以隱藏 IP 位址為基礎之 Ethernet 區域網路安全管理機制

董呈煌
屏東商業技術學院資訊科技系
chdong@npic.edu.tw

黃敏裕
屏東商業技術學院資訊管理系
s92306020@npic.edu.tw

摘要

隨著危害資訊安全手法的推陳出新，使得企業對 Lan 上收送資料的安全性與可靠性愈來愈重視。本篇論文對使用 TCP/IP 協定的 Ethernet LAN 安全性，以網路通訊介面層的“NDIS Hook”技術為基礎，提出一套隱藏 IP 位址為基礎的安全管理機制。該安全管理機制是從 TCP/IP 協定三層次進行各項安全控制，針對各層次可能會洩露 IP 資訊部份加以分析並進行通訊 DES 加密與認證機制。實驗結果顯示，未安裝本安全管制軟體的主機無法與已安裝本管制軟體的主機或閘道器之間進行通訊，即使攔截網路上傳送的封包，也無法獲得相關的 IP 資訊，可確實達成以此管理機制對於 Lan 未經授權的行為進行安全控管的目的。

關鍵詞：Lan、NDIS Hook、TCP/IP、DES

Abstract

The reliability and confidentiality of data transfer over Local Area Networks (LANs) in enterprises are more and more important due to the continually emerging of information security problems. This paper discusses the “NDIS Hook” technology in the network communication layer and how to apply this technology to improve the security of TCP/IP networks. We propose an effective mechanism which focuses on hiding the IP addresses of network interfaces by analyzing the appearance of the IP addresses among different network layers. This methodology manipulates the TCP/IP protocol stack of windows platform to secure a TCP/IP LAN against unauthorized hosts.

Keywords：LAN、NDIS Hook、TCP/IP、DES

1. 緒論

網路管理一直是Lan上管理的重要議題，在Ethernet Lan中收送TCP/IP封包時存在著許多安全上的問題，例如通訊的資料竊聽、封包資料偽裝及內部人員惡意攻擊等[1]。因為TCP/IP協定當初在設計時，認為安全問題應該在應用層中解決，而非在TCP/IP協定中解決，但是隨著軟硬體技術的進步，設計於應用層的安全協定以協助應用軟體提升通訊時的資料隱密性[2]，其安全性已經不敷需求，且TCP/IP本身無法在基本的網路底層之下提供安全的加密通訊及認證機制。

為補強現存作業系統之網路通訊上安全管理缺失，因此在視窗作業系統網路架構平台上實現對網路封包進出的過濾機制以提高安全性[3]。視窗作業系統網路架構基礎是提供NDIS應用程式介面的NDIS裝置驅動器。NDIS是一種由微軟和3Com公司共同發展的網路驅動程式介面規格，其定義了一個標準資料連結層介面，供上層各種協定堆疊同時存取位於NDIS驅動器之下的網路介面實體，微軟提供驅動程式發展工具(DDK)的開發環境來設計相關驅動程式軟體[4,5]。其中，對於網路驅動程式的特殊應用“NDIS hook”技術，是利用驅動程式有高於應用程式的執行權限，優先對特定型態的封包，透過NDIS核心驅動程式來實現對NIC(Network Interface Card)收送的封包進行轉換與過濾[6]。

目前許多研究可以透過ICMP、TCP或未定義使用之連接埠技術來建立隱藏通道，將資料掩藏在IP和TCP等headers中，經由轉換IP識別、初始化Sequence Number或TCP Acknowledge Sequence Number欄位等方式來達到隱藏通訊通道[7,8]。

因此本論文提出加強TCP/IP協定安全功能的機制，並將隱藏Ethernet Lan中傳送的所有TCP/IP封包之IP位址列為首要重點。針對ARP協定[9]與TCP/IP協定[10]的欄位分析其內涵及格式，以及可能洩露IP資訊的應用層，運用DES加解密技術於封包的內容中，使達到隱藏IP資訊及認證身份的效果。

本論文相關章節分述如下，第2節簡介NDIS與Hook的原理與應用，第3節針對使用TCP/IP協定的Ethernet LAN安全加以探討，第4節著重於ARP協定的重要性及應用，第5節著重於IP協定的原理與改良，第6節說明應用層的IP隱藏機制，第7節將上述應用以實驗驗證，第8節結論說明。

2. 相關工作

Microsoft Windows 系統支援網路驅動程式介面規格(NDIS)協定驅動程式，它包含支援 TCP/IP、NetBEUI、IPX/SPX 等相關協定。NDIS 提供網路驅動程式開發的一個標準介面，並且它使得網路驅動程式更好的跨平台。Hooking 是程式攔截和修改正在執行程序的共通機制。Windows 有許多我們無法得知的內部結構。因此，運用該技術便可以進一步了解內部系統架構，甚至在許多情況下，可以解決無法由一般程序所處理的事。如圖 1 所示，假如 NDIS Hook 欲針對 NetBEUI 及 TCP/IP 兩種協定所

