以 NetFlow 為基礎之網路異常流量快速偵測

On Fast NetFlow-Based Traffic Anomaly Detection

[1] 賴守全　　[2] 王夢勳

[1] Shou-Chuan Lai　　[2] Meng-Hsun Wang

[1] 銘傳大學電腦與通訊工程學系
[1] Department of Information and Telecommunications Engineering, Ming Chuan University

[2] 銘傳大學電腦與通訊工程學系
[2] Department of Information and Telecommunications Engineering, Ming Chuan University

摘要

近年來隨著網際網路的普及以及電子商務的蓬勃發展，網路已經成為大多數人生活中不可缺少的一部份。但網路的發展也伴隨著許多安全的問題，例如 DDoS、病毒、蠕蟲等網路攻擊，這些攻擊可能造成大量封包在網路間傳送，造成網路壅塞，危害網路的運作。通常此類的攻擊若不在短時間內進行處理，容易因為惡意程式快速的散播導致更多主機遭受感染，使得問題越來越嚴重。因此對於網路管理者來說，如何在短時間內對網路流量進行監控與分析，並能快速的找出異常的主機是非常重要問題。

傳統的網路流量管理工具通常過於簡單導致無法快速找出異常主機，若使用市面上的流量統計軟體可能需要負擔昂貴的經費及較大容量的儲存空間，有鑑於此，本文提出使用 Cisco 所發展的 NetFlow 來收集網路流量，嘗試透過 NetFlow 收集的檔案分析目前各種網路應用的行為，再藉由正常與異常流量比對的方式找出異常特徵。而本文也將此方法撰寫成一個自動化的系統，此系統能自行產生 NetFlow 檔案並進行判斷，快速的找出網路中的異常主機，管理人員能對於這些造成異常的主機在最短的時間內進行分析及處理，以減少進一步的危害發生。

關鍵字： NetFlow、異常偵測、網路安全

Abstract

As the Internet and e-commerce grow rapidly in recent years, it also brings network security issues, such as DDoS attacks, viruses, worms, and other network attacks. These attacks will generate lots of packets on the network and make the network too busy to operate smoothly. The condition of the network may become worse if the host being infected or under attack cannot be solved immediately. Therefore, how to monitor and analyze network traffic and identify abnormal hosts fast enough has become an important issue for network administrators.

Traditionally, network traffic management tools may be too simple to identify abnormal hosts. Some other tools may need large storage or computing power to monitor lots of packets and the cost of these tools may be not affordable for campuses or small businesses. In this paper, we propose to use Cisco NetFlow to collect network traffic and analyze network behaviors to identify abnormal hosts. We build an automatic detection system which can produce and analyze NetFlow files and find abnormal hosts. The results show that network administrators can make use of the proposed system to reduce network security problems.

Keywords: NetFlow、Anomaly Detection、Network security

1. 前言

網際網路迅速的發展改變了人們的生活，但相對的網路的發展也帶來一些威脅。根據雲端服務供應商 Akamai Technologies 所發佈的 2012 年第四季網際網路現狀報告[8]中對於全球網路攻擊流量統計的結果顯示，台灣位居全球排名最大攻擊流量第五名，攻擊流量占總數的3.7%，因此可以發現網路攻擊時常存在於我們的生活當中。而通常網路出現問題都是經由使用者向管理者反映網路變慢或是不能連上網路等情形，管理人員才會透過設備記錄進行分析，找出影響網路運作的原因。影響網路運作異常的原因有很多種，大致可分為線路異常、硬體設備異常、網路佈署或環境設置錯誤以及最常見的使用占大量頻寬的應用，如：P2P 軟體，或是遭受病毒、蠕蟲或是 DDoS 等惡意程式攻擊等。通常發現並排除問題需要一段時間，在這段時間之內若是蠕蟲或病毒持續的進行傳播，容易造成更多的主機遭感染，而使得網路異常時間延長。因此要如何能在短時間內有效的發現攻擊的存在，並在剛開始攻擊時對於造成問題的主機進行處理，對於網管人員來說是值得關心的議題。

傳統上網路管理者會使用 SNMP[16] (Simple Network Management Protocol) 或 MRTG[11]（Multi Router Traffic Grapher）來觀測網路流量，這兩種工具能夠提供的訊息僅限於設備統計結果，並不易立即辨認出造成問題的主機。若使用封包監聽器 (packet sniffer) 雖然能夠獲得詳細的封包資訊，不過分析封包需要耗費大量的時間和運算資源，且所需的儲存空間也比較大。對於一些大型繁忙的網路來說，可能因為網路太過繁忙而導致影響整體網路的運作。而目前市面上也有許多軟體與設備能夠在大型網路中有效的進行監控，不過這些設備通常需要額外付出相當昂貴的費用而令管理者望之卻步。因此在效能與成本考量之下，本文使用 NetFlow[10]做為網路分析的主要工具。NetFlow是由Cisco發展出來的一套網路流量監測技術，提供封包所組成的流量訊息，其最主要的特點在於只讀取封包標頭的資訊，能夠大幅減少檔案的大小及儲存空間，另外因為目前大部分設備皆有支援NetFlow的運作，不須另外添購新的設備，因此也能減少成本支出。

本文的研究範圍著重於透過NetFlow側錄的流量資訊，對於病毒、蠕蟲及DDoS等惡意攻擊的行為進行觀察與研究，這類的惡意攻擊通常會占用頻寬與資源，導致網路上充滿不必要的封包導致網路壅塞，嚴重可能造成網路無法使用。因此本研究希望透過本文所提出的方法能為管理者在龐大的流量中找出真正對於網路運作具有威脅性的主機，以利於協助管理者在網路異常發生時能快速的找到造成異常的主機並進行處理，以避免災情快速的擴大。

2. 相關研究

本章節將對於本研究所使用的工具以及相關的文獻進行介紹與探討。首先在第一節將會對本論文使用的流量監測工具 NetFlow 進行介紹。第二節則會對於相同使用 NetFlow 進行異常偵測的相關文獻進行探討。

2.1 NetFlow 介紹

NetFlow 是由 Cisco 發展的一套網路流量監測技術，提供每一筆流量的統計紀錄，目前已內建在大部分 Cisco 路由器上，由於被許多人廣為使用，因此其他網路設備供應商也漸漸支援 NetFlow 技術，使其成為一個標準。而 NetFlow 發展至今有許多的版本，最廣泛使用的版本為具有固定的格式第五版，主要支援到 IPv4，而為了因應 IPv6 的運行而發展出的第九版[9]，其具較有彈性以及可擴展性，提供管理人員能自行定義輸出格式。

NetFlow 的運作[12]包含了兩個關鍵的要素，分別是 NetFlow Cache 與 NetFlow Data Export，其原理是利用為封包傳輸時連續相鄰的封包通常是送往相同的目的 IP 位址的特性，配合 Cache 快取機制，當路由器開啟 NetFlow 功能時，路由器會解析接收到的封包標頭取得流量資訊，再將封包彙整成一筆筆的 Flow。最後利用 UDP 封包傳送給接收主機，主機接收後將其轉成 NetFlow 格式輸出，提供管理人員進行觀察與分析。

