IJCSC International Journal of Computer Science & Communication Vol. 1, No. 1, January-June 2010, pp. 171-173

System Design for Packet Sniffer using NDIS Hooking

[1]Muna M. Taher Jawhar & [2]Monica Mehrotra

[1-2] Department of Computer Science, Jamia Millia Islamia, New Delhi, India
[1]muna.taher@gmail.com, [2]drmehrotra2000@gmail.com

ABSTRACT

Intrusion detection is a problem of great significance to protect information system security, especially in view of the worldwide increasing incidents of cyber attacks. The ability of an NIDS to classify a large variety of intrusions in real time with accurate results is important. Every NIDS need to capture the packet of internet by packet sniffer. In this paper a new program for packet sniffer is proposed that will help researchers to capture internet packet in real time and to retrieving information of any field in any headers they want to work on. The program work is in user-mode that makes it easy to use by using NDIS hooking application. This program can monitor, control, add, and modify the NDIS incoming/outgoing packet. This work is based on TCP/IP protocol suite because it is the most famous and important current communication protocol.

Keywords: Network Security, Packet Sniffer, Hooking Method.

1. INTRODUCTION

Due to the great wide spread used of computer network and because most of the institution's computer network are using Local Area Network(LAN) topology and connection to the internet, this connection is not away from dangers caused by the intruding of some persons to discover others privacy and secrets, steeling information from the network and possibly caused damages. Accordingly, network security has become one of the most important interesting areas for researcher. Protecting the network can be done by many mechanisms. Among the most effective one is the network firwall and intrusion detection systems[1].

In all windows OS, the protocol TCP/IP is proprietary. The most important and famous practical communication model is the Transmission Control Protocol/Internet Protocol(TCP/IP)[2][3]. Since the windows OS deals with the network through a network cards and besides there are many different card brands, Microsoft has developed the Network Deriver Interface Specification (NDIS) in order to be as standardize interface between the TCP/IP protocol stack and the network adapter card driver.

NDIS is intended to define a standard API for network interface card (NIC). It includes a library of functions that can be used by higher level protocol driver (such as TCP/IP). These functions facilitate the development of protocol drivers and to hide dependencies on a computer platform[1].

We present in this paper design system for packet sniffer by using hooking application, in this system we can capture the packet and control to its in real time, and we can display any field in any headers protocol we want. This can help the researchers in intrusion detection system. Finlay the proposed system implementation was developed by using Visual C++ language.

2. PACKET SNIFFER TOOLS

When setting up or debugging a Network Intrusion Detection System (NIDS), it is vital to ensure that we are monitoring the traffic for the subnet to which we are connected. There are several program which can do this job, some of the tool are[4][5].

- *Snort:* It is an open source network intrusion prevention and detection system (IDS/IPS) developed by Sourcefire. Combining the benefits of signature, protocol and anomaly based inspection, we can used it as packet sniffer from command line (www.snort.org).
- *Ethereal:* It is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible. The software is available from web site (www.ethreal.com).
- *Tcpdump:* It is a common packet analyzer that runs under the command line. It allows the user to intercept and display TCP/IP and other packets being transmitted or received over a network to which the computer is attached. Tcpdump works by capturing and displaying packet headers and matching them against a set of criteria. Tcpdump works on most Unix-like

operating systems: Linux, Solaris, BSD, Mac OS X, HP-UX and AIX among others. In those systems, tcpdump uses the libpcap library to capture packets. The website is (www.tcpdump.com).

- *Windump:* It is the Windows version of tcpdump, the command line network analyzer. WinDump is fully compatible with tcpdump and can be used to watch, diagnose and save to disk network traffic according to various complex rules. It can run under Windows 95, 98, ME, NT, 2000, XP, 2003 and Vista. WinDump captures using the WinPcap library and drivers and the website is (http://windump.polit.it).

All these tools just display the packet information without availability to change or control the packet and for large networks, it would be necessary to store Gigabytes of event data every day. Our new program can capture the packet of internet in real time, display the field which we want, monitor any field in the specific header, and control the incoming packet. Even we can control the outgoing packet by make some change to the program.

3. NETWORK DRIVER INTERFACE SPECIFICATION (NDIS)

In 1989, Microsoft and 3com jointly developed the NDIS, which lets protocol drivers communicate with network adapter drivers in a device independent manner[10]. So the NDIS device driver is related to the OSI model at the data link layer [5]. NDIS standardizes access to the network card, so that the same software may be used to access any brand of network device [11]. In fact, it become a requirement, when developing device driver for network card in all varieties of windows[5]. The NDIS is a interface between a protocol stack and network adapter card driver[9]. NDIS intermediate drivers can see all network traffic taking place on a system because the drivers lie between protocol drivers and network drivers[4]. Its provides drivers to interface with network adapter hardware via functions call API functions. As shown in figure(1).

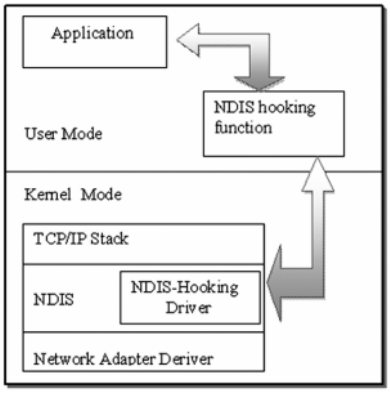

Fig 1: NDIS Hooking Driver with Relation to user Mode

4. NDIS HOOKING ALGORITHM

This program is a user-mode application, it controls the hooking driver, which means controlling the packet traveling.