進出的封包來進行檢查與過濾，如此，系統必須修改原本此兩種協定與 NDIS 之間所建立的連接通道函式介面，並指向 NDIS Hook 所提供的檢查與過濾程序函式，封包路徑便完成 Hooking 動作。

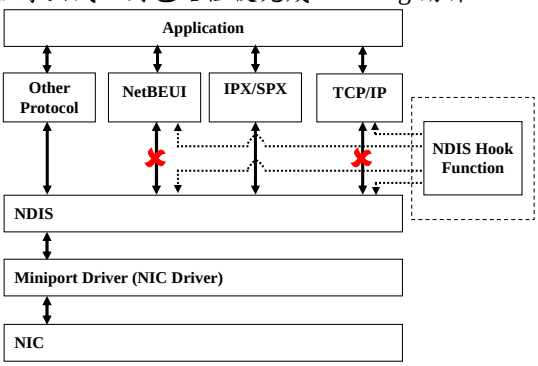


圖1 攔截封包路徑

在 Windows 系統內，各種協定形成一個單向鏈結，是由 NDIS 負責維護這些協定。每種協定結構皆有 NextProtocol 指標指向下一個協定的記憶體位址。當新協定向 NDIS 註冊時，NDIS 便將新協定結構存放於單向鏈結的開端，它只要使用 NextProtocol 指標就能輕易地逐一找尋到所有協定。就前述針對的 NetBEUI 及 TCP/IP 協定為例，在所有協定皆向 NDIS 註冊後，再向 NDIS 註冊另一個 Dummy 協定（如圖 2 中步驟 1），便取得協定鏈結的開端指標，再依據協定結構的 NextProtocol 指標依序比對找出 NetBEUI 協定（如圖 2 中步驟 2），並修改收發路徑的指標指向 NDIS Hook 本身函式（如圖 2 中步驟 3）。接著再依目前的 NetBEUI 協定再依序往下找尋 TCP/IP 協定（如圖 2 中步驟 4），同樣也修改收發路徑指標（如圖 2 中步驟 5）即完成攔截。

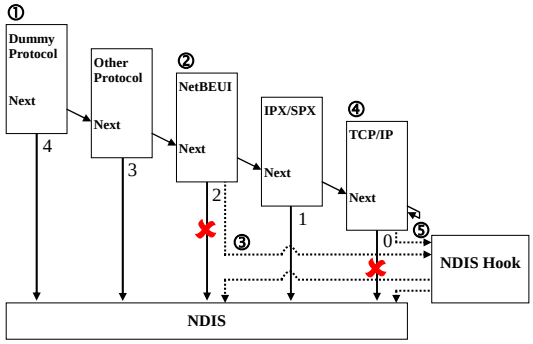


圖2 攔截協定註冊串列結構

3. 使用 TCP/IP 協定的 Ethernet LAN 安全性探討

隨著 Internet 服務的日益興盛，可以經由所在的 Ethernet Lan 連上 Internet，已成為這個 Lan 上使用者的最基本需求。故在一個 Ethernet Lan 上收送 TCP/IP 封包，已成為 Lan 與 Internet 互動的唯一選擇。但是在 Lan 上的使用者，卻因為在 Ethernet Lan 中收送 TCP/IP 封包的不嚴謹性，而經常承受非常大的安全缺失。即 TCP/IP 協定本身無法確認封包來源身份或是檢驗封包是否被授權在 Lan 上傳送，使得未經授權行為所導致的安全管理問題。

最常見到如圖 3 所示，未經授權主機以手動方式設定固定 IP，將會衍生出兩類問題，其一為造成 Lan 中擁有該 IP 之合法主機的 IP 衝突，可能導致兩方皆無法上網；另一為雖未與他人 IP 衝突，但因該 IP 並非經合法分配，若此未經授權之上網引發問題（例如網路犯罪），後續將難以透過追蹤 IP 之使用而找出真正使用者。

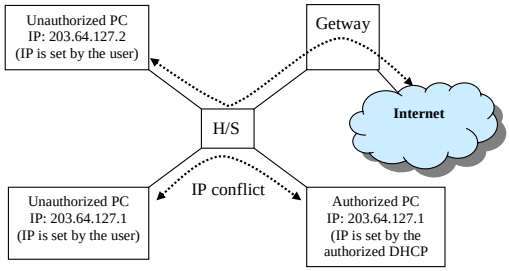


圖3 區域網路未經授權行為衍生問題

TCP/IP 協定在 Lan 中引發的上述問題，常讓網路管理員無法解決。本研究針對 TCP/IP 在 Ethernet Lan 中的特性，提出 TCP/IP 協定適用於 Ethernet Lan 具安全解決的方案，使封包在 Lan 傳送增強安全及認證機制。為發展在 TCP/IP 協定底層資訊保護技術，本研究以隱藏在 Ethernet Lan 中傳送的所有 TCP/IP 封包之 IP 資訊為重點。因為 IP address 是每部電腦在 Internet 中的定位資訊，及 TCP/IP 封包傳送中的重要資訊；若在 Lan 中，互相來往的 TCP/IP 封包無法被其他 host 正確的解讀 IP 訊息，且未經授權的 host 無法在自行設定 IP 後與其他 host 溝通，則可避免對特定 IP 的監控及冒用。