Flow 的定義為一個來源端和目的端之間單方向傳輸的流量資料，每一筆 Flow 當中都包含著相同的欄位屬性，NetFlow 會以下列七個欄位判斷是否屬於同一筆 Flow，這七個欄位分別是來源端 IP(Source IP)、目的端 IP(Destination IP)、來源端通訊埠號(Source Port Number)、目的端通訊埠號(Destination Port Number) 、Layer 3 通訊協定(Protocol)、服務種類(Type of Service)以及路由器接入介面(Router input interface)，路由器會檢視這些欄位來判斷這個封包是否已經屬於任何已記錄的 Flow，如果欄位相同路由器會將其合併，並且將其 Bytes 數與 Packets 數累加在同一筆 Flow 上，若其中一個欄位不相同則建立一筆新的 Flow 並結束上一筆 Flow 的紀錄。當然 Flow 紀錄也會因為某些情形而終止，若符合下列條件之一，路由器就會終止目前的 Flow 並透過 UDP 封包將記錄匯出到使用者事先設定的收集主機上：

1. 此筆 Flow 於指定時間內沒有發出任何封包。
2. 流量達到最長保留時間時自動停止 Flow，預設為 30 分鐘。
3. 路由器暫存已滿時終止此筆 Flow。
4. 收到 TCP Flag 為 FIN 或 RST 時，代表其連線已結束，因此結束該筆 Flow。

NetFlow 最大的特色在於它只讀取經過流量的標頭檔(header)，並將其轉換成 NetFlow 格式輸出，並不會讀取完整封包的內容，因此容量遠小於原始未經壓縮的封包檔案，除了能夠大幅減少儲存的空間之外，也可以減少運算資源的負擔。另外目前大部分設備皆支援 NetFlow 運作，對於網路上許多昂貴的設備來說相對的能夠減少成本支出，也因為其所需容量及運算資源小的特性，並不會影響整個網路的運作，因此適合應用在大型的網路的運作之中。管理人員能夠利用所產生出來的 NetFlow 檔案進行後續的分析或應用，例如：監控設備資源使用情況、流量管理、用戶監控、故障排除、安全和異常偵測等。

2.2 NetFlow異常偵測方法

網路中的異常偵測最常使用的方式就是使用基準線,所謂基準線即是一個或一組標準,也是一個臨界值,只要統計出來的結果超過預先訂定的臨界值範圍,則判定為異常流量。例如一定時間T之內正常所產生的Flow數不會超過250,即可將基準線設定為250,只要某IP在時間T之內的Flow數超過基準線的臨界值,則可認為此IP為異常主機。使用臨界值的方法也會有產生誤判的情況,例如:使用P2P軟體可能會與病毒所產生的流量類似,此時就可能發生誤判。因此為了減少誤判的情形,必須再加上另外的偵測方式搭配基準線,以達到較好的偵測效果。以下兩小節將介紹兩種目前常見的使用NetFlow的偵測方法,我們將其分類為以封包特徵比對為主的偵測方法以及以連線狀態為主的偵測方法兩種。

2.2.1 以特徵比對為主的偵測方法

傳統上大部分對NetFlow異常的偵測方法主要是使用特徵比對為基礎,例如:比對特定IP、Port、封包數、協定、封包大小等。透過觀察每一種異常特有的連線行為將其歸納成攻擊特徵並設立臨界值,當某個IP符合特徵值的總數超過於預設的臨界值時,即判定為異常。此種方式常使用於病毒或蠕蟲的偵測上面,例如:黃文穗、林守仁提出的“利用Netflow建置Code Red Worm偵測系統”[4]當中就是針對早期有名的Code Red Worm紅色警戒病毒進行偵測。受Code Red感染的主機會被植入後門程式,並且持續發送攻擊封包,企圖感染散播至別的主機而造成大量的網路壅塞的問題。管理者透過NetFlow提供的資訊發現此蠕蟲的攻擊行為擁有共同的特性,即是每筆Flow的目的port為80,Packets數為3,size大小為144 Bytes,透過符合特徵進行統計,擁有此特徵數目超過預設的門檻值則認為存在CodeRed的流量。另外,也有一些文章[2][6]提到了許多特定行為的特徵,例如:透過觀察主機所使用埠號欄位可以發現SQL Slammer利用1433、1434 port進行感染,而W32.Sasser則是使用445 port進行散播。另外一篇由Wang Jinsong、Liu Weiwei等人所提出的文章[18]雖然沒有對病毒或蠕蟲進行偵測,不過其利用NetFlow的流量分析出特徵來判斷是否存在P2P的使用者。此篇文章的作者認為P2P在使用上會造成大量的連線,甚至影響網路的正常運作,因此若能偵測P2P的流量能夠對其進行有效的控管。

透過觀察封包特徵的方式雖然簡單快速,但是卻有其缺點存在。主要的缺點有二,第一個缺點在於一次只能判別一種網路行為,假若所有流量進到判別程式當中都需要對各種攻擊特徵進行比對,將會耗費許多的時間。第二個缺點是當有新種類的攻擊行為產生時,因為沒有此種攻擊的特徵,勢必無法比對到新的異常行為存在,管理人員需要花時間進行分析並將新的攻擊特徵手動輸入,而這中間的時間差可能已經造成許多的機器遭受攻擊或感染。因此使用封包特徵比對的方比較適合使針對單一種攻擊的偵測上,並不適合用於整體網路的異常行為偵測。

2.2.2 以連線狀態為主的偵測方法

除了觀測封包特有的特徵之外,另外一個常見的方法是以網路整體連線狀態為主進行分析。不管是蠕蟲、病毒或是DDoS攻擊,為了能在短時間內進行快速的散播,必定會透過對網段主機進行大量掃描的方式以確定主機狀態,因此也勢必會產生出大量的連線,而這樣

的動作又將會於新的主機遭受感染之後重複同樣的行為持續的進行散播。而此種偵測方法就是利用整體連線的特徵進行分析與判斷，找出存在於網路中異常的流量。有些學者[3][5]認為網路流量與連線數會依照不同的時間點而不同，因此透過記錄平時每個時段的正常流量資訊建立基準線，基準線能反應正常行為下所產生的流量，再透過依照各個時段不同的基準線計算出每個時段的動態臨界值，若網路流量資訊大於該動態臨界值則可能代表有異常行為的存在。

另外，由王曠銘、羅孟彥、楊竹星所提出的“基於NetFlow之大型網路蠕蟲偵測系統”[1]當中主要以“連向大量的IP位址上的特定目的port”以及“大量的失敗連線”的方法進行蠕蟲偵測。若某IP符合連向大量的IP位址上的特定目的port即為“掃描”，而大量失敗連線則是使用方向性為逆向且TCP RESET的數量是否超過臨界值來判斷連線失敗的標準，若同時具有掃描以及失敗連線數高於臨界值的特性，則判定為感染蠕蟲。此篇文章必須考慮到Flow的方向性以計算某個IP回應Flow的TCP RESET數目，因此所需花費的時間較長。而我們透過一個十分鐘的NetFlow歷史檔案中發現A主機感染了SQL Slammer蠕蟲，並使用TCP 1433 port向B網段中249個IP進行掃描，而這249個IP卻只有4組IP以port 1433回應A主機，也就是說偵測到失敗連線比例只有1.6%。因此可以發現若攻擊連向網段中許多不存在的IP，這些位址因為主機不存在因此也不會回應TCP RESET給攻擊主機。如此一來對於連向不存在的主機的連線將會無法使用此方法進行偵測。

