A New Hybrid Network Sniffer Model Based on Pcap Language and Sockets (Pcapsocks)

Azidine GUEZZAZ, Ahmed ASIMI, Yassine SADQI, Younes ASIMI and Zakariae TBATOU
Laboratoire LabSiv: Systèmes d'information et vision
Equipe SCCAM : Sécurité, Cryptographie, Contrôle d'Accès et Modélisation
Department of Mathematics and Computer Sciences
Faculty of Sciences, Ibn Zohr University, B.P 8106, City Dakhla, Agadir, Morocco

***Abstract*—Nowadays, the protection and the security of data transited within computer networks represent a real challenge for developers of computer applications and network administrators. The Intrusion Detection System and Intrusion Prevention System are the reliable techniques for a Good security. Any detected intrusion is based on data collection. So, the collection of an important and significant traffic on the monitored systems is an interesting feature. Thus, the first task of Intrusion Detection System and Intrusion Prevention System is to collect information's basis to treat and analyze them, and to make accurate decisions. Network analysis can be used to improve networks performances and their security, but it can also be used for malicious tasks. Our main goal in this article is to design a reliable and powerful network sniffer, called PcapSockS, based on pcap language and sockets, able to intercept traffic in three modes: connected, connectionless and raw mode. We start with the performances assessment performed on a list of most expanded and most recently used network sniffers. The study will be completed by a classification of these sniffers related to computer security objectives based on parameters library (libpcap/winpcap or libnet), filtering, availability, software or hardware, alert and real time. The PcapSockS provides a nice performance integrating reliable sniffing mechanisms that allow a supervision taking into account some low and high-level protocols for TCP and UDP network communications.**

Keywords*—*Network Security; Intrusion Detection; Intrusion Prevention; Sniffing; Filtering; Network sniffer; Libpcap; Libnet; Sockets

I. Introduction and Notations

The sniffing is a technique of monitoring every packet that crosses the network. A packet sniffer is a piece of software or hardware that monitors all network traffic. Network analysis is the process of listening and analysis of network traffic. It controls network communications to identify performance problems, locates security vulnerabilities, analyzes the behavior of the application, and performs capacity planning. The management and the supervision of exchanged data by network systems are a fundamental task that contributes to a reliable intrusion detection and analysis of intrusive activities detected. The sniffing tools are used to listen, monitor, capture, record, and analyze network traffic. They extract the necessary information's to make decision and implement the best strategies to improve the computer security. Many sniffers are available to capture packets circulated in wired networks (Ethernet sniffers, for example) and wireless network. They help network managers to assess and review the data over their networks, to indicate the network problems and to identify some failures monitored network [16]. The sniffing is also exploited by attackers to gather a database of information on victims' networks and hosts that constitute them. They can intercept the data and even the users' passwords [28] [31]. The goal of this paper is to describe a new hybrid sniffer for a relevant collection of data. For this, we proceed as follows: the second section cites an art state on the techniques of network sniffing; a related study will be developed on some sniffers. In the third section, the performances assessment is performed on a list of network sniffers based on various parameters to ensure the computer security objectives, mainly the type of used library, libpcap or libnet to establish the performances and limitations of each library and finally validate our choice. A classification is deducted at the end of the section. The proposed network sniffer works in raw, connected and connectionless mode; it will be described in part four. A detailed description of the new model and its characteristics will be discussed in fifth section. This work will be finished by a conclusion and perspectives. Trough this paper we use the following notations:

IP:	Internet Protocol.
IDS:	Intrusion Detection System.
IPS:	Intrusion Prevention System.
ARP:	Address Resolution Protocol.
NIC:	Network Interface Cards.
MAC:	Media Access Control.
CRC:	Cyclic Redundancy Check.
PDU:	Protocol Data Unit.
BSD:	Berkeley Software Distribution.
BPF:	Berkeley Packet Filter.
LSF:	Linux Socket Filter.
FFPF:	Fairly Fast Packet Filter.
CGF:	Control Flow Graph.
ATM:	Asynchronous Transfer Mode.
ISDN:	Integrated Services Digital Network.
FDDI:	Fiber Distributed Data Interface.
RFMON:	Radio Frequency MONitor.
SSID:	Service Set Identifier.
WEP:	Wired Equivalent Privacy.
WAP:	Wireless Application Protocol.
PSK:	Pre-Shared Key.
OSI:	Open Systems Interconnection.
LLC:	Logical Link Control

DOS:	Denial of Service.
SOCK_DGRAM:	Datagram Sockets.
SOCK_STREAM:	Stream Sockets.
SOCK_RAW:	Raw Sockets

II. Related Work

In this section, we discuss the sniffing types, components of the sniffing tools, used libraries to capture network packets and the used methods to filter the captured traffic.