本研究以下列三項層次達到完全的 IP 資訊隱藏效果：

1. ARP 協定加解密：防止未授權電腦建立 binding address 資訊
2. IP 協定加解密：防止 Ethernet Lan 內的使用者 IP 遭竊伺，並防止未授權電腦使用 IP 協定與其他 host 收送封包
3. 應用層協定加解密：防止應用層協定內容洩露 IP 訊息

經實驗證實，只要能完成以上三項層的 IP 資訊隱藏，如圖 4 中，未授權 host 自訂 IP 所衍生的兩個問題，均因該 host 送出的 TCP/IP 封包與授權 host 所發封包的規格不同，而均不再存在。

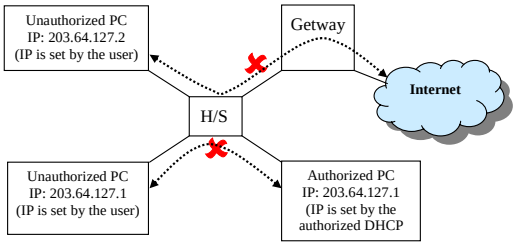


圖4 防止未經授權之行為

4. ARP 原理與應用

Ethernet Lan 中，MAC address 是 Ethernet frame 能否送達的最重要屬性，然而主機若要獲得 MAC address，則它將會送出一個 ARP 的詢問封包，以尋求該 IP 主機的回應，來獲得正確的 MAC address。ARP 協定便是扮演重要的角色，它負責 IP 與 MAC address 之間的對應與轉換功能。因為此 ARP 的功能，使得 ARP 在 Lan 可扮演第一道重要關卡角色。

當某台在 Lan 上的主機並未安裝管制軟體，該主機在 Lan 上與其他已安裝管制軟體的主機即完全無法用 ARP 取得 address binding 資訊，使該主機無法進行任何 IP 封包發送。故本研究首先提出於 Lan 上任兩台主機，於發送端對 ARP 協定加密及接收端對 ARP 協定解密與認證的機制，來達到上述目的。

首先，我們將先分析 ARP 封包的欄位內涵及格式，以決定哪些欄位將在發送端送出前將其加密，以達到保密效果，並在接收端將同樣的欄位解密並予以驗證，只有通過驗證的 ARP 解密封包才需要繼續往上傳送處理，未通過驗證的封包即予丟棄。

ARP 格式如圖 5（含 Ethernet header），ARP 本身格式最重要為發送端及接收端的 MAC address 及 IP；與 Ethernet header 相比，我們發現在發送端及接收端的 MAC address 是完全重複，這使得我們提出在 ARP request 封包發送前可進行的加密程序。

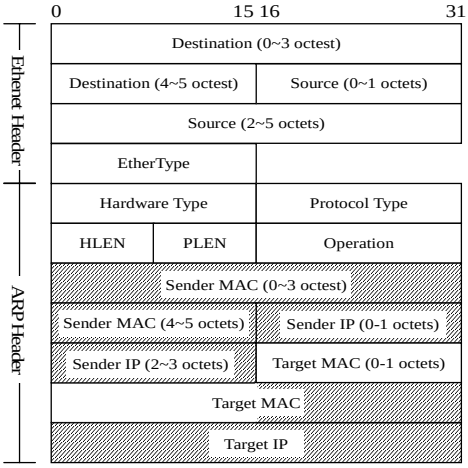
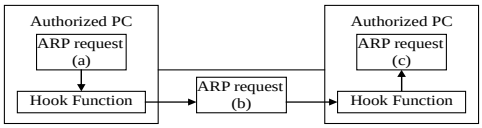


圖5 ARP 訊息格式（含 Ethernet header）

首先，如圖 6(a)在發送端已將一 ARP request 封包準備完成後，Hook function 將這封包攔下加工處理，我們使用 DES 加密技術（被加密內容以 8 bytes 為單位），將 ARP 格式之中傳送端 MAC(6 bytes)、傳送端 IP(4 bytes)、接收端 IP(4 bytes) 及接收端 MAC 最後 2 Bytes（共 16 bytes），如圖 5 描述的 ARP 訊息格式斜線部份，進行二次 DES 加密工程，加密後的密文按原順序填回原欄位中。