3. 研究方法

本章以下將分成三個小節對於整個研究進行探討，第一節將先對整個實驗環境與樣本的選擇進行說明，包含一開始的實驗環境架設說明以及正常、異常所使用的樣本選擇。第二部分則是對整個實驗的流程進行說明，從一開始的NetFlow檔案產生到系統分析結果將會進行詳細的解說。最後第三個部分則是本研究對於異常偵測所提出的方法介紹，本文提出了一個系統，此系統包含三種異常流量辨識方式，分別是檔案的大小、每個 IP 總連線數以及每個 IP 的 Flow 未回應比例，詳細內容將於下面三個小節進行詳細的介紹。

3.1 實驗環境說明與實驗樣本選擇

在此小節我們將分成兩個部分進行解說。第一部分將對於實驗環境進行說明，包含了NetFlow 常用架設方法與本研究實際架設 NetFlow 的方式，另外還有本研究中所使用的軟體、硬體環境架設進行解說；第二部分則為實驗樣本選擇。

3.1.1 實驗環境說明

在實驗開始之前我們必須準備一個能夠輸出 NetFlow 格式的環境，一般來說產生NetFlow 的環境架設方法主要有兩種，第一種是透過支援 NetFlow 功能的 Cisco Router 直接產生 NetFlow 檔案，並透過 NetFlow UDP Packets 傳送給接收主機進行分析。若沒有支援此功能的設備則可透過網路設備的 Port Mirror 功能或是使用網路分流器 (TAP, Test Access Point) 將流量鏡像或複製到其他設備，同時執行能夠將複製的流量進行轉換的工具將流量轉成 NetFlow 格式，最後再將流量轉送給收集主機。

本文使用上述之 TAP 搭配 nProbe[13]及 nfdump[14]的方式架設實驗環境，圖 1 為本研究進行正常流量側錄的實驗環境架設圖，PC1 放置於 TAP 的監聽端口，其主要的功能為側錄與監聽，內部作業系統為 Ubuntu 12.10[17]，並在其中架設能夠輸出成 NetFlow 格式的 nProbe 與 nfdump 進行側錄工作。PC2 為主要製造各種正常流量的主機，其上面配有固定 IP。PC3 與 PC4 為相同網段的主機，主要是當 PC2 必須與另一方進行互相連線(例如使用 MSN、Skype 進行通訊，或是模擬社群網路中互相回應訊息等)時，這兩台主機能與 PC2 進行相互的連線。

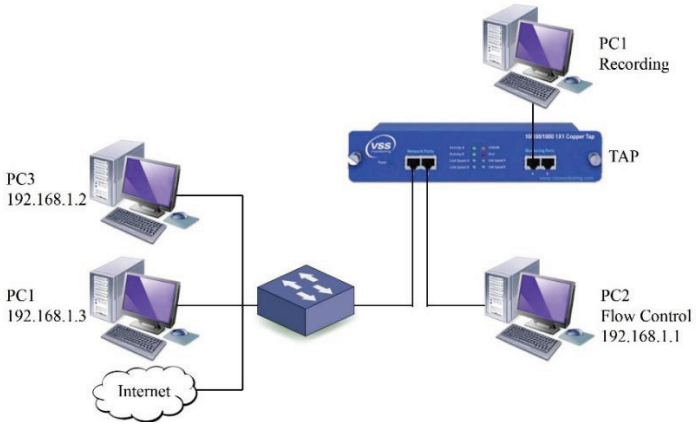

圖 1 正常流量實驗環境架設圖

分析程式使用 Perl 程式語言[15]對 NetFlow 檔案進行分析與判斷。Perl 的原來的功能主要是用來對文字進行處理，並且產生出所需的報表。Perl 借取了很多其他程式語言的特性，其中最重要是其內部整合了正規表示式的功能，這也是 Perl 大量被拿來使用做為文字處理的原因之一。除了 Perl 原有的功能之外，在 Unix-like 的環境下也能使用以 Perl 為基礎的 Shell，讓程式能達到自動化存取 NetFlow 並進行分析，使得管理者在網路的管理方面更加的方便。本研究使用之各種軟體版本如表 1。

表 1 研究使用各軟體版本

軟體名稱	軟體版本編號
Ubuntu	12.10
nProbe	6.2.1
nfdump	1.6.8
tcpdump	4.3.0(內建)
libpcap	1.3.0
Perl 5	14(內建)

本研究系統流程如下圖 2，此系統架設於圖 1 的 PC1 主機中，使用者只須執行啟動程式系統便能完全自動化進行，包含一開始產生 NetFlow 檔案並依照不同時間儲存於不同資料夾到最後分析結果，使用者不需在其中執行額外的指令或動作。

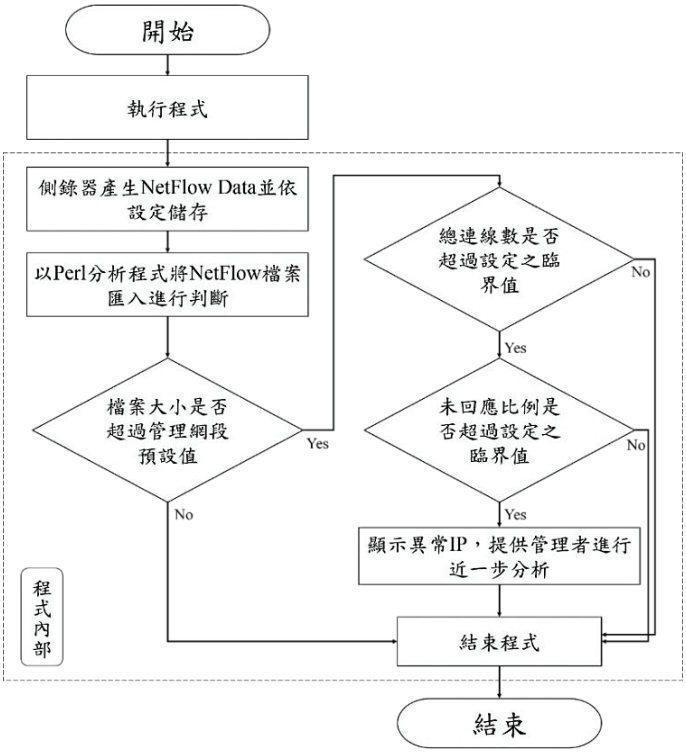

圖 2 系統流程圖

內部分析程式的運作首先在使用者執行啟動程式後，程式會自動執行nProbe與nfdump以產生NetFlow檔案並輸出至收集主機。接著程式能夠自動將NetFlow檔案轉換成txt文字檔並匯入Perl分析程式進行判斷。分析程式可分為三大部分，分別進行檔案大小、IP總連線數以及IP未回應比例分析，分析程式執行完畢若發現有異常威脅則會對管理者發出警報。時間選擇方面若時間取得太短可能造成程式無法快速負荷，若取太長則無法達到快速偵測的效果。因此本研究中產生的NetFlow檔案以十分鐘為基礎時間，分析程式必須在這段時間內分析完畢，才能夠達到快速偵測的效果。