The network monitoring is a difficult and demanding task. It is an essential part in the use of network administrators who are trying to maintain the good operating of their networks and need to monitor the traffic movements and the network performances [21] [29]. The sniffing listens to public conversations in computer networks. It is used by network managers to manage and ensure the network security. It can also be used by unauthorized users. Mostly, this device is placed between the server and the clients web pages, it listens and analyzes all sent and received requests by the server. Sometimes, a network sniffer is called a network monitor or a network analyzer [30]. There are different types of sniffing packets [3] [30]:

- IP sniffing: collects all IP packets traveled through a network corresponding to the IP addresses of supervised entities.
- MAC sniffing: captures the corresponding frames to supervised interfaces MAC addresses.
- ARP sniffing: intercepts the ARP packets used to query the ARP cache during network communication.

Generally, the sniffing is divided into two major classes: passive sniffing that collects raw traffic circulated in the network without treatments, and active sniffing that intercepts and treats the collected traffic [3][28]. The sniffers monitor a wide sent and received information by computer networks. There are many commercial and no commercial tools, hardware and software that enable to intercept packets [30]. The copies of captured packets are stored in a temporary (buffers) or permanent memory (database server). They are analyzed to extract the useful information or specific models (patterns). The amount of captured traffic depends on the location of the controlled host as well a primary server in a computer network intercepts a significant traffic than isolated system client. The sniffers operate in two distinct ways: with filtered way to capture the data containing a specific elements and an unfiltered way to collect all the raw network traffic. Some network topologies such as Ethernet are designed so that all machines connected to a network segment share the same transmission media, thus, the hosts that are connected to the same network segment will be able to see all traffic passing through that segment. Ethernet hardware is designed to filter the traffic passed, it captures the traffic which concerns it or has a broadcast addresses and ignores all other traffic. This is done using the MAC address. To copy all traffic, the host network cards have to be implemented in promiscuous mode [3] [16]. The hardware sniffers use the standard adapter's NIC, otherwise they may face problems in the CRC error, voltage and cabling problem. The analysis of captured packets is often done in real time. The captured traffic may be submitted to a decoding operation to be descriptive and understandable text for easy interpretation. Sometimes, the sniffers edit the packets and transmit them to the network. The security aspect done by sniffers is represented by their availability to monitor and capture the traffic in and out of the network taking into account the clear text passwords and user names [4]. Besides, the network sniffers participate in detecting and identifying of the intrusions by monitoring the activities of networks and systems [14] [16] [31]. They are constituted by the components described by Clincy & Abi Halaweh in [3] [16]:

- Hardware: is represented by a NIC, activated in sniffing mode.
- Driver: starts the capturing data from the network cards, applies a number of filters on traffic and stores it in a memory.
- Buffer: stores the captured traffic or transfers it to permanent storage.
- Analyzer: is software responsible for analyzing the traffic in real time taking into accounts the criteria and analysis needs.
- Decoder: receives a stream of bits and interprets them to finally build a descriptive texts format.
- Editor: is available in some sniffers, it changes the traffic using a unified format and then converts it and retransmits it in the network.

The sniffers can be used effectively for teaching and learning networking concepts regardless of the technical context. They are presented to understand the model and protocols of network layers [26]. They allow to:

- Examine the format of a protocol data unit (PDU) to each layer in the network model.
- Examine the message exchanges for two TCP or UDP connections.
- Examine the messages transferred between a client application and a server.

The simulation with packet sniffers is used in learning of computer networking, allows a good understanding of network concepts, topologies and explains the functions and the roles of a hub, a bridge or switch and a router. It shows how a data packet is transmitted into LAN and illustrates the encapsulation and decapsulation operations while going through the protocol stack [31]. The main capture libraries are libnet and libpcap [1] [2] [22]:

