International Journal of Electronics Communication and Computer Engineering
Volume 3, Issue 6, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

The Design and Implementation of a Packet Sniffer (PSniffer) Model for Network Security

Awodele Oludele
Babcock University, Ilishan-Remo, Ogun State, Nigeria

Otusile Oluwabukola
Babcock University, Ilishan-Remo, Ogun State, Nigeria

***Abstract* — This paper presents another type of packet sniffer software that captures network data as well as provides sufficient means for the decision making process of an administrator. This work designed a new model and defined its benefits over existing packet sniffers; the model was developed in Java totally. The aim of this model is to rewrite C language sniffer models into Java, and also develop a model that consumes little memory on the hard disk. This model comprises of five independent modules that handles different tasks efficiently using Winpcap and JPCAP for sniffing. ARP cache poisoning method is used for sniffing in this model. The proposed system does not transmit any data onto the network, uses 1MB of the hard disk space, friendly GUI and it is very easy to install.**

***Keywords* — Network traffic, Packets, Packet capture, Packet sniffer.**

I. INTRODUCTION

In computer communication, packets can be defined as a quantity of data of limited size. In Internet all traffic travels in the form of packets, the entire file downloads, Web page retrievals, email, all these Internet communications always occur in the form of packets. The packet is a formatted unit of data carried by a packet mode in computer network

A packet is a series of digital numbers basically, which conveys the following: The source IP address and port; the destination IP address and port; error checking information; and usually some sort of information about the type and status of the data being sent [20].

In many networking protocols, transmitted data gets split into small segments, or packets, and the Internet Protocol address of the destination computer is written into the header of each packet. These packets then get passed around by routers and eventually make their way to the network segment that contains the destination computer. As each packet travels around that destination segment, the network card on each computer on the segment examines the address in the header. If the destination address on the packet is the same as the IP address of the computer, the network card grabs the packet and passes it on to its host computer [18].

A packet analyzer sometimes called a network analyzer, protocol analyzer or sniffer or Ethernet sniffer or wireless sniffer [21][25], is a computer program or a piece of computer hardware that can intercept and log traffic passing over part of a network [1].

II. WHY THE USE OF A NETWORK SNIFFER

The information running through networks is a valuable source of evidence for network administrators to fish out intruders or anomalous connections. The need to capture this information has lead to the development of packet sniffers.

A number of research works exist in the development of packet sniffers. However, the search for the ideal packet sniffer continues. Psniffer will come with additional functionalities such as 3D pie charts, a GUI and with little memory requirements.

Psniffer when installed in a network will help monitor network traffic and keeps log of all connections to the network, which is then analyzed for the detection of suspicious activities.

III. PACKET SNIFFER TOOLS

Several tools exist that can monitor network traffic, usually such tools will put the network card of a computer into promiscuous mode, this enables the computer to listen to the entire traffic on that section of the network. Filtering of this packets can be done based on the IP related header data present in the packets, usually such filtering specifies simple criteria for the IP addresses and ports present in the packets. These passive network sniffing programs have been developed for either wired or wireless network measurement; the best-known are tcpdump and Wireshark.

i. Tcpdump By McCanne, Leres and Jacobson

It is one of the most popular packet sniffers. Tcpdump is accompanied by the libpcap library. It was originally written in 1987 at the Lawrence Berkeley National Laboratory and published a few years later and quickly gained users attention.

Libpcap is a C library for capturing packets. The procedures included in libpcap provide a standardized interface to all common (UNIX-based) operating systems, including Linux and FreeBSD. The interface of the libpcap is usable even under Windows but there the library is called winpcap [23].

Tcpdump is a common packet analyzer that runs under the command line and parsing tool ported to several platforms. 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.

It runs on most UNIX-like operating systems - e.g. Linux, BSD, Solaris, Mac OS X, HP-UX and AIX amongst others making use of the libpcap library to capture packets.

ii. Wireshark by Gerald Combs

It is a free and open- source packet analyzer and it is written in C. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Originally named **Ethereal**,

in May 2006 the project was renamed Wireshark due to trademark issues [22].

Wireshark is very similar to tcpdump, but has a graphical front-end, plus some integrated sorting and filtering options.

It allows the user to put network interface controllers that support promiscuous mode into that mode, in order to see all traffic visible on that interface, not just traffic addressed to one of the interface's configured addresses and broadcast/multicast traffic. However, when capturing with a packet analyzer in promiscuous mode on a port on a network switch, not all of the traffic traveling through the switch will necessarily be sent to the port on which the capture is being done, so capturing in promiscuous mode will not necessarily be sufficient to see all traffic on the network. Port mirroring or various network taps extend capture to any point on net; simple passive taps are extremely resistant to malware tampering.

