot的特性。	5
線上遊戲 + 環境流量	小封包為主要內容，具有部分P2P bot的特性，但連線數少。	5
瀏覽器 + 環境流量	封包大，連線數少，和bot流量差別較大。	10

4. 實驗結果與討論

4.1 初步的實驗結論

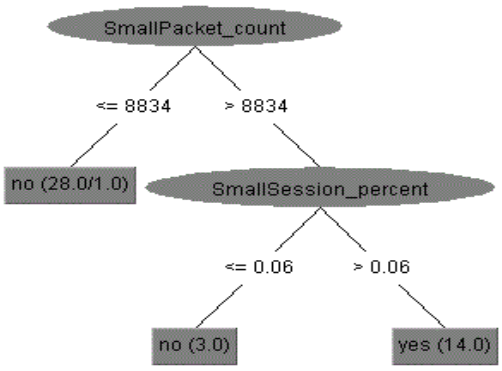
把實驗數據送進 WEKA 分析，並選擇預設的 test option：Cross-validation Folds 設定為 10，可得到表三的結論。三種演算法的準確率分為 98%、89%、87%的正確率，其中 J48 具有較好的分辨比率，正確的在 45 筆資料中分辨出 44 筆的資料。NaiveBayes 和 BayesNet 也有接近九成的分辨率，但 BayesNet 演算法發生了三筆 false positive，顯見此方法搭配 BayesNet 演算法有改善空間，其不理想的原因可能是參數的選擇上不夠好，也有可能是分群大小需要調整。J48 演算法是決策樹演算法的一種，可把分群決策樹視覺化方式呈現如圖八。由該圖可以看到經過 training 之後後，J48 演算法判定在眾參數裡面，真正關鍵的因素是 SmallPacket_count 與 SmallSession_percent 這兩個參數，只要 Small Packet 大於 8834 個，且 Small Session 的比例大於 6%，就可以判定該批流量內含 P2P 殭屍網路的流量。因此初步的結論是我們所提出的方法論可以成功的分辨出感染了殭屍病毒的電腦的流量與沒有感染殭屍病毒的流量，而就我們所挑選的演算法中是以 J48 演算法比較適合。

4.2 探勘參數最佳化

由於 NaiveBayes 和 BayesNet 初步實驗的結果並不理想，故我們用三種方式來尋求改善。首先是使用 J48 決策樹所顯示的關鍵參數來測試 NaiveBayes 和 BayesNet 演算法，接著使用 WEKA 內建的 exhaustive Search 演算法

表三：三種演算法的比較

演算法	正確比例	正確筆數	錯誤比例	錯誤筆數
J48	98%	44	2%	1
NaiveBayes	89%	40	11%	5
BayesNet	87%	39	13%	6



圖八：本資料探勘之 J48 決策樹

來挑選其參數，最後是調整 Cross-validation Folds 的大小來測試是否有所改善。首先使用 J48 所顯示的關鍵參數: SmallPacket_count 與 SmallSession_percent 來做測試，NaiveBayes 的正確率為 86.67％，並沒有得到提昇。而 BayesNet 則受惠於參數的變動正確率上升為 91.11%。而 exhaustive search 挑選的關鍵參數為：Max_Session、Session_pkt_avg、SmallPacket_count、Null_packet_count 這四個參數。使用該參數再進行分群運算，則 NaiveBayes 正確率為 86.67％，false positive 有兩筆，BayesNet 則改善為 93.33%。最後我們試著去調整 Cross-validation Folds 的大小，結果發現當 Folds 設定為 12 的時候，NaiveBayes 演算法的正確率便提升到了 93.33％，BayesNet 設定為 11 個 Folds 時，正確率也提升到了 93.33％，false positive 也減少為一筆。足見原有設計的探勘參數應可應付分類的需求。透過實驗的驗證，本研究達到了不看封包的 payload，不需等到實際攻擊行為出現的前提下，只需要觀察五分鐘其網路行為，就可以成功的抓出殭屍電腦的存在並達到可接受的正確率的目標。

5.結論與未來方向

本篇論文提供了一個靠著在網路閘道端監聽流量，使用資料探勘技術分析連線行為的作法來偵測 P2P 殭屍網路的存在的方法。除了實驗的結果證明了本方法可以達到一定程度上的準確度之外，本研究也達到了以下的目的：1. 使用 Anomaly detection 偵測技術，避免掉了使用 Misuse detection 偵測技術的侷限性，也不受限封包是否加密。2.不需要在電腦上安裝軟體，不必更動網路架構，佈署可行性高 3.不把攻擊行為(如 syn flood，port scan)納入探勘參數，只針對 P2P 殭屍網路的聯繫行為做研究，盡量做到事前預警而非事後檢討。4.不被異種流量混雜所干擾，即使各種流量混雜在同一台電腦上面也可以成功找出其中是否有 P2P 殭屍網路的流量。本研究目前僅針對 LAN 的環境做研究，相較於 internet 仍為分散式的偵測架構，若要大規模的消滅 P2P 殭屍網路就必須將本方法佈署在 ISP 端。在 ISP 端的佈署就必須考量到網際網路上普遍使用 NAT 技術，而 NAT 技術勢必會把該 LAN 的所有電腦流量導成同一個 IP 的流量，這會造成 P2P 殭屍網路流量特徵被極端稀釋導致判定上的困難。此問題待後續研究做進一步的探討。

參考文獻

[1] Y. Al-Hammadi, U. Aickelin, and J. Greensmith, "DCA for Bot Detection," *IEEE Proceeding of Congress on Evolution Computation (CEC)*, 2008.

[2] D. Dittrich and S. Dietrich, "Discovery Techniques for P2P botnets" *Stevens CS Technical Report 2008-4*, September, 2008.

[3] D. Grabowski, "Global Network Pandemic – The Silent Threat," *Global Telecommunications Conference*, 2008.

[4] J. B. Grizzard, V. Sharma, C. Nunnery, and B. B. Kang, "Peer-to-Peer Botnets: Overview and Case Study," *First Workshop on Hot Topics in Understanding Botnets (HotBots '07)*, 2007.

[5] S. Kumar and E. Spafford, "An Application of Pattern Matching in Intrusion Detection," *Purdue University, Tech. Rep.*, 1994.

[6] C. Langin, H. Zhou, S. Rahimi, B. Gupta, M. Zargham, and M. R. Sayeh, "A Self-Organizing Map and Its Modeling for Discovering Malignant Network Traffic," *IEEE Computational Intelligence in Cyber Security (CICS '09)*, 2009.

[7] C. Livadas, R. Walsh, D. Lapsley, and W. T. Strayer, "Using Machine Learning Techniques to Identify Botnet Traffic," *IEEE LCN Workshop on Network Security (WoNS'2006)*, 2006.

[8] M.M. Masud, T. Al-khateeb, L. Khan, B. Thuraisingham, K. W. Hamlen, "Flow-based Identification of Botnet Traffic by Mining Multiple Log Files," *International Conference on Distributed Framework and Applications (DFmA 2008)*, 2008.

[9] W. T. Strayer, R. Walsh, C. Livadas, and D. Lapsley, "Detecting Botnets with Tight Command and Control," *IEEE LCN Workshop on Network Security (WoNS'2006)*, 2006.

[10] WEKA, http://www.cs.waikato.ac.nz/ml/weka/

[11] Wireshark, http://www.wireshark.org/