TABLE I. LIBPCAP AND LIBNET LIBRARIES

	Libpcap	***Libnet***
Capture level	• Captures the packets in low level. • Extracts the packet so raw kernel without treatments.	• Manipulates a high level traffic. • Can manipulate a several low level networking routines.
Used mode	• Conected mode (TCP) • Connectionless mode (UDP)	• Conected mode (TCP) • Connectionless mode (UDP)
Filtering	• Compatible filtering with the BPF filter. • Initializes and configures filters. • Receives the packets using a loop.	• Doses not provide a packet filtering.
Supported protocols	• Supports almost of networking protocols.	• Injects any kind of an IP packet. • Manipulates a network firewall (IP filter, ipfw, ipchains, pf, PktFilter, ...). • Offers the addresses manipulation functions, the ARP cache and routing tables. • Manipulates an IP tunnel (tun BSD / Linux, Universal TUN / TAP device).
Errors management	• Provides the functions to manage errors.	• Provides the functions to manage errors.
Supported platforms	• Supported by all platforms.	• BSD (OpenBSD, FreeBSD, NetBSD, BSD / OS), Linux (Redhat, Debian, Slackware, etc.), MacOS X (Windows NT / 2000 / XP), Solaris, IRIX, HP-UX, Tru64.

The filtering is an essential operation to classify the captured packets using filters according to the needs of capture. When the packets are intercepted, a filtering is applied. The packet that respects the filter is stored. The capture filters are useful to limit the captured packets when concentrated on a specific packet type, the packets that meet the filter criteria are elected [32]. Among the criteria are used to filter a packet, we find: type packet used (IP, TCP, UDP, ICMP, ...), address of input or output interface, address of source or destination of packet, the number of source or destination port of application, …. The filter is a Boolean function which returns true if the traffic is accepted; otherwise, returns false (the traffic is ignored or rejected). For example, to apply the filtering, the operating system use a packet filter like the BSD Packet Filter for Open BSD systems [13] and the LSF filter for Unix platforms. To improve the filtering operation, several filters are implemented; we cite a result of a recent research on filters packets, the rapid filter FFPF (Now Streamline) [15]. Multiple filters can be loaded simultaneously in FFPF. To design a filter, two basic approaches are available: tree model and direct acyclic graph (CGF) model used by Berkeley Packet Filter [13]. The filtering can be classified into two types:

- The static filtering initializes the filter parameters to be applied in advance. It is provided for example by the pcap language [6] maintained and developed by researchers at the Lawrence Berkeley National Laboratory and enables the use of simple rules to remove the unwanted packets.
- The dynamic filtering implements the parameters that change during running. The filter Swift or Fast Dynamic Packet Filter is an example of dynamic filter [12] [27].

III. STUDY AND PERFORMANCES ASSESSMENT

This section will study a performance evaluation of a proposed list of sniffers setting up parameters related to computer security. It is completed by a classification.

A. Assessment Parameters

Normally, to realize an assessment performances and classify the various sniffers which use wired and/ or wireless networks, many criteria are available such as, supported platforms, operating systems and interfaces, user interface, number of protocols that the network sniffer can decode, available utilities to enable the user to personalize capturing and displaying network packets, support for customized protocol decodes, readability of captured data, provided statistical information, decoding captured data, …. Our main objective in this work is to propose an approach to improve the security level. So, our study is based on parameters related to computer security that test the sniffers availability and their reliability. It is useful to recall that the sniffing requires the activation of interfaces in promiscuous mode for wired networks and in rfmon mode for wireless networks.

To compare and evaluate the proposed tools, we focus on evaluation characteristics dependent on computer security cited in [16] [28] [33].

- Availability: to test the availability of a sniffer, three parameters are cited:
  - Operating time of the sniffer.
  - Memory size allocated to the implementation of the sniffer (as the size increases, the treatment requires a lot of time).
  - Maximum controlled flow by the sniffer [34]. The table II bellow illustrates the different network technologies with their supported maximum flows.

TABLE II. NETWORK STANDARDS AND MAXIMUM FLOWS

Network	Standard	maximum flow
Wired Networks	Ethernet	Megabits, gigabits
	ISDN	Low flow Services: 64Kbps to 2Mbps
	ATM	High flow Services: 10Mbps to 622Mbps.
	FDDI	100 Mbps
	Token Ring	4 Mbps to 16 Mbps
Wireless Networks	802.11a	54 Mbit / s with a range of 10 m.
	802.11b	11 Mbit / s with a range of 10 0 m.
	802.11g	54 Mbit / s with a range of 100 m.
	802.11n	Frequency 2.4GHz and 5GHz

- Filtering: verifies the existence of a filtering system to filter the traffic.
- Used library: determines the used library by the sniffer to capture traffic: libnet or libpcap.
- Supported protocols: means the number of protocols taken by a sniffer.
- Alert: an alert will be produced, if a problem exists in the controlled segment,
- Real Time: the treatment in real time is a parameter of an effective sniffing [3] [25].