IV. Limitations of Existing Packet Sniffer Software

Tcpdump is a command-line network sniffing and parsing tool ported to several platforms. Wireshark is similar to tcpdump, but with a graphical user interface and many advanced sorting and filtering options. TcpDump is very economical in terms of memory since its installation file size is just 484 KB. TcpDump does not have a user friendly Graphical User Interface (GUI). So the user has to study those commands and get acquainted with the command prompt like screen [23].

That limitation may play a key role in not choosing it for use. On the other hand Wireshark has a very good user friendly GUI, but its installation file size is 18 MB and after installation it will consume 81 MB in Windows and a hefty 449 MB in Linux. So in terms of memory requirements, it is very expensive.

V. The Proposed System

The proposed system will be written in Java unlike the other Sniffers that are written in C language. It will capture packets and size of the packet and source and destination machine IP addresses which are involved in the packet transferring. It can show this process in graphical manner. It also shows the working of different layers in graphical manner. It gives complete information about the captured packets; like which layers are involved and which protocols are in use at a particular time. Finally, it will have a facility to store the information of the packets.

VI. The Feature of PSniffer

Psniffer is a customized software application that has a number of features. These features enable:

- Administrators to show statistics of received packets
- Administrators detect malicious IP addresses according to its number of ARP requests in previously specified time
- Administrators to view all network interfaces and enable them to capture data from that interface and consequently save captured packets.
- Administrators generate reports that aid effective and efficient decision making.

The proposed sniffer will be totally developed in Java™[14][15]. This application will be designed in five independent modules which will take care of different tasks efficiently.

1. User Interface Module.
2. Packet Sniffing Module.
3. Analyze layers Module.
4. Free Memory Module.
5. Protocol Analysis Module.

1. **User Interface Module**: Actually every application has one user interface for accessing the entire application. The user interface for the Psniff application is designed completely based on the end users. It provides an easy to use interface to the users. This user interface has an attractive look and provides ease of navigation. Technically, the swing is used in core java for preparing this user interface.
2. **Packet Sniffing Module**: This module takes care of capturing packets that are seen by a machine's network interface. It grabs all the packets that goes in and out of the Network Interface Card (NIC) of the machine on which the sniffer is installed. This means that, if the NIC is set to the promiscuous mode, then it will receive all the packets sent to the network.
3. **Analyze layers Module**: This module contains the code for analyzing the layers in the system. Mostly in this module we have to discuss about three layers Transport layer, Application Layer, Network Layer. The module shows the graphical representation of the usage of different layers in packet capturing time. It can show the graph in two manners like line graph and pie graph.
4. **Free Memory Module**: This module analyzes computer memory usage at the time of packet capturing. It can show the memory size in number format as well as graphical representation.
5. **Protocol Analysis Module**: This module analyzes the protocols of the layers. Like TCP, UDP, HTTP etc. It can show the source port, destination port and packet length of the system of each protocol.

VII. Installation

Installation on Windows requires WinPcap software which can be downloaded from winPcap website [24]. Jpcap is a set of Java classes which provide an interface and system for network packet capture; it is required for packet capture in Java and built upon Libpcap which is a packet capture library in C language. Java Runtime Environment (JRE) 5.0 or higher will also be required to run this Java application. JFreeChart is another java library required for rendering 3D pie chart for captured packet

International Journal of Electronics Communication and Computer Engineering
Volume 3, Issue 6, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

statistics. More space may be required to store the captured packets since the required space on hard disk for installation is less than 1MB.

The fig above shows the user interface of the sniffer, where the device captured model will be determined either Wan or Lan.

VIII. SCREENSHOT OF PSNIFFER

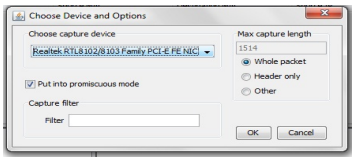

Fig.1. The main GUI of PSniffer

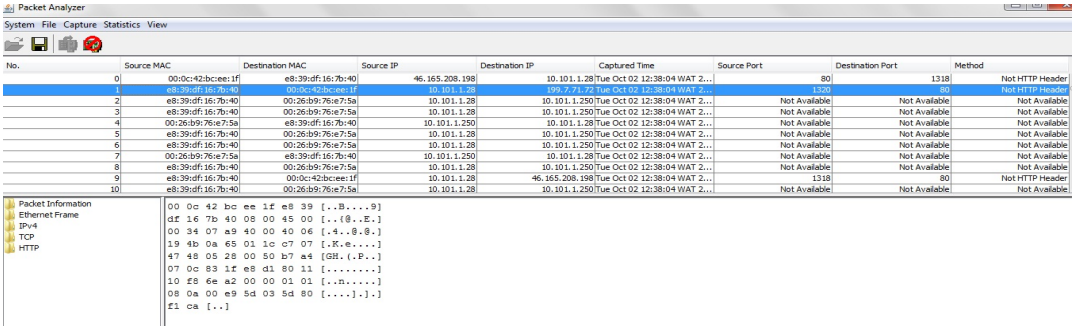

Fig.2. Captured packets containing necessary information

The figure above shows details of the captured packets showing the Source Mac and IP addresses, Destination Mac and IP addresses and methods of system on the network as at the time it was sniffed.