3.1.2 實驗樣本選擇

我們將所需要的實驗樣本分為正常與異常兩種，正常樣本參考了「2012年台灣寬頻網路使用調查報告」[7]，其中一項“全國地區12歲以上曾經有使用寬頻上網民眾常使用寬頻上網功能”的調查，從這之中歸納出幾種網路常見的應用與行為，再從這些常見行為當中挑選幾種不同的應用與行為做為正常流量側錄對象，詳細正常側錄樣本選擇如下表2所示。

表 2 正常流量側錄種類

	種類	網路應用與行為		種類	網路應用與行為
1	搜尋資訊	使用 HTTP 之網頁	5	傳送即時短訊	LINE
2	網路社群	Facebook	6	電子檔案傳送	FTP、BitComet
3	線上觀賞影片	YouTube、風行網	7	網路電話/視訊	Skype
4	收聽網路電台、音樂	KKBOX	8	電子布告欄	BBS

而異常流量樣本選擇的部分因為本研究無法取得能夠造成網路壅塞的病毒、蠕蟲或是DDoS攻擊等樣本，因此使用過去側錄有關於遭受惡意攻擊的NetFlow流量檔案進行分析，期望能夠透過比對正常與異常的網路行為找出異常流量的特性。

3.2 異常偵測方法介紹

本文提出了三種對於異常的檢測方法，分別是透過NetFlow檔案大小、IP總連線數以及IP的Flow回應比例來偵測異常流量。首先透過接收檔案大小判斷此檔案內是否有足以影響網路運作的流量，若未達標準則將其過濾。接著透過實際計算連線總數能過濾大部分正常IP 。最後透過計算每個IP的連線未回應比例，找出在固定時間內發送大量連線導致網路發生異常的主機，即可進行進一步問題的處理與排除。以下三個小節將進行的詳細說明。

3.2.1 NetFlow檔案大小

當管理者從網路上成功側錄到網路封包資訊時，第一個能夠觀察與判斷是否有異常的特徵即是NetFlow檔案的大小。NetFlow的原理是具有相同欄位的封包會聚集成一筆Flow，而其packet數與Bytes數會累加到同一筆Flow當中，只要有其中一個欄位不相同，即會被記錄到下一筆新的Flow。因此依照Flow產生的特性，對於以每十分鐘產生並匯出的一個NetFlow檔案來說，正常的使用者產生出來的流量會於一定的大小之內，假若單一個NetFlow檔案的大小超出預期正常的數值，則可能代表連線種類數過多，例如：相同來源IP連線到許多不同的目的IP，或是使用不同的port進行連接，使得NetFlow的欄位數不相同，導致Flow數增加。因此當檔案中具有異常主機發送足以影響網路運作的攻擊時，NetFlow產生出的檔案大小勢必也會上升。

為了能夠了解一般使用者使用網路大約會產生多大的檔案大小，本研究設計一套實驗，主要是對各種正常應用進行流量側錄，分析每一種應用產生的檔案大小有何特性。而對於每種不同的管理環境來說，並不是固定一組標準值就能適應各種環境，因此本文提出一個方法能夠對於各種管理範圍訂定出適用於該範圍的正常檔案大小。計算公式為(1)：

$$fs = n \times m \times s \tag{1}$$

fs為該網段之檔案大小標準值，n為在管理者所管理的網段下所分配出去有在使用的IP總數，m為每十分鐘該網段每個IP所產生的平均紀錄筆數，m值會根據每個管理網段不同而有不同的大小，因此管理者能夠根據本身管理網段套用該網段的m值。而s則為NetFlow轉成文字檔時其中每一筆紀錄所佔的大小，經過我們實際測試發現，其每一筆紀錄所產生的大小皆是固定的，每筆大約為152 Bytes。因此透過將每個網段IP數、每十分鐘於該管理網段所產生之記錄筆數以及每筆記錄所佔的大小這三者相乘即為此管理網段的檔案大小標準值。藉由此方法能夠觀察設備產生出來的NetFlow檔案的大小是否超過此網段設定的標準值，管理者能夠對於大小超過預期標準的檔案進行進一步的分析與判斷即可，並不需要對於每個NetFlow檔案進行分析，如此一來能減少管理人員與設備的負擔。

3.2.2 每個 IP 總連線數

除了檔案大小之外，對於異常行為的觀測另一個最直接的方法即是觀察每個IP的連線數。本研究我們不觀察整體網路的連線情形，而把每一個來源IP都當成獨立的事件進行統計，其原因是因為觀察整體網路雖然能找出異常，卻無法立即知道是哪一個IP帶有異常的流量。另外一個原因是我們認為每個IP所產生出來的連線都是獨立的事件，因此當進行分析時必須將不同的 IP 分開進行判斷。

在本文分析程式中使用{來源 IP、來源 port、目的 IP、目的 port}這四個欄位為一組做為唯一的一種連線，只要完全符合上述欄位的Flow會被累加到同一種連線當中。而對於總連線數 TC 定義為：單位時間 t 內，某個 IP 總共具有 n 種以該 IP 為發送方的{來源 IP、來源 port、目的 IP、目的 port}組合。根據正常使用者產生的連線大部分具有連向特定目的的特性，若是 TC 值過大，則可能是使用了 P2P 的軟體導致具有許多不同的連線組合產生，或者是因為蠕蟲、病毒等攻擊，其為了能夠快速傳播必須先進行大量的掃描以得知網路上主機使用狀況，此種掃描的行為也會造成大量的連線組合的產生。藉由計算出每一個IP的連線種類數目能夠觀察出正常與異常的連線數的不同，藉此能快速找出有問題的主機。

3.2.3 每個 IP 未回應比例

對於一般正常的網路使用行為來說，使用者通常會具有偏好的使用行為，例如：連線到常用的網頁、使用常用的應用等，正常的連線行為通常具有接收連線請求的一方會進行回應的特性，即使對方因為網路問題或是某些原因導致連線失敗，正常的使用者也不會產生超過人為能製造出的連線請求，例如：大量連向根本不存在的目的主機或是port等。因此本文認為計算每個IP的Flow未回應比例，能夠有助於分析異常流量的存在與否。而對於正常的使用者來說Flow未回應比例應該會是非常低的，較有可能產生誤判的情況為 P2P的使用者所產生出來的流量，因為P2P接收連線的節點有可能關閉或斷線，因此也具有一定機率產生沒有回應的情形。不過P2P與攻擊所產生的流量來說比例應該會低上許多，因為後者通常會對於網路進行大量掃描的動作，因此連線到並不存在的主機或是沒有分配到的IP網段的機率會比一般正常的使用者來的高，也比較容易具有較高的未回應比例。

在計算回應比例之前我們必須先定義什麼樣的Flow才算是沒有回應，透過程式的分析能夠得到每個 IP 的每筆連線組合所送出以及回應的 Flow 數，例如以 192.168.0.1:53 -> 192.168.0.2:52646 這組連線來說，有可能送出的 Flow 數為 4 筆，回應的 Flow 數為 3 筆，也有另外的可能是送出的 Flow 數為 4 筆，回應的 Flow 數為 0 筆。在這兩種組合皆並不具有100%的回應比例，不過因為前者具有 3 筆的回應 Flow，並不能完全定義為失敗的連線。因此本文對於回應比例計算的定義為：只要有任何一筆回應的Flow存在即歸類為成功的連線，相反的若回應的 Flow 數為 0 筆，則定義為沒有回應的連線。透過此計算方法，我們能得到一個 IP 未回應的異常連線總數 k，以及前面統計出的 IP 總連線數 n，透過這兩個值我們能夠計算每個 IP 的 Flow 未回應比例值NR，計算方式如公式(2)：

$$NR = \left(\frac{k}{n}\right) \times 100\% \tag{2}$$

下表3為本文實際透過每種不同網路使用行為側錄的NetFlow檔案中計算出的NR值，由此表可發現在正常使用的網路之下所計算的NR值為較低百分比，另外也可以看出雖然BT所產生的總連線數較高，但透過NR值的計算能夠被過濾掉。透過NR值我們能夠將因為連線總數過高所過濾出的流量進行未回應比例分析，找出真正對於網路具有影響的主機，而濾掉像是P2P這類雖然會產生大量連線數但對網路並沒有嚴重影響的IP。