B. Classification of the Sniffers

We refer to the study treated in [3] [4] [5] [6] [7] [8] [9] [10] [11] [16] [17] [18] [19] [25] [26] [28] and [32] and we deduce the classification of sniffers according to the characteristics and proposed parameters:

TABLE III. CLASSIFICATION OF SNIFFERS

Network sniffers	S/ H	Library	Filtering	Flow	Availability	Alert	Real time
Tcpdump	S	Libpcap (Winpcap)	++	Flow of Ethernet networks	Very economical installation file size: 484 KB	- -	- -
Wireshark	S	Libpcap (Winpcap)	++	Flow of Ethernet and wireless networks.	81 MB after installation.	- -	++
PACKETYZER	S	Libpcap (Winpcap)	++	Flow of Ethernet, FDDI, PPP, Token Ring and wireless networks.	-supports 483 protocols. -Decodes and edits packets.	- -	++
Netflow CISCO	S H	Libpcap	++	High flow networks (Gigabit).	Very high (provides valuable information about users, network applications, peak hours). -2GH Dual processor. -2GO Memory.	++	++
Colasoft Capsa	S	Libpcap	++	Flow wired and wireless networks over 802.11a, 802.22b, 802.11g and 802.11n	-No Tolerant with the attacks: ARP, TCP port scanning, -Signals DOS attacks - Free version is available with limited features.	++	++
PRTG Network Monitor	S H	Libpcap	++	High flow	-653 MB on after windows 7 installation. -Integrates SNMP, Packet (Sniffing and Net flow). -monitors 24/7 network. - Includes over 200 types of sensors. -Less than 30 protocols (Free). - More than 30 protocols (Com)	++	++
Kismit	S	Libpcap	++	Flows of wireless networks 802.11n, 802.22b 802.11g and 802.11a	High (supports any wireless card rfmon)	++	++
Scapy	S	Libpcap and Libnet	++	Injectes the 802 frames	- Generates and receives quick and accurate traffic. - Decodes packets of a number of protocols.	++	++
OmniPeek	S H	Libpcap	++	Ethernet, Gigabit, 10 Gigabit, 208.11 a / b / g / n / ac wireless, VoIP, Video, MPLS and VLAN	-captures on multiple networks simultaneously. - Several hundred protocols - WPA, WPA2 and PSK Decoding.	++	++
ETHERAP	S	Libpcap	++	Flows of Ethernet, FDDI, Token Ring, ISDN.	- Is only available for GNU / Linux systems.	- -	++
Soft Perfect Network Protocol Analyzer	S	Libpcap	++	Flows of Ethernet networks	-Analyzes of fragmented floors. -defragments and reassembles the packets. - Size of the installation file 4.87 Mb.	- -	++
Airodump	S	Libpcap	++	- Wireless networks 802.11. - Supports 4.2 GHz channels	-Identification the coordinated access points. -Writes the several files containing details of all seen access points and clients.	- -	++

Com: Commercial license. Free: Free license. ++: Available. - -: Not available.
H: Hardware. S: Software.

C. Discussion of the Results

This section cites the architecture of many sniffers, their characteristics and their operating. We assess their performances based on parameters related to security objectives: authentication, confidentiality, integrity, availability and rapidity. Really, it's difficult to meet this assessment, because normally the goal of sniffing is not to indicate the problems and attacks but to collect the circulated traffic in the networks and sometimes to inform the state of the monitored network excepting some IDS sniffers that can detect intrusions. The majority of these sniffers use libpcap library to intercept traffic and include a filtering system. They are highly available to monitor wired and wireless networks with a high flows supporting a large number of protocols. The treatments are often in real time and the detected problems are alerted by some sniffers. On the other side, this study helps us to discover certain limitations of those sniffers. They are based on a passive sniffing. Sometimes, they are exploited for unauthorized uses, for example, Airodump that is designed to crack WEP and WPA encryption algorithms; it is used to encrypt traffic on wireless networks. The implementation of software sniffer by interpreted languages such as Python presents a slow in their performance and increases consequently the system. The encrypted and fragmented packets are intercepted by sniffers but they are not analyzed. The hardware sniffers have adaptation and compatibility problems [3]. In the next section, we describe in detail the new model of a network sniffer.

IV. Our Proposal Scheme Pcapsocks

In this section, the proposed model of sniffing is cited. We prove that our proposal takes into account the benefits of a reliable collection of traffic to satisfy the current expectations. It is time to formulate a new proposition of network sniffer. Our model, called PcapSockS, based on pcap language and sockets satisfies. It decodes the intercepted traffic to prepare it for the analysis step and finally built a collection database for automatic intrusion detection. Specifically, it ensures two major tasks:

- Collects the data traffic in high and low level.
- Builds a database for the new proposed IDS/IPS.