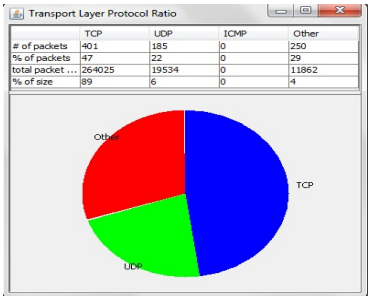

Fig.3. 3D pie chart showing received packet characteristics on transport layer

The figure above shows the pie chart (percentage, total packets and size) of protocols used on the transport layer as at the time the network was sniffed.

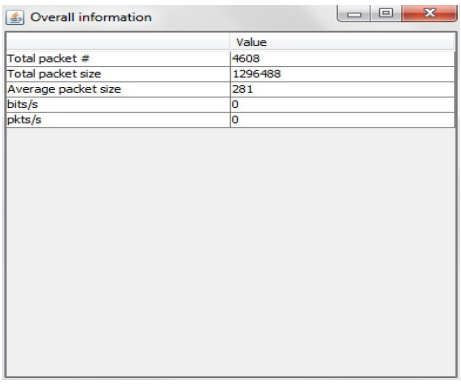

Fig.4. Overall information sent

The figure above shows the overall information of the packets sent over the network as at the time it was sniffed.

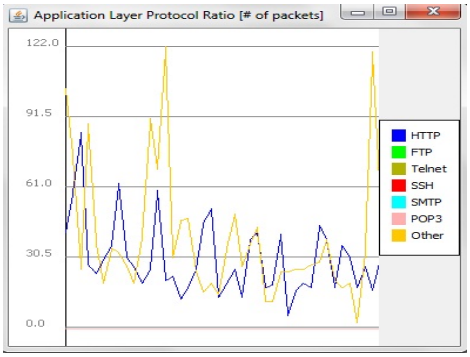

Fig.5. Graph showing the ratio of used supported application

The figure above shows the graph of the protocols (http, ftp, telnet) used on the application layer as at the time it was sniffed.

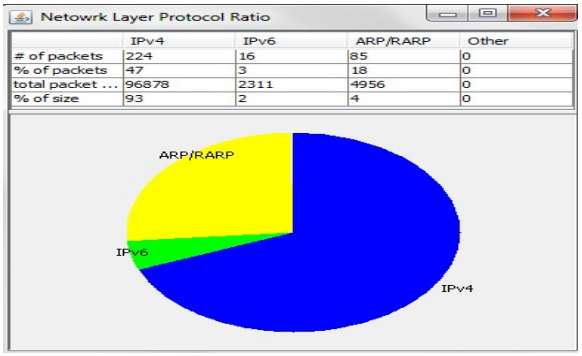

Fig.6. 3D pie chart showing received packet characteristics on Network Layer

The fig above show the Pie Chart (percentage, total packets) of the internet protocol type used on the network as at the time the network was sniffed.

IX. RESULTS

Compared to similar works this models show the layer involved in sniffing and the protocols. In this model sniffing is done based on layers unlike other models that analyzes based on protocols only. This model displays individual graphs of the layers that is sniffed showing the protocols, size of the packet, percentage and total number of the packets sniffed at real time.

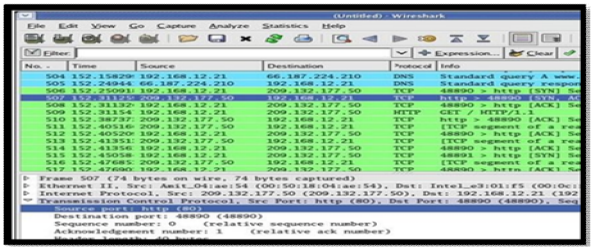

Fig.7. Output display of Wireshark

The fig above displays the Wireshark model, showing the packet information and the protocols without showing the layers in which the protocols run on.

X. CONCLUSION AND FUTURE WORK

There are many available tools used to capture network traffic, but there are limitations in some of the tools. Some tools only capture network traffic without analysis, while some require large memory size for installation therefore the researcher has to use other tools for analysis to get the traffic features as required and also consider the memory size of the system in use. Our system captures network traffic and analyzes it and allows the user to take only the features he needs. Our system requires little memory size for installation and enables the user to store his/her selected features in a file for later use in his/her work. Consequently, this will reduce the memory that is used to store the data. Finally, PSniffer contains additional functionalities like 3D pie chart statistics and possible malicious IP address detection.