表3 正常流量側錄種類十分鐘所產生記錄筆數

種類	總連線數	異常數	NR	種類	總連線數	異常數	NR
FTP	72.25	13	17.99%	HTTP	400	7.25	1.81%
FB	75.25	11.5	15.28%	BBS	76.25	7	9.18%
Funshion	338	42	12.43%	LINE	173.25	7.75	4.47%
KKBOX	109.75	6.5	5.92%	SKYPE	37	8.25	22.30%
YouTube	111.25	13.75	12.36%	BT	6479.25	1820.75	28.10%

透過以上三個循序漸進的方法，從一開始接收檔案進行檔案大小的分析，接著將需要處理的檔案進行每個IP的總連線數統計，再將連線數過高的IP進行未回應比例的分析，經過這三個步驟最後過濾出的IP即是對於整個管理網段具有較大影響的主機，管理者能夠透過警報中的這些主機進行更仔細的判斷與處理，若真的有類似病毒、蠕蟲等攻擊，能夠在剛開始攻擊時就把異常IP進行封鎖，以免造成更多主機遭受感染而使問題更加嚴重。

4. 研究成果

目前的網路上的攻擊手法有許多種，每一種攻擊行為皆有不同的特徵，因此若使用特徵的方式判斷勢必只能針對某一種攻擊行為進行分析。對於網管人員來說，任何一種網路威脅都有可能造成網路災害的發生，因此是哪一種威脅種類其實並不重要，只要能快速找出可能造成威脅的主機，針對這些主機進行處理即可避免更多的災害發生。因此本文提出於NetFlow所提供的資訊，搭配Perl進行分析，程式必須能夠在下一個檔案產生之前執行完畢，並預期能夠準確的找出大部分具有影響網路運作的IP位址。以下將對於本文所提出的三個研究方法進行結果分析。

4.1 檔案大小結果分析

在檔案大小的分析方法中，我們已知管理IP數目與每筆紀錄的Bytes數(152 Bytes，此為使用 nPorbe以及nfdump所產生出來之NetFlow檔案每筆記錄大小)，因此實驗將對十分鐘內每個IP大約會產生多少筆的連線紀錄進行分析與探討。首先對於前面提到的十種正常流量種類進行流量的側錄，每一種應用與行為在側錄時根據不同日期分別側錄四個檔案並進行平均。每一個檔案皆是以十分鐘為基準，因此我們能得到每一種應用在十分鐘之內所產生的連線數，包含由本身IP送出的Flow數以及由對方回應的Flow數，實驗數據如下表4。

表 4 正常流量側錄種類十分鐘所產生記錄筆數

種類	送出 Flow	回應 Flow	種類	送出 Flow	回應 Flow
FTP	102	80.75	HTTP	448.25	437.5
FB	86	71.5	BBS	92.25	78.5
Funshion	536.75	485.5	LINE	197.25	185.5
KKBOX	204.5	186	SKYPE	70.25	58.75
YouTube	186.25	155.25	BT	7971.5	5980.25

根據表4的數據我們能夠觀察到除了BT之外，其他大部分的應用每十分鐘所產生的送出流量記錄筆數最多大約在537筆之內，而這其中又除了風行網以及HTTP會發送大約400~500筆流量記錄之外，其餘皆在200筆記錄左右，甚至在每十分鐘只會發送100筆以內的Flow筆數。由回應Flow的部分則可以看出大部分應用對於送出Flow的都有正常的回應Flow與之相對應。在這些應用當中唯一的最大不同就是BT所產生出來的流量，因為BT本身會向散落在其他各地的位置進行連線，以能達到快速將所需資料下載完成的特性，因此其本身在十分鐘之內會產生的記錄筆數相對地就會比較多，而回應的部分因為網路上其他的主機並不一定有存活，因此BT所產生的回應Flow相對於其他應用來說會比較少。

因為BT所產生的流量筆數與病毒蠕蟲較為相似，因此在檔案大小的計算中本實驗不加上BT的流量進行計算。我們將除了BT之外的其餘九種應用所產生出來的記錄筆數進行平均，能夠得到每十分鐘之內一個IP大約產生214筆記錄，而回應的平均紀錄為193筆，也就是在一個十分鐘的NetFlow檔案之內每個IP大約會產生出407筆的記錄，每個IP在10分鐘之內大約會產生的檔案大小約為60.4KB。而此種使用平均值的計算方式在IP的數目上有些限制，我們以一個IP於十分鐘內會產生60.4KB的方式計算在IP數目不同時所產生的檔案大小，以及其具有異常流量時檔案大小的放大倍數，詳細數據如表5所示。

表 5 具有異常流量下檔案大小放大倍數

管理 IP 數	檔案大小 (單位：KB)	異常流量 (922KB) 放大倍數	異常流量 (1640KB) 放大倍數
50	3020	1.17	1.41
100	6041	1.09	1.20
150	9062	1.06	1.14
200	12082	1.04	1.10
256(Class C)	15466	1.03	1.08
512(2 倍 Class C)	30932	1.02	1.04
768(3 倍 Class C)	46398	1.01	1.03
65536(Class B)	3959296	1.00	1.00

在表5中我們使用兩個感染疾風蠕蟲的IP所產生出來的記錄筆數進行測試，兩個檔案分別送出6212筆與11046筆向大量IP的135 port進行連線的Flow，兩者皆沒有任何回應的Flow存在，因此所產生的檔案大小為922KB與1640KB。透過此表我們能觀察到雖然每個IP所產生的檔案大小為60.4KB，與異常流量所產生的大小相差很多，不過當既有的IP數過多時，該異常流量在於整體的放大倍率就越少，尤其若本身一個異常IP流量所發動的攻擊流量數較少時，會因為其異常筆數佔整體比例太低而導致檔案大小變動並不明顯，例如以一個Class B網段65536個IP數來說，即使有一個IP產生異常的流量，對於整體的檔案大小並不會有太大的改變，除非該範圍內有大量的異常IP存在，才有可能對於改檔案大小有影響的作用。因此本研究的檔案大小計算方式建議使用於大約一個Class C的網段IP數當中，若IP數再增加可能會導致整體檔案大小過大而使異常流量不明顯。另外本計算方法未將BT納入計算範圍，因為BT會與病毒、蠕蟲等惡意攻擊產生類似的大量連線，因此若是該網段具有較多的IP數或者是該網段開放使用BT或主機成員時常使用BT的情況下，建議能夠忽略此計算檔案大小的方法直接進行以下的總連線數以及未回應比例分析即可。