The new design focuses on the combination of current performances of high sniffers and minimization of various limitations. Thus, we propose a distributed model consisted by two main components:

- The kernel is composed by two processors, the first to capture the traffic and the second for filtering.
- The operator decodes the elected traffic using the functions and treatments of normalization.

These components are described in the figure1 below:

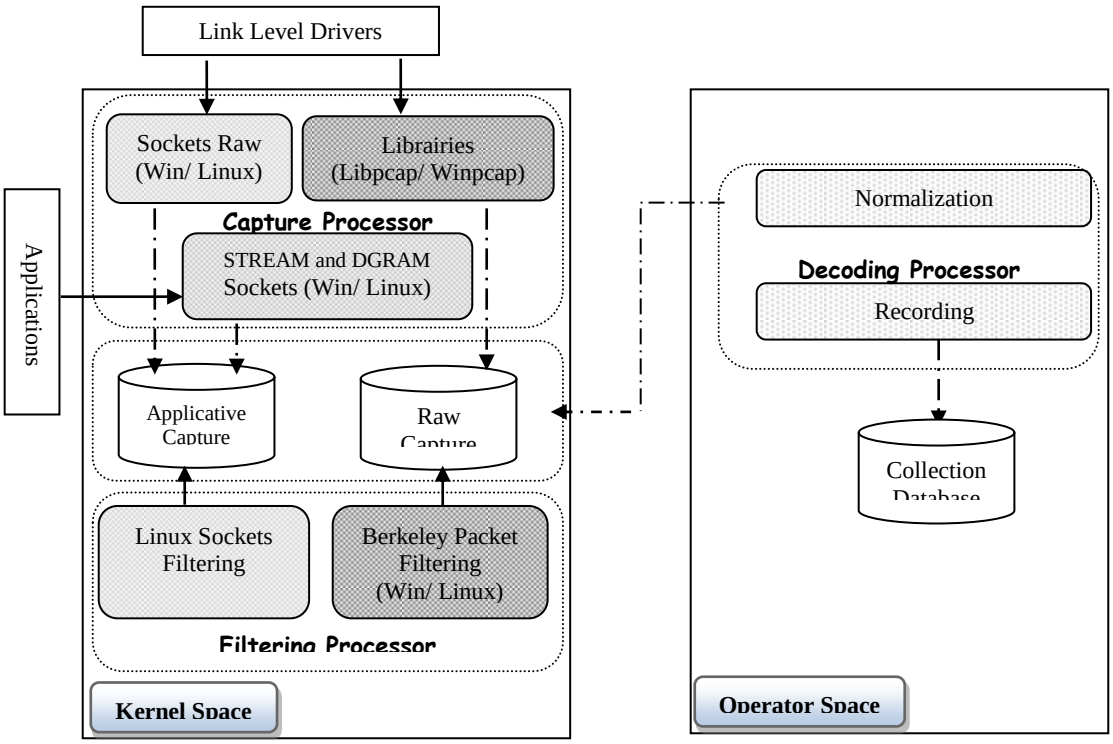

Fig. 1. Model of a network sniffer PcapSockS

The above design can be implemented in the Linux and Windows platforms, for the Berkeley Packet Filtering filter is an extension of Linux Sockets Filter [23]. So, the provided functions by the LSF are taken into account by the PBF in the case of windows. With this new design, we provide an optimal sniffer for capturing, filtering, optimization and decoding traffic while enabling the large satisfaction of various specificities and open the horizons for other works trying to improve the computer security techniques. The figure2 shows the flows exchanged between the various processors:

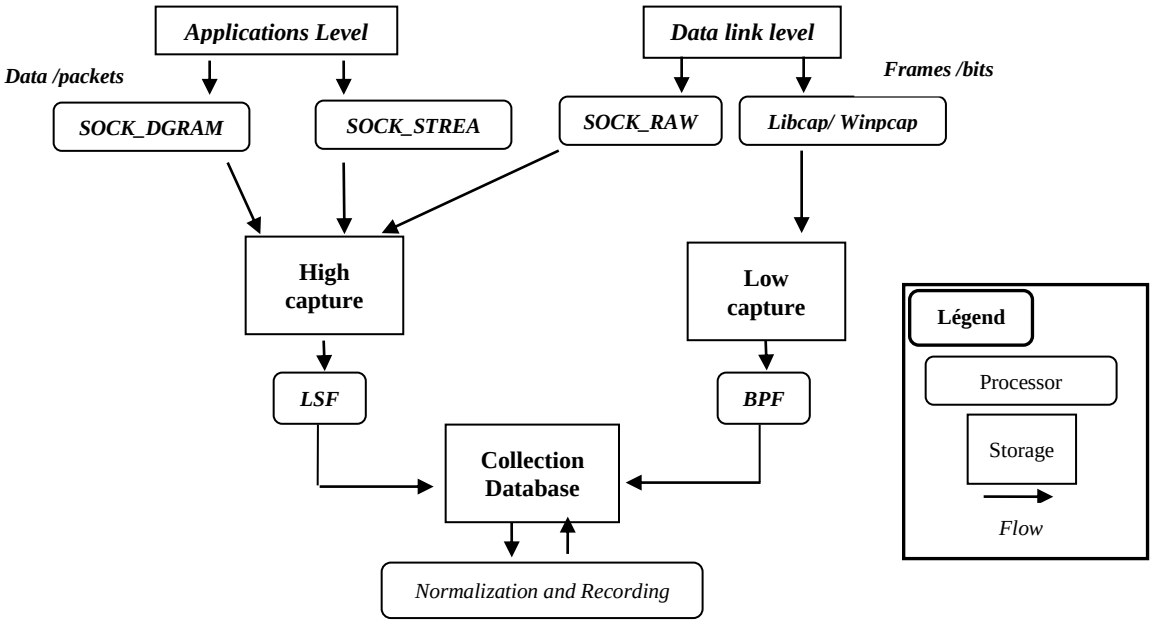

Fig. 2. Data flow diagram

Our new model provides different performances:

- Combining libpcap and sockets functions to capture the packets.
- Filtering traffic taking into account the capture needs.
- All treatments are in real-time.
- Encryption of transactions between the sniffer and Collection database.

The next section details the decoding operations used by the PcapSockS, it shows the performances provided by this new probe by comparing it with the Scapy and Wireshark which are considered the most famous sniffers in the moment.

V. Detailled Description

A. Processing Operations

The libpcap library is an open source library written in C that provides a programming interface from which the packets are intercepted [22]. It relies on a low level language, includes the functions that can be associated with the user request and provides a powerful and abstract interface for the capture process [24]. The process used by libpcap is defined by the following figure:

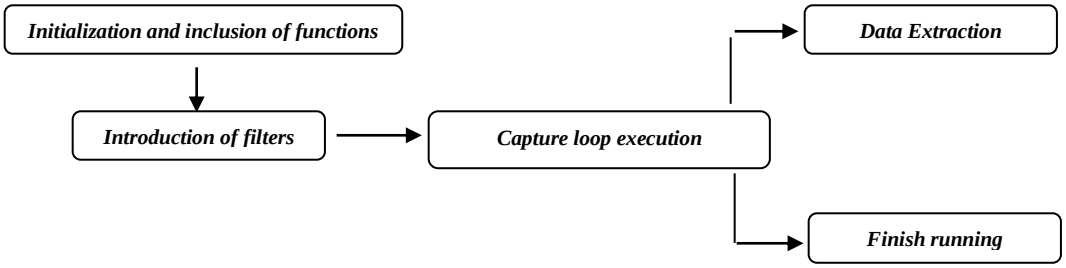

Fig. 3. Capture process provided by libpcap

The Sockets are the objects for sending and receiving messages between processes. They were developed by Berkeley in 1982 as part of the Berkeley version of Unix. The Sockets are the specific original Unix systems; they ensure the communication between various processes, applications and network layers. The main socket types are:

- SOCK_DGRAM: connectionless sockets (UDP messages).
- SOCK_STREAM: connection oriented sockets (TCP packets).
- SOCK_RAW or Raw sockets (frames and bits): The IEEE 802.2 protocol defines the sub layer LLC of the data link layer.

The sockets in connectionless and connection oriented mode are inserted between the layers 3 and 4 of the OSI model. The raw sockets are positioned in layers 1 and 2 [23].

The filtering is an essential process of checking the integrity of the kernel traffic. The copies of the collected traffic can be minimized by deploying a kernel agent called a packet filter that rejects unwanted packets [13]. The traffic can be ignored and blocked using one of the techniques used for the blocking of data [20].