The full implementation of this work is in progress, which would be seen in the upcoming paper

REFERENCES

[1] Ansari, S., Rajeev, S., & Chandrashekar, H. (2002). Packet Sniffing: A Brief Introduction. IEEE Potentials (Vol. 21, Issue 5, pp. 17-19).

[2] Asrodia, P. & Patel, H. (2012). Network Traffic Analysis Using Packet Sniffer International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com (Vol. 2, Issue 3, pp.854-856)

[3] Brozycki, J. (2010). "Capturing and Analyzing Packets with Perl".

[4] Chan, C. Y. (2002). *A network packet analyzer with database support*. Retrieved from http://www.cs.rpi.edu/~szymansk/theses/chan.ms.02.pdf

[5] Dabir, A. & Matrawy, A. (2007). "Bottleneck Analysis of Traffic Monitoring Using Wireshark", 4th International Conference on Innovations in Information Technology, 2007, IEEE Innovations '07 (pp. 158 – 162)

[6] Deri, L. (n.d.). Improving passive packet capture: Beyond device polling. Retrieved from

[7] http://www.net-security.org/dl/articles/Ring.pdf

[8] Dhar, S. (2002). "Switchsniff". Retrieved from http://www.linuxjournal.com/article.php

[9] Flor, N.V. & Guillory, K. (2011). Technology Corner: Internet Packet Sniffers Journal of Digital Forensics, Security and Law, Vol. 6(1).

[10] Fuentes, F. & Kar, D. (2005). "Ethereal vs. Tcpdump: A Comparitive Study on Packet Sniffing Tools for Educational Purpose," Computer Journal of Computing Sciences in Colleges, (Vol. 20, Number 4, pp. 169-176).

[11] JFreeChart. (n.d.). JFreeChart. Retrieved from http://www.jfree.org/jfreechart/download.html

[12] Jpcap. (2011). Jpcap. Retrieved from http://jpcap.sourceforge.net/

[13] JRE. (n.d.). JRE. Retrieved from http://www.oracle.com/technetwork/java/javase/downloads/index.html

[14] Kjell, B. (n.d.). Introduction to Computer Science using Java.

[15] Lewis, J. & Loftus, W. (2001). Java Software Solutions, Addison Wesley.

[16] McCanne, S. & Jacobson, V. (1992). The BSD Packet Filter: A New Architecture for User-level Packet Capture.

[17] Muna, M., Jawhar, T. & Mehrotra, M. (2010). System Design for Packet Sniffer using NDIS Hooking, International Journal of Computer Science & Communication (Vol. 1, No. 1, pp. 171-173).

[18] Niphadkar, S. (2006). *Analysis of Packet Sniffers – TCPdump VS Ngrep VS Snoop*

[19] Parmar, R. & Patel, H. (2011). *NetCap: A Packet Sniffer in Java*, International Journal of Computer Science and Technology (Vol. 2, Issue 3).

[20] Senthil, K.P. & Arumugam, S. (2012). *Establishing a valuable method of packet capture and packet analyzer tools in firewall*, International Journal of Research Studies in Computing 2012 April, (Vol. 1, Number 1, pp. 11-20)

[21] Spangler, R. (2003). Packet sniffer detection with antisniff. Retrieved from http://www.linux-sec.net/Sniffer.Detectors/snifferdetection.pdf

[22] TcpDump. (2009). Overview of TcpDump. Retrieved from http://www.tcpdump.org/

[23] Wireshark. (2009). Wireshark: Introduction. Retrieved from http://www.wireshark.org/

[24] Winpcap. (2009). Sniffers: Wincap. Retrieved from http://www.wipcap.org/download

[25] Wikipedia. (2012). *Packet Sniffer*. Retrieved from http://en.wikipedia.org/wiki/Packet_sniffer

[26] Wikipedia. (2012). *Ethernet*. Retrieved from http://en.wikipedia.org/wiki/Ethernet

[27] Wikipedia. (2012). Packet analyzer. Retrieved from http://en.wikipedia.org/wiki/Packet_analyzer

AUTHOR'S PROFILE

Oludele Awodele Ph.D.
is presently the head of the department of computer science & mathematics, Babcock University, Ilishan-Remo, Ogun State, Nigeria. His research areas are Software Engineering, Data Communication and Artificial Intelligence. He has published works in several journals of international repute. He can be contacted at delealways@yahoo.com.

Otusile Oluwabukola
Received a B.Sc. degree in Computer Technology from Babcock University 2009, and currently awaiting M.Sc. degree in Computer science from Babcock University 2013. She can be contacted at buhkieotusile@yahoo.com