4.2 總連線數結果分析

此小節將對於總連線數與未回應比例進行分析，首先實驗樣本我們使用過去於2004年3月4日整天所側錄的144個檔案進行選擇，在這之中我們選擇三個檔案進行總連線數與未回應比例的分析樣本，此三個檔案分別是於8:30、14:30、20:00三個時間點所側錄的流量，大約是一個校園當中早中晚三個時段，本實驗將以此三個檔案進行分析，期望能找到一個較適合的標準以提供管理者進行過濾。

首先針對此三個測試檔案必須先將校內與校外IP進行分離，因為對於管理人員來說主要還是管理校內IP的使用情形，因此本實驗測試仍以校內IP所產生的數據為主。透過計算得到每個校內IP所產生出來的總連線數與未回應比例，再透過每個校內IP的數據計算出每個檔案的平均值與標準差以提供後續分析使用。詳細平均值與標準差資訊如下表6所示。

表6 三個測試檔案的平均值與標準差

	校內IP數	總連線數平均值	總連線數標準差	NR平均值	NR標準差
083001	3478	97.54	359.90	12.64	25.87
143000	5185	79.56	322.46	10.14	23.14
200000	5521	73.06	315.93	9.924	22.87

接著我們對於此三個檔案找出其中具有異常流量的IP數量，之後透過上述計算出來的平均值與標準差進行總連線數標準差(TCS)與未回應比例標準差(NRS)測試實驗。首先是總連線數的標準差(TCS)測試實驗，我們將NR的標準暫時固定為25%，再利用0.5-3倍TCS觀察其過濾的狀況，測試在每種不同標準差之下是否能夠成功找到異常的IP，以及計算出使用25%的NR值所產生的誤報IP數。接著提高NR過濾的比例，分別以2倍NRS以及2.5倍NRS進行誤報IP過濾實驗，觀察在哪一種NRS之下能過濾掉誤報的IP，若誤報IP

本身的 TC 值小於測試 TCS 且 NR 值小於測試 NRS 則會被過濾，不符合上述標準則代表此誤報 IP 無法被過濾掉。此測試範圍不包含 3 倍 NRS 測試，原因為 3 倍 NRS 在一開始進行異常 IP 過濾時已經會找不到某些異常的 IP，因此不採用。我們將使用 2.5 倍 NRS 與 2 倍 NRS 搭配各種 TCS 將誤報 IP 過濾的情形繪製成圖，如圖 3、圖 4。

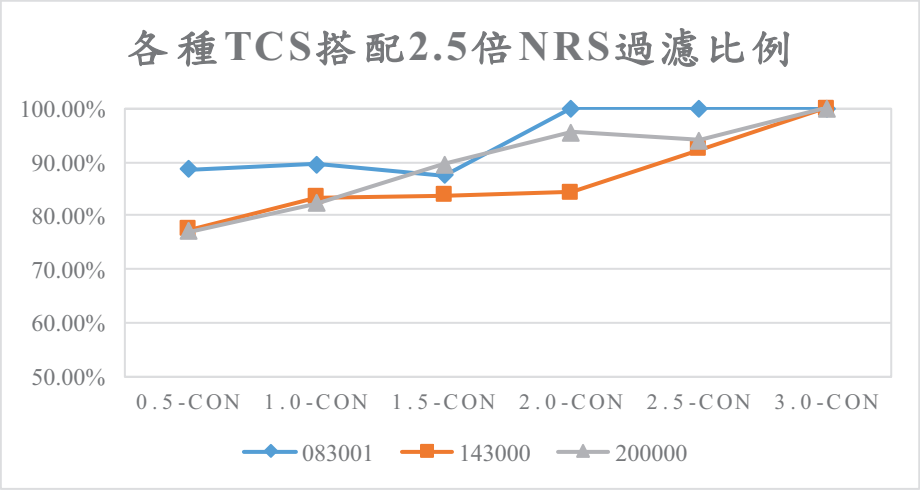

圖 3 各種 TCS 搭配 2.5 倍 NRS 對誤報 IP 過濾比例

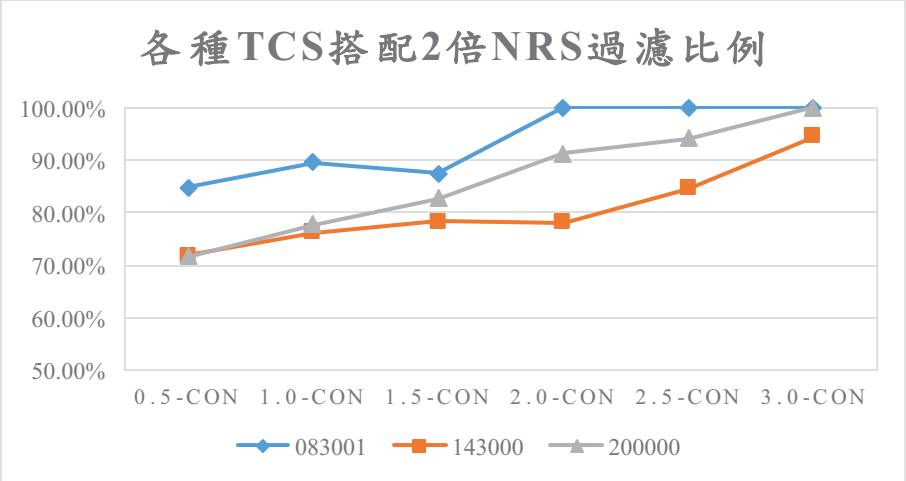

圖 4 各種 TCS 搭配 2 倍 NRS 對誤報 IP 過濾比例

由上面兩張圖我們可以觀察到在 TCS 部分最好的情況是 3 倍的 TCS 差搭配 2.5 倍 NRS，此種組合能將誤報的 IP 完全過濾掉，若使用 3 倍的 TCS 搭配 2 倍 NRS 雖然會少濾掉一個 IP，不過也是具有較高的過濾比例，而其他的組合則會有較多 IP 未被過濾，因此較不建議採用。

4.3 未回應比例結果分析

在 IP 未回應比例的部分所進行的實驗也是使用標準差交叉比對的方式，首先針對三個測試檔案找出裡面有異常行為的 IP，並將總連線數的標準暫時設定為 250，此為較低的標準。接著使用 0.5-2.5 個 NRS 進行測試，並記錄每種 NRS 是否能確實過濾出上述的異常 IP，

藉此能找出每個檔案在不同標準差之下誤報IP數。而本實驗不針對3倍NRS進行實驗，原因在於一開始的實驗中因為標準差太高已經會有忽略某些較嚴重IP的情形出現，因此不採用。接下來使用每個檔案各自的3倍、2.5倍以及2倍TCS進行過濾，觀察哪一種組合過濾的效果較好，若誤報IP本身的TC值小於測試TCS且NR值小於測試NRS則會被過濾，不符合上述標準則代表此誤報IP無法被過濾掉。不同的TCS值搭配結果如圖5、圖6以及圖7，分別是使用3倍、2.5倍、2倍TCS過濾的結果，過濾比例越高則代表偵測效果越好。