進行加密後並送出的 ARP request 封包如圖 6(b)，雖在發送者及接收者 MAC address 在 Ethernet header 中可被察覺，但因 ARP 格式中的 MAC 及 IP 均為密文，所以可以在網路傳送過程中全程將 IP 保密，使他人在網路上攔截封包者，不能獲得任何 IP 及 address binding 的任何訊息。在授權的接受端主機收到封包後，再經 Hook function 將該 ARP request 加密封包以 DES 將同樣欄位解密後回填即為圖 6(c)，回填後的 ARP 格式中之傳送端的 MAC(6 bytes)即為驗證欄位，若與 Ethernet header 的來源 MAC(6 bytes)相同，則為授權且正確的封包，並向上層傳送做進一步處理，反之則將封包攔截丟棄。



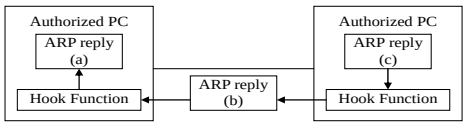
00000000: ff ff ff ff ff ff 00 e0 18 e4 99 d4 08 06 00 01
00000010: 08 00 06 04 00 02 00 11 43 3e 02 c9 cb 40 7f 2d
00000020: 00 e0 18 e4 99 d4 cb 40 7f 21
(a)

00000000: ff ff ff ff ff ff 00 e0 18 e4 99 d4 08 06 00 01
00000010: 08 00 06 04 00 02 **60 32 67 46 82 3d 7e f0 d7 8b**
00000020: 00 e0 18 e4 **56 b2 a5 8b 4f f5**
(b)

00000000: ff ff ff ff ff ff 00 e0 18 e4 99 d4 08 06 00 01
00000010: 08 00 06 04 00 02 **00 11 43 3e 02 c9 cb 40 7f 2d**
00000020: 00 e0 18 e4 **99 d4 cb 40 7f 21**
(c)

圖6 ARP request 封包加解密內容

同理，該 ARP request 封包經接收端處理後，接收端對原傳送端將以 ARP reply 封包回覆如圖 7(a)，該封包於傳送前亦於接收端以 DES 以相同程序加密後如圖 7(b)內容在網路上傳送，再於原傳送端以相同程序解密及驗證圖 7(c)的封包，決定是否接受或丟棄。在 Lan 上，某台主機並未本加解密管制軟體，該主機送出之 ARP request 封包為完全未加密，若被安裝本加解密管制軟體主機接收，在該主機以 DES 解密後，驗證即未能通過而被丟棄，故該未安裝本加解密管制軟體主機無法以 ARP request 封包獲取任何 address binding 訊息。同樣的，若由未安裝本加解密管制軟體主機接收經過加密的 ARP request 封包，因為 IP 訊息均已被加密，其無法有效回覆來獲取 address binding 訊息。綜合上述，對 ARP 封包進行上述處理已可完全達到 IP 訊息隱藏及管制 address binding 的成效。



00000000: 00 e0 18 e4 99 d4 00 11 43 3e 02 c9 08 06 00 01
00000010: 08 00 06 04 00 01 00 e0 18 e4 99 d4 cb 40 7f 24
00000020: 00 00 00 00 00 00 cb 40 7f fe
(a)

00000000: 00 e0 18 e4 99 d4 00 11 43 3e 02 c9 08 06 00 01
00000010: 08 00 06 04 00 01 **60 32 67 46 82 3d 7e f0 d7 8b**
00000020: 00 00 00 00 **56 b2 a5 8b 4f f5**
(b)

00000000: 00 e0 18 e4 99 d4 00 11 43 3e 02 c9 08 06 00 01
00000010: 08 00 06 04 00 01 **00 e0 18 e4 99 d4 cb 40 7f 24**
00000020: 00 00 00 00 **00 00 cb 40 7f fe**
(c)

圖7 ARP reply 封包加解密內容

5. IP 原理與應用

當主機透過 ARP request 封包的傳送，來達到獲取 address binding 訊息後，如同包裹已有了實際地址(MAC address)，後續包裹只要填入收件者姓名(IP address)便可將包裹寄送出去。如此，若 ARP 是 Lan 中扮演的第一道重要角色，那麼 IP 便是扮演第

二道重要角色。

在 Lan 上任何未安裝本管制軟體的主機，若正常運作之下是無法完成 address binding 資訊而達到傳送封包目的。但無可避免的是使用者可能經其他方法得知 binding table 中相關 IP address 與 MAC address 對應關係，以手動方式新增於該主機的 binding table 中。使該主機不需經 ARP request 來認證，但該主機與其他安裝管制軟體的主機也無法進行任何 IP 封包的發送。故本研究提出於 Lan 中任兩台主機上，於發送端對 IP 協定加密及於接收端對 IP 協定解密與認證機制，來達到上述目的。