B. Description of Solutions

The PcapSockS Sniffer integrates libpcap to intercept traffic from the low-level, physical and data link layers of the OSI model. This traffic is composed of a set of bits and frames, it's saved in a temporary basis to apply the BPF filter and then meet adequate collection conditions. Libpcap provides the possibility to introduce the filters to filter traffic: PBF, SWIF [12]. It applies the filters on traffic in the basis in order to choose the elected packets. This latter is redirected to the operator space. The decoding processor normalizes and stores the chosen traffic in the collection Database. In the high level, we use the sockets mechanism to ensure a reliable collection. The TCP and UDP sockets are implemented for this purpose. Raw sockets are used to reinforce the interception to the low level with libpcap. The collected traffic is saved in a temporary basis to apply the filter LSF and redirected directly to Collection Database. So, our sniffer collects data in three modes:

- Connection oriented mode requires a prior connection establishment between communicating entities; this connection is defined by a logical relationship between the parts which exchange data.
- Connectionless mode cannot guarantee a reliable connection, insertion errors, wrong delivery, duplication, or non sequencing delivery packets. These faults can be reduced by providing a reliable transmission service to a protocol layer of the highest level.
- Raw mode can provide both services in connection oriented and connectionless mode.

The filtering provides a considerable gain; it avoids the congestion and the saturation of memory. The filtering is a very useful to meet the various network services using mainly in intrusion detection [14] [31]. The PcapSockS Sniffer implements the filtering operations on collected traffic taking into account the parameters and attributes characterizing the monitored entities. The treatments are in real time. Take into account the time constraints which are as important as the accuracy of the results for this system synchronizes multiple tasks that take place and the possibility of including several shorter threads in a single process [25].

To show the performances provided by the PcapSockS Sniffer, it is very useful to compare it with other sniffers which have demonstrated their reliability, we cite Scapy and Wireshark.

TABLE IV. COMPARISON OF PCAPSOCKS WITH SCAPY AND WIRESHARK

Sniffer	Platforms	Low capture	High capture	Low filtering	High filtering	Network
Scapy	-Win -Linux -Mac OS	-Libpcap	-Libnet -Python Functions	-PBFfilter	-No	-Wired -Wireless
Wireshark	-Win -Linux	-Libpcap	-No	-PBF Filter	-No	-Wired -Wireless
Pcap.Sock Sniffer	-Win -Linux	-Libpcap -Raw Sockets	- Sock_Stream -Sock_Dgram	-PBF Filter	-LSF Filter	-Wired

VI. CONCLUSION AND PERSPECTIVES

There are many available tools used to capture network traffic that researchers use in their work, but there is a limitation in their functions. Some tools capture network traffic only without analysis. Therefore, the researchers have to use another tool for analysis to get the traffic feature like it is need of his work. In this article, we studied in detail the discipline of sniffing which is an interesting task but is difficult to put in place taking into account the various needs. The sniffing enables improved security of computer networks and systems that compose them.

This study provides a list of popular sniffers to evaluate and to deduct the existed limits. Thus, a classification is provided based on the parameters cited in the second section, related to computer security: availability, traffic filtering, real time, used library and flow.

We propose this software for the data collection part of our new intrusion prevention system approach based on neural network. So, we describe in detail its objectives, roles of its various components and nature of modes used for sniffing. Our future work is to implement and validate the steps of PcapSockS Sniffer and integrate this sensor in IDS/IPS.

REFERENCES

[1] Yan Grunenberger, Thesis "Réseaux sans fil de nouvelle génération : architectures spontanées et optimisations inter-couches", 7 Jun 2010.

[2] http ://libdnet.sourceforge.net/.

[3] ms.sonali, a.karale, ms.punam, p.harkut, "packet sniffing" international journal of pure and applied research in engineering and technology IJPRET, 2014; Volume 2 (9): 654-661.

[4] Pallavi Asrodia and Hemlata Patel, "Analysis of Various Packet Sniffing Tools for Network Monitoring and Analysis", Department of Computer Science and Engineering, Jawaharlal Institute of Technology, Borawan, Khargone, (M.P.), International Journal of Electrical, Electronics and Computer Engineering 1(1): 55-58(2012).

[5] All about Wireshark [Online] Available http://www.wireshark.org/.

[6] All about Tcpdump [Online] Available http:// www.tcpdump.org/.

[7] http://www.colasoft.com/capsa/.

[8] https://www.kismetwireless.net/.

[9] http://www.paessler.com/prtg.

[10] http://www.secdev.org/projects/scapy/.

[11] David Rideau, "Outils de collecte pour réseaux gigabits Une alternative à la technologie Cisco Netflow", Département Réseau du CICG (Centre Inter-Universitaire de Calcul de Grenoble).