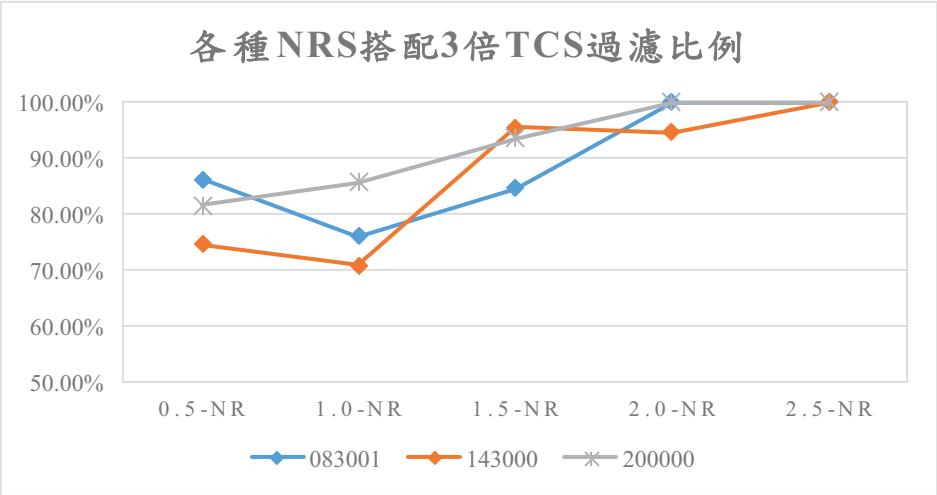

圖 5 各種 NRS 搭配 3 倍 TCS 對誤報 IP 過濾比例

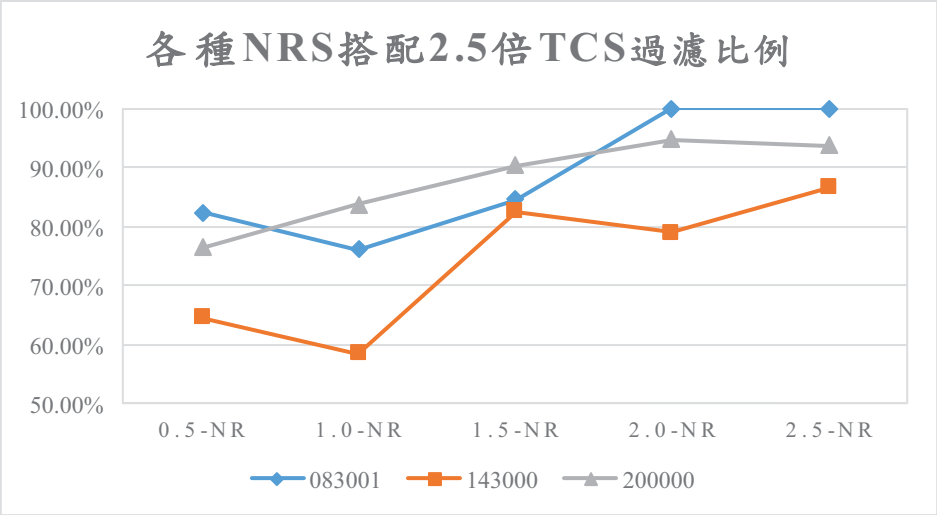

圖 6 各種 NRS 搭配 2.5 倍 TCS 對誤報 IP 過濾比例

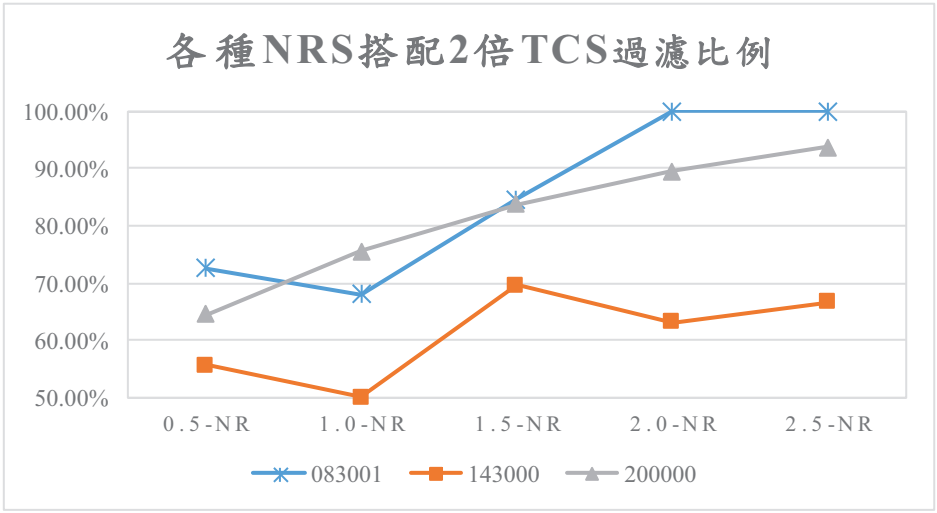

圖 7 各種 NRS 搭配 2 倍 TCS 對誤報 IP 過濾比例

經由上面三張圖可清楚的發現，使用 3 倍 TCS 時在 2 倍以上的 NRS 都能達到較好的過濾效果，而 TCS 若降到 2.5 倍以下時即使使用 2.5 倍 NRS 進行過濾，其效果會比較差一些，因此分析結果建議使用 3 倍 TCS 搭配 2.5 倍或是 2 倍的 NRS 會達到較好的偵測效果。此結果與上一小節對於 TCS 分析結果相同。

4.4 總連線數與未回應比例實際測試

為了測試上述標準差分析結果所得到的標準值是否能正確找出異常主機，本實驗將針對 3 倍 TCS 搭配 2 倍/2.5 倍 NRS 所得到的標準進行實際測試。首先使用 2004 年 3 月 4 號一整天所側錄所得到的 144 個 NetFlow 檔案(每個檔案皆為十分鐘)進行統計，計算出每個檔案所得到的平均值與標準差，再將所有標準差進行平均，因此得到 144 個檔案 3 倍 TCS 的平均值為 1111，2 倍 NRS 的平均值為 60，2.5 倍 NRS 為 73。本實驗將使用得到以上標準值進行異常檔案的過濾，觀察推論標準值是否能正確過濾出有異常的主機。

測試檔案選擇不同年份的三個帶有異常流量的 NetFlow 檔案，分別是於 2004 年 3 月 8 日上午 11 時、2005 年 6 月 3 號下午 3 時以及 2013 年 5 月 3 號下午 4 時 40 分於清華大學校內網路側錄的三個 NetFlow 檔案。實驗將使用推論的標準對此三個檔案進行過濾，觀察其過濾情況。測試檔案中帶有異常主機情形如下表 7。

表 7 實際測試檔案異常 IP 數

	威脅性較高 IP 數	威脅性較低 IP 數
ft-v05.2004-03-08.110001	8	5
ft-v05.2005-06-03.150001	7	2
ft-v05.2013-05-03.164000	4	4

在此表中又依照對於網路運作影響的程度高低進行分類，這些威脅性較高的主機中包含有 Sasser 蠕蟲(透過通訊埠 TCP 445)、疾風 Blaster 蠕蟲(利用通訊埠 TCP 135)、SQL Slammer