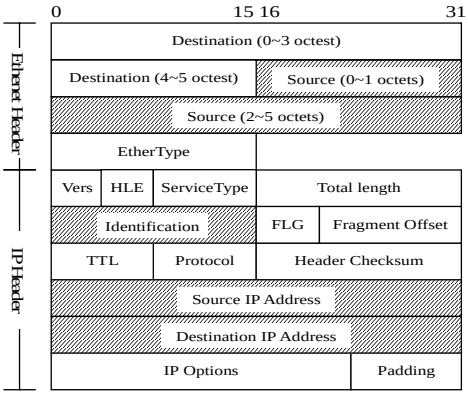
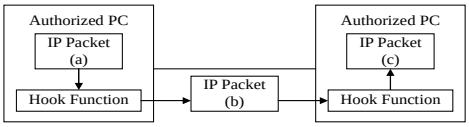


圖8 IP 加密與認證格式

我們對 IP 封包欄位內涵及格式進行分析，來決定哪些欄位要在發送端傳送前必須將其加密，以達到保密效果，並在接收端將相同的欄位解密並予以驗證，同樣的只有通過驗證 IP 封包才需要繼續往上傳送處理，未通過驗證的封包即給予丟棄。

IP 格式如圖 8(含 Ethernet header)，在 IP Header 與 Ethernet Header 中會顯示出兩組 address binding 訊息，分別是發送端的 MAC address、IP address 及接收端的 MAC address、IP address 兩組。使我們提出在 IP 封包發送之前可進行加密，但接收端的 MAC address 是給 NIC 接收識別用不能加密。



00000000: 00 0c 30 6a 3d 3f 00 e0 18 e4 99 d4 08 00 45 00
00000010: 00 3c 01 36 00 00 80 01 14 4b c0 a8 00 64 3d db
00000020: 26 59

(a)

00000000: 00 0c 30 6a 3d 3f **5e d4 86 6c e2 ec** 08 00 45 00
00000010: 00 3c **36 61** 00 00 80 01 **2c 0f c4 40 2d 45 0e 15**
00000020: **d9 b6**

(b)

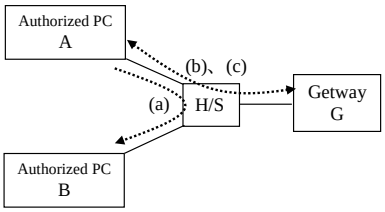
00000000: 00 0c 30 6a 3d 3f **00 e0 18 e4 99 d4** 08 00 45 00
00000010: 00 3c **01 36** 00 00 80 01 **14 4b c0 a8 00 64 3d db**
00000020: **26 59**

(c)

圖9 IP 加解密封包內容

當發送端在把將要發送的 IP 封包準備完成後即為圖 9(a)內容，而 Hook function 中會將 IP 封包攔下並使用 DES 加密技術處理，將 Ethernet Header 中接收端 MAC address(6 bytes)和 IP Header 的發送端 IP (4 bytes)、接收端 IP (4 bytes)及 Identification 的 2 bytes(共 16 bytes)，如圖 8 描述的 IP 訊息格式斜線部份，進行二次 DES 加密工程，加密後的密文以原順序置回原欄位中。而在 IP header 中，Checksum 欄為封包在傳輸過程中可能遭受破壞所使用的保護機制，由於 IP header 部份欄位已被 Hook function 加密，所以最後必須重新計算。

圖 9(b)封包顯示，若他人要在網路傳送全程中，從這些經加密過的 IP 封包中是無法獲得這兩組 address binding 資訊，雖然接收端的 MAC address 是可以從 Ethernet Header 中被得知的，但是其他的 MAC 及 IP 均為密文，所以儘管獲得這些資料也是一無用處。同樣的經授權的接收端主機收到封包後，會由 Hook function 將該已加密的 IP 封包以 DES 將資料正確解密後填回原欄位即為圖 9(c)。



	Source		Destination	
	Ethernet	IP Header	Ethernet	IP Header
(a) A→B	A's MAC	A's IP	B's MAC	B's IP
(b) A→G	A's MAC	A's IP	G's MAC	G's IP or External IP
(c) G→A	G's MAC	G's IP or External IP	A's MAC	A's IP

圖10 區域網路傳送及封包內容表示

從 Lan 中，我們從傳送方向與其 IP 封包內容表示方式，在傳送方向歸納分別為三種：主機傳送給閘道器、閘道器傳送給主機及主機與主機之間互傳。圖 10 描述各形式傳送 IP 封包所展現的 IP Header 與 Ethernet 各發送端和接收端內容。