The hooking algorithm have the following functions:

i. Loading the hooking driver from the specific adapter.

ii. Controlling the original NDIS driver in order to give the ability for moving the packet from kernel-mode to user-mode.

iii. Continuing receiving operation.

The function in the user mode which do this jobs are programmed in a Dynamic Link Library(DLL) file using visual C++ language. The first step in the hooking application is loading the function from DLL then the hooking driver is loaded from the specified adapter, by using the API function:

GetTcpipBoundAdaptersInfo()

In order to know that the packet is reached to the specified adapter, an event must be created and banded with the specified adapter. So when packet event occurred, then the application knows that a new packet has received, which is the NDIS packet, by using the following function:

CreateEvent

If we failed to create the event, we stop listing to the adapter.

And then check if the packet is incoming and not outgoing because we want only incoming packet, after that, remove the packet to the buffer then we began read the field which we want from the specific headers like IP header, TCP header, UDP header, and ICMP header. The field which we want to retrieve from the headers as we need in the next paper is as following:

From IP header following information can be achieved:

- Source-IP address
- Destination-IP address
- IP-protocol
- Check-sum

From TCP header get:

- Source-port
- Destination-port

From UDP header following information can be achieved:

- Source-port
- Destination-port

From ICMP header get:

- *Message type.*

The next figure show the procedure of the program.

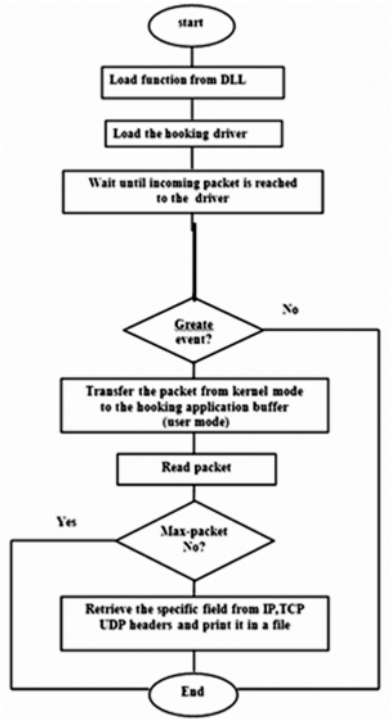

Fig 2: The Hooking Procedure

The information of the field was saved in file type (dat) as result of the program to used it in the IDS. As description in the following figure(3).

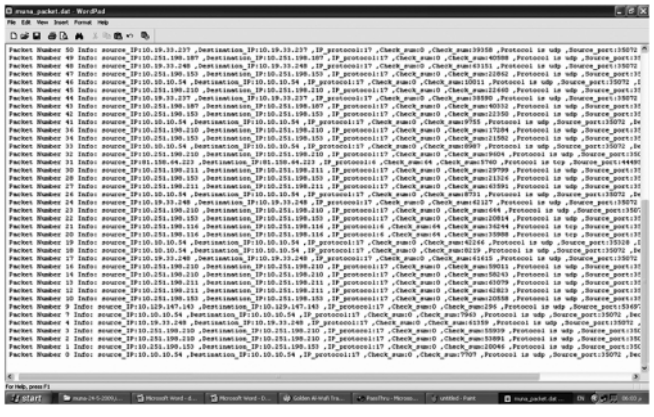

Fig 3: The Result of Hooking Program

5. CONCLUSION AND FUTURE WORK

There are many available tools used to capture network traffic that researcher used in their work, but there is a limitation in their work. Some tools only capture network traffic without analysis, therefore the researcher have to use another tools for analysis to get the traffic feature like it is need in his work. Our system capture network traffic and analyze it and allows user to take only the feature as he need and store it in file to use it later in his work, then this will reduce the memory that is used to store the data.

In future work we want to design network intrusion detection system using neural network and we need to capture network traffic in real time, and we will use this system to capture the network traffic and get only the feature that we want it and store the data in a file to use it later in our NIDS.

REFERENCES

[1] AL-Dabbagh, Omar, "Implementation and Analysis of a Software System for protection of Local Area Network from internal Intruder", Ph.D. thesis, Department of Computer Science, University of Mousl, Iraq, 2006.

[2] Anderson F. and Karlsson M., "Security Jini Services in Ad Hoc Networks", Royal Institute of Technology, 2000.

[3] Forouzan B., "TCP/IP Protocol Suite", third edition, Tata McGraw-Hill, 2006

[4] Andrew R. Baker, Brian Caswell, Mike Poor, "Snort 2.1 Intrusion Detection Second Edition", Shroff Publishers & Distributors PVT. LTD., 2004.

[5] Kerry Cox and Cbristopher Gerg, " Managing Security with Snort and IDS Tools", 2004.

[6] Ries C., "Defeating Windows Personal Firewalls: Filtering Methodologies, Attacks, and Defenses", 2005

[7] Dhawan S., "Network Device Drivers", Van Nostrand Reinhold, 1995.

[8] Wolthusen S., "Tempering Network Stacks", Security Technology Department Fraunhoferstr, Germany, 2004.

[9] Oney W., "Programming the Microsoft Windows Driver Model", Microsoft Press, 1999.

[10] Barkley W. and Macdonald D., "Microsoft Windows 2000 TCP/IP Implementation Details", Microsoft Corporation, 2000.

[11] Moller J. and Donbaek T., "Internal Network Security", Department of Computer Science at the University of Aarhus, 2001.