[12] Zhenyu Wu, Mengjun Xie, Member, IEEE, and Haining Wang, Senior Member, IEEE "Design and Implementation of a Fast Dynamic Packet Filter", IEEE/ACM TRANSACTIONS ON NETWORKING, 2011.

[13] S. McCanne and V. Jacobson, "The BSD packet filter: A new architecture for user-level packet capture," in Proc. Winter USENIX Tech. Conf., 1993, pp. 259–269.

[14] V. Paxson, "Bro: A system for detecting network intruders in Real-Time", vol. 31, no. 23–24, pp. 2435–2463, Dec. 1999.

[15] H.Bos, W.de Bruijn, M. Cristea, T. Nguyen, and G. Portokalidis,"FFPF: Fairly fast packet filters," in Proc. USENIX OSDI, 2004, pp. 347–363.

[16] Clincy; Abu Halaweh. "A taxonomy of free network snifers for teaching and research", Journal of Computing Sciences in Colleges, Volume 21 , Issue 1, pp 64-75, 2005.

[17] All about soft perfect network protocol analyzer [Online] Available http://www.softperfect.com/products/networksniffer/

[18] http://etherape.sourceforge.net/

[19] http://www.aircrackng.org/doku.php?id=airodump-ng

[20] B.Suneel Kumar, S.V.V.D Venu Gopal, M.Satish Kumar, "Blocking Technique of Dataflow in Networks", ASR Engineering College, Tanuku, W.G Dist, and Andhra Pradesh.

[21] AlishaCecil, "A Summary of Network Traffic Monitoring and AnalysisTechniques", acecil19@yahoo.com

[22] Alejandro L´opez Monge , "Aprendiendo a programar con Libpcap", kodemonk@emasterminds.net, 20 de Febrero de 2005

[23] Christophe Gimenez,"MiniSniff Application de capture de trames,V.A.E Algorithmique N1/N2 – Réseaux", DESS C.C.I. 2003-2004

[24] Fulvio Risso and Loris "An Architecture for High Performance Network Analysis", Degioanni Dipartimento di Automatica e Informatica – Politecnico di Torino Corso Duca degli Abruzzi, 24 – 10129 Torino, Italy

[25] Manas Saksena, "Conception de logiciel en temps réel – Progrès actuels et défis à venir", Université Concordia, et Bran Selic, ObjecTime Limited

[26] Bruce P. Tis , "Using packet sniffing to teach networking concepts", Simmons College, Boston Ma Journal of Computing Sciences in Colleges, Volume 30 Issue 6, June 2015, Pages 67-74

[27] Zhenyu Wu, Mengjun Xie and Haining Wang "Swift: A Fast Dynamic Packet Filter", The College of William and Mary, NSDI '08:5th USENIX Symposium on Networked Systems Design and Implementation

[28] Dr. Charu Gandhi, Gaurav Suri, Rishi P. Golyan3, Pupul Saxena, Bhavya K. Saxena, "A Packet Sniffer – A comparative study", VOL. 2, NO. 5, MAY 2014, 179–187 Available online at: www.ijcncs.org ISSN 2308-9830

[29] SB .A. Mohammed, Dr.S.M Sani, Dr. D.D. DAJAB, "Network Traffic Analysis: A Case Study of ABU Network", Computer Engineering and Intelligent Systems, ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online), Vol.4, No.4, 2013

[30] Rupam, Atul Verma, Ankita Singh, "An Approach to Detect Packets Using Packet Sniffing", International Journal of Computer Science & Engineering Survey (IJCSES) Vol.4, No.3, June 2013

[31] X.Yuan, P.Vega, Jinsheng Xu, Huiming Yu, and Stephen Providence "An animated simulator for packet sniffer", , Department of Computer Science, North Carolina A&T State University, 1601 East Market St., Greensboro, NC 27411

[32] L.Chappell, "Wirehark Network Analysis",The Official Wireshark Certified Network Analyst™ Study Guide Second Edition San (Version 2.1b).

[33] Y.Farhaoui, A.Asimi, "Performance method of assessment of the intrusion detection and preventionsystems", International Journal of Engineering Science and Technology (IJEST), ISSN : 0975-5462, Vol. 3 No. 7 July 2011.

[34] Claude Duvallet, "Les réseaux informatiques", Université du Havre UFR Sciences et Techniques 25 rue Philippe Lebon - BP 540 76058 LE HAVRE CEDEX Claude.Duvallet@gmail.com.