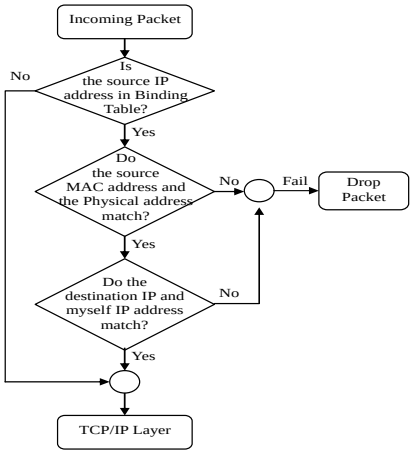


圖11 IP 認證流程

故本研究針對 IP 封包提出驗證的方法，來加強 IP 封包認證流程如圖 11 描述。當接收端接收進入的封包時，先檢查 IP header 的發送端 IP address 是否存在於該主機上的 Binding table 內？倘若 IP 不存於表格內，表示主機尚未獲得 address binding 資訊或者 IP 可能為外部 IP，因此直接將封包交付由上層決定是否需要對此 IP 再發送 ARP Request 封包或其他處理。若發送端 IP 存在於表格中，但是表格所對應的 MAC address 卻與 Ethernet header 的發送端 MAC address 不符時，表示封包的來源端主機已

被異動，即使將封包向上層傳送做進一步處理，其後續若有回覆封包也是無效的，所以將該封包丟棄。最後比對 IP header 與接收端本身 IP 是否相符，IP 以確定封包目地是否正確。

當主機傳送的 IP 封包是未經加密的 IP 封包，若被安裝本加解密管制軟體閘道器接收並以 DES 解密後，經驗證即未能通過而被阻攔丟棄，故該未安裝本加密管制軟體的主機無法透過閘道器連上 Interne。若該未安裝本加解密管制軟體的主機接收到加密過的 IP 封包，也會進行無效回覆，要是他人要從這些 IP 封包了解區域網段 IP 資訊也是無法獲得。針對 IP 封包進行上述處理已可完全達到 IP 訊息隱藏與認證機制。

6. UDP/TCP 與應用層原理與應用

由於 UDP 或 TCP 標頭後所裝載的資料大小會依上層應用程式服務所需而有所不同。如圖 12 中 UDP 格式所描述(含 IP header)，在 IP header 中定義整個封包總長度(Total length)及 IP header 長度(HLE)，相減後即為 IP 裝載資料大小，而且 IP header 也定義後續所使用上層協定類型，若 IP 上層協定為 UDP 協定，則將 IP 裝載大小值再減去固定長度的 UDP（8 bytes），即是 UDP 傳輸協定之後的應用層加密部份，以 DES 加密技術進行多次加密，在加密過程不足 8 Bytes 部份則從 TCP header 的來源埠(2 bytes)、目的埠(2 bytes)和序號(前 3 bytes)等共 1~7 Bytes 作為附加的加密部份，如圖 12 斜線部份。

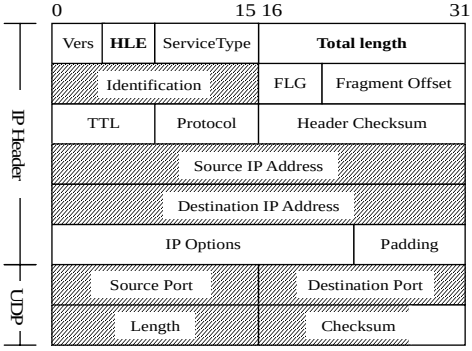


圖12 UDP 傳送格式

倘若 IP 上層協定為 TCP 協定，則將 IP 裝載大小值再減去 TCP header 定義 header 長度，如圖 13 的 TCP 格式(含 IP header)，即是 TCP 傳輸協定後的應用層加密部份，但並非全部 TCP 傳輸協定之後的應用層都要加密，只有 FTP 服務在傳送中，有 POST 指令時會要求對方 IP address 部份才進行加密工程。加密區域相同以 DES 加密技術進行多次加密。不足部份則必須從前面的 UDP header 前 1~7 Bytes 作為附加加密部份，如圖 13 TCP 斜線描述區域。

封包加解密是一項耗時間的處理動作，為避免將全部 UDP 或 TCP 傳輸協定之後的應用層皆運用加解密工程，而且又必須隱藏 IP 以達到保密效果。所以，我們針對圖 14 表格中的協定進行加密即可。

當 Lan 全面安裝本加解密管制軟體，網路上傳

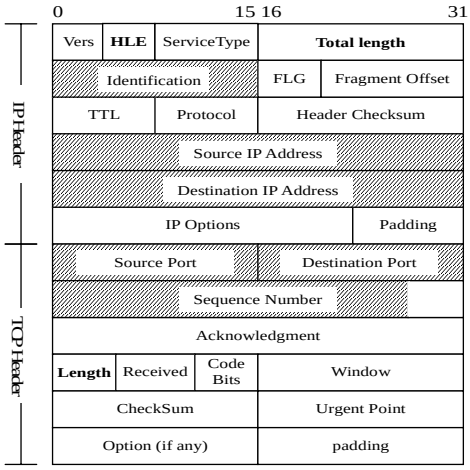


圖13 TCP 傳送格式

送的封包皆已加密。由於 ARP request 及 reply 封包，因為 IP 訊息均已被加密，其無法從中獲取 address binding 訊息；而在於 IP 封包的傳送端及接收端 IP 資訊也均已被加密，也無法獲取 Lan 的 IP。假使想從應用層著手去獲得資訊，也因本軟體針對這些服務協定進行加密，將許多重要資訊在傳送前先行加密，如此可達到全面性 IP 訊息隱藏及管制。