蠕蟲(利用通訊埠 UDP 1434、1433)、Code Red 紅色警戒病毒以及大量對網段進行掃描等，而這些異常流量則是本實驗中希望能夠過濾出的主機。

實際測試結果如表 8,透過此實驗我們可以觀察到在使用 3 倍 TCS 搭配 2.5 倍 NRS 於三個檔案的過濾結果大部分皆能正確找出帶有較高威脅性的主機,而威脅性不那麼高的主機則會被過濾掉;相對的若是使用 3 倍 TCS 搭配 2 倍 NRS 所得到的結果當然在較高威脅性的部分能夠完全的找出異常主機,不過因為標準較寬鬆的關係,容易另外找到一些威脅性較低的主機,若是某個 NetFlow 檔案有許多低威脅性主機,則此種搭配方式將會過濾出較多的 IP,因此管理人員需要在短時間內對於較多的 IP 進行分析。而若使用 3 倍 TCS 搭配 2.5 倍 NRS 所得到的結果雖然未能過濾出威脅性較低的異常主機,不過卻也因為過濾出的主機都具有較高威脅性，管理人員能夠針對這些主機進行分析與處理即可。

表 8 實際測試檔案於各種標準差下所過濾異常 IP 數

	1111/60 過濾威脅性較高 IP 數	1111/60 過濾威脅性較低 IP 數	1111/73 過濾威脅性較高 IP 數	1111/73 過濾威脅性較低 IP 數
ft-v05.2004-03-08.110001	8	2	8	0
ft-v05.2005-06-03.150001	7	0	7	0
ft-v05.2013-05-03.164000	4	0	3	0

以上兩種標準提供給管理者自行選擇,若需要找出具有潛在威脅的主機則可以使用 2 倍 NRS,但相對的管理者需要花較多時間對顯示出的 IP 進行分析;而若使用 2.5 倍 NRS 則是能夠找到威脅性較高的主機,雖然忽略了潛在威脅的主機,不過管理者相對的不需要花許多時間對於潛在威脅的主機進行分析，只需針對高威脅性的主機進行處理。

由以上的推論與結果可以發現使用本文所提出的過濾方式能夠在短時間內快速的於大量的流量當中計算出每個 IP 的總連線數與未回應比例,再利用實驗所推論的標準能正確找出檔案中具有異常流量的主機,並提供警報給管理者。本文所提出的過濾方式並不限定於異常的種類,因為大部分的具有影響網路運作威脅性的攻擊具有高連線數與高未回應比例的特性,因此不管是哪一種攻擊總類,只要具有此特性皆可被找出。而過去使用的方法中若是使用特徵比對的方式大部分只能針對某一種異常進行偵測,若需要同時針對不同的異常種類則需要將每種特徵全部進行比對,不僅需要花較多時間且當有新種類的攻擊發生時因為無該異常特徵因此無法立即偵測。而觀察整體網路狀態雖然能依照過去的流量觀察出目前是否產生異常,不過在發現異常之後卻無法知道造成異常的主機,因此需要再更深入的分析才能確認異常主機並進行處理。當然目前市面上也有許多的產品能夠達到很好的偵測效果,不過通常這些軟體或設備需要額外負擔較昂貴的費用。因此透過本文所提出的自動化系統中,管理者透過指令的執行開啟流量側錄並進行判斷,程式能夠於下一個檔案產生之前分析完畢並提供警報,因此在十分鐘之內即可判斷目前存在的異常主機,達到快速偵測的效果。

5. 結論

在龐大的網路流量當中，雖然管理者能夠透過使用者的反應或是利用某些工具觀察到網路有所異常，但要是無法在短時間內無法明確的知道造成異常的主機並進行處理，將會造成災害持續的擴散，直到發現及解決問題點時，異常狀態已經持續一段時間，也可能已經造成不少的網路災害發生。因此本研究使用不須占大量空間，卻又能保有相關細節的NetFlow進行正常及異常流量的偵測，經由觀察正常的網路行為，以觀測出異常流量所具有的行為特徵。透過本論文提出的方法，從接收到NetFlow檔案時立即進行檔案大小的分析，協助管理者進行最初的判斷是否有異常行為存在，接著透過每個IP的連線總數找出可能影響網路運作的主機。當然產生具有較大的連線總數有可能是使用了P2P軟體，因此再透過未回應比例找出具有較高比例的主機，藉此判斷可能為異常的攻擊。而透過實際NetFlow檔案進行測試，結果顯示確實能利用本文所提出的方法取得連線數以及未回應比例，利用此組標準值能於十分鐘之內分析出目前網路是否有異常，因此能有效的達到快速偵測異常的效果。

目前系統已經能夠完全自動化的執行，使用者只須執行程式，程式首先會開啟NetFlow以產生NetFlow檔案，接著自動抓取檔案並在下一個檔案生成前執行完畢並提供警報給管理者。而程式部分建議將NetFlow轉換成txt文字檔的部分利用其他方法進行取代，因為使用I/O會耗費比較多的時間，因此若能簡化這部分將能提高程式執行的效率。另外如果能夠透過機器學習的方式根據不同環境自動調整參數，例如：當學校放暑假時能在可接受的時間之內將參數調整成適用於暑假假的參數值，若能完成此部分對於管理者來說不需要隨著環境改變而手動更改參數值，更能達到本文所希望得到的自動化偵測的效果。

6. 參考文獻

[1] 王曠銘、羅孟彥、楊竹星，基於 NetFlow 之大型網路蠕蟲偵測系統，台灣網際網路研討會(TANET)，2005。

[2] 徐偉智、王明輝，利用 Netflow 即時偵測蠕蟲攻擊，台灣網際網路研討會(TANET)，2006。

[3] 陳彥錚、張嫄媗、張家瑋、王士豪，基於網路訊務動態基線分析之網路蠕蟲偵測機制，第十屆資訊管理暨實務研討會，2004。

[4] 黃文穗、林守仁，利用 Netflow 建置 Code Red Worm 偵測系統，台灣網際網路研討會(TANET)，頁 471-474, 2001。

[5] 劉俊華，基於 NetFlow 之網路異常偵測系統，碩士論文，中興大學電機工程學系所，2007。

[6] NetFlow 網路分析應用, http://www.tcci.com.tw/information01.aspx?id=MjAwOTAxMTUwMDI=

[7] 2012 年台灣寬頻網路使用調查報告(3 月), http://www.twnic.net.tw/download/200307/20120709d.pdf

[8] Akamai, The State of the State, 4th Quarter, pp.6-10, 2012. http://www.akamai.com/stateoftheinternet/

[9] B. Claise et al., “Cisco Systems NetFlow Services Export Version 9” RFC3954, October 2004.

[10] Introduction to Cisco IOS NetFlow, http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555/ps6601/prod_white_paper0900aecd80406232.pdf

[11] MRTG, http://en.wikipedia.org/wiki/Multi_Router_Traffic_Grapher

[12] NetFlow Services Solutions Guide, http://www.cisco.com/en/US/docs/ios/solutions_docs/netflow/nfwhite.pdf

[13] nProbe, http://www.ntop.org/products/nProbe/

[14] nfdump, http://nfdump.sourceforge.net/

[15] Perl Mongers, http://www.perl.org/

[16] SNMP, http://en.wikipedia.org/wiki/Simple_Network_Management_Protocol

[17] Ubuntu, http://www.ubuntu-tw.org/

[18] Wang Jinsong, Liu Weiwei, Zhang Yan, Liu Tao, and Wang Zilong, “P2P Traffic Identification Based on Netflow TCP Flag”, International Conference on Future Computer and Communication, pp.700-703, 2009.