Network Layer	Transport Layer	Application Layer	IP Information
IP	UDP	DNS (53)	DNS: Response
		WINS (137)	WINS: Command & Response
		DHCP (67,68)	DHCP: Request & Reply
		NetBIOS (138)	BROWSER: NetBIOS Datagram
	TCP	FTP (21)	FTP: POST **xxx,xxx,xxx**

圖14 應用層

7. 實驗

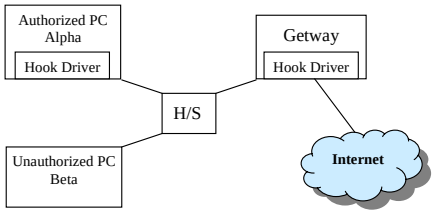


圖15 區域網路實驗架構

基於以上所提出對 TCP/IP 協定進行 DES 加解密技術與認證機制，我們建置小型 Lan 為實驗對象，實際撰寫出一套封包加解密管制軟體程式。我們在實驗 Lan 中，分別將其中一台扮演未經授權的主機名為 Beta；另一台則視為授權的主機名為 Alpha，並與閘道器皆安裝此封包加解密管制軟體程式，並透過 Hub 相互連結，利用 Sniffer 軟體截取封包內容，如圖 15 描述。

首先，我們將實驗的 Lan 內 Alpha 及 Beta 兩台主機的 TCP/IP 協定皆設定為自動取得 IP address 並完成重新開機。經由封包顯示，Alpha 送出 DHCP Request 封包到網路上，以尋求 DHCP 伺服器指派的 IP 及相關設定的 DHCP Response 封包回覆，如

圖 16(a)；反觀 Beta 經過多次 DHCP Request 封包發送，卻無法獲得 DHCP 伺服器有效回覆，如圖 16(b)。

Source Address	Dest Address	Summary
[202.23.251.87]	[217.12.211.34]	Expert: DLC source address multicast
		UDP: D=14080 S=44963 LEN=308 (missing data?)
[202.23.251.87]	[217.12.211.34]	UDP: D=14080 S=44963 LEN=308 (missing data?)
[202.23.251.87]	[217.12.211.34]	Expert: DLC source address multicast
		UDP: D=14080 S=44963 LEN=308 (missing data?)
[202.23.251.87]	[217.12.211.34]	UDP: D=14080 S=44963 LEN=308 (missing data?)
[202.23.251.87]	[217.12.211.34]	Expert: DLC source address multicast
		UDP: D=14080 S=44963 LEN=308 (missing data?)
[202.23.251.87]	[217.12.211.34]	Expert: DLC source address multicast
		UDP: D=14080 S=44963 LEN=308 (missing data?)
[202.23.251.87]	[217.12.211.34]	Expert: DLC source address multicast
		UDP: D=14080 S=44963 LEN=308 (missing data?)

(a)

Source Address	Dest Address	Summary
[0.0.0.0]	[255.255.255.255]	DHCP: Request, Message type: DHCP Discover
00112FF90E07	Broadcast	ARP: C PA=[192.168.0.9] PRO=IP
[192.168.0.1]	[255.255.255.255]	DHCP: Reply, Message type: DHCP Offer
[0.0.0.0]	[255.255.255.255]	DHCP: Request, Message type: DHCP Request
00112FF90E07	Broadcast	ARP: C PA=[192.168.0.9] PRO=IP
[192.168.0.1]	[255.255.255.255]	DHCP: Reply, Message type: DHCP Ack

(b)

圖16 DHCP 封包實驗結果

假使 Beta 自行建置 DHCP 服務或以手動設定方式獲得 IP address。我們測試 Alpha 和 Beta 兩台對於閘道器可否建立連線，執行 ping 閘道器指令。Alpha 本身因 Binding table 不存在閘道對應的 MAC address 資訊，所以 Alpha 自動送出 ARP Request 廣播封包到網路上。當閘道器接收到 ARP Request，則會將它相關資訊以 ARP Reply 封包有效回覆給 Alpha，如圖 17(a)。在 Alpha 完成 address binding 後，即發送 ICMP Echo 封包，且閘道器以 ICMP Echo reply 封包回應，如圖 18(a)；反觀，Beta 經多次發送 ARP Request 封包，依然未獲有效回覆封包，如圖 17(b)。在此顯示本加解密軟體成功的阻擋未授權 ARP 封包，並將經授權 ARP 封包有效回覆。

Source Address	Dest Address	Summary
AsustkE499D4	Broadcast	ARP: C PA=[248.250.155.252] PRO=IP
00112FF90E07	AsustkE499D4	ARP: R PA=[65.60.147.146] HA=2848CEE0D4DA PRO=IP

(a)

Source Address	Dest Address	Summary
AsustkE499D4	Broadcast	ARP: C PA=[192.168.0.1] PRO=IP
AsustkE499D4	Broadcast	ARP: C PA=[192.168.0.1] PRO=IP
AsustkE499D4	Broadcast	ARP: C PA=[192.168.0.1] PRO=IP
AsustkE499D4	Broadcast	ARP: C PA=[192.168.0.1] PRO=IP

(b)

圖17 ARP 封包實驗結果

假設當 Beta 知道閘道器相關的 Address binding 資訊，並藉由 arp 指令以手動方式將 Address binding 資訊加入 Binding table 中。所以，我們在 Beta 完成上述動作後再使用 Ping 指令功能，如圖 18(b)顯示。

Source Address	Dest Address	Summary
[252.252.204.53]	[37.156.83.135]	Expert: Source Address is Broadcast
		ICMP: Echo reply
[252.252.204.53]	[37.156.83.135]	Expert: DLC source address multicast
		ICMP: Echo reply
[252.252.204.53]	[37.156.83.135]	Expert: DLC source address multicast
		ICMP: Echo reply
[252.252.204.53]	[37.156.83.135]	Expert: DLC source address multicast
		ICMP: Echo reply

(a)

Source Address	Dest Address	Summary
[192.168.0.100]	[192.168.0.2]	ICMP: Echo
[192.168.0.100]	[192.168.0.2]	ICMP: Echo
[192.168.0.100]	[192.168.0.2]	ICMP: Echo
[192.168.0.100]	[192.168.0.2]	ICMP: Echo

(b)

圖18 IP 封包實驗結果

該 Beta 此次並未發送 ARP request 封包，而直接發送 ICMP Echo 封包，經多次的發送相同 ICMP Echo 封包，卻無法獲得閘道器的有效回覆 ICMP Echo reply 封包。此實驗結果顯示，儘管以手動方式將 address binding 資訊加入 Binding table 中，本加解密管制軟體仍然能有效阻擋 IP 封包的傳送。

最後，我們在實驗的 Lan 進行一段長時間的封包截取，並將可能在應用層顯示 IP 資訊的服務協定以未加密及加密方式描述。經 DES 技術加密後在網路上傳送的封包，完全無法看出原始內容為何，也確實達到封包 IP 資訊隱藏。

8. 結論及未來研究

本實驗藉由 TCP/IP 協定套件並以 DES 作為加解密轉換機制，建置一套支援 Windows 作業系統的封包管制軟體，並建置於整個 Lan 中經授權的主機及閘道器內。本系統能夠將 Lan 上傳送的封包內全面 IP 資訊加以隱藏，並給予認證機制來確認封包的合法性。幫助網管人員阻擋未經授權的主機在網路隨時隨地存取網路資源之行為。經由建構實驗環境的實驗結果，的確達到攔截這些未經授權主機所傳送的封包，且無任何有效回覆，並對經授權主機所傳送的封包給予有效轉換與回覆處理。

本篇論文未來可將單一金鑰加密技術擴充應用於通訊鑰匙加密技術，使通訊雙方於通訊前透過 ARP 的發送取得 Session Key 做為加密使用。

參考文獻

[1] Rinat Khoussainov, Ahmed Patel, "LAN security: problems and solutions for Ethernet networks", Computer Standards & Interfaces, Vol.22, No.2, 2000. pp.191-202.

[2] Timothy G. Shoriak, "SSL/TLS Protocol Enablement for Key Recovery", Computers & Security, Volume 19, Issue 1, 2000, Pages 100-104.

[3] Divine, T. F. (2002). "Windows Network Data and Packet Filtering", I. Printing Communications Assoc. http://www.ndis.com/papers/winpktfilter .htm

[4] Microsoft's Driver Development Kit (DDK) on line documentation - Network Drivers.

[5] Floroiu, J.W., Ionescu, T.C., Ruppelt, R., Henckel, B., Mateescu, M., "Using NDIS intermediate drivers for extending the protocol stack. A case study", Computer Communications , April 1, 2001, Vol 24, Issue: 7-8, pp. 703-715

[6] C. H. Tung, Wei-nan Lin, "Research on Windows Kernel and Network Drivers", Master's thesis of Department of Information Management of National Pingtung Institute of commerce, 2004

[7] Rowland, C. H. (1996). "Covert Channels in the TCP/IP Protocol Suite". http://www.firstmonday .dk /issues/issue2_5/rowland/

[8] WJ Buchanan and D Llamas, "Covert Channel Analysis and Detection with a Reverse Proxy Servers using Microsoft Windows", School of Computing, Napier University, EH10 5DT, Scotland, UK

[9] D.C. Plummer, "An Ethernet Address Resolution Protocol", Internet Standard RFC826, 1982.

[10] J. B. Postel, "Transmission Control Protocol, RFC 793", Information Sciences Institute, Sept. 1981.